Recommendation System for Next Generation of Smart TV

MoE Alexandra Posoldova

BA, MA, MPhil

Griffith University, School of Information and Communication Technology

Submitted in fulfilment of the requirements of the degree of Doctor of Philosophy

Submission date: 18.06.2017

Abstract

This research thesis is dedicated to the design of recommendation system for television. Nearly every household has at least one TV and pays a broadcast provider for a number of channels.. Orientation in program offer can be challenging and having a real time overview of hundreds of channels is impossible. This means the money paid for service is ineffective if not wasted completely. Recommendation systems are designed to save time and effort when searching for a suitable content. The system learns user preferences from past observations and suggests a content fitting these preferences. There are 5 basic recommendation techniques, each using different kind of knowledge for their prediction and their hybrid combinations. In this thesis, two most commonly implemented approaches are described in detail, namely, content and collaborative filtering. The main focus of this work is on handling categorical data as electronic program guide can provide a rich description of a program. State-of-the-art methods associated with this two recommendation approaches are described and some of them are extended to improve their performance. For my design, I choose to apply a probabilistic approach allowing comprehensive manipulation of the categorical data as well as providing insight into feature relationships of the content description. Graphical models meet all the requirements and because of this reason they become the primary approach the design on is built on. A novel approach based on transfer learning is applied to the graphical network in this thesis. This approach is able to benefit from user group information, therefore overcoming the issue of insufficient user data. The proposed recommendation system is applied to a television environment involving the emerging technique of hybrid broadband and broadcast (HBB) transmission of TV content. HBB is a standardized platform combining and harmonizing streams from broadband and broadcast sources allowing a simple implementation of entertainment services to enhance the user experience. Recommendation engine is one of the interactive services in this framework allowing the user to have an overview of favourite programs. The recommendation is made based on the estimation of a rating prediction. The item with the highest predicted rating is then recommended. This makes an accurate rating prediction crucial for the performance evaluation of the model. Because of this, beside the commonly used mean absolute error (MAE), a new metric to measure the performance of a recommendation engine is proposed which focuses on the importance of rating prediction. Experiments are performed on real world data set provided by Yahoo Labs. It is a collection of movies with their description categorized in a number of features and user ratings. The item description is often incomplete with many feature values missing. This is

common for many data sets. Another typical issue encountered by this data set is the sparseness of the user-item matrix and item-feature matrix. It is beneficial if the recommendation system is designed in a way that these issues are either minimized or the model is robust enough that the system design is not affected by them. The model proposed in this work incorporates a method for missing value estimation and does not suffer from the sparsity issue.

Statement of Originality

This work has not previously been submitted for a degree or diploma in any university. To the best of my knowledge and belief, the thesis contains no material previously published or written by another person except where due reference is made in the thesis itself.

(Signed)_________[redacted]______

Alexandra Posoldova

Table of Contents

List of figures

List of tables

List of abbreviations

AIT	Application Information Table
AODE	Averaged one-dependence
A/V	Audio/Visual
CB	Collaborative filtering
CF	Content-based filtering
DAG	Directed Acyclic Graph
DSM-CC	Digital Storage Media – Command and Control
DVB	Digital Video Broadcasting
EIT	Event Information Table
EM	Expectation maximization
EPG	Electronic Program Guide
HBB	Hybrid broadband and broadcast
IPTV	Internet Protocol television
KL	Kullback-Leiber
kNN	K-Nearest Neighbour
MAE	Mean Absolute Error
MAP	maximum a posteriori probability
OWL	Web Ontology Language
RDF	Resource Description Framework
SKOS	Simple Knowledge Organization System
SVD	Singular Value Decomposition
SVM	Support Vector Machine

WRE	Weighted Recommendation Error
XML	Extensible Markup Language

Acknowledgement

Foremost, I would like to express my sincere gratitude to my supervisor Associated Professor Alan Liew for the continuous support of my Ph.D study and research, for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis.

A very special gratitude goes out to Griffith University for providing financial support covering tuition fees as well as living allowance. This way I was able to focus on research taking away financial worries. I am also grateful for travel funds provided by School of Information and Communication Technologies and GAP to attend international conferences.

I am also thankful for English help, student services and library services provided by Griffith University.

I would further like to thank to my external supervisor Professor Gregor Rozinaj for his insightful comments and encouragement as well as for the opportunity to apply my research findings into international projects HBB NEXT and NEWTON and so being part of research group based at Slovak University of Technology.

My gratitude also belongs to Beyond Success Academy for helping me in my professional development and though understanding of emotional intelligence to be able to work more effectively and knowingly.

My special thanks goes to my parents, grandparents, partner and friends, who supported me mentally throughout my studies.

Chapter 1. Introduction

Knowledge and information has grown exponentially in recent time. What we accumulated until the end of World War II has doubled in the next 25 years. Nowadays it doubles every 13 months and it is predicted that this rate will increase to every 12 hours with the emerging internet of things technology [Schilling 2013]. The explosive growth of information available overwhelms users. The vast amount of choices leads to poor decisions. It is understood that while choice is good, more choice is not always better [Ricci 2011].

