A GRAPH-BASED CORE MODEL AND A HYBRID RECOMMENDER SYSTEM FOR TV USERS

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

BY

ARDA TAŞCI

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

FEBRUARY 2015

Approval of the thesis:

A GRAPH-BASED CORE MODEL AND A HYBRID RECOMMENDER SYSTEM FOR TV USERS

submitted by **ARDA TAŞCI** 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. Hakan Ferhatosmanoğlu _____________
Computer Engineering Dept., BILKENT UNIVERSITY

Assoc. Prof. Dr. Pınar Karagöz _____________
Computer Engineering Dept., METU

Dr. Ayşenur Birtürk _____________
Computer Engineering Dept., METU

Date: 05.02.2015

ABSTRACT

A GRAPH-BASED CORE MODEL AND A HYBRID RECOMMENDER SYSTEM FOR TV USERS

Taşcı, Arda
M.S., Department of Computer Engineering
Supervisor: Prof. Dr. Nihan Kesim Çiçekli

February 2015, 92 pages

This thesis proposes a core model to represent user profiles in a graph-based environment which can be the base of different recommender system approaches as well as other cutting edge applications for TV domain. The proposed graph-based core model is explained in detail with node types, properties and edge weight metrics. The capabilities of this core model are described in detail. Moreover, in this thesis, a hybrid recommender system based on this core model is presented with its design, development and evaluation phases. The hybrid recommendation algorithm which takes unique advantages of different types of recommendation system approaches such as collaborative filtering, context-awareness and content-based recommendations, is explained in detail. The introduced core model and the hybrid recommendation system are evaluated and compared with a baseline recommender system and the results are presented. The evaluation results show that the core model and the recommender system presented in this work produce remarkable results for TV domain.

Keywords: Recommender systems, collaborative filtering, context-aware recommendation, content-based recommendation, graph-based recommendation, TV program recommendation, graph-based user profiling.

ÖZ

ÇİZGE TABANLI BİR ÖZ MODEL VE TV KULLANICILARI İÇİN MELEZ ÖNERİ SİSTEMİ

Taşcı, Arda
Yüksek Lisans, Bilgisayar Mühendisliği
Tez Yöneticisi: Prof. Dr. Nihan Kesim Çiçekli

Şubat 2015, 92 sayfa

Bu çalışmada TV kullanıcılarını temsil etmek için çizge tabanlı bir altyapı oluşturulmuş ve bu çizge tabanlı alt yapının yetenekleri açıklanmıştır. Bunun yanı sıra, bu çizge tabanlı altyapı üzerine melez bir öneri sistemi geliştirilmiş ve tasarım, geliştirme ve değerlendirme aşamaları detaylı olarak aktarılmıştır. Geliştirilen melez öneri sistemi içerik tabanlı, bağlan duyarlı ve işbirlikçi öneri sistemleri gibi farklı öneri sistemi yaklaşımlarının olumlu yanlarını kullanarak kullanıcılara öneri sunan bir sistemdir. Geliştirilen bu sistem ile aynı veriler üzerinde geliştirilen temel bir öneri sistemi karşılaştırılmış ve önerilen sistem dikkat çekici sonuçlar ortaya koymuştur.

Anahtar Kelimeler: Öneri sistemleri, işbirlikçi öneri sistemleri, bağlam duyarlı öneri sistemleri, içerik tabanlı öneri sistemleri, çizge tabanlı öneri sistemleri, TV program öerni sistemleri, çizge tabanlı kullanıcı modelleme.

To my family *with love and respect*…

ACKNOWLEDGEMENTS

First and foremost, I would like to express my sincere gratitude to my advisor Prof. Dr. Nihan Kesim Çiçekli. I am grateful for her endless support, patience and encouragements and for sharing truthful and illuminating views on a number of issues related to the research. Without her assistance and dedicated involvement in every step throughout the process, this research would have never been accomplished. Having the chance to work together with her has reshaped my point of view on academic disciplines and opened up my horizon intellectually.

I also take this opportunity to express my deep gratitude to every member of Arçelik Ar-Ge Tasarım Ofisi. Being a part of this innovative, communicative and energetic team has added my life the value of trusting the people I work for, having pride in what I do and enjoying the people I work with. Especially, I would like to express my special appreciation and thanks to Mr. Burak Demirtaş who has been a tremendous mentor, great leader and true friend for me. Without his support, guidance and illuminating thoughts, this research would be worthless.

I would like to thank Scientific and Technological Research Council of Turkey (TUBITAK) for providing partial financial support throughout this study with grant TUBİTAK EEEAG-112E11. Moreover I am also thankful to Ministry of Science, Industry and Technology for the support throughout this research which is a part of the SAN-TEZ project number, 1651.STZ.2012-2.

I am deeply indebted to my colleagues and life-long friends Emrah Şamdan, Yunus Emre Işıklar and Sedat Marangoz for their support, encouragement and collaboration.

My heartfelt gratitude goes to my friends and supporters in this journey, Zeynep Yalman, Dinçer Demirci and Ece Bulut.

Last but not the least; my special thanks go to my family. Words cannot express how grateful I am to my parents and sister for their patience, support, prayer and life-long love which sustained me thus far.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVATIONS

API	(Application Programming Interface)
PC	(Personal Computer)
TV	(Television)
BOW	(Bag of Words)
ORM	(Object Relational Mapping)
TF	(Term Frequency)
IDF	(Inverse Document Frequency)
DVB	(Digital Video Broadcast)
EPG	(Electronic Programming Guide)
NLP	(Natural Language Processing)
VOD	(Video On Demand)
XML	(Extensible Markup Language)
JSON	(Javascript Object Notation)

CHAPTER 1

INTRODUCTION

For many years, TV has been the most used conventional media tool which enables the users to access mass information about their interests. On many different areas of subjects, there have been lots of programs that are being broadcasted each hour of each day. In Turkey, while the media landscape continues to evolve with digital alternatives such as video on demand services, social media etc., traditional media still remains to be the widely used entertainment service among people [1]. In fact, the time that people spent in front of TV increases each year [2]. Moreover, with the evolution of internet technology, TV met the internet connection which evolves regular TV into connected TV, smart TV, and IPTV each of which are different concepts. Internet connected TVs open the door of exploring new content over web besides the standard broadcast content provided by TV channels. A large amount of information, products and services are now available for the users to satisfy smarter entertainment while watching TV.

Meanwhile, broadcasting technology is also getting improved day by day which brings new channels to born and start casting. National satellites are placed in order to serve more number of channels in better quality. In fact, most of the national channels are now broadcasting HD contents. As a consequence of the expansion of TV content and digital networks, hundreds of TV programs are broadcast each day. Among this massive amount of content, users are getting lost to find the relevant content that fits to their interests.

Since internet connectivity on TV market is improving, recommendation systems can be the key solution for finding relevant TV programs. Tracking user behavior over the internet connection of TV enables constructing user models including the metadata of TV programs.

A TV program has its own attributes such as genre, actor, director, description, start and end time of broadcast, etc. These attributes can be employed in order to generate TV program listings specific for each user. Different from other domains, TV programs which are broadcasted from conventional TV channels are available only at a certain time. Therefore it is necessary to analyze and process this time-based data in an efficient way to satisfy TV programs every time users would like to have personalized TV program listings. The suffering that the TV users face with shall be overcome by employing personalization and recommendation systems designed for TV domain.

In this thesis a core model and the usage of this core model on personalization and recommendation systems is presented. The presented core model is a graph-based model to represent the users and the TV programs with their attributes. It is evaluated in a case study of a hybrid recommender system.

1.1. Contribution of Thesis

The approach presented in this thesis contributes to the literature of TV recommendation systems in the following ways:

- We have created a graph-based core model which comprises users, TV programs, TV program attributes especially for TV domain. However the model can be adapted to be used in personalization and recommendation systems in different domains.
- A sparse data of TV programs are collected from web, which is preprocessed to be used in our research.

- The core model is employed to create a complex recommender system which comprises collaboration approach, content-based approach, semantic enhancements and context awareness.
- The evaluation is conducted and compared with a baseline system created over the same data and the proposed system produced compelling results.

1.2. Organization of the Thesis

The organization of thesis is as follows:

Chapter 2 presents general information about recommender systems. The common recommendation approaches in the literature and the similarities and differences of these approaches are presented in detail. Moreover, the core models which are presented in the literature are examined in detail and their differences from the approach presented in this thesis are exposed. Finally, the recommendation systems for TV domain are examined in detail.

Chapter 3 presents the graph-based core model which is created to be the base of different approaches for recommendation and personalization systems. The core model is explained in detail with its capabilities including the explanations which describe how to be used in different approaches.

Chapter 4 presents a hybrid recommendation system for smart TV users based on the core model presented in Chapter 3. The recommendation approaches which were hybridized are explained in detail with a set of formulas and figures.

Chapter 5 introduces the evaluation metrics and the evaluation results of the system presented in Chapter 4. The evaluation results are compared and discussed with the baseline method.

Chapter 6 includes the conclusion remarks about the work presented in the thesis and it proposes future work that can be done over this work.

CHAPTER 2

RELATED WORK

Recommender systems are software tools and techniques employed to find relevant items for users who are faced with a huge amount of items to select. Recommender systems are used to obtain relevant items using people's previous decisions in a self-driven way [3]. The easiness on the process of decision making provided by recommender systems helps users in various domains such as what product to buy, what news to read, what music to listen [4]. A recommender system not only helps people not to get lost in a huge amount of items, but also learns from users' interests and habits to decide the values and customization of these items according to their audience [5].

To achieve this goal, recommender systems conduct some identification on user interests, habits and item properties. Since recommender systems are used in many areas to help users find the most relevant items in an item space, the metrics used for relevance measurements change according to the domains which the recommender systems are designed for. Basic recommender systems use non-personalized techniques to suggest items according to item properties such as popularity and demographic information. On the other hand, most of the advanced recommender systems generate profiles for each user to keep their taste and habits to measure the similarity between items and users. [4]

In this chapter, the related work about the common recommender system approaches is presented and discussed. Moreover, the basic modeling techniques to create user profiles and item representations are presented. Since this thesis is about TV program recommendation and user modeling on TV domain, the approaches on TV program recommendations are presented and discussed in detail.

2.1. Common Approaches for Recommender Systems

In the literature, well known approaches for recommender systems are *content-based approaches*, *collaborative filtering approaches*, *knowledge-based approaches*, *context aware approaches* and *hybridization* of those to create stronger recommender engines. In this section, these approaches are presented in detail.

2.1.1. Content-Based Recommendation

Content-based recommender systems suggest items to users by analyzing the item descriptions in order to identify which items are of interest to a particular user. The recommended items are similar in content to the items that the user was previously interested. Thus, item representation and user profiling are main concerns of the content based recommender systems [6].

There are several ways to represent items and users for content-based approaches. Content-based item representations are based on item descriptions which has semantic information about the items. The description of an item is formed to calculate the similarity between items and user profiles. User profiles, similarly, are formed using the descriptions of users' previous item preferences. Vector space model is the common model to represent users and items which is similar with representing documents and queries in information retrieval systems. A document is represented as an m-dimensional vector whose dimensions are the terms that the document includes. Each term is kept with a weight which is the measure of how much this term represents the content of the document. The common approach used for item representation is TF-IDF (term frequency-inverse document frequency). The weight $w_{i,j}$ of a term t_i, in a document j is calculated using TF-IDF measure as follows [7]:

$$w_{i,j} = tf_{i,j} \cdot \log\left(\frac{n}{df_i}\right)$$

where $tf_{i,j}$ is the term-frequency, df_i is the document frequency of the term t_i and n is the number of documents. The term-frequency is the number of occurrences of a term in a document, document frequency of a term is the number of documents which the term occurs at least once.

TF-IDF measurement is based on the assumptions that,

- The more a term occurs in a document, the more relevant this term with the document.
- The more a term occurs in different documents, the less discrimination between documents can be estimated using this term.

The vector representation, described above, can be used to calculate the similarity between items and users if items and users are also described with terms. There are several methods for calculating the similarity between vectors such as Cosine Similarity and Euclidean Similarity [8]. A distance is calculated by using the term weights in user profiles and item representations. Lower distance between user-item pairs indicates higher relevancy or higher degree of interest this user shows on that item [9].

In this thesis content-based approach is employed in a different manner. The item representation and user profile generation are different from the standard content-based approaches. Moreover, TF-IDF measurement is also used for modeling items and users in a different approach. We employed a graph-based data structure to represent users and items. The edge weights between users and items are constructed using TF-IDF measurements.

2.1.2. Collaborative-Filtering Approaches

Collaborative-filtering methods, without any need for content information about items, can recommend items to the users based on the similar users' interests or habits. Resnick et al. presented collaborative filtering on recommender systems first time in 1994 assuming that the users who previously have the same idea, according to their subjective perspective, on past item preferences are likely to agree on the future items again. The user ratings on the same item are calculated first and predictions are made on how well the new items will be liked based on similar users [9].

The collaborative-filtering methods use feedback coming from user-item interactions. These interactions are employed as the inputs of the system in two different ways either *explicit feedback* or *implicit feedback.* Explicit feedback is an input where users directly report the interest on items via likes, rating stars etc. However, in some domains such as TV domain, explicit feedback cannot be gathered, instead, implicit feedbacks are used. Implicit feedback is indirect feedback coming from users in different ways such as duration of watch, browsing history, search history etc.

User feedback on items can alternatively be categorized as follows:

- Scalar feedback: Scalar such as giving 1-5 stars to an item or ordinal ratings such as giving strongly agree, agree, disagree and strongly disagree with an item.
- Binary ratings: Two option feedback such as Like/Dislike or Good/Bad.
- Unary Ratings: User has observed or purchased an item [10].

In this thesis, since explicit feedback is not available in TV domain, implicit feedback is used in order to create the core model. We have used the duration of watching a TV program by a user as the implicit feedback method.

Collaborative-filtering algorithms aim to recommend new items or make a prediction of how likely an item is interesting for a particular user based on the user's previous actions or based on other users' actions whose previous actions are similar to the user. Collaborative-filtering methods can be applied using both item based and user based. Item based methods are based on similarly rated items. Sarwar et al., state that the similarities of the items are measured using Pearson Correlation [11]. Let the set of users who both rated items *i* and *j* are denoted by U then the correlation similarity is given as follows:

$$sim(i,j) = \frac{\sum_{u\in U}\left(\left(R_{u,i} - R_i\right)\bullet\left(R_{u,j} - R_j\right)\right)}{\sqrt{\sum_{u\in U}(R_{u,i} - R_i)^2}\bullet\sqrt{\sum_{u\in U}(R_{u,j} - R_j)^2}}$$

Here $R_{u,i}$ denotes the rating of user u on item i, R_i is the average rating of the i-th item [12].

In order to truly represent the user-item interaction and create an appropriate user preference model baseline predictors (a.k.a. biases) are used. Baseline predictions can be calculated in the following way. Let m be the overall average rating, for estimating the rating which is denoted by $b_{u,i}$, the observed deviation of user u is shown as b_u and the observed deviation of item is shown as b_i in the formula below.

$$b_{u,i} = \mu + b_u + b_i$$

Baseline predictors reduce the effect of user independent interactions caused by systematic tendencies caused by giving higher ratings than the others, therefore, normalization is applied on the ratings to overcome such tendencies [9].

2.1.3. Knowledge-Based Approaches

Burke, in his work [13] stated that the knowledge-based recommender systems are good fit for ever-expanding information resources. A knowledge-based recommender system is a system which include predefined utilities by user or system itself. According to these utilities, system filters the items to be suggested and solves the basic problem for users to face with huge amount of items. Further, these systems also help the users understand the domain. Users are integral part of the system and the user need is directly gathered by the system. Knowledge-based recommender systems are not based on statistical data about rated items or users. In the large scale systems, the items are not weighted according to users' previous feedback. Knowledge-based approach can be a powerful complimentary of the other approaches. Methodologies based on knowledge based approach are *constraint-based* and *utility-based* recommender systems.

A *constraint-based* recommender system is presented as a process of constraint satisfaction for users. Some of these constraints come from users; some of them come from the product domain. Those constrains are aimed to be defined to filter the options presented to the user. Particularly, such recommender systems are distinguished in functional knowledge about an item meets a particular user need. [14]

A *utility-based* recommender system does not keep any long term generalization about users, rather than that these systems compute the utility for each item for the user. Therefore, a user profile is a set of utilities computed for each item in the system. The advantage of utility-based system is that it makes possible the trade-off analysis for items [13].

2.1.4. Context-Aware Approaches

Contextual information is recognized by researches and practitioners in many disciplines as improving the quality of the recommender systems. Although most of the recommender systems focus on the relevance of the items, some attributes such as time and place should be taken into consideration in order to produce successful recommendations [4].

Contextual information helps information retrieval which is proven by Jones et al. in their work [15]. Jones showed that the returned results are more successful when results are filtered according to context-relevancy.

Tuzhilin et al. in their work [4] indicated that although majority of the work on recommender systems relies on user and item pairs which gives the rating of that item according to that user, when the context-awareness is added as a third dimension to indicate the rating of an item by a user in a specified context, the success of the recommendation systems is increased remarkably. They also introduced the filtering methods using contextual information. The filtering operations can be applied as either pre-filtering or post-filtering in order to reduce the result set to match the current contextual information. Their work gives comprehensive case studies over context-aware recommender systems and constitutes a newly developing and promising research area for recommender system problems.

In this thesis, besides content, the contextual information is also taken into account using pre-filtering operations. Two dimensions are selected as content which is described in detail in Chapter 3.

2.1.5. Hybrid Approaches

Hybrid recommender systems are systems that combine the approaches presented above. The hybridization of those systems aims to use advantages of these systems and excluding the disadvantages of these systems.

Collaborative- filtering approach suffers from incoming items with no ratings. If an item has not rated by a user yet, that item does not have any user set to be recommended. However in the content-based systems when a new item appears in the system, using the content of that item, it can be recommended to users who are interested in similar items. On the other hand, content-based methods suffer from recommending a standard set of items. Users cannot have recommendations of the items which might likely be in the interest set of the user. Hybridization of these two methods can overcome these two problems at the same time. Using content-based user profiles, whenever a new item included in the item set, it can be recommended to the users according to the item representation. In the meantime, using collaborative methods, users have richer recommendations with the items that they have not shown interest to, but have a high probability to like [16]. Burke et al. presented the evaluation results of different approaches on collaborative-filtering which have five conditions: a random, the average rating of all users, inter-user correlation, the sparse metric using cosine similarity and the heuristic metric. The result of their work is shown in the Figure 2-1.