Recommendation system or engine is a tool for prioritizing and sorting the vast amount of information available. It assists a user to choose an item, which would fit his/her needs and preferences. Item in this case is a general term and can represent music, book, movie or other content. Recommenders have a wide spectrum of applications in e-shops, movie databases, online radio, e-commerce, e-learning and many others. Popular companies like Facebook, Amazon, Pandora and Netflix employed recommendation systems in their services. In this work, the focus is on application to the hybrid broadband and broadcast (HBB) TV environment. The HBB platform specifies transmission from different data sources in order to unify and harmonize them. This technology aims to enhance the user watching experience via providing a spectrum of entertainment services through connected TVs, set-top boxes and multiscreen devices. Recommendation system is one of the services a customer can highly benefit from. There are only a few experimental recommendation systems implemented in television as I will show in the overview of the current situation. There are publications either considering the technical aspect of the recommendation system or the application aspect of it into the HBB environment. To my knowledge, there is no research considering both. This is the gap I addressed in this thesis and that is why the implementation of recommendation system into this newly emerging technology was chosen.

The important part of a recommendation is to find out what the user preference is and what aspect affects his/her choice. There are five basic recommendation techniques and their hybrid combinations. We often rely on a recommendation of our friends or colleagues when buying a product or hiring a person. The first type of recommendation system approximates this behaviour. Many e-commerce web sites started to implement this service helping customers to choose from a wide range of available products. This is called collaborative filtering (CF) and over the past years it has got a lot of attention. The technique uses past feedback of users to find and group users with similar taste. We may be familiar with the sentence: “People who like/buy this also buy…”. This method however, cannot make recommendation for a new user or a new item, because there is no feedback information to base the recommendation on. Because of this, other ways of recommendation were developed. The second most popular is content based filtering (CB) that uses a content description of an item to find a similar item to the one user liked or bought in the past. Rating or feedback from other users is not needed. Therefore it can also recommend a new item nobody has ever rated.

These two techniques are the ones this thesis pays the most attention to. There are, however, other three that are also described briefly. Among them, community based method has grown in popularity because of the social networks. It is assumed that people within a community share similar interests. This recommendation is based on preferences of our friends. On top of these five basic techniques, hybrid combinations are often implemented to minimize drawbacks of each other. Especially, collaborative and content based filtering methods are commonly employed together as one brings novelty and the other deals with the new item issue.

Great focus is paid to approaches that can deal with categorical data. Considering the application into TV environment, categorical data extracted from electronic program guide is the primary and a rich source of information about a program. These methods did not get much attention in the past and that is another research gap addressed in the model design in this thesis. Graphical models proved to be the best choice as they provide insight into user preferences and how item features affect user rating pattern. Graphical based techniques are described in this work together with their enhancements such as smoothing technique and algorithm for missing values estimation.

The thesis is organized as follow. First, an overview of related work that has been done in this area is given. Then, research gaps are identified and the questions addressed in this thesis are drawn. The next chapter continues with recommendation system description and its application in the hybrid broadband and broadcast television. The thesis then elaborates on recommendation systems from a methodology point of view and describes basic approaches. The next chapter is dedicated to content based filtering methods. I pay close attention to graphical models, namely Naïve Bayes classifier, Averaged One-Dependence estimator, and PC algorithm. In the following chapter, the principles of transfer learning and application of this novel approach to Naïve Bayes classifier is explained. In this approach, user data is enriched by information from other users with similar taste and knowledge is transferred from a group of users into individual members of the group. Other popular techniques are then briefly described and compared with the proposed approach. The rest of the thesis is dedicated to experiments, where I look at an algorithm’s performance from different perspectives. A real world data set provided by Yahoo Labs is used for the experiments. A new way of precision measure is proposed based on weighted recommendation error, which is designed for recommendation systems based on rating prediction. Finally, a summary of the research is given and conclusions are drawn.

Chapter 2. Related Work on TV Recommendation Systems

A collaborative filtering based program recommendation system designed for IPTV that groups users according to genre preference was proposed by [Kim 2011]. The advantage of this work is the implicit feedback of users without the need to ask a viewer to rate the content. As was pointed out in this work, only 15% of users provide explicit feedback, causing a rating sparsity issue. It was also noted that unlike recommenders designed for e-commerce, TV viewers do not seek as much novelty. The reason is that while in e-commerce, people are unlikely to buy the same product over and over again, when it comes to watching television, people tend to watch TV content they have been accustomed to. This supports my assumption that the content based filtering method is more suitable for this environment.

In Japan, an experiment was conducted on 5 students tracking their watching history and the TV program and description they downloaded from Internet Electronic Program Guide [Xu 2006]. Polynomial kernel Support Vector Machine (SVM) was adopted in their work. The authors claimed that the model demonstrated good dynamically adaptive capability. The paper, however, lacks comparison with state of the art methods.

Another paper using collaborative filtering method to design recommendation system for TV uses automatic context tagging to solve the issue of new user and item, which is typical for CF methods [Lee 2014]. They compare the proposed method with only one algorithm, which seems to be based on Slope One.

There are very few designs using content information to recommend TV content. [Uluyagmur 2012] tried to estimate user rating based on features such as genre, actor, director and others. They offered 30 movies to users and tried to measure the precision of recommendation. The performance was evaluated as the ratio of the number of movies the user decided to watch to all recommended movies. Another example is a recommender designed for Japan video service [Ikawa 2010]. The recommendation is based on the ratio between movie features with high frequency of occurrence in user watched history to frequency of these features in all the movies. Both systems use ranking of programs and those with the highest rank are recommended.

There have been a couple of designs considering also context information like [Dobrowsky 2013]. This work proposes a design of context aware recommendation system for IPTV-

Internet Protocol Television services from an implementation point of view. As context, they considered information describing user, such as gender, age, agenda, usage history and activities, device/terminal specification, network and service description like content description, language, channel, actor, director, studio, release year, and others. Also [Song 2012] considered context information in IPTV service to personalize user watching experience while also providing the benefit for service and network provider.

A few hybrid combinations of CB and CF method have been proposed for TV recommenders and movie databases. TV program predictor was designed for HBB TV combining a number of techniques [Krauss 2013], such as cosine similarity as a content based component to find similar programs, Pearson Correlation and Slope One as a CF component to highlight favourite programs of similar users, enriched with a clustering method for increasing performance, association rules for analysing item relations, and SVM to identify patterns in user behaviour. Cosine similarity as a CB component was also used by [Barragans-Martinez 2010] to create the item-item similarity matrix. Then Singular Value Decomposition (SVD) was applied to this matrix to reduce the number of dimensions leading to more co-rated dimensions.

Several hybrid designs have been proposed for movie recommendation. Content-boosted collaborative filtering was used by [Melville 2002] to fill the user rating matrix with missing ratings. They used the harmonic mean weighting and self-weighting to fill missing ratings. Pearson Correlation was then applied for rating prediction and recommendation. In [Campos 2010], graphical model was used in combination with canonical weighted sum to estimate the relationship between items and users combining content and collaborative filtering.

Because watching TV is often a social activity, a couple of papers look at group recommendations. In [Amolochitis 2014], a recommendation system for video-on-demand is designed considering multiple users using one account. The design combines content and collaborative filtering into a hybrid model that is able to respond fast to user requirements. A content based group recommender proposed in [Pera 2013] uses tags for capturing the contents of movies considered for recommendation and group members' interests. Software implementation of a recommendation system for digital television is outlined in [Sotelo 2012], where the system stores a user profile but the audience is modelled as a group of viewers.

Television and commercials are closely interrelated. Because of that, some predictors, instead of recommending TV program, estimate what percentage of TV households is tuned to a specific TV station [Pagano 2015]. This yields a more effective spending on TV advertisement saving thousands of dollars.

A number of researches deal with software implementation of recommendation system. In [Zhang 2005], the system uses the TV-Anytime metadata to extract program description in the form of XML. This was designed for digital television. In the HBB TV environment, recommendation system will work as an application users can download. Implementation of a Java based application on digital TV devices was elaborated in [Kuzmanovic 2012]. Necessary modifications to the underlying operating system as well as JavaScript plug-ins and modules related to digital TV are described in order to have the HBB TV features enabled. While [Zhang 2005] and [Kuzmanovic 2012] dealt with the software implementation only, [Smyth 2001] proposed an architecture as well as a collaborative filtering method to design personalized electronic program guide.

2.1 Research questions

The main issue this research addresses is the information overload a user is facing while watching TV. Based on a literature survey into recommendation system, I have identified some research questions that I would like to address. Previous research about recommendation system in TV environment approaches it either from an implementation point of view or from a methodology point of view. In this research, I design a recommendation system considering its application in TV environment and outline its implementation according to the HBB TV standard.

The primary accuracy measure of rating based recommendation systems is mean absolute error (MAE). My research founds this measure highly insufficient. Therefore a new measure called the weighted recommendation error is proposed. Because items with a high predicted rating are recommended, this measure penalizes inaccurate predictions based on the actual user rating.

Another common issue in any data analysis is missing information. To my knowledge, no recommendation system in TV environment deals with this. Moreover, commonly used similarity based methods cannot be applied in TV environment due to the time delay as it will be explained later. An estimation of missing inputs using empirical distribution is proposed. This approach is simple and fast and is suitable for TV application and uses only user training data.

Last but not least is the size of user training set. Although Naïve Bayes classifier is known to perform well also on small data sets compared to other methods like logistic regression, in my data set users often have 8 or less samples in their training set. It is difficult to draw any assumption about user preferences from such a small set. Because of that, a novel approach is proposed. The approach groups users according to their similarities and then using transfer learning, prediction for individual users using the group as an extension of user training set is made. This is a novel approach which combines principles of collaborative filtering with the content information. This approach is not affected by the cold start problem, which is the main weakness of collaborative filtering approaches as will be explained later.

Chapter 3. Recommendation system as HBB TV application

Hybrid broadband and broadcast television is a new platform combining two types of transmission. This new way of watching TV brings new challenges as well as opportunities for service development [ETSI TS 2012]. It allows users to have personal profiles and to adjust TV settings to their convenience. A user can be identified by a camera [Jirka 2014] or a microphone [Kacur 2014]. Furthermore, a camera can be used to perform gesture recognition and personal gestures can also be created [Vanco 2013]. Therefore, HBB television offers a truly personalized approach. The recommendation system is another personalized service enhancing user experience in HBB TV.