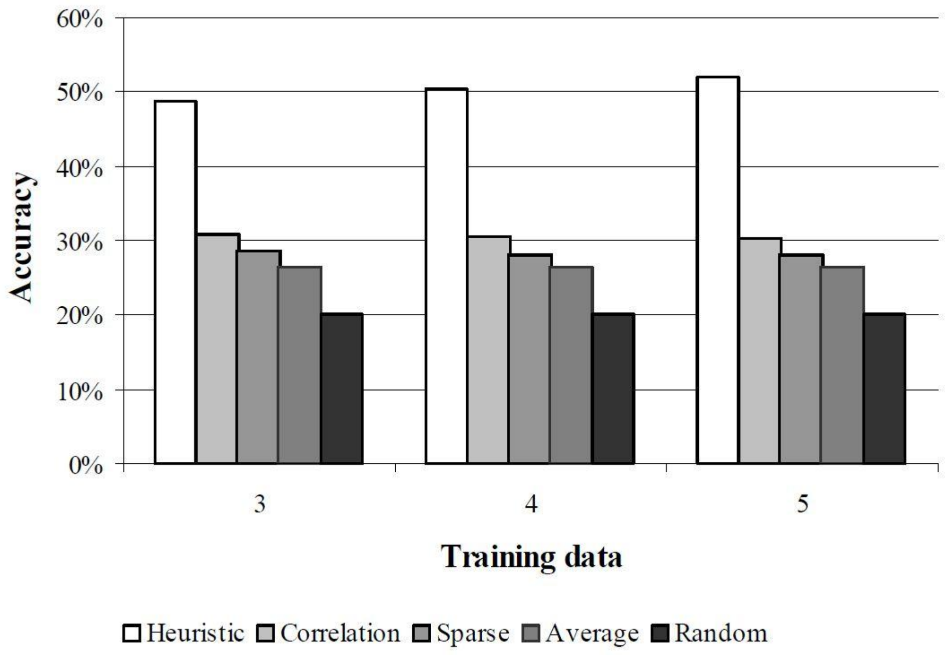


Figure 2-1: Random Ranking Based on Different Metrics [16]

Knowledge-based approaches can be added as a complementary approach to other recommender system approaches in order to help to track users' current needs and transforms other approaches sensitive to the changes on user needs. The hybridization of the knowledge-based approaches and other approaches can also be useful in personalized search of the items. In a huge amount of item sets, the result set can include excessive amount of items. In such cases, when content-based and collaboration methods are used, by hybridization of knowledge-based approaches, items can be presented in a ranked way to the user which helps the user find the most relevant items above in the listings [17].

2.1.6. Comparing Recommender System Approaches

In the survey paper by Sarwer et al., different approaches on recommendation systems are compared [11]. *Content-based* approaches have some advantages over collaboration filtering approaches. In content-based approaches users are independent from each other. User profiles are generated using solely either implicit or explicit ratings of the user. Unlike collaborative-filtering approaches, there is no need for the ratings other users, to recommend items to the users. Transparency is another advantage of content-based approaches over collaboration which enables users to decide whether to trust a recommendation system. The reason of each recommended item can be defined according to the item content; however, in *collaborative-filtering* approaches there is no specific reasoning other than similar user preferences. New item problem is a common problem for collaborative filtering approaches. In content-based recommenders there is no new item problem raised by unrated items. On the other hand, in order to create a content-based recommender system, domain knowledge is needed since item representation and user profile generation are highly effective on the success of the system. Another drawback of using content-based approach is getting stuck in user interests. If a user has not rated an item that may be interesting for him/her, content-based approaches cannot detect such items in item sets. However in collaborative filtering method, users can discover interesting new items independent from their current interests [11].

In the *knowledge-based* approaches, the domain specific constraints are taken into account to generate item recommendations according to users' current needs. In these systems, unlike others, the user specifies what s/he wants at this time and items are recommended over the user needs which ease the filtering operations over large amount of items.

The comparison of the existing approaches is summarized in **Error! Not a valid bookmark self-reference.**.

Table 2-1: Comparison of Different Approaches [16]

Technique	**Background**	**Process**	**Pros**	**Cons**
Content-Based Filtering Approach	Features of items	Generates a classifier that fits user rating behavior and use it on items	No new item problem	Stuck in user preferences, New User Problem, Insensitive to preference changes
Collaborative-Filtering Approach	Ratings of items by users	Identify users in the system which are similar	Items according to similar users can be suggested, not stuck in users preferences	New item problem, New user problem, Insensitive to preference changes
Knowledge-Based Approach	Features of items Knowledge of how these items meet a user’s need	Inferring a match between user need and item features	No statistical user profiling, Sensitive to preference changes	Knowledge engineering, Suggestion ability is static, Not effective on large item sets

In this thesis, although content-based recommendation is employed mainly, the advantages of collaborative-filtering approach and similarity measurements are also integrated into the core model. Thus a hybrid method is implemented.

2.2. Core Models of Recommendation Systems

Recommender systems include learning methods for complex human behavior over time. In order to perform this task, user actions are observed to make inferences about the relationships between users and between items [18].

In his work [18], Zuckerman, presents several user modeling techniques to be employed in content-based and collaborative-filtering recommender systems. He divided the user modeling strategies into several different statistical models. These models are: Linear models, TFIDF-based models, Markov models and neural networks.

Linear Models: in linear model, learning is easy and highly extendable and generalizable. These models use the weighted sums of known values to compute a value for unknown quantity. The sum of weights of user-item pairs can be used in similarity calculation in both content-based and collaborative systems. For example in content-based systems, the attributes of items and the user ratings on these item properties can be employed to calculate the similarity between users and items. In collaborative systems, these weights can also be used to calculate the similarity between users.

TFIDF – based models: Term frequency inverse document frequency based object modeling is the most common approach in the field of information retrieval to extract the most relevant document which matches the user query [19]. Objects (a.k.a documents in information retrieval domain) are modeled by a vector of weights where each of these weights are gathered from the term in a document. Using this model, the similarity between documents is calculating by distance calculation such as cosine distance or Euclidean distance. Moukas et al. applied TF-IDF based models in content-based systems in order to recommend similar documents to the users. Later they extended this approach with genetic algorithms to adapt the changes in user preferences [18].

Markov Models: The use of markov chain models on collaborative recommender systems is presented to predict the users' requests on World Wide Web [18]. In his work, Bastevros calculated the probability of a user to search for a particular document in the future while Zukerman et al. made a comparison on prediction performance of different markov models. In these works, the events happening sequentially are observed and the next event is predicted using probability distribution over the following sequences of events.

Neural Networks: Jennings and Higuchi (1993) presented a neural network approach to employ in content-based systems to represent users' preferences on news articles. There are outgoing edges from a user node to the terms extracted from the documents that the user liked previously in the created neural network. The relationship between the user and the terms shows the strength of the association between the user and terms [20].

These models are used in different approaches in recommender systems. In this thesis, although the core model resembles a neural network model, the TFIDF-based models are also embedded into the model.

2.2.1. Graph-Based User-Item Modeling Approaches

Content-based and collaborative-filtering methods are applied together to build better recommendation systems. Modeling items and users in a graph structure is a natural way to apply these two approaches in one framework [21]. Instead of representing users and items as vectors of attributes, in graph-based data stores, users, items and attributes can be modeled as nodes and the relationship between the nodes are weighted edges which show the relatedness of the connected nodes.

Huang et al. presented a method for keyword search and recommendation for digital libraries using two-layered graph architecture. [21] The first layer of the graph includes nodes of customer type and the other layer includes nodes of book type. The relationships in the graph layers show the similarity between customers and books accordingly. The layered structure of the graph including book and customer nodes on different layers is shown in Figure 2-2.

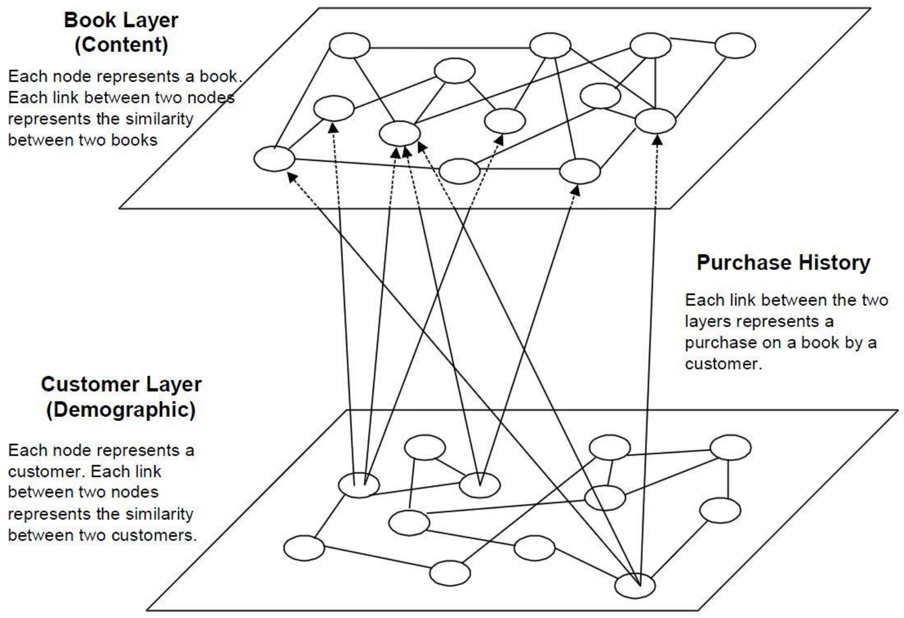

Book Layer (Content)
Each node represents a book. Each link between two nodes represents the similarity between two books
Purchase History
Each link between the two layers represents a purchase on a book by a customer.
Customer Layer (Demographic)
Each node represents a customer. Each link between two nodes represents the similarity between two customers.

Figure 2-2: Graph Structure in Two Layers [21]

Another research conducted by Baluja et al., improves the video recommendation of YouTube, one of the most popular video provider networks [22]. In their work, they stated that searching over an enormous number of videos is an overwhelming process for users. They proposed a personal search system based on a graph-based data model to encounter this problem. They modeled the data as two graphs one of which is user-video view graph and the other one is video co-view graph. In user-video view graph, edges between user and video nodes are weighted using the users’ view history. In the video-video co-view graph, nodes are of type video and the edges between nodes are weighted according to the number of times the two videos are viewed by the same user. They label the unlabeled nodes by propagating information in the graph using random-walk algorithm. Cicekli et al., proposed an algorithm to a hybrid video recommendation by enhancing the attributes semantically in their algorithm [23].

Another video recommender system presented by Bogers et al. uses contextual information to build the graph-based data model. In the contextual graph, the node types are users, movies, tags, actors and genre. A utilization of random walk algorithm, namely ContextWalk is applied to calculate the similarity between node types. The main advantage of this approach is that the similarity between different and same node types can be examined using their graph-based data model [24].

In this thesis, a graph-based data model is created including the node types of users, programs and program attributes. Different from Bogers approach, the proposed method in this thesis includes the co-occurrence relations and weighted user rating edges between nodes. The traversal algorithm commonly used in the approaches mentioned above is random walk, however, in this thesis, the traversal and item similarity calculations are gathered using a utilization of activation spreading method [25].

2.3. Recommendation Systems for TV Programs

The availability of TV channels is increased by both conventional and IPTV channel providers. TV users confronted with a large amount of TV programs available at the same time [26]. Based on the assumption that a recommender system for TV programs is beneficial for TV users, a considerable amount of work has been presented on TV program recommendation recently.

TV program recommendation has its own challenges. Standard data modeling techniques do not fit into TV program recommendation applications. Turrin et al. in their work [27] presented a set of applications using traditional methods over TV domain. In their work collaborative-filtering and content-based methods are employed in order to generate recommendations for TV users. However, these algorithms are not fitted exactly for different type of services. Since a TV program is available only for a certain period of time in conventional TV channels, collaborative-filtering approaches are not fitted well to generate the probabilities of user program matches. In collaborative-filtering approaches, user similarities are generated based on their preferences and users are recommended items that similar users preferred. However, collaborative-filtering approaches suffer from new items which are continuously added into dataset without being rated. Therefore these approaches are not suitable for TV program recommendation. Therefore, since in conventional channels, contents are available for a certain period of time, the availability of the previously rated items is very low. Moreover, collaborative-filtering approach cannot suggest a future TV program according to similar users since that particular future program has not been watched by any users yet.

Turrin et al. employed content-based approaches to create a recommender system for TV programs. TV programs have their particular attributes such as, start time, end time, genre, title, description, actors and director. They stated that the quality of metadata is directly effective over a successful content-based TV program recommendation. Each metadata gathered from EPG is either a string or a list of strings. They weighted each metadata differently based on the heuristics of importance and the attributes containing sentences are tokenized, filtered, stemmed and the words in these sentences are extracted. Furthermore, the weight of the metadata attributes is calculated using TF-IDF values. As core data model, they used TF-IDF based model which keeps the dimensions as vectors with weights. Bag of words (BOW) data model is also used in order to keep the tokenized sentences in the vector. Similarly, user profiles are kept using same core model which is filled according to user ratings on programs. In order to measure the closeness of programs and users, they transform users and programs into a latent space. In the latent space, the distance is inversely proportional with closeness of a program with the user.

The research conducted by Kim et al., shows that collaborative-filtering approaches may also be applied to TV program recommendation. To create the core model, they used a TF-IDF scoring called CF-IUF (category frequency-inverse user frequency) [28]. They combined genre, provider into a single attribute namely category which is the input of the system to calculate the frequency for each user as follows:

$$s_{uq} = \begin{cases} \left(1 + \log f_{uq}\right) \times \log \dfrac{N}{h_q}, & if\ f_{uq} \neq 0 \\ 0 \quad , & if\ f_{uq} = 0 \end{cases}$$

$$F_u = \langle s_{uq_1}, s_{uq_2}, s_{uq_3}, \cdots, s_{uq_m} \rangle$$

Here, s_{uq} represents CF-IUF of user u for category q as given in the equation. f_{uq} denotes the number of views of category q by user u, h_q represents the number of users who viewed the category q. N represents the total number of users. As core model they applied TF-IDF based vector notation. In order to cluster the users, k-means clustering method is not the most fitted option since the number of clusters is not well defined. Instead, they employed ISOData algorithm which determines the optimum number of clusters. After creating clusters, collaborative-filtering methods are applied on these clusters instead of applying them on users, which improves the effectiveness of their system.

Kurapati et al. employed both explicit and implicit feedback from users in their TV program recommender system [29]. Implicit feedback is the watching preference collected from user devices. They combine watch and not watched preferences of users to create user profiles. Bayesian classifier is applied over user profiles by means of collection of features (attribute-value pairs) together with a count of how many times a user has watched and not-watched the content. The weakness of their approach rises due to new items that have not been broadcast before. In order to compute the program recommendation scores, they applied decision tree approach. The explicit feedback coming from users also brings in the user-specified ratings in a rage of one to five. They generate recommendations using combined score coming from implicit and explicit feedback of users. The experiments showed that when the distance between implicit and explicit feedback scores are apart, explicit feedback is considered to be more reliable to use. They created a system including user feedback and implicit inferring according to user behavior which is shown in the Figure 2-3.

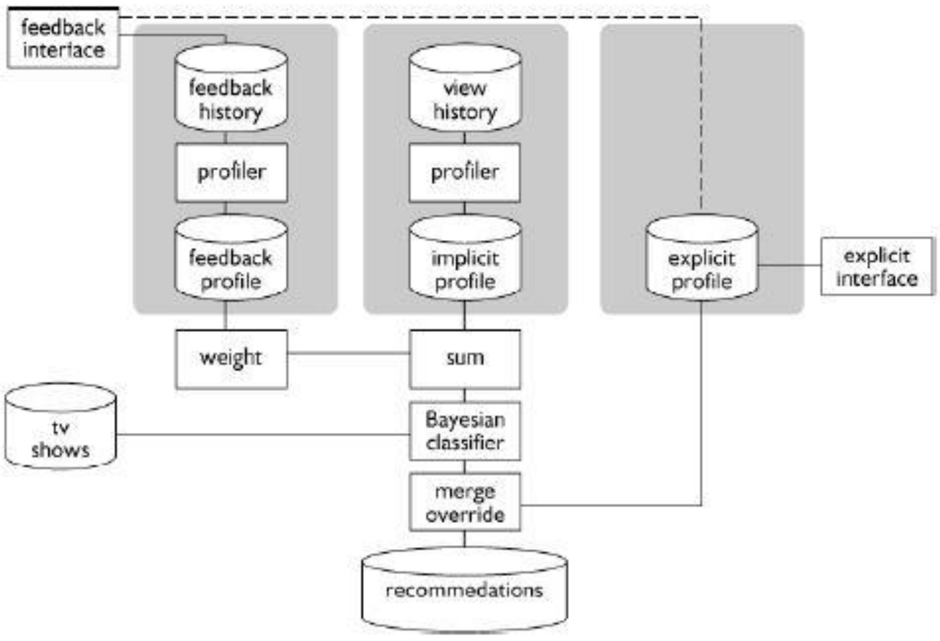


Figure 2-3: System for both Implicit and Explicit Feedbacks [29]

Woo et al., presented a socially aware recommender system for TV program recommendation [30]. They claim that TV programs are consumed simultaneously by a group of users so TV program recommendation systems should take into account the user groups in front of TVs. In order for a TV program recommendation to meet both group and individual user preferences, they propose a system which linearly combines user profiles with common interests. They have three strategies to recommend TV program to the users: automatic selection with recommendation, TV program recommendation and category recommendation. The first case is that the group of people has an agreement on one program. In this case the algorithm chooses the best matching item in the item list using the highest group preference. In other words the system treats the group of users as a single user. In the second case the group of similar users whose interest on TV programs are similar but not the same, programs are recommended individually to each user with scores coming from all the users in the group. The third strategy is employed when all the users in the group have different interests. In this case system recommends only category instead of program. The application of this algorithm includes a user interface where users can input their interest individually. When a single user is in front of the TV, the system generates recommendations in standard way.

Another conducted project on TV content recommendation is NoTube project which is funded by European Union 7th Framework Program [31]. Their aim is “to close the gap between Web and TV by semantics”. They merge the user driven data on the web with TV related data to recommend TV programs. They created a system to take the social network behavior of the users into account to generate more precise recommendations without having any cold-start problem. Cold-start problem occurs when there is a new user who has not rated any item yet.. The wide resources of social web via APIs make it available to create such a system. Another perspective is the reverse data flow from TV to social web. Users may want to enrich their TV content with more information gathered from DBPedia[1] or FreeBase[2] or may want to share the watching habits over a social network such as Facebook[3]. Enhancing user profiles with social web accounts enables the system to recommend TV programs in a more precise way. Cross platform data can be enhanced over time with further research.

[1] http://dbpedia.org/
[2] https://www.freebase.com/
[3] https://www.facebook.com/

CHAPTER 3

THE GRAPH-BASED CORE MODEL

In this thesis, we present a graph-based core model to represent users and the details of TV programs broadcast by different channels. The core model is a graph-based model which comprises different node types and weighted/unweighted edges, representing users, terms, named-entities, genres and the relatedness between them. The constructed core model can be the base data model for different types of applications. In this chapter, the graph-based core model and its capabilities are explained in detail.

3.1. Graph-Based Item Representations

The descriptions of TV programs and interests of users to these TV programs are modeled as a graph which contains different types of nodes. Nodes are connected by either weighted or unweighted edges depending on the relationship between them. A weighted edge between two nodes represents the degree of relatedness between these two nodes. Each edge weight is calculated using different metrics based on the type of the nodes that they are connecting.

A user profile is described as *a sub-graph which includes a user node and all the other nodes and edges obtained by a traversal starting from this user node*. Similarly, a program profile is *a sub-graph which includes a program node and all other nodes and edges obtained by a traversal starting from this program node*. Thus, a user profile and a program profile may contain all types of nodes.

Different graph traversal algorithms can be applied on this core model in order to generate profiles. Different traversal strategies may produce profiles which include different depths of information according to the decision of when to stop the traversing. In this manner, it is up to the application layer to decide what depth of information the profiles should embrace.

The core model is illustrated in the Figure 3-1.

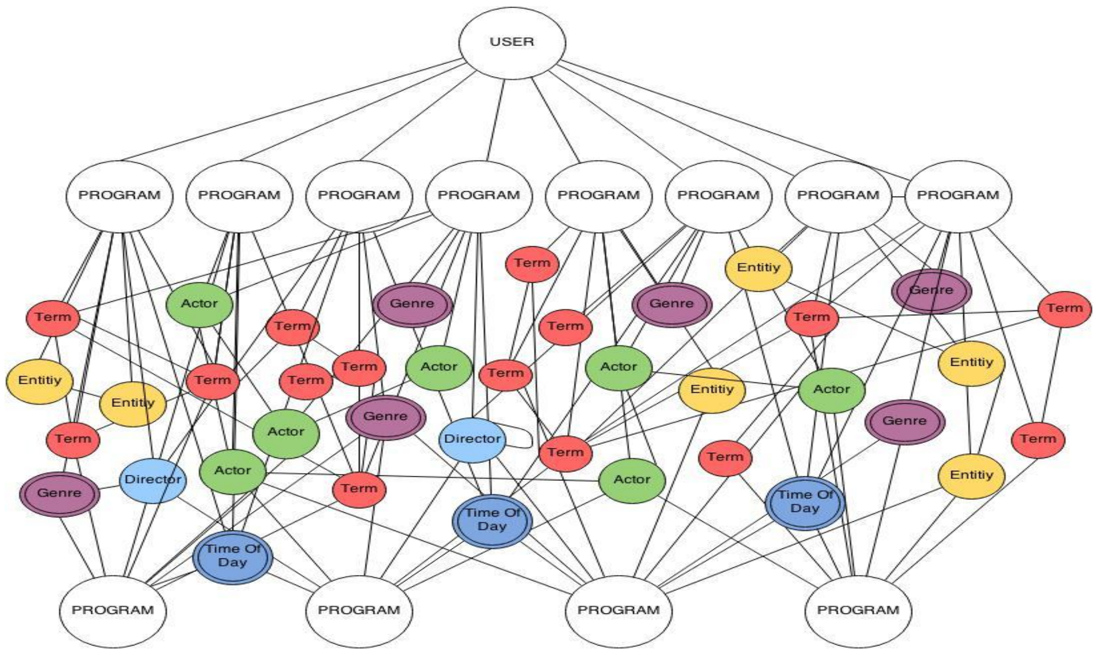


Figure 3-1: The Core Model Representation

In this section, *node types*, *relation types*, *properties* and *edge weight metrics* are presented in detail.

3.2. Node and Relation Types and Properties

In the core model, there are eight types of nodes which can be examined in four different categories as *identity nodes*, *descriptor nodes*, *attribute nodes* and *context* *nodes*. Identity nodes can be *user* or *program* node types to represent users and TV programs in the data set.

The descriptor nodes represent the semantic content of the TV programs to which they are connected. Descriptor node types in this core model are *terms* and *named-entities*. These node types can also be connected to the same node types with co-occurrence relation representing how many times these items occurred in a TV program together.

Actors and *director* of a TV program are represented as attribute nodes in the core model. In order to be employed in sophisticated applications, the core model contains co-occurrence relations between actor nodes representing how many times these actors play in the same TV program together.

The context nodes of type *genre* and *time of day* represent the contextual information about TV programs. The node types grouped by categories are listed below:

- Entity Nodes
    - User nodes
    - Program nodes
- Descriptor Nodes
    - Term nodes
    - Named-Entity nodes
- Attribute Nodes
    - Actor nodes
    - Director nodes
- Context Nodes
    - Genre nodes
    - Time of day nodes

The nodes and the relationships between these nodes are presented in detail below.

3.2.1. Entity Nodes

Entity nodes are either user nodes or program nodes. The attributes of entity nodes and content of TV programs (terms, named entities etc.) constitute the other types of nodes. An entity node together with other types of nodes reachable from that entity through a specific traversal forms the profile of the entity node.

User nodes represent the real users and they are connected to the program nodes with a weighted edge, namely *user_program,* and they are connected to other node types through program nodes.

Program nodes correspond to TV programs on the graph-based core model. Program nodes with other types of nodes constitute to a profile of a TV program. Program nodes represent particular TV programs connected with descriptor nodes, attribute nodes and context nodes according to the content of the TV program that the node represents.

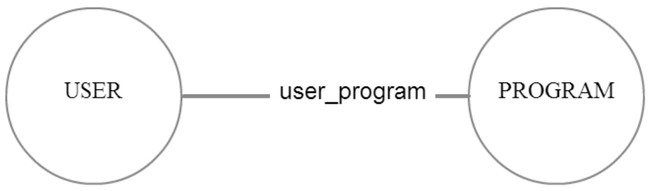


Figure 3-2: Entity Nodes

3.2.2. Descriptor Nodes

Descriptor nodes and their connection between program nodes represent the semantic relations between concepts to form into the knowledge representation of TV programs. These nodes may represent items which have semantic meanings such as terms, named entities, word annotations etc. In the scope of this thesis, terms and entities are selected to create descriptor nodes. The edges between these node types and the program nodes are weighted edges showing the TV program representation capability of these node types.

Term nodes represent the words in the TV program profiles connected with program nodes and other term nodes that occurred together. The type of an edge between a program node and a term node is named as *program_term* and the edge between two term occurred in the same TV program at least once is called *term_cooccurance.*

Named-entity nodes represent the annotated entities extracted from TV program descriptions. Entities are connected to TV program nodes and other entities occurred at least once in a program together. The type of an edge between a program node and an entitiy node is called as *program_entity* and the edges between two entities occurred in the same TV program is called *entity_cooocurance*.

Edge weight metrics of these relations are presented in Section 3.3.

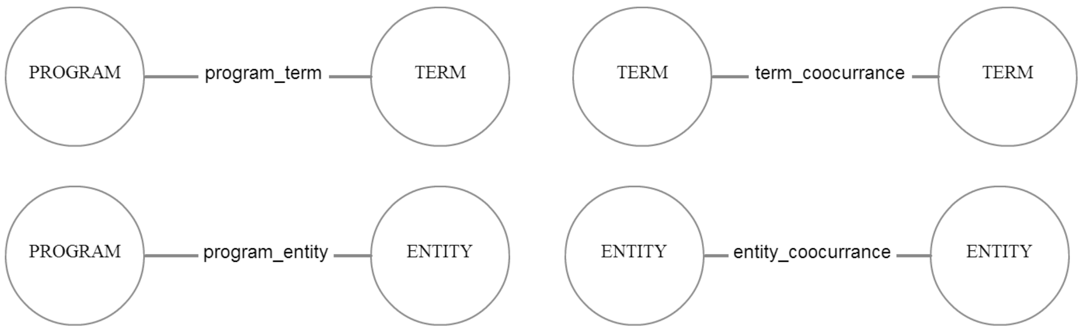


Figure 3-3: Descriptor Nodes

3.2.3. Attribute Nodes

Attribute nodes are projections of the TV program properties over the core model. These node types represent the properties of TV programs in the core model. These node types can be actors, directors, channels, producers etc. In the scope of this thesis, only actors and directors are selected as attribute nodes. The relation of these node types with program nodes are not weighted since these nodes are employed to define the properties of TV programs.

Actor nodes represent the actors played in TV programs which are connected to TV program nodes and other actors occurred at least once in a TV program. The type of an edge between a program node and an actor node is called as *program_actor* which is an unweighted edge, since the ability of an actor to represent a TV program cannot be measured unlike descriptor nodes. Similarly, co-occurrence measurements are employed for the edges between the actor nodes which are called *actor_co-occurence* connecting two actor nodes played at least once in the same TV program. Director nodes represent the directors of TV programs which are connected to TV program nodes with unweighted edges namely *program_director*.

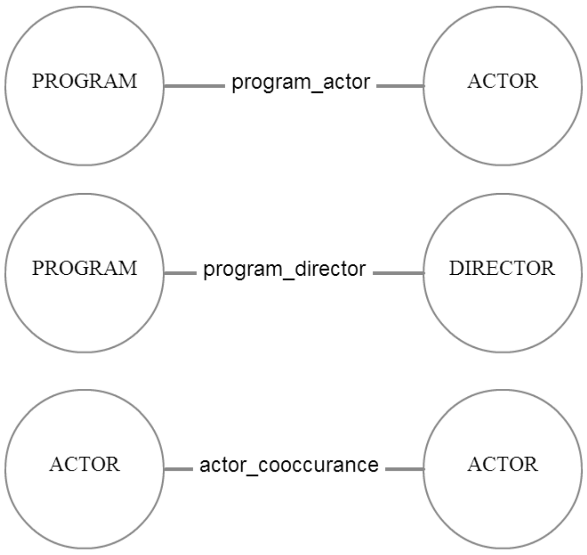


Figure 3-4: Attribute Nodes

3.2.4. Context Nodes

Context nodes are projections of the contextual information which can be about TV programs or users. The node types in context nodes category can be genre of TV programs, time of day that TV programs are broadcasted, the location of the users, the age of the users etc. Context nodes can serve for pre or post filtering the items to fit in a defined context. According to the TV domain, the context can be any attribute of TV programs or users. For example, if one needs to create a TV program recommendation over this core model, time of day is an important feature to select as a context. In the scope of this thesis, genre and time of day is selected as context nodes to be employed in the graph-based core model.

Genre nodes represent the genre as a context of TV programs. Genre nodes are connected to TV program nodes with unweighted edges of type *program_genre*. Time of Day nodes represent the day time partition of the TV programs which are called as *program_timeofday*. Time of day nodes are connected to TV program nodes with unweighted edges like genre nodes.

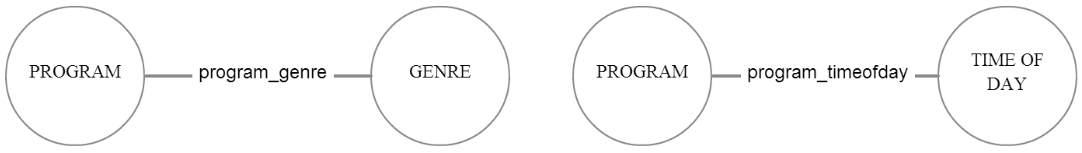


Figure 3-5: Context Nodes

3.3. Edge Weight Metrics

The edges between the nodes in the core model can be categorized into 4 categories which are *rating weighted edges*, *tf-idf weighted edges*, *co-occurrence weighted edges* and *unweighted edges* according to how they can represent the correspondence of the nodes that they connect. In this section, the edge weight metrics are presented in detail.

3.3.1. Rating-Weighted Edges

The relation between a user node and a TV program node represents how much that program is of interest to that user. In order to measure the interest of a user to a TV program, a normalized value of how much time that user has spent in watching that

TV program is used as a metric. This metric is called *rating* which is calculated using the following formula:

$$rating = \frac{duration\ of\ watch}{overall\ duration\ of\ TV\ program}$$

The rating shows the interest of a user to a TV program by calculating the proportion of the user watch time over the overall duration of the program.

3.3.2. TF – IDF weighted Edges

The relation between a program node and descriptor nodes is weighted intended to reflect semantically how important that term or named-entity is for that program. The metric used for that purpose is TFIDF weighting [32] which is commonly employed in Information Retrieval [33] systems. The TFIDF is product of two statistical measures namely *tf* (term frequency) and *df* (document frequency). The weight $w_{i,j}$ to represent the correspondence between a TV program and descriptor nodes is calculated using the formula below.

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

where $tf_{i,j}$ is the frequency of *i* in *j*, df_i is the number of programs containing *i* and *N* is the total number of programs.

3.3.3. Co-occurrence Weighted Edges

The relation between same-typed nodes is represented with co-occurrence weighted edges. The actors played in the same TV program, the entities occurred in the same TV program and the terms occurred in the same TV program are connected with the co-occurrence weighted edges. The normalization is applied on these edge weights using the formula below. Let $w_{n_1,n_2}$is the normalized co-occurrence weight of the nodes $n_1$and n_2, $c_{n1,n2}$ is the number of TV programs that $n_1$and n_2 occurred together, $C_{max}\,(n_1)$ is the maximum co-occurrence among the nodes and $C_{min}\,(n_1)$ is the minimum.

$$w_{n_1,n_2} = \frac{c_{n1,n2} - C_{min}\,(n_1)}{C_{max}\,(n_1) - C_{min}\,(n_1)}$$

One may easily realize that w_{n_1,n_2} and $w_{n_2,n_1}$are not equal to each other since the normalization is applied using the outgoing edges of a starting node. Co-occurrence values are changed according to the start node and the relations of this node with other nodes. Therefore, in the core model, the co-occurrence relations are represented as directed edges which represent the normalized co-occurrence weight of start node with the end node.

3.3.4. Un-weighted Edges

There are un-weighted edges connecting the program nodes with context and attribute nodes since these relations represent only the existence of an attribute or a context in a TV program. For example, a TV program can be directed by a director or not, an actor can play in a TV program or not, a TV program can be in a genre or not. These real world relations are projected to the core model as un-weighted edges showing the presence of a correspondence between two nodes.

3.4. Core Model Capabilities

The core model presented in this thesis can be the base of different approaches to be applied on TV domain. The core model is formed as a graph-based model in order to represent the correspondence between the concepts in TV domain. These concepts are represented as particular node types connected with weighted or un-weighted edges. The attributes used in the model are the most common attributes used for TV domain in the literature [34] [35] [36].

The presented core model comprises descriptions and attributes of the content as well as the contextual information which turns this model into a perfect candidate to be the base of different recommendation and personalization methodologies applicable on TV domain.

This core model is a graph-based model which takes the advantages of representing the real world entities as linked data leveraging the efficiency and scalability of querying. Time-varying data collected from high-resolution user activities on TV can be projected with the presence and interactions of individual nodes of concepts. Furthermore, besides modeling the concepts as a graph, the relation between concepts are weighted using information retrieval methodologies which makes the content based approaches applicable on this model.

Described node types can be extended according to the dimensions of the dataset. Each new node type can be mapped into the categories which are entity, descriptor, attribute or contextual. The relations between these node types can be extended and improved with different metrics. The node types presented in this thesis relies on the domain knowledge gathered from previous works and the dataset we used. This core model is scalable with new node types and relations to be applied in other domain specific applications.

In this section, we present a variety of applications which can be supported by our graph-based core model in connected TV domain.

3.4.1. Content-Based TV Program Recommendations

Content-based recommender systems analyze item descriptions to filter the items that are of particular interest to the users. Item description analysis can generate the words or named entities significant to that item. The main aim of content-based recommendation systems is to recommend items that have similar item representations with the items that users show their interests in the past.

Standard content-based approaches in the literature recommend items according to their content by using contents of the other items that users have rated before. On the other hand, more sophisticated systems employ machine learning techniques such as Bayesian classifiers, cluster analysis, decision trees, and artificial neural networks to measure the probability that whether the user is going to be interested in the items or not. The presented model in this thesis can be classified as an artificial neural network [37] by means of its structure including semantic relations between items.

The core model presented in this thesis provides the content information that the users have shown their interests before. The content information is represented as *terms* and *entities* as the content and *actor, director* and *genre* as other attributes of TV programs. The edges between descriptor nodes and programs are weighted using TF-IDF measurements in order to represent the content information in a more accurate way.

According to the contents of TV programs that have been previously watched by users, new programs can be recommended by employing different classifier and clustering algorithms. These algorithms can be implemented on a graph-based database instead of bag of words representations.

As shown in the Figure 3-6, a content-based recommendation can be implemented on the graph using some traversal algorithms. Unwatched programs can be suggested to the user by a traversal starting from a user node, moving to the previously watched programs over the weighted rating edges and then from these programs moving through the attribute and descriptor nodes layer.

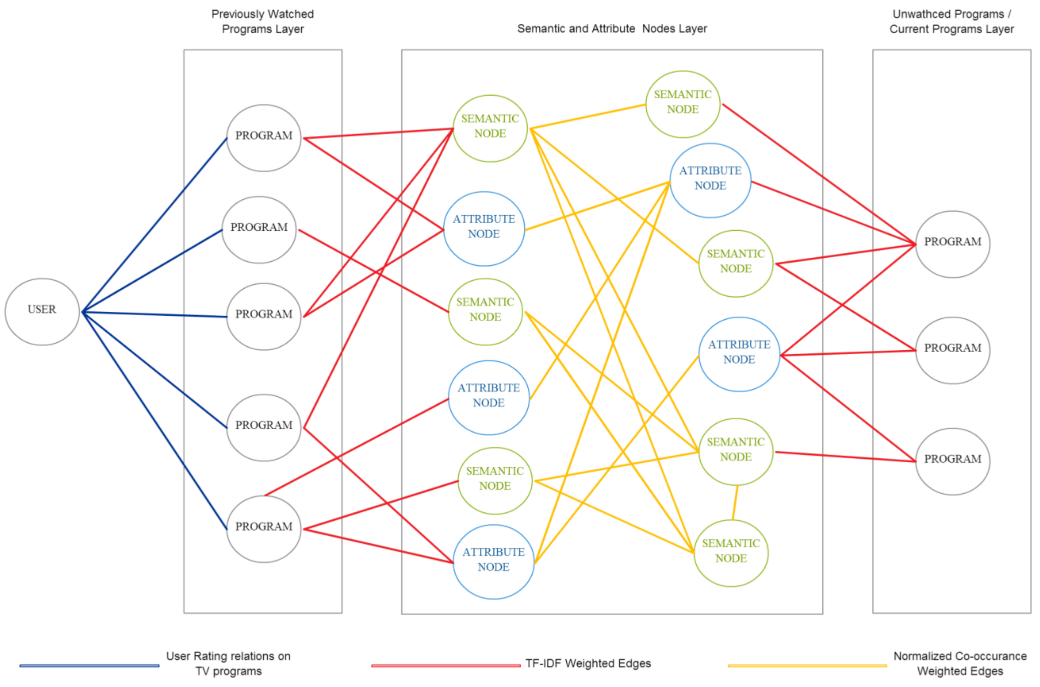

Previously Watched Programs Layer
Semantic and Attribute Nodes Layer
Unwathced Programs / Current Programs Layer
USER
PROGRAM
SEMANTIC NODE
ATTRIBUTE NODE
User Rating relations on TV programs
TF-IDF Weighted Edges
Normalized Co-occurance Weighted Edges

Figure 3-6 : Core Model Visualization for Content-Based System

3.4.2. Collaborative-Filtering TV Program Recommendations

Collaborative-filtering employs techniques involving collaboration among multiple agents, viewpoints, data sources, etc. by filtering the information or patterns. In a narrow sense, collaborative recommendation is a set of approaches that filter items to be recommended according to interests of similar users. Although the recommended item set is specific for the users themselves, the information of interests are gathered from many different users.

Bayesian networks, clustering models, latent semantic models such as singular value decomposition and probabilistic latent semantic analysis are common approaches for developing a collaborative-filtering recommender system. Most of the models are based on a classification or clustering technique to identify similar users.