,,The HbbTV specification was developed by industry leaders to effectively manage the rapidly increasing amount of available content targeted at today's end consumer. It is based on elements of existing standards and web technologies including OIPF (Open IPTV Forum), CEA-2014 (CE-HTML), W3C (HTML etc.) and DVB Application Signalling Specification (ETSI TS 102 809) and DASH.'' [HbbTV 2016]

Figure 1 is a diagram describing connection of a hybrid TV terminal or setup box, plus a companion device, which can be a smart phone, tablet or any other device supporting HBB transmission, and broadband and broadcast service providers. This figure is from HbbTV 2.0 Specification [HbbTV 2015].

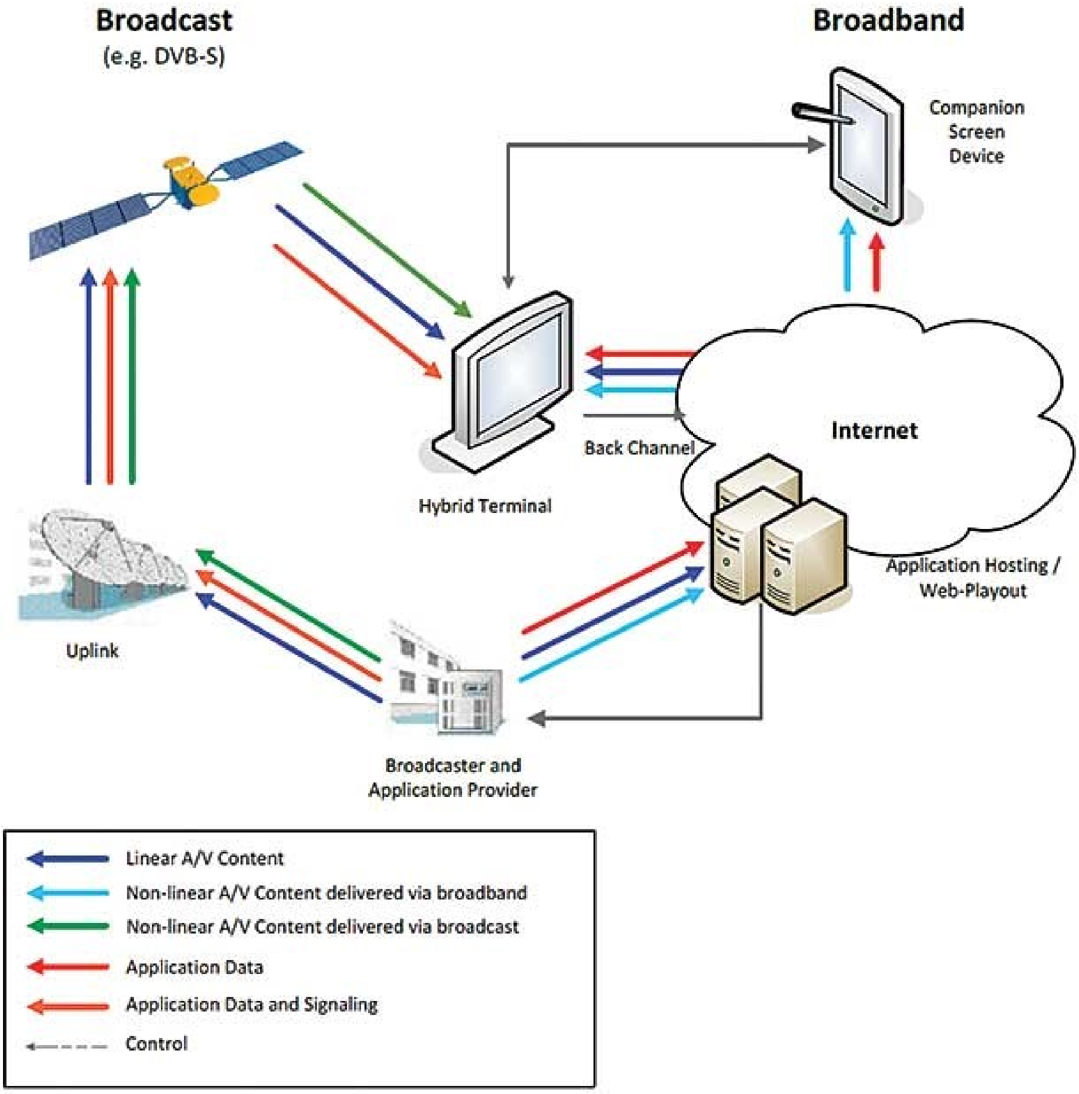
Broadcast
(e.g. DVB-S)
Broadband
Companion Screen Device
Hybrid Terminal
Back Channel
Internet
Application Hosting / Web-Playout
Uplink
Broadcaster and Application Provider
Linear A/V Content
Non-linear A/V Content delivered via broadband
Non-linear A/V Content delivered via broadcast
Application Data
Application Data and Signaling
Control

Fig. 1 Diagram connecting a TV terminal, companion device and connection to the broadband and broadcast service [ETSI TS 2012].

The recommendation engine can be described by the block diagram shown in Figure 2. It comprises 5 basic parts. Up to date user preferences are stored in the user profile. As user taste can change over time, the preferences are updated in the profile learner according to user feedback on recommendation done for the user. How to deal with the preference change can be found in [Xu 2006]. User feedback is very important for learning user preference, therefore this is further elaborated later on. Because the system is applied into the HBB domain combining two data streams, a data collection and unification process is needed. This process is described in the following section. Once data is processed and unified, it is fed to the recommendation block to make a suggestion, where information from the user profile is considered. The majority of this research is devoted to the recommendation engine design. Therefore the next chapter is dedicated to this topic.