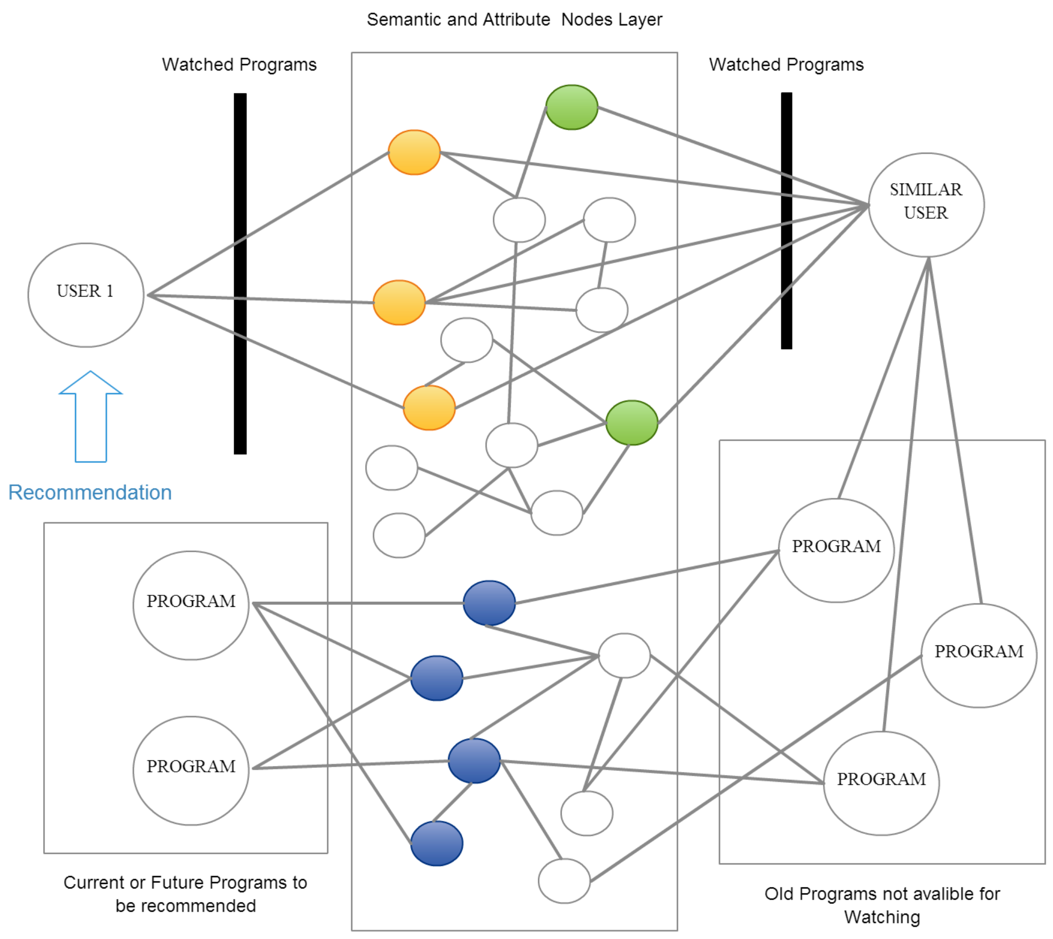
The core model presented in this thesis can be a base model for a collaborative-filtering recommendation approach since a user profile defined in this system includes other user nodes which can be accessible with a graph traversal algorithm starting from that user node. The basic issue in a collaborative recommendation system is calculating the similarity between users based on their previous interests.

When a user profile is gathered, similar users can also be extracted which makes it possible to generate recommendations based on similar users.

Storing users, programs and other items in a graph database provides convenience for implementing collaborative-filtering algorithms. Starting from a user node, the closest accessible user nodes are the most similar users to the starting user. After filtering the similar users, the programs that these users have watched but the starting user has not watched yet, can be recommended to the starting user. In a different sense, after filtering similar users, the programs including the actors, directors, genres, terms and entities which are interesting for similar users, can also be recommended for the starting user Figure 3-7 illustrates how a collaboration filtering can be applied over the proposed model. Starting from USER 1 similar users can be accessed by a traversal over program and content nodes such as actors, directors, terms and entities.

In TV domain, since a TV program is available for watching only in a particular time period, standard collaborative filtering (i.e. recommending programs that similar users have watched but the user has not) is not applicable.

Instead of using the program nodes, the contents of these programs can be used for collaborative-filtering. The system in Figure 3-7 represents a collaborative recommendation. Traversal over the descriptor and attribute nodes from USER 1 to similar users is shown in the figure. Same content consumed by other users and accessible content via co-occurrence relations are used to find similar users. After finding similar users, the accessible unwatched or current TV programs being watched by similar users can be recommended to the user.



Semantic and Attribute Nodes Layer
Watched Programs
Watched Programs
SIMILAR USER
USER 1
Recommendation
PROGRAM
PROGRAM
PROGRAM
PROGRAM
PROGRAM
Current or Future Programs to be recommended
Old Programs not avalible for Watching

Figure 3-7: Core Model Visualization for Collaborative System

In the core model we presented, collaborative filtering for a user U can be applied in the following manner: "*Users who are similar to user U are those who like the same content. Then, the user U is recommended those programs which are liked by the similar users*". Same reasoning can be applied to actors, directors or genres as well: *"Users who are similar to user U are those who like the same actor(s). Then, the user U is recommended those programs which are liked by the similar users* ".

3.4.3. Knowledge-Based TV Program Recommendations

Knowledge-based recommender systems are built on the explicit knowledge given by the users. Most of the knowledge-based recommender systems are employed for assisting other approaches (content-based, collaborative) or as an alternative for the applications where these approaches cannot be applied.

Knowledge-based recommendation systems are good candidates for the systems where cold-start problems occur frequently since the users explicitly define what they want to get. Moreover, these systems are also used for searching items among huge set of items.

The core model we presented in this thesis also supports knowledge-based search systems based on users' inputs. For example, users may want to search for the programs including the specified actors, directors, genres, terms and entities. The presentation of the items in the result set of this search can be restricted and ranked according to the relevancy of the items in the result set to the users' previous interests.

Moreover, the items retrieved from content-based and collaborative-filtering methods can be pre or post filtered according to the input given by the user. This filtering mechanism can be achieved by applying knowledge-based methodologies on the core model presented. For example, a user may want to watch a TV program including an actor specified by this user. In this case, the TV programs may be filtered using that actor and the retrieved results can be presented to the user in a ranked way using the previous watching habits of that user.

Another capability of the core model is that when a user searches for an item, the other related items may also be retrieved. For example, when a user searches for an actor, the other programs in which the actor played can be gathered, the other actors playing in the same programs frequently can be gathered, the directors working with that actor frequently can be gathered and so on.

The efficiency of retrieving items is a desired property for a core model. The graph-based systems are highly efficient to retrieve items and other related items. Whenever a user searches for an item, the close items to this item can be gathered in an efficient way.

The rationale explained above shows that the core model we have presented is a perfect candidate for knowledge-based recommender systems.

3.4.4. Context-Aware Systems

User preferences may differ with context which can be time, location, genre etc. Context-aware recommender systems take the contextual information into account when suggesting items to users.

In different contexts, such as different times of the day, users are interested in different types of items [4] . An example of a context aware system is presented by Adomavicius et al. as a news recommender system [32]. While users prefer articles about business in the morning, they prefer leisure articles in the afternoon.

Based on the domain, different context variables are employed in recommender systems to achieve more accurate results. TV domain has its particular contexts like time of the day and the genre of the TV programs.

The presented core model involves context nodes to support context-aware recommendations. Time of day and genre are chosen as context variables. Other context variables may also be included in the core model based on different recommender system domains.

3.4.5. Group Recommendations

Television is common equipment that is used in both private and public areas. TV is generally viewed by a group of people sitting together. A sophisticated TV program recommendation system should be aware of the heterogeneity of viewers and suggest TV programs combining different interests of viewers [38].

Current research conducted for recommending items to a group of users focus on merging individual user profiles to build a group profile. The profile of the group is used to filter most relevant items to be recommended.

In our core model, a user profile is described as a sub graph which comprises a central user node and other related nodes and edges gathered by traversal algorithms. Similarly, a group of users can be modeled as a node type connected with user nodes representing the existence of the users in that group. A group profile in this sense becomes a sub graph which comprises a central node whose type is group and other related nodes and edges gathered by traversal algorithms. The traversal algorithm which creates a group profile can take into account the weights of each user interest into account. Therefore, the core model presented in this thesis can be employed for group recommendations as well by including the information about the membership of users to a group as edges between group nodes and user nodes.

3.4.6. Personalization for TV Users

One of the most common problems about TV domain is the limited capabilities of TV controller devices. Although new applications are developed and presented to connected TV users frequently, users suffer from the lack of user-friendliness of TV devices. Being cheap and easy, standard remote controllers are preferred by manufacturers instead of sophisticated ones.

Convenience of controlling TV devices can be achieved by personalization techniques developed based on the core model presented in this thesis. The core model keeps the information of most interested items to the users. These items can be represented in different types of nodes such as genre, actor, director, named-entity, term, channel etc. User interests gathered from this core model may be employed to create a personalized experience for users in different manners.

Generally, users organize the channels manually based on their channel preferences. However, by the virtue of the presented model, channels can automatically be organized based on the preferences of the viewer. The channels may dynamically be organized according to the continuous interest of users and the changed content of TV programs in these channels.

Zapping through channels to find the most suitable TV program for viewers is another exhausting process. A more sophisticated system may organize and present the current programs based on the users' interests. Changing the shape of standard channel oriented presentation into relevant program listings may improve the user satisfaction.

The core model presented in this thesis is able to perform the personalization approaches described above. The distance between user nodes and channels or current TV programs shows the relevance of these items to the user. For example, an automated channel organization according to users' interests can be achieved by computing the distance between user nodes and program nodes using the channel attribute of program nodes.

Another example of personalized usage of TVs is to present the personalized TV program listings to the users. When the current TV programs are inserted to the graph-based system, the geodesic distance between user nodes and program nodes can be calculated with or without using edge weights.

3.4.7. Recommending other types of items

As well as recommending TV programs to the users, the core model can also be used for recommending different types of items. Different types of items can be represented with different node types in this model. The contents of these node types can be connected to the attribute and descriptor nodes in the model. By inserting different types of nodes, such as cinemas, theaters, news, books etc., such types of items can also be recommended based on the preferences of users in TV domain.

Moreover, other types of items may include advertisements. Targeted advertisement is an emerging concept in both web and TV domain. If the advertisements are projected into this core model, the advertisements can be targeted to specific users, according to the TV program preferences of the users.

CHAPTER 4

A HYBRID RECOMMANDATION SYSTEM FOR SMART TV USERS USING GRAPH-BASED CORE MODEL

In this chapter, a case study is presented to illustrate capabilities of the core model. For this purpose, a hybrid recommendation system is implemented for the use of smart TV users. The recommendation system has context-aware, collaborative and content-based features. In the following sections, the main modules of the system, which are data collection and preprocessing, creation of graph database and traversal via spreading activation are presented in detail.

4.1. Data Collection and Preprocessing

The graph-based model is constructed using the user logs supplied by Arçelik A.Ş. The user logs include channel usage data obtained from Arçelik, Beko and Grundig brand connected TVs. The user logs are collected from real users in one month interval between 01-Dec-2013 and 0-Jan-2014. In that time interval, 10 million logs are collected from 2938 distinct users.

Another data collection process was the collection of TV program contents from external resources. The contents of the TV programs could be gathered from different content resources such as EPG (Electronic Program Guide) [39], web and third-party content provider companies. EPG data is embedded in DVB [40] protocol and supplied by broadcasters. However, in Turkey, EPG data is not adequate enough to extract content attributes and descriptors, and contextual information. Although third-party companies supply detailed content about TV programs, they demand very high prices for their services.

Web crawling [41] and web scraping [42] on the other hand, are the most suitable options for our purposes. Web crawling and scrapping methods are techniques to collect sparse and unstructured data on the web to create structured information to be stored and analyzed in local databases. In this thesis, web crawling and web scraping methodologies are applied in order to collect, analyze and store TV program information. After crawling web resources to obtain contents of the programs that were broadcast during the time interval matching the user logs, we created a data set which contains:

- 3769 distinct TV programs,
- 36 distinct genres,
- 1653 distinct actors,
- 469 distinct directors,
- 676 distinct named entities,
- 3159 distinct terms.

Since the user logs and TV program content information were collected from different resources, two independent data sources needed to be aggregated in order to build the user profiles. The aggregation of TV program information and user data is facilitated by matching the common attributes such as channel name, start time and end time. In the following sections, data collection processes and the aggregation of these data are presented in detail.

4.1.1. User Data Collection

4.1.1.1. Client Module

A reporter module is implemented and integrated to Arçelik, Beko and Grundig connected TV software in order to collect usage logs of users who grant permission to data sharing which is a part of the terms and conditions statements of Arçelik A.Ş. The data which is agreed to be collected by users are transmitted from TV devices to the database servers through this module running on client devices.

The module is implemented in a modular way that not only the channel usage logs of the users but the connected TV application usage logs are also collected in data servers. TV software is running Ubuntu as operating system and the modules are implemented as processes which are running over a service layer and communicating with the drivers. The reporter module is implemented using C++ including the embedded system libraries.

During system initialization of TV software, the reporter module is also initialized right after the system boot. Reporter module keeps the logs of channel change information on client-side in JSON format and sends these logs to the server over the internet every 5 minutes. The data kept in the client side includes *channel_name, frequency, tsonss, source_type, start_time* and *end_time*. Each record is enqueued in the memory to be sent to the servers. The attribute *tsonss* is a triple id which is used as identifier for channels [43].

The attributes in these logs are supplied from different modules of TV software. The name of the channel is an attribute of an EPG stack which is always supplied by channel broadcasters. Other attributes which are frequency*, tsonss* and *source_type* is gathered from the tuner module which is an embedded modulation component responsible for the modulation of DVB stack. The *start_time* and *end_time* attributes are computed at run time using UNIX time operations. The types of these attributes are described below:

- channel_name : *String*
- frequency : *Long Integer*
- tsonss : *Long Integer*
- source_type : *Integer*
- start_time : *Unix Timestamp*
- end_time : *Unix timestamp*

In order to reduce the network load between client and the server, instead of using strings to identify the attributes source_type and tsonss, an enumeration is defined as a protocol for the communication. Another issue taken into consideration for reducing the network load is that the accumulated logs are compressed into a GZIP file before sending to the server which reduces the data size in a remarkable way.A sample log including two channel usage data coming from a client is shown in Figure 4-1 as JSON format after extracted from GZIP format.

```
{
   "channels" : [
      {
         "end_time" : 1299855394,
         "frequency" : "746000",
         "name" : "France 5",
         "sourceType" : "0002",
         "start_time" : 1299855390,
         "tsonss" : "2314098422262
      },
      {
         "end_time" : 1299855398,
         "frequency" : "746000",
         "name" : "France 3",
         "sourceType" : "0002",
         "start_time" : 1299855394,
         "tsonss" : "2314098422261"
      }
]
}
```

Figure 4-1 : Log Sample

The security between the client devices and the server is another concern. User logs should be transmitted over a secure network to avoid the man-in-the-middle attacks emitting the data stream. In order for the communication to be safer, SSL protocol is used instead of standard http requests. The transmitted data is encrypted by certificates loaded uniquely to each device on the production line, and the server decrypts the data using the server-side certificate. The overall architecture of the client system is shown in Figure 4-2.

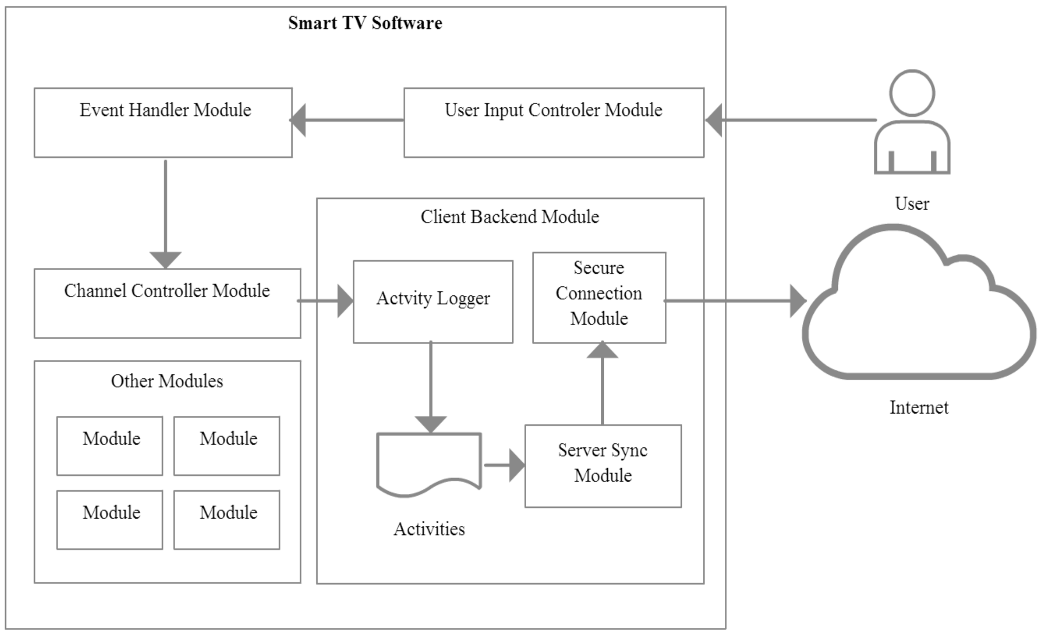


Figure 4-2 : Client Side Software Architecture

4.1.1.2. Server Backend Module

Server backend module is running on Ubuntu Server operating systems. These servers and the end-points created for handling client requests are implemented using *ruby* [44] language and *Ruby on Rails* [45] framework. On the upper layer of the server rack structure, *Nginx* [46] is employed to handle the requests and SSL certificate validation operations. Since there are lots of clients connected to the servers, a load balancer module is needed to distribute the requests over two replicated server nodes. Database layer of the server architecture, which connects and manages both relational and non-relational databases, is installed on these machines. The relational databases are not the most suitable option to keep the user logs. On the other hand, schemaless databases known as *Not only SQL (NoSQL)* are efficient agents to keep the user logs [47]. As the relational database *MySQL* and as non-relational database *MongoDB* is employed to keep the user data and user logs respectively.

Right after the logs coming from client devices are decrypted and forwarded to the associated end-point in the application layer of the server structure, the logs are uncompressed and inserted into the database. Before the insertion step, using the session information, the unique *device_id* and the unique *user_id* are merged with the related log. After this operation, user logs are ready to be queried. The dimensions of the stored collection are *user_id, device_id, channel_name, start_time, end_time, frequency, source_type and tsonss* as well as a unique *id* field incrementally set by database agent.

In this thesis, only channel_name, user_id, start_time and end_time attributes are used to build user profiles. Schema definition with the description of each of these attributes is given in Table 4-1.

Table 4-1: The Attribute Descriptions of Collected Logs

Attribute	**Description**
id	Unique id which is set by database agent
user_id	Unique id of the user
channel_name	Name of the channel
start_time	Start time of the watch event
end_time	End time of the watch event

Channel identification was another problem to solve. In different countries there are channels with the same name, which makes the channel_name attribute useless to identify the channels uniquely. In a global system, one should identify channels using a hash function considering frequency, tsonss, source_type, channel_name and identifier of the country. Although identifying channels using only their name is not possible in a global system, it is sufficient to use only channel_name attribute as an identifier in our system, since we are interested in only the users in Turkey. The research conducted by this thesis used the channel_name attribute as an identifier which is based on the assumption that in Turkey the channel names are unique. The detailed architecture of the server side modules is represented in Figure 4-3.

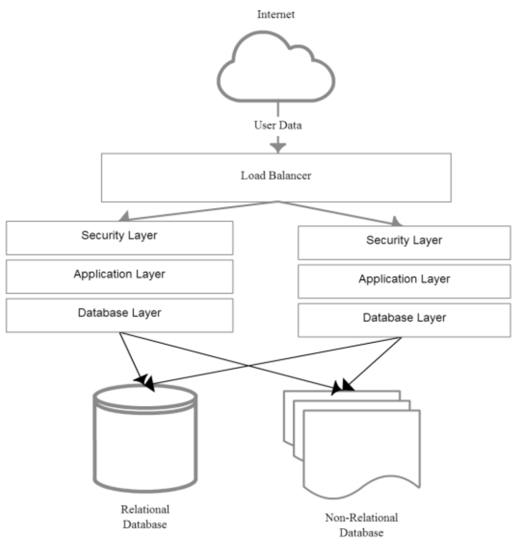


Figure 4-3: Server Backend Structure

4.1.2. Preprocessing of User Data

The data stored on the server needs to be cleaned before processing. Due to network problems redundant duplicate data can be stored in the database. For example, for a particular user, there are some rows where start_time, end_time and channel_name attributes are same which means redundancy since a user cannot watch TV more than once in a particular time period. A record based cleaning is applied to overcome the duplicate data problem.

Another data cleaning process needs to be done on time attributes. Precise timing is highly important to aggregate user data with the TV program information. Therefore, a time based cleaning is applied on the user log data to overcome the problems caused by wrong and imprecise time logs. Start time and end time attributes of a channel usage are gathered from system time of client devices which turns the precision of system time in client software into a critical component for the accuracy. The process checks the following conditions for time attributes in the user logs and excludes the redundant and imprecise log entries from the dataset:

- The entries where start_time > end_time.
- The entries where start_time < 2012 (the user log collection starts in 2012)
- The entries where end_time > now (applied while saving records)
- The entries where end_time - start_time > 8 Hours

In the scope of this thesis, only TV programs involving Turkish content has been taken into consideration. The brands Beko and Grundig are global brands which have been in the market in Europe, Russia and Middle East, however, user logs transmitted from devices sold in the countries other than Turkey are also excluded from the data set used in this study.

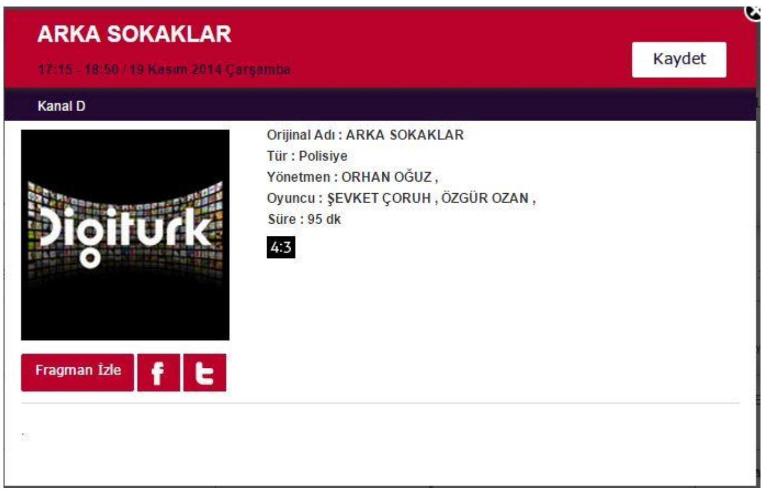
4.1.3. Program Data Collection

The contents of TV programs are extracted from the crawled websites. Digiturk[4] and Radikal[5] are chosen as the websites to be used as content providers. The data gathered from these websites are publicly available in different formats. Since an API is not provided for these websites and only publicly available content is used in the scope of this thesis, it was not necessary to be granted any permission from these resources.

[4] http://www.digiturk.com.tr/yayin-akisi
[5] http://www.radikal.com.tr/tvrehberi/

In order to collect the unstructured data provided by these websites, a software module is implemented which crawls these websites, collects the unstructured TV program data and migrates them into a predefined data structure which is constructed to keep the TV program information in the local servers of Arçelik.

Digiturk and Radikal contents were compared according to the variety of the attributes supplied by these websites for a TV program in a benchmarking stage. Although Digiturk provides descriptions of TV programs, these descriptions are not detailed enough to extract the descriptive attributes or named entities. Moreover, other attributes such as actors and directors which can be employed for a recommendation or a personalization system are not provided efficiently on their website. An example of a TV program content which is supplied by Digiturk is shown in Figure 4-4.


ARKA SOKAKLAR

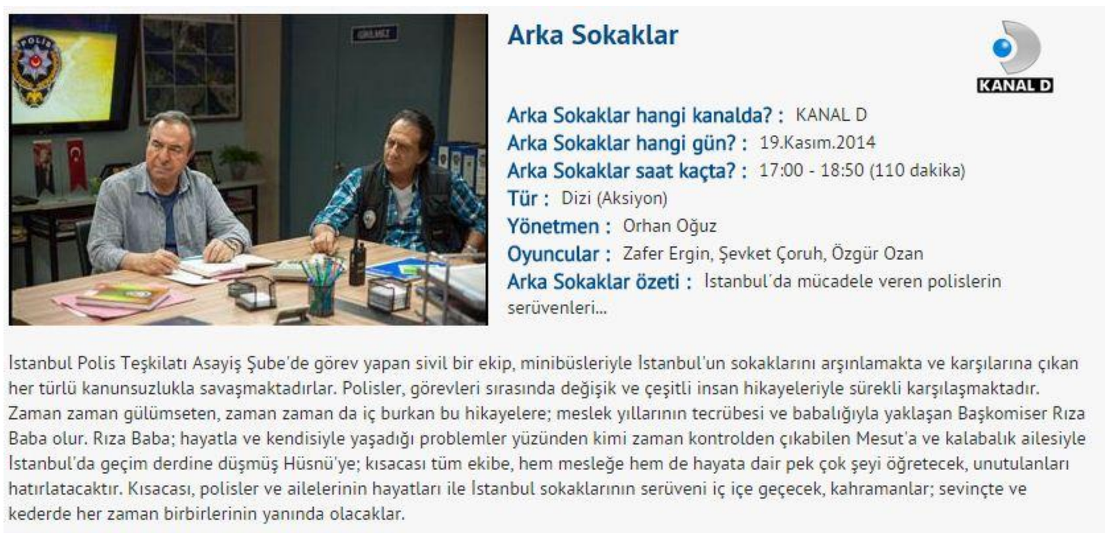
17:15 - 18:50 / 19 Kasım 2014 Çarşamba

Kanal D

Orijinal Adı : ARKA SOKAKLAR
Tür : Polisiye
Yönetmen : ORHAN OĞUZ ,
Oyuncu : ŞEVKET ÇORUH , ÖZGÜR OZAN ,
Süre : 95 dk

Figure 4-4: TV Program Content in Digiturk TV Guide

Radikal TV guide, when compared with Digturk TV guide, has much richer content of TV programs including director, actors and genre in a reliable way. The benchmark we conducted showed that Radikal TV guide is more viable for our purposes since the descriptions are more detailed and other attributes are sufficient and adequate. An example of the same TV program content obtained from Radikal TV guide is shown in Figure 4-5.



Arka Sokaklar

KANAL D

Arka Sokaklar hangi kanalda? : KANAL D
Arka Sokaklar hangi gün? : 19.Kasım.2014
Arka Sokaklar saat kaçta? : 17:00 - 18:50 (110 dakika)
Tür : Dizi (Aksiyon)
Yönetmen : Orhan Oğuz
Oyuncular : Zafer Ergin, Şevket Çoruh, Özgür Ozan
Arka Sokaklar özeti : İstanbul'da mücadele veren polislerin serüvenleri...

İstanbul Polis Teşkilatı Asayiş Şube'de görev yapan sivil bir ekip, minibüsleriyle İstanbul'un sokaklarını arşınlamakta ve karşılarına çıkan her türlü kanunsuzlukla savaşmaktadırlar. Polisler, görevleri sırasında değişik ve çeşitli insan hikayeleriyle sürekli karşılaşmaktadır. Zaman zaman gülümseten, zaman zaman da iç burkan bu hikayelere; meslek yıllarının tecrübesi ve babalığıyla yaklaşan Başkomiser Rıza Baba olur. Rıza Baba; hayatla ve kendisiyle yaşadığı problemler yüzünden kimi zaman kontrolden çıkabilen Mesut'a ve kalabalık ailesiyle İstanbul'da geçim derdine düşmüş Hüsnü'ye; kısacası tüm ekibe, hem mesleğe hem de hayata dair pek çok şeyi öğretecek, unutulanları hatırlatacaktır. Kısacası, polisler ve ailelerinin hayatları ile İstanbul sokaklarının serüveni iç içe geçecek, kahramanlar; sevinçte ve kederde her zaman birbirlerinin yanında olacaklar.

Figure 4-5: TV Program Content in Radikal TV Guide

One can observe the differences of the content between these content resources from Figure 4-3 and Figure 4-4 above. According to the benchmarking result conducted on two different TV program content provider websites, Radikal is chosen to be the main content provider for the research presented in this thesis.

A module is implemented using JAVA to crawl Radikal website and extract the data into local servers over internet. Since the selected user log set is between the dates 01-Dec-2013 and 10-Jan-2014, only the TV programs broadcasted in this time period were taken into consideration. The data crawled from web was transformed into JSON formatted objects and stored in the file server. The collected content is illustrated in Figure 4-6.

```
[
  {
    "name": "Gece Gündüz",
    "date": "05.Aralık.2013",
    "channel_name": "NTV",
    "startTimeStr": "00:30",
    "endTimeStr": "00:55",
    "duration": 25,
    "startTime": 1383604200,
    "endTime": 1383605700,
    "genreList": [
      "Yaşam",
      "Hobi"
    ],
    "imageUrl":
"http://i.radikal.com.tr/TvRehberi/320x240/2013/11/05/50582704212013
11050030.jpg",
    "summary": "Kültür ve sanatın gündemini yakalamak isteyenler
için haberler...",
    "longDescription": "1997 yılından bugüne sırasıyla Pınar
Demirkapı, İnci Türkay, Haşmet Topaloğlu, Ayşe Tolga ve son olarak
Yekta Kopan'ın sunumuyla sanatseverlere rehber olan Gece Gündüz,
bundan böyle Gülay Afşar'la devam ediyor. NTV Kültür Sanat ekibi
tarafından hazırlanan Gece Gündüz yeni sezonda da yıllardır
kemikleşen kaliteli çizgisini sürdürerek izleyicisiyle buluşuyor."
  }
]
```

Figure 4-6: TV Program Information in JSON Format

4.1.4. Program Data Enhancement

TV program content which is collected from web is semantically enhanced. The fields *longDescription*, *startTime* and *endTime* are employed to enrich the content with new attributes and contexts.

The core model presented in this thesis is designed to be used for TV program domain which has context types different from other domains. TV programs are broadcast in a certain period of time; therefore, it is important to take the time constraints into consideration. Another important approach about the time of TV programs is that, in different times of a day, the concepts of TV programs change. According to a research conducted by Deloitte Turkey in 2014 [48], for example, there are mostly news programs on air between 18:00 and 20:00 in Turkey. Moreover, the people in front of TV also vary during the day. While most of the family members are watching TV together in the evening, only one or two members of the family are watching TV in the middle of the day, which makes the TV program watching habits of people to differ based on whom they are watching it with. Therefore the time of the day information is used as contextual information in TV domain. TV program contents are examined according to their time of broadcast and context information is merged into the TV data.

The day is divided into 6 time slots based on a day parting research conducted on TV domain [49]. These partitions are used as context labels and the corresponding time periods are shown in Table 4-2.

Table 4-2: Day Partitions According to Time

Label	Time Period
NIGHT	00:00-04:00
EARLY MORNING	04:00-07:00
BREAKFEAST	07:00-09:00
LATE MORNING	09:00-13:00
DAYTIME	13:00-18:00
EVENING	18:00-20:30
PRIME TIME	20:30-24:00

Another enhancement operation is applied on the description field of TV programs. The descriptions of TV program are analyzed to generate *terms* and *entities*. Terms are generated using Natural Language Processing (NLP) methods applied on description field. Terms are stemmed words whose type is noun grammatically. The process for generating terms is shown in Figure 4-7.

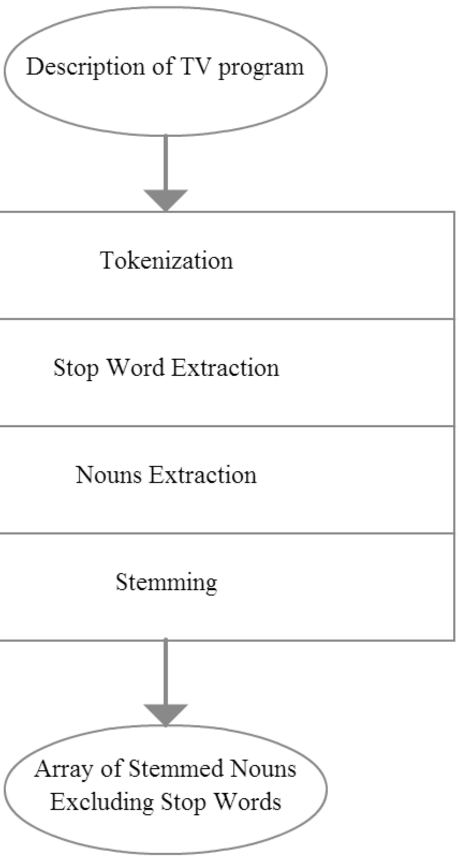

Description of TV program
Tokenization
Stop Word Extraction
Nouns Extraction
Stemming
Array of Stemmed Nouns Excluding Stop Words

Figure 4-7: Enhancement Process of TV Program Descriptions

First, the description is tokenized into words which are kept as a list of words. Stop words are the words which have no semantic meaning. These words are excluded from the tokenized list of words in order not to affect the semantic profile of TV programs. Turkish stop words are obtained from an open source project [50]. After extracting the stop words, based on the assumption that nouns have more semantic meaning than other types of words, nouns are filtered to be used in further operations. Each of these nouns is stemmed in order to generalize the same semantic meaning coming in different formats.

For the operations described above, one of the most popular open-source NLP projects conducted on Turkish language, ZEMBEREK is used [51]. Zemberek is an open source, platform independent, general purpose Natural Language Processing library and toolset designed for Turkic languages, especially Turkish. Zemberek is officially used as spell checker in Open Office Turkish version and Turkish national Linux Distribution, Pardus. After stemming, the words are ready to be used as terms in the TV program profiles.

The set of terms is not sufficient enough to model the semantic meaning of a description. Instead, more explicit terms or term groups which imply more sharp and straight meanings need to be spotted on description. Named entity extraction [52] methodologies are employed with the help of DBPedia APIs [53] to extract the Turkish named entities from descriptions [54]. Description fields of TV programs go through a process of annotation over public APIs of DBPedia. For example, considering the TV program description "*Mehmet Yaşin lezzet rotasını bu kez çok uzaklara, Avrupa'nın çatısı Norveç'e çeviriyor. 3 bölüm sürecek olan uzun Norveç gezisinin ilk durağı, dünyanın en kuzeyinde, kuzey kutup noktasından önce üzerinde insan yaşamı olan son ada Svalbard.*", the result set coming from DBPedia is shown in Figure 4-8 as XML.

```
<annotation text="Mehmet Yaşin lezzet rotasını bu kez çok uzaklara,
Avrupa'nın çatısı Norveç'e çeviriyor. 3 bölüm sürecek olan uzun
Norveç gezisinin ilk durağı, dünyanın en kuzeyinde, kuzey kutup
noktasından önce üzerinde insan yaşamı olan son ada Svalbard.">
<surfaceForm name="Mehmet Yaşin" offset="0"/>
<surfaceForm name="Norveç" offset="114"/>
<surfaceForm name="Svalbard" offset="230"/>
</annotation>
```

Figure 4-8: DBPedia Result for Specified Description

The surface forms spotted by DBPedia APIs have lexical units and have a page in Wikipedia [55]. Surface forms consist of a word or a group of words matching lexical units like *Anadolu* or *Mustafa Kemal Atatürk*. The description fields of TV program data are processed in order to extract the surface forms by the help of

Wikipedia[6] knowledgebase. These surface forms are aggregated with the TV program data structure as a list of *named-entities*.

The enhanced version of TV program is transformed into the enhanced version of TV program profiles presented in Figure 4-9. The enhanced version of a TV program content structure includes *stemmedWords*, *annotatedEntities* and *timeOfDay* fields for further usage in the generation of user profiles.

```
[
  {
    "name": "Gece Gündüz",
    "date": "05.Aralık.2013",
    "channel_name": "NTV",
    "startTimeStr": "00:30",
    "endTimeStr": "00:55",
    "duration": 25,
    "startTime": 1383604200,
    "endTime": 1383605700,
    "genreList": [
      "Yaşam",
      "Hobi"
    ],
    "imageUrl":
"http://i.radikal.com.tr/TvRehberi/320x240/2013/11/05/50582704212013
11050030.jpg",
    "summary": "Kültür ve sanatın gündemini yakalamak isteyenler
için haberler...",
    "longDescription": "1997 yılından bugüne sırasıyla Pınar
Demirkapı, İnci Türkay, Haşmet Topaloğlu, Ayşe Tolga ve son olarak
Yekta Kopan'ın sunumuyla sanatseverlere rehber olan Gece Gündüz,
bundan böyle Gülay Afşar'la devam ediyor. NTV Kültür Sanat ekibi
tarafından hazırlanan Gece Gündüz yeni sezonda da yıllardır
kemikleşen kaliteli çizgisini sürdürerek izleyicisiyle buluşuyor.",
    "stemmedWords": [
      "sıra",
      "pınar",
      "inci",
      "haşmet",
      "tolga",
      "yekta",
      "sunum",
      "sanatsever",
      "rehber",
      "gündüz",
      "böyle",
      "devam",
      "kültür",
      "sanat",
      "ekip",
      "taraf",
```

[6] http://www.wikipedia.org

```
       "gündüz",
       "yeni",
       "sezon",
       "kemik",
       "kalite",
       "çizgi"
     ],
     "annotatedEntities": [
       {
         "keyword": "İnci Türkay",
         "keywordTypes": [
           "Agent",
           "Http",
           "Person",
           "Artist",
           "Actor"
         ]
       },
       {
         "keyword": "Ayşe Tolga",
         "keywordTypes": []
       },
       {
         "keyword": "NTV",
         "keywordTypes": [
           "Agent",
           "Organisation",
           "Broadcaster",
           "TelevisionStation"
         ]
       }
     ],
     "timeOfDay": [
       "NIGHT"
     ]
   }
]
```

Figure 4-9: An Enhanced TV Program Content

4.1.5. User-Program Data Aggregation

User profile building using the data coming from different sources is another process in this thesis. The user logs and the corresponding TV program profiles are aggregated to build context-aware content-based user profiles. Aggregation process is applied by matching the *channel_name*, *start_time*, *end_time* attributes of user logs and TV program profiles. However the channel names coming from client devices and TV program guides are different syntactically. For example, while the name of the channel ATV HD in user logs could be “ATVHD”, “AtvHD”, “ATV HD” and “Atv HD”, in TV program guide it is “ATV HD”. For the purpose of user profile generation a mapper hash is created in order to match the channel names of the data gathered from these different resources. Sample channel name mapper hash for the channel “ATV HD” is shown in Table 4-3.

Table 4-3: Channel Name Mapper for Channel “ATV HD”

User Log – Channel Name	TV Program Attribute – Channel Name
ATVHD AtvHD ATVHD Atv HD	ATV HD

Besides the channel_name attribute, the interval between start_time and end_time of user logs are also employed to point the right TV programs to be added to user profiles. TV programs are filtered by channel_name and queried by start_time and end_time to extract the TV programs that the user watched. After aggregation operation, the user profiles are ready to be inserted to the graph database to build the core model.