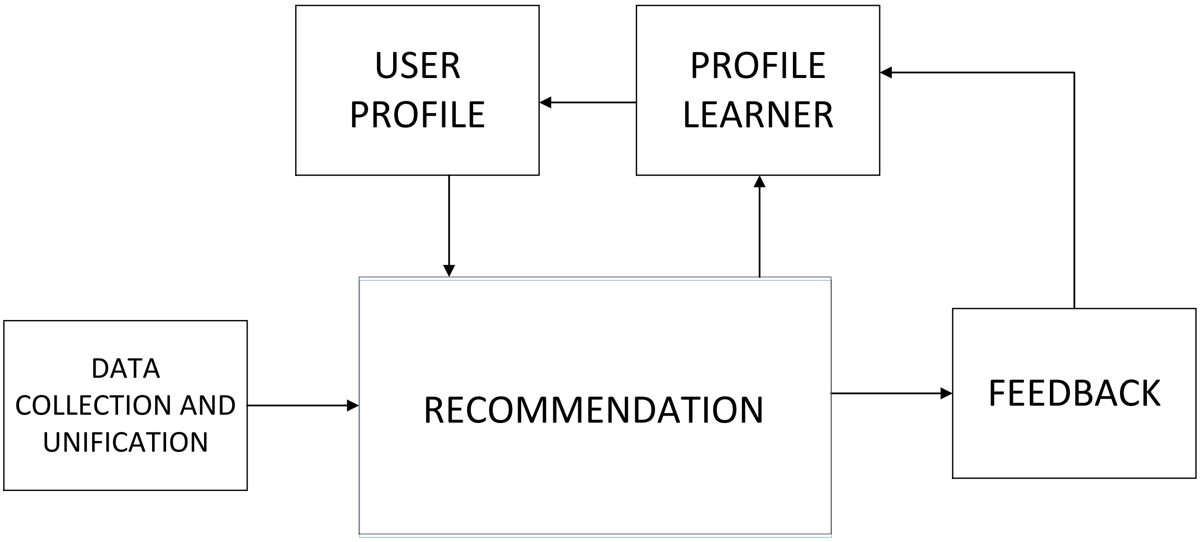

Fig. 2 Recommendation engine as block diagram designed for HBB television.

3.1 Data collection and unification

In this section, the process of data collection and unification for hybrid broadband and broadcast TV is outlined. In order to have a quick and easy access to information, the data needs to be pre-processed and unified. The data format suitable to work with is XML. It is the preferred data format due to its simplicity, wide application and easy readability by computers as well as humans. It has been applied to several application program interfaces and some Electronic Program Guides (EPGs) allowing downloading a program in the XML TV format. Data can be further modified and translated to other file format to make the access more effective [Bellekens 2011]. Easy and fast access is especially important in TV application as the recommendation needs to be done in real time. Block diagram in Figure 3 describes the process of data collection from four basic data sources, namely Internet, network, device and EPG. Data can be further divided into two logical categories. Data giving information about a program, such as actor, title, running time, genre and others, and data giving information about a context, such as transmission conditions, location, device specification, day in week, weather and others.

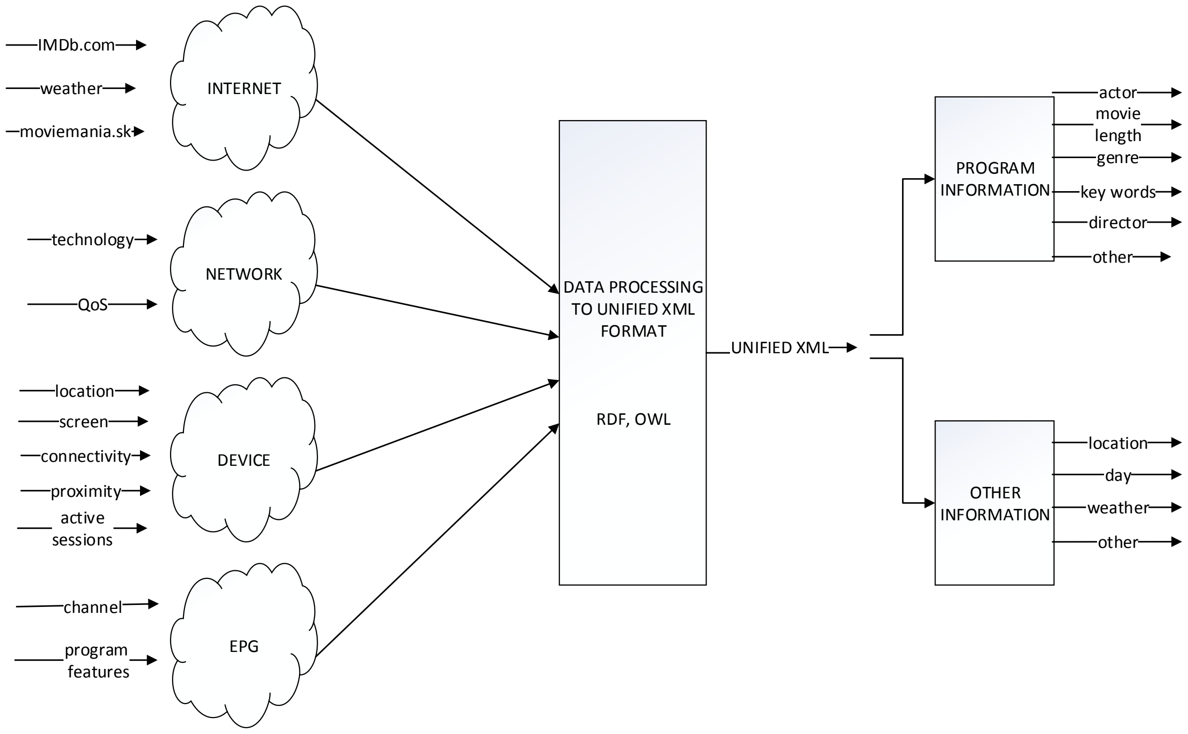

Fig. 3 Block diagram describing data collection, pre-processing and unification.

Network and device information can be gathered by some physical or logical sensor located in a user device. For more information about how data is collected and what protocols are used I refer to the document [ETSI TS 2008].

Internet is another source of information about program as well as context. Several online movie data bases provide complex information about content, which can be complementary to data collected form EPG. Moreover, part of the recommendation system can be based on collaborative filtering. If a user has a profile in one of the online movie data bases, we can find other users with similar tastes so as to enrich the recommendation to provide novel suggestions.

3.1.1 Data unification

Information collection for recommendation purposes from different data sources was described earlier. As multiple data streams are considered, every download can have different ways of tagging and ordering. Moreover, it often contains a lot of duplicates. These make the access slow and less effective. To prevent this, and make access easier, it is better to unify the data [Lovinger 2007] using RDF or OWL method:

- Resource Description Framework (RDF) – This approach is part of W3C specifications designed to describe information from different data sources. It uses three tags composition to describe data such as: <Subject>, <Predicate> <Object>. Then, movie features are assigned to those tags.

- Web Ontology Language (OWL) – Similar to RDF, although, it has more tags can be used to describe features, this makes the description more complex.

To avoid duplicates in files, Simple knowledge organization system (SKOS) [W3C 2012] can be used to identify and remove similar expressions.

3.2 Feedback

User feedback is used to evaluate recommendation. If the feedback is strongly positive, then the recommendation of this program was well predicted and will strengthen an existing believe about user preferences. In general, the recommendation with a positive feedback reflects that the system is well trained and knows user taste. On the other hand, strongly negative feedback will reflect a bad decision and the system has to reconsider feature suggestions, retrain the model in the profile learner block from Figure 1 and update information in the user profile. Of course, it does not depend only on one positive or negative feedback. But consistent positive or negative feedback gives a clear picture of how the system is trained and what adjustments needs to be done. There are two main approaches of feedback, implicit and explicit.

Explicit feedback can be expressed as [Lops 2011]:

- Like/dislike: It is the simplest explicit feedback. A simple binary rating scale is used to distinguish between relevant and not relevant content.
- Rating: Compared to like/dislike approach, this one offers a wider scale. For instance, it can be a scale from 1 to 5 to label the level of relevance.
- Text comments: User can leave a short comment about what he/she liked or did not like about the recommended program. For instance, the genre of movie was of user preference, but the actor was not. Other two approaches just returns good or bad feedback and it will take longer to recognize which feature affected the recommendation in a negative way. This approach offers the most complex feedback boosting the profile learning process. However, the techniques of text classification have to be implemented.

Unlike explicit feedback, implicit feedback does not require user to be involved, which might be more convenient for viewer, but bias is more likely to occur. Feedback can be estimated by:

- Actions: For instance, switching channel or turning off TV is considered as an action resulting in negative feedback. This approach was used in [Uluyagmur 2012]. Viewer can however switch a channel just because of break, not because he/she does not like the channel. This type of actions has to be distinguished from others negative ones. This approach is generally used for rating web sites.
- Emotions: This is a more complex way to evaluate the relevance of recommended program. When a user profile is created, user is asked to watch a short movie, which combines all types of genres. The camera captures his/her emotions and these will be used as the implicit feedback in future. In other words, if the recommended program is supposed to be scary, this type of emotion should be recognized in order to assign a positive feedback. Systems for human emotions detection were proposed in [Kudiri 2012] and [Luoh 2010].

3.3 Recommendation system as application in HBB TV environment

A TV terminal allowing hybrid connection has the capability to communicate with two networks in parallel, broadcast and broadband. The broadcast network transfers standard audio/video content, application data and signalling information. If the TV terminal is not connected to broadband network, it can still receive broadcast-related content via broadcast transmission. The connection to broadband transmission allows the terminal to receive internet content via bi-directional communication with the application provider. This connection allows the terminal to receive the application related data and audio/video on demand content. Non-real time download of this content can also be supported. It is through this connection the companion devices can be connected to TV terminals through the same local network [HbbTV Association 2016].

The recommendation engine belongs to the broadcast-related application group as it is associated with broadcast services and events. The main reason for choosing this platform is due to its easy implementation and friendly environment towards applications. In the past, TV set came with a pre-defined set of applications and it was mainly a one-way medium. These applications needed constant maintenance by vendors if they wanted to keep them up to date. Moreover, these could change from one system to another [Kuzmanovic 2012]. Connecting set-top boxes to the internet opened up possibilities for more services within standard digital television. HBB TV standardizes and defines this hybrid transmission combining broadcast and broadband network in order to enhance user experience.