4.2. Graph Based Model Construction

After data collection and preprocessing step, for each user in the data set, the enhanced profiles of watched programs are inserted to a graph database with different node / relation types and edge weighting metrics as described in Chapter 3. In this section, the engines and the methodology used to construct the graph based model are presented in detail.

4.2.1. Graph Based Database Engine

NoSQL (Not only SQL) is a paradigm used for database engines which store data in the formats different from schema based tables used in relational databases. Graph databases gain popularity among NoSQL databases among the developers of Web 2.0 applications as these graph-based databases premise to deliver superior performance when handling highly interconnected data. The rapid growth of NoSQL graph databases improved the availability of supplied documentation and support with increasing stability [56].

In this study, as database engine Neo4j[7], is installed and configured. Neo4j is an open-source graph-database, implemented in Java [57]. Neo4j is widely used especially in high scale web applications [58].

There are different query languages for Neo4J engines, one of which is *Cypher* [59], a declarative graph query language that is an expressive and efficient language intended to be employed in real world applications. The simplicity satisfied by Cypher allows focusing on the domain rather than getting lost in database access. An example Cypher query to extract most popular actors is shown in Figure 4-10.

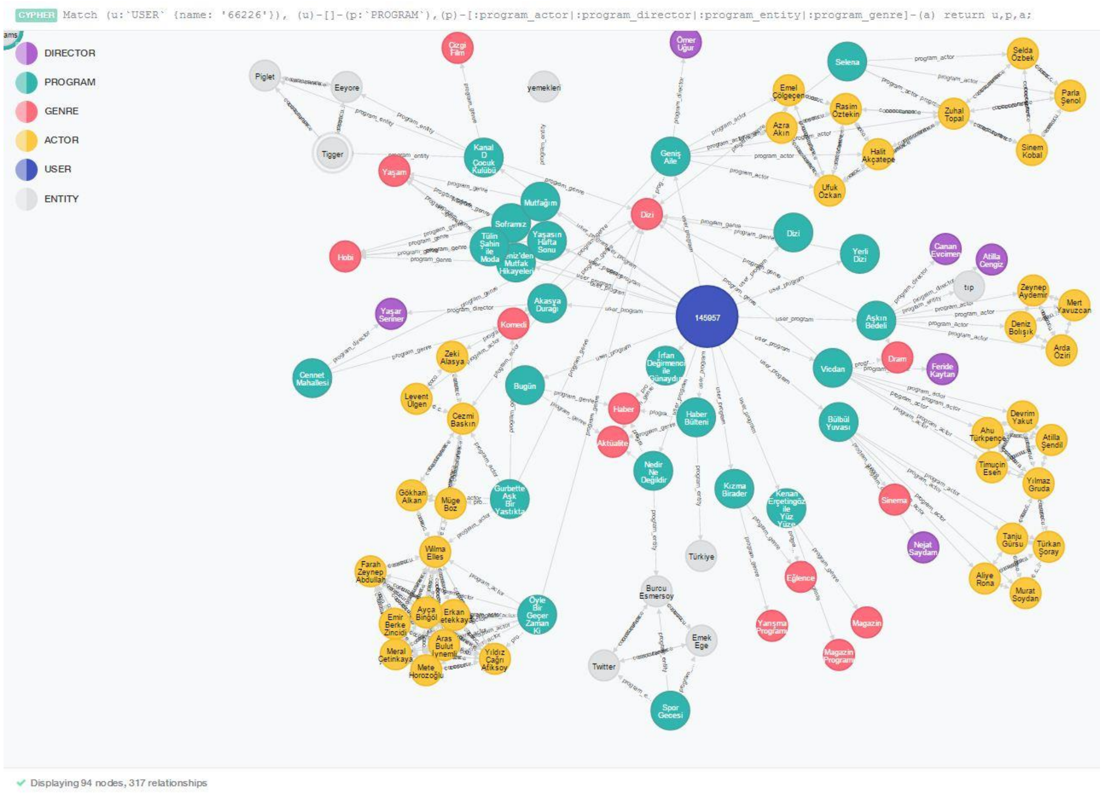
```
Match (u:`USER`)-[r1:`user_program`]-(p:`PROGRAM`)-
[r2:`program_actor`]-(a:`ACTOR`) with  count((a)) as popularity,
a.name as name order by popularity DESC return name, popularity
limit 10;
```

Figure 4-10: A Sample Cypher Query

[7] http://www.neo4j.org

4.2.2. Constructed Graph Based Profiles

The aggregated data is inserted into Neo4J graph database as different node types and edges using Cypher. Each node is indexed to be queried more efficiently. A partial profile of a user whose id is 145957 is shown in Figure 4-11. Not only the watched programs, but not watched programs having relationship with the watched ones are also shown in the figure.

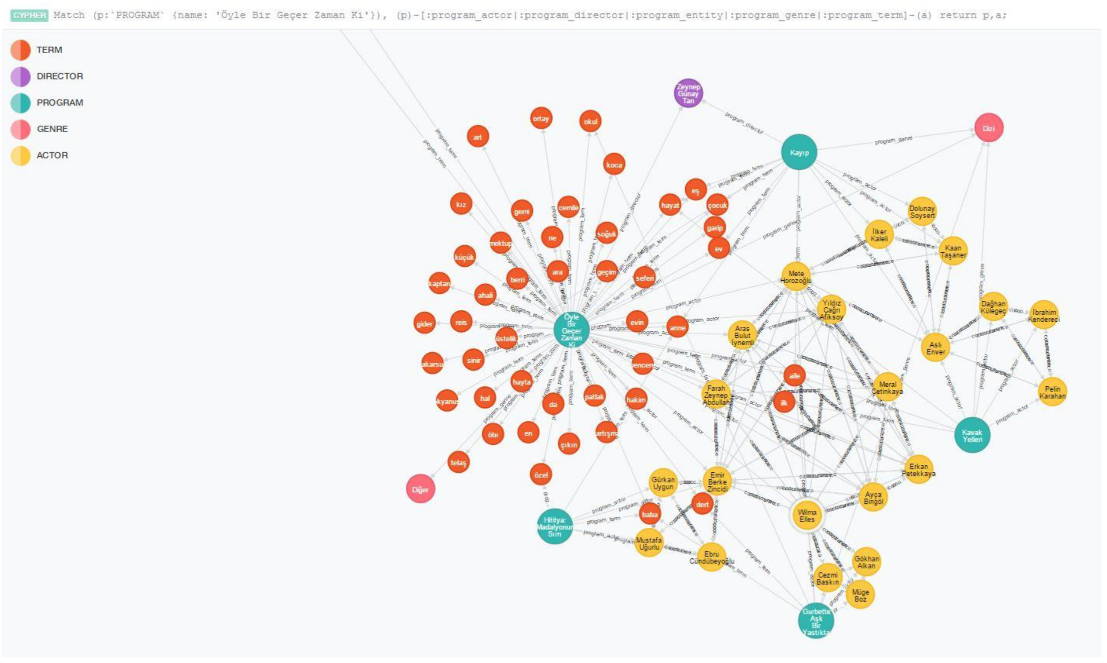


```
CYPHER Match (u:`USER` {name: '66226'}), (u)-[]-(p:`PROGRAM`),(p)-[:program_actor|:program_director|:program_entity|:program_genre]-(a) return u,p,a;
```

Displaying 94 nodes, 317 relationships

Figure 4-11: A Graph-Based User Profile

Similarly, an example of a program profile is shown in the Figure 4-12. In the figure, the program named “Öyle Bir Geçer Zaman Ki” is shown in graph structure with connected term, named-entity, actor and director nodes. Moreover, as seen in the figure, other programs accessible over connected nodes are included in the profile of that program. In the same sense, the profiles of other types of nodes can be constructed with the accessible nodes by deciding what depth the traversal continues.



```
CYPHER Match (p:`PROGRAM` {name: 'Öyle Bir Geçer Zaman Ki'}), (p)-[:program_actor|:program_director|:program_entity|:program_genre|:program_term]-(a) return p,a;
```

Figure 4-12: A Graph-Based Program Profile

4.3. Recommendation Algorithm

In order to verify the capabilities of the core model, a hybrid recommender system which comprises context awareness, collaborative and content filtering approaches is designed and implemented by means of graph traversals.

Traversal through the graph including graph-based user profiles and graph-based item representations is applied to recommend TV programs to the users based on their previous interests. Modified Spreading Activation [25] is applied over graph-based profiles of users to collect relevant items in a ranked way. Traversal visits the content nodes which turns the system into a content-based recommendation. Moreover, pre-filtering based on genre and time of day nodes is applied for context awareness. The co-occurrence relations are also employed to gain the collaboration between items.

4.3.1. Spreading Activation

Spreading activation is an algorithm designed for searching over associative networks, neural networks or semantic networks. Standard methodology offers labeling nodes with a weight called “activation” and propagating over other connected nodes. While propagating the nodes are labeled with an activation value which decays over each propagation. Activation process may originate from different paths. Spreading activation models are used in cognitive psychology [60] to represent the fan out effect. Moreover, in information retrieval systems, spreading activation can also be applied on documents which are represented as a graph that comprises of nodes and edges [25]. The theory of spreading activation proposes that each semantic concept is activated at the same time with the related concepts. The activation of one concept triggers the activation of the related concepts. According to the strength of the relations between activated concepts, some related concepts are activated faster than others. Spreading activation theory seeks to explain how the human mind processes the related ideas, especially semantic or verbal concepts. The priming effect [61], is introduced by psychologists as the tendency to recall concepts faster when introduced with a related concept. Presenting any concept to a person leads to prepare or *prim* him/her to recall information related to them [62].

In computer science, spreading activation algorithm is presented in order to model the human perception computationally. In a graph-based system the associations are weighted according to the relatedness of those items. An item is labeled with an activation value and the associated items are also activated with decreasing weight according to the association weights. In the end of propagating over items, the related items are gathered in a ranked way according to the activation values of these items.

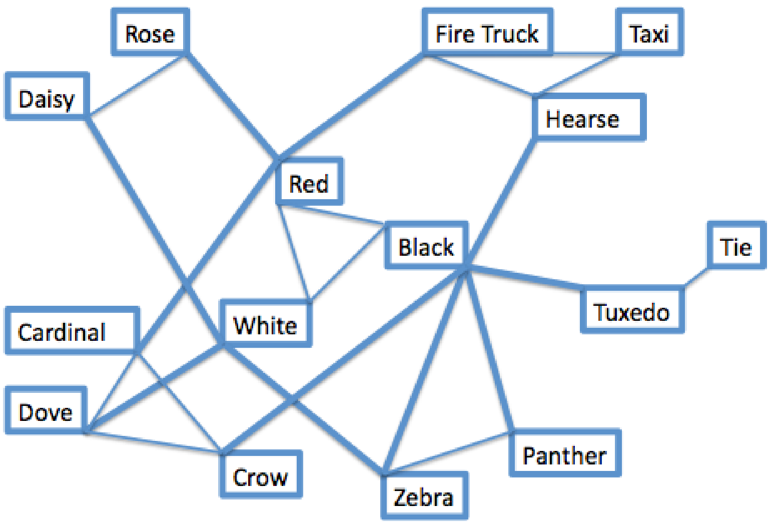


Figure 4-13: Spreading Activation Model (adapted from Collins and Loftus – 1975) [63]

Collins and Loftus have suggested the earlier hierarchical spreading activation model in their work [25]. In this model objects (e.g., fire truck), features (e.g, red) and the links between all of these are treated as concepts with distinct nodes. Their model is extended by Ferrand et al. and their visualization is shown in Figure 4-13. The main idea behind spreading activation models is that whenever a node of a concept is activated a pulse of this activation spreads through the links connected to that node and it activates all the nodes that this pulse passed through. The activation of these nodes is proportional to the closeness of these nodes and this pulsing process continues over links until it dissipates completely.

A mechanism for applying spreading activation over a hypothetical network shown in Figure 4-14 which operates starting from activating the nodes A and B , modulated by the edge weights. In this hypothetical network there are intermediate paths between the nodes which enable that the accumulation process can start at certain points. In Figure 4-14 node C is accumulated by the activation processes initiated from the nodes A and B.

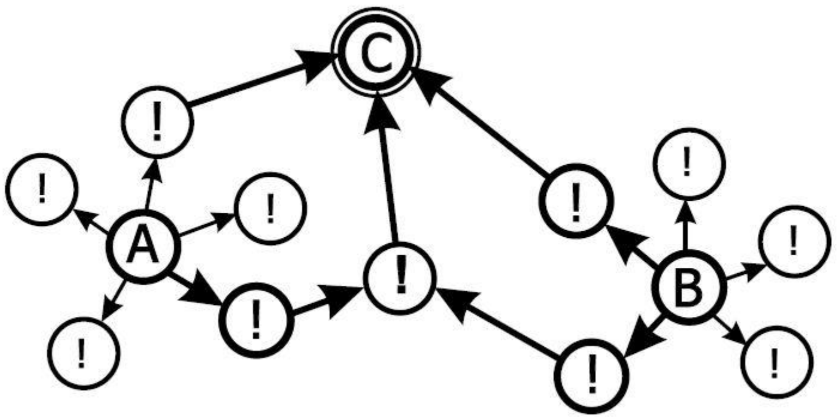


Figure 4-14: Activation spreads from the activated nodes A and B to accumulate in node C

Spreading activation is applied over the created graph database in order to generate TV program recommendations for users. The interests of the users on TV programs are modeled as a linked graph and generating the recommendations is implemented using the spreading activation technique.

Activating a user node and traversing through the linked nodes to spread the activation value, the system collects the closer TV programs for that user. The spreading activation algorithm is applied in the following way:

Let the set of nodes in the core model be

$$V = \{v_0, v_1, \dots, v_n\}$$

and the weighted graph defined on this set of nodes be

$$G = (V, E, W)$$

for which

$$E = V \times V$$

The function W for weight represents the closeness of the connected nodes as real values as described in Chapter 3. The weights over all connections in the graph G can be represented by $n \times n$ matrix M whose entiries $w_{i,j}$ are the values

$$w_{i,j} = W(a_i, a_j)$$

The state of the activation spreading process of the core model at time t will be represented by the vector A_t whose entries $a_i \in \mathbb{R}$ as

$$0 \leq a_i \leq 1$$

which indicates the activation value of each node v_i at iteration t. The vector is dynamicaally updated at each iteration as

$$A_{t+1} = M \cdot A_t$$

which shows the states of the vector.

The value of activation is determined at each iteration as the sum of the previous activations of the nodes and the new activation value coming from the iteration according to the edge weight.

$$a_t = \sum_{j=0}^{n} a_j \times w_{ij}$$

The spreading activation can be though as an iterative process of multiplication of A_t by the matrix M.

4.3.2. Modification of Spreading Activation According to TV Domain

The spreading activation algorithm is needed to be modified based on the needs of the domain specific attributes in order to produce more effective results. The context variables are needed to be taken into consideration while applying the algorithm. The variables, genre and time of day, are employed to filter out the similarity calculations between the TV programs of different genres or broadcast at different times of days. Another modification is to choose the decay factor which is a loss of energy between nodes over the iterations. For example, moving through co-occurrence relations the spreading activation is reduced since the nodes are getting far from the user nodes showing reduction of user interest on these concepts. Moreover the unwatched or not broadcast TV programs need to be marked since in TV domain recommending a past program is not acceptable.

4.3.2.1. Starting Activation

To recommend a TV program to the users, the activation process is started from a user node. The spreading process accumulates the previously watched TV programs by passing through the user_rating relations which are weighted as the normalized value of the interest of that user to these TV programs. These TV programs are accumulated using the formula below where $w_p$is the activation value of TV program p initiated by user u. The function $rating$ gives the normalized weight of the user_program edge between user node u and program node p.

$$w_p = rating(u, p)$$

4.3.2.2. Context Based Pre-Filtering

After determining the activation value of the program p, the context based pre-filtering is applied using the genre nodes and time of day nodes. The unwatched and not broadcasted program nodes which have the same genre and time of day connections in the system are labeled as traversable to be collected as recommendations.

4.3.2.3. Moving Through Program Nodes

After pre-filtering operations, spreading moves through the other connected nodes with the program p. Other nodes are actors, directors, term and entities in the core model. The descriptive nodes which are term and named-entity nodes are connected with a weighted edge to the program nodes, on the other hand, actor and director nodes are connected with un-weighted edges which differs the behavior of spreading algorithm on these nodes.

Descriptor nodes are accumulated by activation spreading taking the edge weights into consideration. The determination of weights of the descriptor nodes is shown below where $w_s^{'}$ is the new activation value of the descriptor node, w_p is the activation value of the program node, $decay\_factor$ is loss of passing which is set 0.6 heuristically, $tfidf(p,s)$ is the edge weights of the edges program_term and program_entity and w_s is the previous activation value of the descriptor node s. By this calculation the weight of the program node is distributed over the connected nodes. Moreover, the previous weight of the descriptor node s was taken into consideration since the descriptor nodes could be accumulated from different paths of the spreading algorithm (e.g. two different TV programs that the user u has watched may contain the same named-entity in their descriptions). The $decay\_factor$ makes the attribute nodes more effective than descriptor nodes for the similarity of programs.

$$w_s^{'} = tfidf(p,s) \times w_p \times decay\_factor + \ w_s$$

Attribute nodes which are actor and director nodes in this core model are accumulated in a similar way with the descriptor nodes. The determination of weights of the attribute nodes is shown below where $w_a^{'}$ is the new activation value of the attribute node, w_a is the previous activation value of the actor node, w_p is the activation value of the program node and $decay\_factor$ is loss of passing which is set 0.6 heuristically.

$$w_a' = w_p \times decay\_factor + w_a$$

After passing over attribute and descriptor nodes, the spreading process passes thorough the co-occurrence relations which connect same type of nodes in a weighted manner. However, co-occurrence relations do not show the direct user interest on the connected nodes but makes the system collaborative over items. A new factor is determined namely loss factor which is 0.4 (less than decay factor chosen). The determination of the weights can be seen in the formula below where w_n' is the new activation value of node n, $w_{n'}$ is the activation of the incoming node and w_n is the old activation value of node n.

$$w_n' = w_{n'} \times loss\_factor + w_n$$

The spreading iteration continues in the similar way on all connected nodes to the user node u for each program preference of that user until the activation values dissipates and becomes 0.2. After the spreading process is finished, the marked unwatched and not broadcasted TV program nodes are collected with their activation values which are used for ranking in future processes.

Applying activation spreading, the current and unwatched TV programs are collected for specified users. TV programs are collected according to their semantic contents by means of term and named-entity nodes which turns the system into a *content-based* recommender system. Moreover, before collecting the TV programs, genre and time of day contexts are employed for pre-filtering operations which turns the system into a *context-aware* recommender system. Similarly, *collaborative filtering* approach is applied by means of the co-occurrence between terms, entities, actors and directors.

The results are produced by a hybrid algorithm which is a combination of different approaches. The core model we presented in this thesis is verified as applicable for different recommender system approaches.

CHAPTER 5

EXPERIMENTAL RESULTS

Several experiments are conducted in order to evaluate the performance of the recommender system and the core model presented in this thesis. In this chapter the evaluation strategy and the metrics used for the evaluation of the system are introduced. The results are presented in detail and discussed by comparing them with the results of a baseline method. The effects of context-awareness and collaboration approach are analyzed and presented.

5.1. Evaluation Strategy and Metrics

5.1.1. Evaluation Strategy

In order to evaluate the success of our recommender system, *k-fold cross-validation* strategy is employed [64]. Cross-validation also known as rotation estimation is a set of techniques used mainly for validating the results of predictive systems by means of running the system on pre-collected data to estimate its performance in practice. Since further samples of data can be hazardous, costly or not possible to collect, the results of the k-fold validation techniques are preferred to estimate the real-world accuracy of a predictive system.

In cross-validation strategy the pre-collected data is divided into two complementary parts one of which is called *training* data and the other one is called *test* data. In order to evaluate our system, 3-fold strategy is selected where the evaluation is conducted in 3 iterations selecting different parts of the total samples as train and test partitions and the overall evaluation result is the average of the results of iterations.

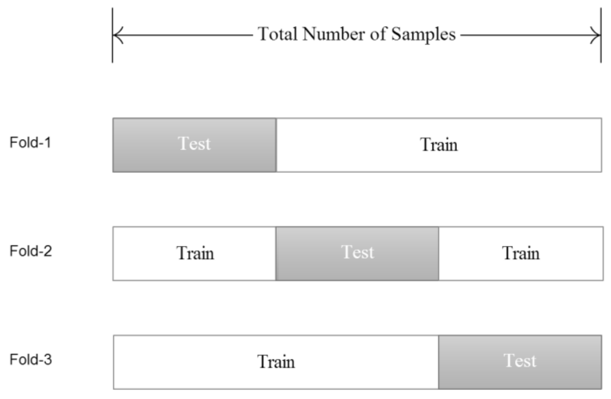


Figure 5-1: 3-Fold Data Partitions

The overall samples in our system are the TV programs watched by particular customers. The evaluation process is applied three times for each customer selecting different parts of the data.

5.1.2. Evaluation Metrics

Most common metrics that are used to evaluate the recommender systems are *precision* and *recall* which are set-based metrics showing the fraction of relevant items over retrieved and all relevant items. The combination of these metrics gives the *f-measure* which measures the success of recommender systems.

Moreover, these metrics are fit into a *table of confusion* which can also be called as *confusion matrix*. Confusion matrix is a 2x2 contingency matrix whose cells represents *false positives*, *false negatives*, *true positives* and *true negatives* in order to calculate the success of predictive systems merely.

Table 5-1: Confusion Matrix

		Predicted TV Programs	
		positive	*negative*
Known Labels	*positive*	True Positives	False Negatives
	negative	False Positives	True Negatives

The confusion matrix is filled with the actual users' watching history of TV programs and the predicted TV programs. True positives are the number of **correct** predictions, false negatives are the number of **missed** items that are watched by users but not predicted for users. False positives show the number of predicted TV programs that are not watched by users. True negatives show the number of items that are not predicted and not watched by users.

5.1.2.1. Precision

In pattern recognition and information retrieval systems, the fraction of retrieved relevant instances over all instances returned by the system is called precision. The metric precision shows how many of the returned programs are predicted correctly.

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

5.1.2.2. Recall

In pattern recognition and information retrieval systems, the fraction of relevant instances over all relevant instances is called recall. The metric recall measures how many of the watched TV programs are returned correctly.

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

5.1.2.3. F-measure

Harmonic average of precision and recall gives the f-measure which shows the success of the system according to all returned items and all the items that the user has watched.

$$f - measure = 2\ x\ \frac{precision\ x\ recall}{precision + recall}$$

5.1.3. Experiments

The experiments are conducted on a data set which contains 2938 distinct users. The average number of watched programs per user is 32.24. Total number of TV programs in the system is 3769.

For each user, the set of watched TV programs is extracted for partitioning as train and test sets. The train set is given to the core system to create graph-based user profiles before applying the recommendation algorithm by means of spreading activation. The result set gathered from the recommendation system is compared with the test set of this particular user. The set is treated as if they were unwatched and the programs that have not broadcast yet in order to simulate the real world applications.

3-fold strategy is employed for each user to apply the evaluation methods on the dataset and in each of the folds different test and train samples of data is selected. According to the correct and false predictions, precision, recall and f-measure metrics are calculated.

5.2. Results and Discussion

5.2.1. Evaluation Results

The results of the 3-fold cross evaluation are shown in Table 5-2. The proposed system produced the results below:

- Average precision: 0.72356,
- Average recall: 0.67976,
- Average f-measure: 0.69409.

Table 5-2: Precision, Recall and F-measure Result of our System

	Precision	Recall	f-measure
Fold-1	0.7534	0.7011	0.72631
Fold-2	0.6875	0.6398	0.66279
Fold-3	0.7298	0.6984	0.69309
Average	**0.72356**	**0.67976**	**0.69409**

5.2.2. Discussion

The results of the recommendation system presented in this thesis are compared with baseline methods and other works presented in TV domain. Moreover, the effects of the context awareness and co-occurrence relations are presented.

5.2.2.1. Comparison with Baseline Method

In the work [65] presented by Marangoz, as baseline method, using the same dataset content-based user profiles are built which is kept in user x item vectors. The long term preferences of the users are collected using the words in TV program descriptions. Apriori algorithm [66] is employed to build the user profiles based on user x item vectors. An example of user x item vector is shown in Table 5-3.

Table 5-3: User x Item Vector

User	Matched Terms
User1	"belgesel", "aslan","göl", …
User2	"spor","ispanya","gol" …

After generating user x item vector, the Apriori algorithm is employed in order to generate the frequent items for users. The result of Apriori algorithm is an inverted index of user item pairs with item weights according to the users. The inverted item representations for the users are shown in Table 5-4. As seen in the table, User1 has watched TV programs including the word "belgesel" 23 times, User2 has watched 35 times and so on.

Table 5-4: Inverted Index for User x Item Representation

Item	User Ratings
"belgesel"	User1 => 23,, User2 => 35, User3 => 13, User4 => 4 …

When a new program comes into the system, the terms appearing in that program are compared with the inverted index matrix and according to the ratings (interests) of users on these terms, the user set who has remarkable interest on these terms are collected and the program is recommended to this user set.

This method is commonly used as baseline methodology for evaluating recommendation systems in information retrieval domain [67]. Most of the recommender system approaches are built based on this model. The core model of this system is based on the transactional and document based databases.

The baseline method is also evaluated by 3-fold cross validation on the same data data set [65]. The precision of the baseline recommender is compared with the recommender system presented in this thesis (see Figure 5-2).

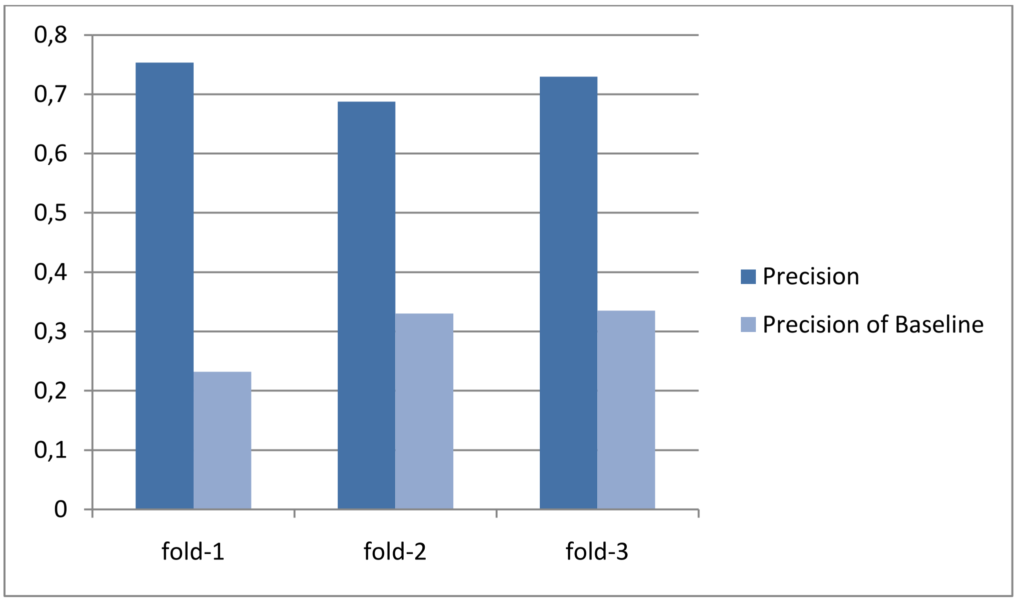


Figure 5-2: Precision of Our System vs Precision of Baseline Method

The results of comparing the baseline method with the recommender system presented in this thesis w.r.t. recall metric are shown in Figure 5-3.

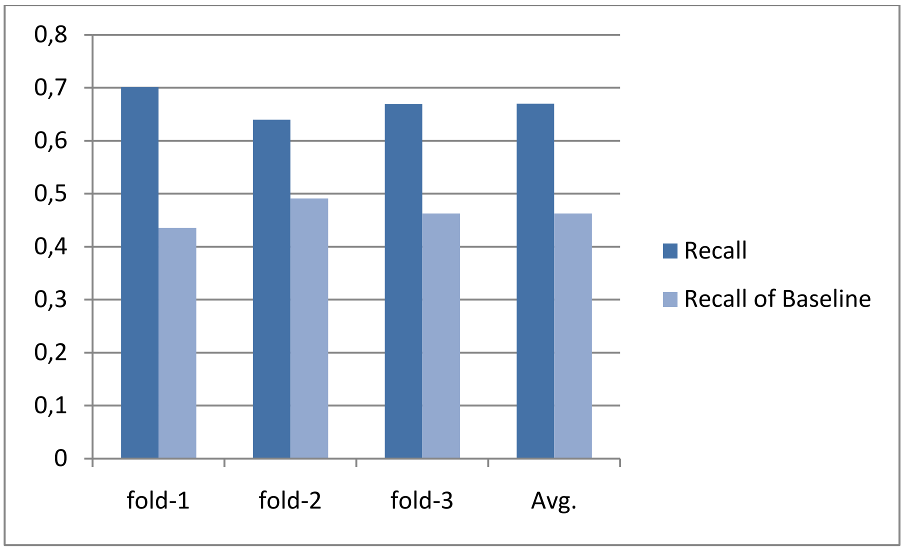


Figure 5-3: Recall of Our System vs Recall of Baseline Method

The results of comparing baseline method with the recommender system presented in this work w.r.t. f-measure metric are shown in Figure 5-4.

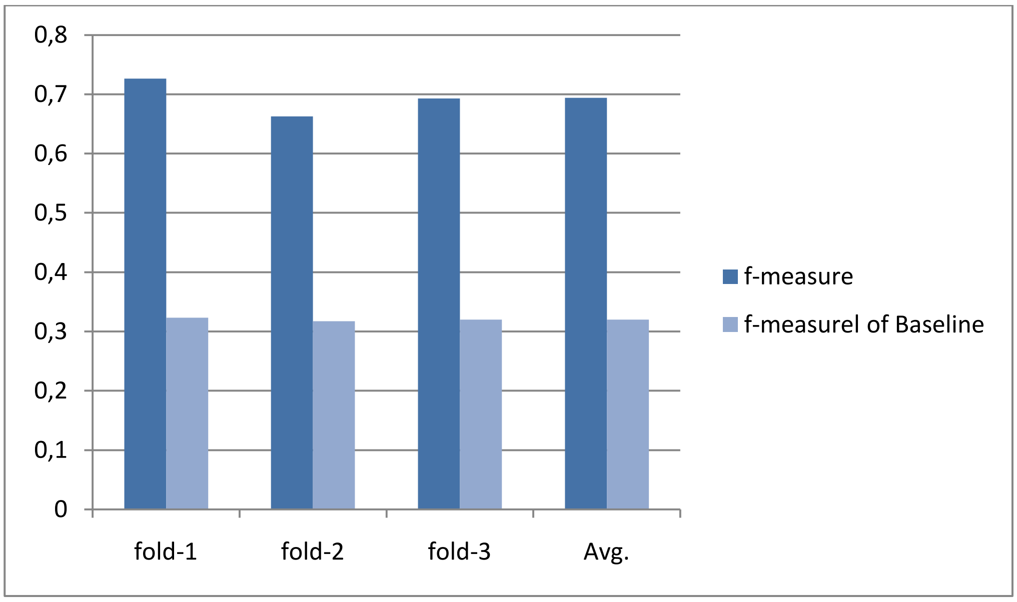


Figure 5-4: F-measure of Our System vs F-measure of Baseline Method

As seen in the results, our proposed methodology shows much more successful results based on the evaluation metrics.

5.2.2.2. The Effect of Context

The effect of the context variables on the success of the system is analyzed with further experiments. The same methodology is employed to generate TV program recommendations with and without pre-filtering operations. In the first experiment, all the pre-filtering operations (context-awareness) are disabled. After that pre-filtering is applied only the genre attributes of the programs. Finally, the time of date context added version is observed. In the Figure 5-5, the f-measures are shown in 3-fold strategy.

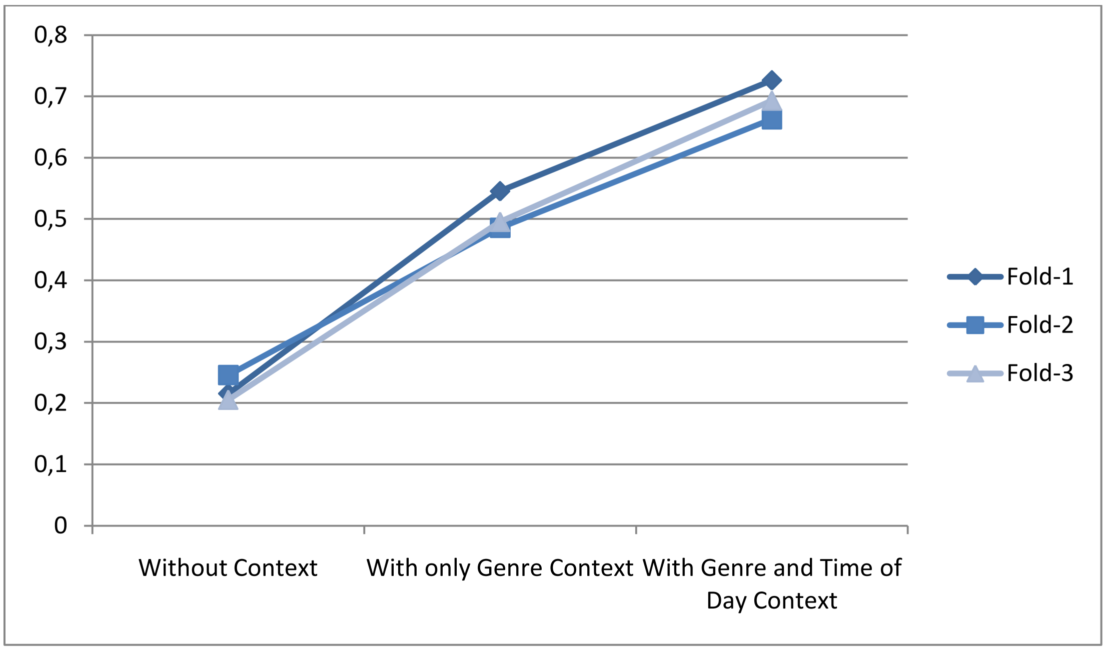


Figure 5-5: Context-Awareness Effect

As seen in the results, context-awareness improves the success of the recommendation algorithm in a remarkable way.

5.2.2.3. The Effect of Co-Occurrence Relations

The collaboration effect is satisfied using the co-occurrence relations in the core model presented in this thesis. The effect of the co-occurrence relations on the success of the system presented is analyzed with further experiments.

In the core model, the co-occurrence typed edges exist between the actor-actor nodes, term- term nodes and named-entity - named-entity nodes. In the spreading activation algorithm the co-occurrence relations are taken into consideration in order to add the sense of collaboration into recommender system. The spreading algorithm is applied without passing through the co-occurrence relations to measure the effect of these relations. The results are presented in the Figure 5-6.

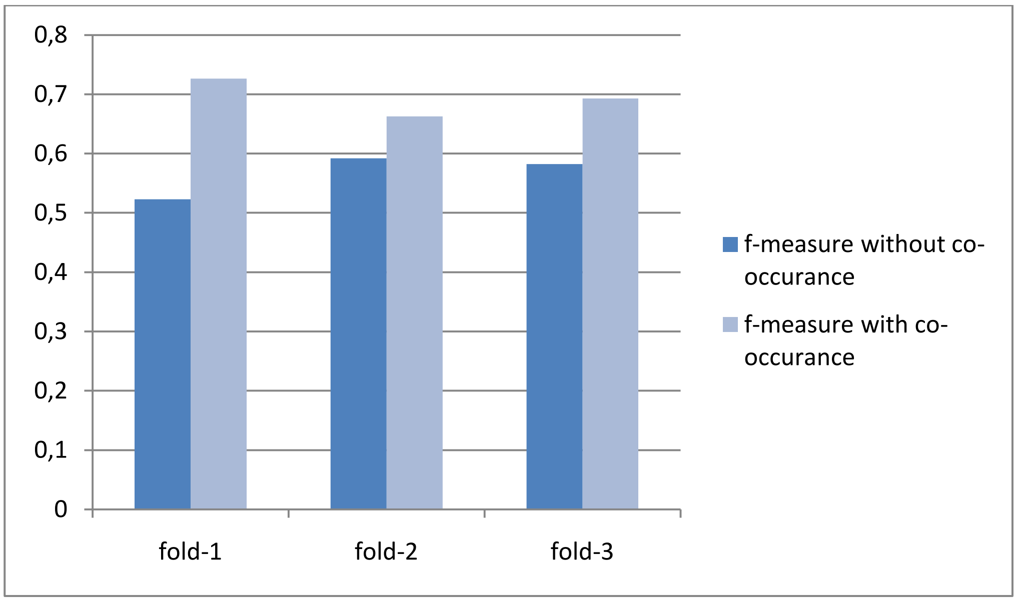


Figure 5-6: F-measure with and without co-occurrence relations

As seen in Figure 5-6, co-occurrence relations improve the success of the system not a remarkable but a reasonable way.

5.2.2.4. Overall Improvement

When the context was excluded from our system but the co-occurrence relations were kept, the results were less successful with respect to the baseline method. Because of the co-occurrence relations, the activation process reached unrelated TV programs since there was no restriction based on any context. This was an expected result since the context awareness was excluded and spreading activation algorithm moved through irrelevant TV programs throughout co-occurrence relations. The improvement based on the effect of context with respect to baseline method is shown in the Figure 5-7. As seen in the Figure 5-7, the context is highly effective on the success of the system.

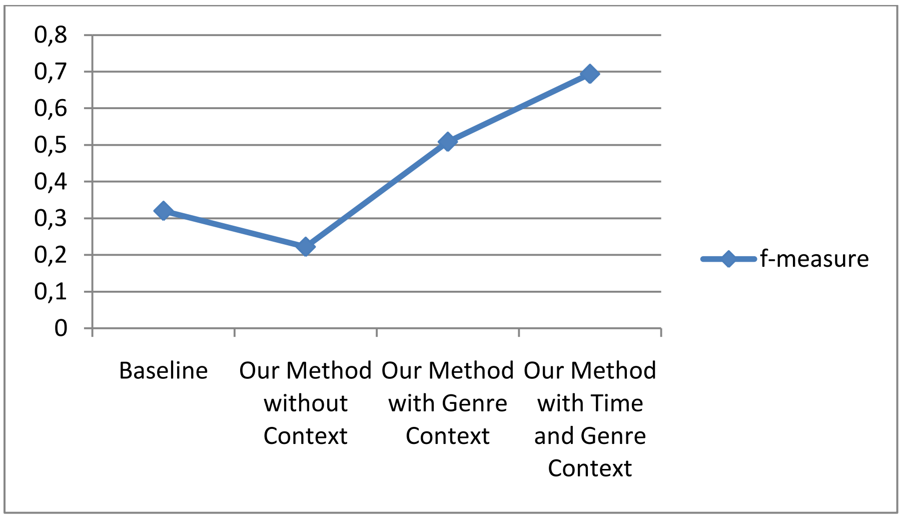


Figure 5-7: The Improvement Based on Context w.r.t Baseline Method

When the co-occurrence relations were ignored but the context awareness (pre-filtering based on time and genre) is kept, the improvement satisfied by co-occurrence relations is shown in Figure 5-8.