3.3.1 Accessing Recommendation System

Recommendation system works as an application installed in a hybrid terminal. It is preferred that the application starts automatically as the terminal is turned on, unless this is changed by user. According to the HBB standard [ETSI TS 2012], the “red button” icon indicates that the application is available to the user. Considering a standard remote control, when the red button is selected, the recommendation system application displays the full user interface. Unless the user sets it otherwise, the auto start broadcast related application will not display full screen automatically. In general, there are three states of this type of applications:

1. the “Red Button” notification is displayed to notify the user that the application is available,

2. no user interface display, the application is running on background,

3. full user interface display with list of recommended items.

The red button is used to switch between stages. If the remote control is equipped with the EXIT button, this can be used to terminate the application. Channel change event can also be used to start the application or to notify the user about availability of the recommendation service. All these incidents are recorded in the Application Information Table (AIT). Note that the recommendation system can also suggest items transmitted via broadband.

The service can be implemented in the terminal by manufacturer or can be downloaded via broadband and access to data is via broadband or broadcast transmission.

The functional components of a hybrid terminal are illustrated in Figure 4 [[ETSI TS 2012]. The terminal receives AIT and application data that stream events together with linear and non-real-time A/V content via the Broadcast Interface. Application data and stream events are transferred by the Digital Storage Media – Command and Control (DSM-CC) carousel. The Runtime Environment, consisting of the Application Manager and Browser, is the representation of component where the application is presented and executed. The Application Manager is responsible for evaluating the data from AIT and controlling the lifecycle of the application. Browser presents and executes the application.

The terminal is connected to the Internet via the Broadband Interface. This connection is the second source of application data from application providers and on demand A/V content. The Internet Protocol Processing component provides all the functionalities for the terminal to handle Internet data as well as application data for the Runtime Environment. The Runtime Environment controls the Media Player receiving A/V content. It can be implemented into the user interface as an application. The Media Player in collaboration with the Synchronization Manager harmonizes content delivered to the hybrid terminal via both interfaces, i.e. the Broadband and Broadcast Interface [HbbTV Association 2016].

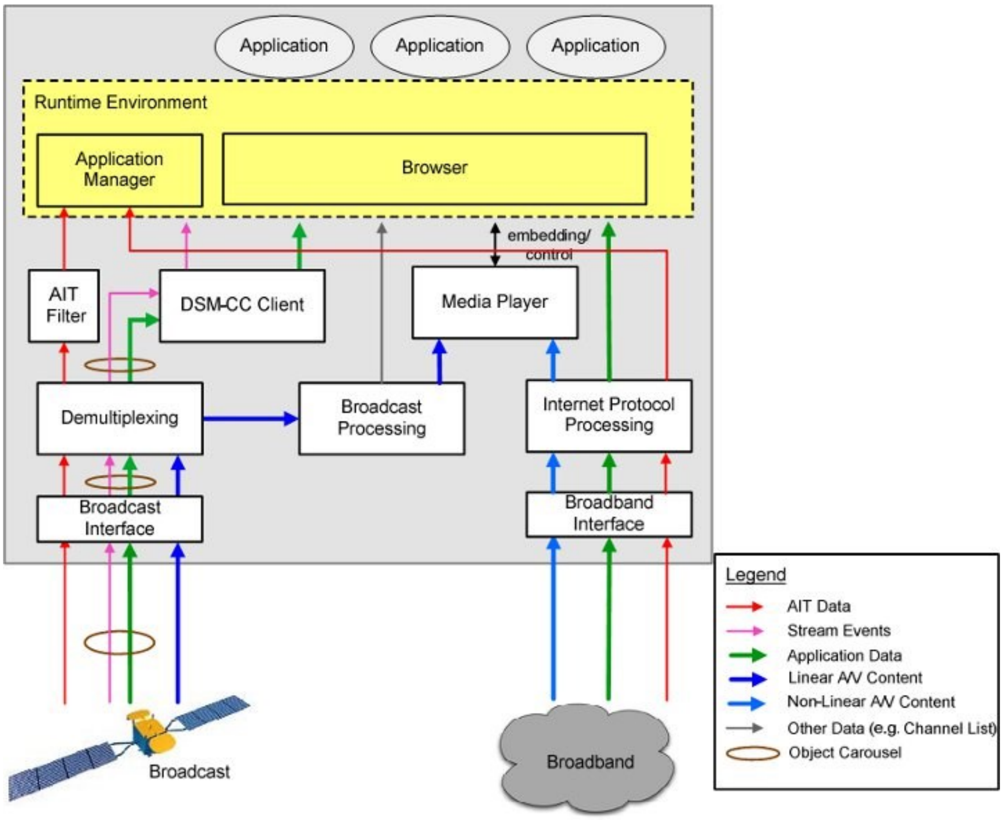

Fig. 4 Functional components of a hybrid terminal [ETSI TS 2010].

The recommendation engine uses a user's watching history to train the model. The signalling is used to capture these past events. Standard TS 102 809 [ETSI TS 2010] defines the application signalling and its transport via broadcast of HTTP.