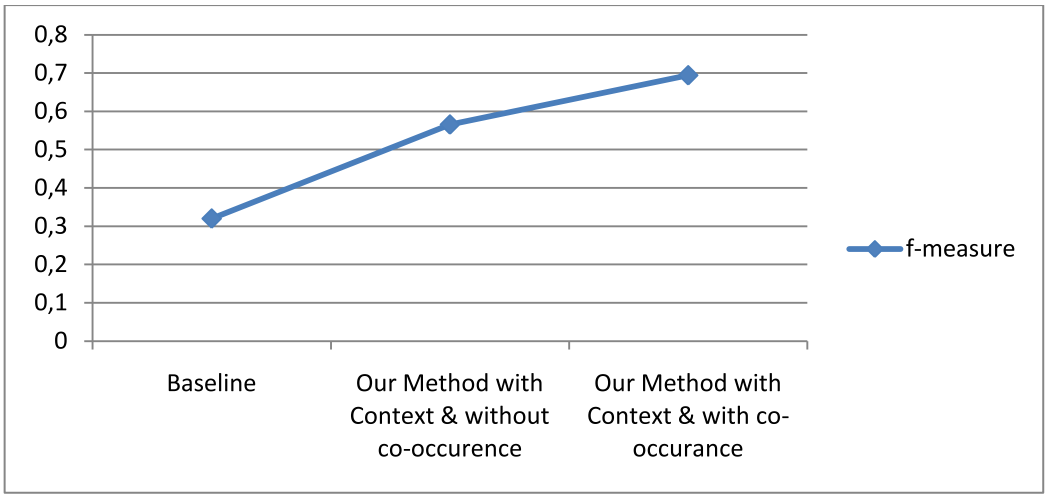


Figure 5-8: The Improvement Based on Co-occurance w.r.t Baseline Method

Overall improvement of our system w.r.t baseline method can be seen in Table 5-5 and Figure 5-9: Overall Improvement of Our System w.r.t Baseline.

Table 5-5: Overall Improvement of Our System w.r.t Baseline

Method	**f-measure**
Baseline	0.3201
Our Method without context & with co-occurence	0.2221
Our Method with Genre Context & with co-occurance	0.5087
Our Method with Time and Genre Context & without co-occurence	0.5655
Our Method with Time and Genre Context & with co-occurance	0.6940

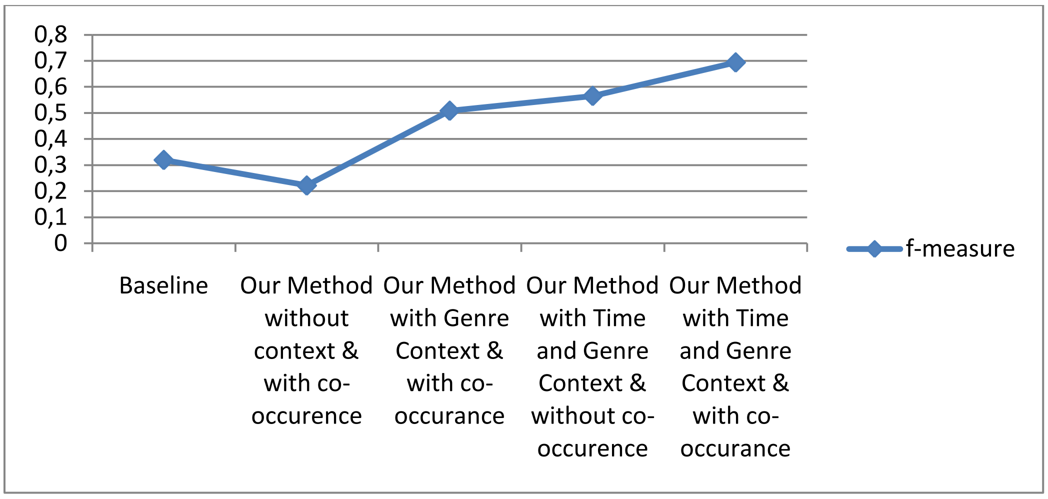


Figure 5-9: Overall Improvement of Our System w.r.t Baseline

CHAPTER 6

CONCLUSION AND FUTURE WORK

In this thesis a graph-based core model for representing users and TV programs with their attributes is presented. Moreover a hybrid recommender system based on this core model is explained in detail. The recommender system is evaluated by comparing it with a baseline method and the results show that the proposed system produces remarkable results.

A sparse data is collected over web with web crawling and web scrapping methodologies to form a structured database of TV program listings. The user data is collected and pre-processed to create graph-based user models.

After the sparse data of TV programs is formed and matched with user log data gathered from connected TVs, a graph-based core model is created to represent the users and the TV programs with their attributes. In order to project the real world entities into the graph-based model, entities such as user, program, actor, director are categorized as entity nodes, descriptor nodes, context nodes and attribute nodes. The edge weights are built based on different metrics according to the node types that they are connecting. The capabilities of this core model are explained in detail with examples.

A hybrid recommender system is created based on the core model to verify the capabilities and to measure the success of the system. As a traversing algorithm, the spreading activation is chosen to create the TV program recommendations for the users.

After creating the recommender system, the evaluation is applied based on the metrics f-measure, precision and recall. The results are compared with the baseline method created with the same data. The produced results of our system are better than the results of baseline method.

The system can be extended or improved in the following ways:

- Performance improvements by importing social media profiles of users,
- Performance improvements by importing demographic information of users,
- Creating personalized TV user interfaces,
- Creating targeted advertisements based on TV program preferences of the users,
- Recommending cinemas, theaters or shows based on TV program preferences of the users,
- TV program, actor, director, genre etc. rating estimations.

REFERENCES

[1] R.T.U.K., “RTUK Araştırmaları - 2013,” 2013. [Online]. Available: http://www.rtuk.org.tr/Icerik/DownloadReport/13. [Accessed: 01-Dec-2014].

[2] R.T.U.K., “RTUK - TV İzleme Alışkanlıkları,” 2013. [Online]. Available: http://www.rtuk.org.tr/Icerik/DownloadReport/3691. [Accessed: 01-Dec-2014].

[3] G. I. Webb, “Posterior Probability,” *Encyclopedia of Machine Learning*, vol. 56, no. March. p. 780, 2010.

[4] G. Adomavicius, B. Mobasher, F. Ricci, and A. Tuzhilin, “Context-Aware Recommender Systems,” in *Recommender Systems Handbook*, 2011, pp. 217–253.

[5] J. Ben Schafer, J. Konstan, and J. Riedi, “Recommender systems in e-commerce,” *Proceedings of the 1st ACM conference on Electronic commerce EC 99*, vol. 2001, pp. 158–166, 1999.

[6] M. J. Pazzani and D. Billsus, “Content-Based Recommendation Systems,” *The Adaptive Web*, vol. 4321, pp. 325–341, 2007.

[7] “TF-IDF - Wikipedia, the free encyclopedia.” [Online]. Available: www.wikipedia.org/wiki/tf-idf. [Accessed: 01-Dec-2014].

[8] P. Lops, M. De Gemmis, and G. Semeraro, “Content-based Recommender Systems: State of the Art and Trends,” in *Recommender Systems Handbook*, 2011, pp. 73–105.

[9] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, “GroupLens : An Open Architecture for Collaborative Filtering of Netnews,” in *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, 1994, pp. 175–186.

[10] J. Schafer, D. Frankowski, J. Herlocker, and S. Sen, “Collaborative filtering recommender systems,” *The adaptive web*, pp. 291–324, 2007.

[11] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, “Item-Based Collaborative Filtering Recommendation Algorithms GroupLens Research Group / Army HPC Research Center,” *Proceedings of the 10th …*, vol. 1, pp. 285–295, 2001.

[12] N. Tintarev and J. Masthoff, “Designing and Evaluating Explanations for Recommender Systems,” in *Recommender Systems Handbook*, vol. 22, 2011, pp. 479–510.

[13] R. Burke, “Knowledge-based recommender systems,” *Encyclopedia of library and information systems*, vol. 69, no. Supplement 32, pp. 175–186, 2000.

[14] A. Felfernig and R. Burke, “Constraint-based recommender systems: technologies and research issues,” in *Proceedings of the 10th international conference on Electronic commerce ICEC ’08*, 2008, vol. 8, pp. 1–10.

[15] G. J. F. Jones, “Challenges and opportunities of context-aware information access,” in *Proceedings of the International Workshop on Ubiquitous Data Management , UDM 2005*, 2005, pp. 53–60.

[16] R. Burke and R. Burke, “Hybrid Recommender Systems: Survey and Experiments 1,” *User Modeling and User-Adapted Interaction*, pp. 1–29, 2002.

[17] Li-Cai, Xiang-Wu, and Yu-Jie, “Context-Aware Recommender Systems,” *Journal of Software*, vol. 23, pp. 1–20, 2012.

[18] I. Zukerman and D. W. Albrecht, “Predictive statistical models for user modeling,” *User Modeling and User-Adapted Interaction*, vol. 11, pp. 5–18, 2001.

[19] G. Salton, A. Singhal, M. Mitra, and C. Buckley, “Automatic text structuring and summarization,” *Information Processing & Management*, vol. 33. pp. 193–207, 1997.

[20] R. van Meteren and M. van Someren, “Using Content-Based Filtering for Recommendation,” in *ECML/MLNET Workshop on Machine Learning and the New Information Age*, 2000, pp. 47–56.

[21] Z. Huang, W. Chung, T.-H. Ong, and H. Chen, “A graph-based recommender system for digital library,” *... conference on Digital libraries*, pp. 65–73, 2002.

[22] Y. Jing, S. Baluja, R. Seth, J. Yagnik, D. Sivakumar, S. Kumar, D. Ravichandran, and M. Aly, “Video Suggestion and Discovery for YouTube: Taking Random Walks Through the View Graph,” in *Proceeding of the 17th international conference on World Wide Web*, 2008, pp. 895–904.

[23] G. Öztürk and N. K. Cicekli, “A hybrid video recommendation system using a graph-based algorithm,” in *Proceedings of the 24th international conference on Industrial engineering and other applications of applied intelligent systems*

conference on Modern approaches in applied intelligence-Volume Part II, 2011, pp. 406–415.

[24] T. Bogers, “Movie recommendation using random walks over the contextual graph,” in *Proc. of the 2nd Intl. Workshop on Context-Aware Recommender Systems*, 2010.

[25] A. M. Collins and E. F. Loftus, “A spreading-activation theory of semantic processing.,” *Psychological Review*, vol. 82. pp. 407–428, 1975.

[26] L. Ardissono, A. Kobsa, and M. T. Maybury, “Personalized digital television,” *Human-computer interaction series*, vol. 6, 2004.

[27] P. Cremonesi, Y. Koren, and R. Turrin, “Performance of recommender algorithms on top-n recommendation tasks,” in *Proceedings of the fourth ACM conference on Recommender systems - RecSys ’10*, 2010, p. 39.

[28] M.-W. Kim, E.-J. Kim, W.-M. Song, S.-Y. Song, and A. R. Khil, “Efficient recommendation for smart TV contents,” in *Big Data Analytics*, Springer, 2012, pp. 158–167.

[29] Y. Blanco-Fernández, J. Pazos-Arias, A. Gil-Solla, M. Ramos-Cabrer, B. Barragáns-Martínez, M. López-Nores, J. García-Duque, A. Fernández-Vilas, and R. Díaz-Redondo, “AVATAR: An Advanced Multi-agent Recommender System of Personalized TV Contents by Semantic Reasoning,” in *Web Information Systems – WISE 2004* , 2004, pp. 415–421.

[30] C. Shin and W. Woo, “Socially aware TV program recommender for multiple viewers,” *IEEE Transactions on Consumer Electronics*, vol. 55, pp. 927–932, 2009.

[31] B. Schopman, D. Brickly, L. Aroyo, C. Van Aart, V. Buser, R. Siebes, L. Nixon, L. Miller, V. Malaise, M. Minno, and Et Al., “NoTube: making the Web part of personalised TV,” *Scenario*, pp. 1–8, 2010.

[32] G. Adomavicius and A. Tuzhilin, “Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 17. pp. 734–749, 2005.

[33] “Information Retrieval - Wikipedia, the free encyclopedia.” [Online]. Available: http://en.wikipedia.org/wiki/Information_retrieval. [Accessed: 01-Dec-2014].

[34] E. Kim, S. Pyo, E. Park, and M. Kim, “An automatic recommendation scheme of TV program contents for (IP)TV personalization,” *IEEE Transactions on Broadcasting*, vol. 57, pp. 674–684, 2011.

[35] N. Chang, M. Irvan, and T. Terano, “A TV program recommender framework,” in *Procedia Computer Science*, 2013, vol. 22, pp. 561–570.

[36] R. Sotelo, Y. Blanco-Fernández, M. López-Nores, A. Gil-Solla, and J. J. Pazos-Arias, “TV program recommendation for groups based on muldimensional TV-anytime classifications,” *IEEE Transactions on Consumer Electronics*, vol. 55, pp. 248–256, 2009.

[37] “Artificial Neural Network - Wikipedia, the free encyclopedia.” [Online]. Available: http://en.wikipedia.org/wiki/Artificial_neural_network. [Accessed: 01-Dec-2014].

[38] L. Boratto, S. Carta, and M. Satta, “Groups identification and individual recommendations in group recommendation algorithms,” in *CEUR Workshop Proceedings*, 2010, vol. 676, pp. 27–34.

[39] “EPG - Wikipedia, the free encyclopedia.” [Online]. Available: http://en.wikipedia.org/wiki/Electronic_program_guide. [Accessed: 01-Dec-2014].

[40] “DVB - Wikipedia, the free encyclopedia.” [Online]. Available: http://en.wikipedia.org/wiki/DVB. [Accessed: 01-Dec-2014].

[41] “Web Crawling - Wikipedia, the free encyclopedia.” [Online]. Available: http://en.wikipedia.org/wiki/Web_crawler. [Accessed: 01-Dec-2014].

[42] “Web Scraping - Wikipedia, the free encyclopedia.” [Online]. Available: http://en.wikipedia.org/wiki/Web_scraping. [Accessed: 01-Dec-2014].

[43] J. T. J. Penttinen, P. Jolma, E. Aaltonen, and J. Väre, *The DVB-H Handbook: The Functioning and Planning of Mobile TV*. John Wiley & Sons, 2009.

[44] “Ruby Language - Wikipedia, the free encyclopedia.” [Online]. Available: http://en.wikipedia.org/wiki/Ruby_(programming_language. [Accessed: 01-Dec-2014].

[45] “Ruby On Rails - Wikipedia, the free encyclopedia.” [Online]. Available: http://en.wikipedia.org/wiki/Ruby_on_Rails. [Accessed: 01-Dec-2014].

[46] “NGINX - Wikipedia, the free encyclopedia.” [Online]. Available: http://wiki.nginx.org/Main. [Accessed: 01-Dec-2014].

[47] E. F. Codd, “Relational database: a practical foundation for productivity,” *Communications of the ACM,* vol. 25. pp. 109–117, 1982.

[48] Deloitte, “TV Media Report in Turkey,” 2014. [Online]. Available: http://www.deloitte.com. [Accessed: 01-Dec-2014].

[49] "Day Parting for TV - Wikipedia, the free encyclopedia." [Online]. Available: http://en.wikipedia.org/wiki/Dayparting. [Accessed: 01-Dec-2014].

[50] Google, "Stop Words for turkish Language." [Online]. Available: https://code.google.com/p/stop-words/. [Accessed: 01-Dec-2014].

[51] A. A. Akın and M. D. Akın, "Zemberek, an open source nlp framework for Turkic languages," *Structure*, 2007.

[52] P. P. Talukdar, T. Brants, M. Liberman, and F. Pereira, "A context pattern induction method for named entity extraction," *Proceedings of the Tenth Conference on Computational Natural Language Learning CoNLLX 06*, p. 141, 2006.

[53] P. N. Mendes, M. Jakob, A. García-silva, and C. Bizer, "DBpedia Spotlight : Shedding Light on the Web of Documents," in *Proceedings of the 7th International Conference on Semantic Systems (I-Semantics).*, 2011, vol. 95, pp. 1–8.

[54] Y. E. Işıklar, "A TV Content Augmentation System Exploiting Rule Based Named Entity Recognition Method," Middle East Technical University, 2014.

[55] M. Krötzsch, D. Vrandečić, M. Völkel, H. Haller, and R. Studer, "Semantic Wikipedia," *Web Semantics*, vol. 5, pp. 251–261, 2007.

[56] F. Holzschuher and R. Peinl, "Performance of graph query languages," in *Proceedings of the Joint EDBT/ICDT 2013 Workshops on - EDBT '13*, 2013, p. 195.

[57] T. Hoff, "Neo4j - A Graph Database That Kicks Buttox," 2009. [Online]. Available: http://highscalability.com/neo4j-graph-database-kicks-buttox. [Accessed: 01-Dec-2014].

[58] D.-E. Society, "DB-Engines Ranking of Graph DBMS," 2014. [Online]. Available: http://db-engines.com/en/ranking/graph+dbms. [Accessed: 01-Dec-2014].

[59] F. Holzschuher and R. Peinl, "Performance of graph query languages: comparison of Cypher, Gremlin and native access in Neo4j," in *16th International Conference on Extending Database Technology, EDBT' 13*, 2013, pp. 195–204.

[60] J. R. Anderson, "A spreading activation theory of memory," *Journal of Verbal Learning and Verbal Behavior*, vol. 22. pp. 261–295, 1983.

[61] A. Pannese and J. Hirsch, "Self-specific priming effect," *Consciousness and Cognition*, vol. 19, pp. 962–968, 2010.

[62] P. R. Cohen and R. Kjeldsen, “Information retrieval by constrained spreading activation in semantic networks,” *Information Processing & Management*, vol. 23. pp. 255–268, 1987.

[63] L. Ferrand and B. New, “Semantic and associative priming in the mental lexicon.,” in *Mental lexicon: Some words to talk about …*, 2003, pp. 25–43.

[64] T. Fushiki, “Estimation of prediction error by using K-fold cross-validation,” *Statistics and Computing*, vol. 21, pp. 137–146, 2009.

[65] S. Marangoz, “DEVELOPMENT OF A SYSTEM FOR SMART TV VIEWERS USING DATA MINING TECHNIQUES,” Dokuz Eylül University, 2014.

[66] J. Gu, B. Wang, F. Zhang, W. Wang, and M. Gao, “An improved apriori algorithm,” in *Communications in Computer and Information Science*, 2011, vol. 224 CCIS, pp. 127–133.

[67] J. Beel, S. Langer, M. Genzmehr, B. Gipp, C. Breitinger, and A. Nürnberger, “Research Paper Recommender System Evaluation: A Quantitative Literature Survey,” *RepSys*, 2013.