Chapter 4. Recommendation Systems

An overview of recommender engines is given in [Ricci 2011]. Two most commonly used approaches, content based and collaborative filtering based recommenders are chosen for the purpose of this research. The advantages of both are often combined to create hybrid systems and minimize their drawbacks. The main issue of each system is the so called cold start. It is the initial stage of the system when there is no information about a user or content. The solution for this is different for each type of recommender and is described along with the RE method description.

Making a recommendation with high accuracy is important, as trust between user and system is crucial. The user interface creates a bridge between algorithms used for program recommendation and system communication with viewer and the way results are presented. The aim is to build a positive relationship and trust. This makes user feel like he/she can rely on the suggestions and increases their frequency of using the system. The more a recommender is used, the better it becomes trained and the more accurate the predictions it makes.

Five main recommendation techniques are described in the following sections with the focus on the two most popular implementations, namely content and collaborative filtering. These two methods are the most suitable for television application and most state of the art research is using them.

4.1 Content-based filtering

CB approach finds and recommends items similar to those items that user rated in the past. The user profile is described by his/her preferences drawn from an item's content description. When recommendation is made, the item with the content description closest to the user profile is chosen. For numerical variables, the most commonly used method is support vector machine (SVM) or k-nearest neighbour (kNN) approach, where features are represented in N-dimensional feature space [Ricci 2011]. Although the content description often involves non-numerical data, there are very few CB designs that can handle categorical data. Most of them use text classification methods, like bag of words, to discover user preference [Melville 2002]. Approach based on counting the number of occurrences of a feature and applying some kind of weighting and normalization was used in [Ikawa 2010], [Uluyagmur 2012]. Cosine similarity has also been used to define items correlation using the item content description [Krauss

2013]. Graphical networks, mainly Naïve Bayes based, are used in hybrid systems with strong CF component. In [Campos 2010], graphical model was used in combination with canonical weighted sum aiming to predict user rating. Unlike previously mentioned methods, this method works much faster as there is no need to search through the whole database to find and match terms.

Content based filtering provides an easy solution for the cold start problem when a user is new to the system. User can fill up a short survey about their preferences, so that the system has an initial profile to work with. Another advantage is that it provides a transparent explanation to user about how the recommendation works. It is clear that the next recommendation will be made based on watching history. This increases trust to the system, which is important in order for user to rely on system recommendations. Compared to collaborative filtering based recommenders that are based on ratings collected from unknown people, which would appear like a black box to the user, content based filtering provides a transparent insight.

The main disadvantages of the content based filtering are over-specialization, lack of serendipity and novelty. Content based recommenders are not able to recommend something unexpected or novel because they are trained to suggest a content similar to the one labelled as relevant in the past. In other words, only similar programs will be offered to the user. While for online movie watching this can be a significant drawback, for TV watching this seems to be less of a concern. Ikawa [Ikawa 2010] showed that people are likely to stick to the same TV shows, and the recommendation can be made using channels and time slots they have accessed.

4.2 Collaborative filtering

Collaborative filtering is based on searching similarities between users and group of users with the same preferences. Dataset usually comprises of set of users and items they rated in the past. No content description is used. CF usually build an user-item matrix and finds users who rated the same items as the active user the prediction is made for. This matrix is typically very big and sparse. Some kind of similarity of users is assessed and a rating is estimated or a recommendation made based on users with the rating pattern most similar to that of the active user. Dataset usually comprises of set of users and items they rated in the past. For example, if there are two users who watched the same item and gave it high rating, these two users would be grouped as they share similar interests and would be recommended items that are highly rated and not yet seen. It is sometime called “people-to-people correlation”.

The most popular method for collaborative filtering is kNN because of its simple implementation and accurate predictions. Improved kNN algorithm was used in a well-known winner of Netflix price [Bell 2007]. This approach implements global weighting computed from all the users to better explore similarities between them. Other widely used approaches to measure the user similarity are slope one, Pearson correlation or cosine similarity [Krauss 2013], [Melville 2002].

Another commonly used approach is singular value decomposition (SVD) method, which is found to bring better results than kNN according to [Lops 2011]. Latent feature space is used to describe users and their ratings in order to explore a relationship between users and products.

CF method is well known for its so called cold start problem. Many publications deal with this issue [Bobadilla 2012],[Sedhain 2014]. It appears in two cases:

- **new item**: when the item is new, no user can provide feedback about it and therefore it cannot be recommended.
- **new user:** when the user is new to the system and has not rated any item, there is no information for CF to base its predictions on.

Another shortcoming of CF is that it relies too much on other user's rating and does not work well for people with unique taste. This and the cold start problem can be solved by combining content based and collaborative filtering based approaches. Also, the issue of sparsity can be overcome with the addition of content information as proposed in [Melville 2002]. Another way of dealing with the sparsity issue of user-item matrix is to incorporate some technique of dimensionality reduction as proposed in [Barragas-Martinez 2010]. Dimensionality reduction also minimizes impact of scalability. This issue arises as the number of users and items increases. It leads to unacceptable latency during the recommendation process.

4.3 Demographic

As the name of this technique suggests, a recommendation is made based on demographic information of user. Although many e-commerce service uses demographic information for basic recommendations, like different products are recommended to man and women or it selects product availability based on the country user requests the service from; this method is usually used in combination with other stronger recommendation technique.

4.4 Community Based or Social Recommender Systems

This technique is similar to collaborative filtering, but rather than relying on a recommendation from a stranger, this technique examines how people are connected and recommendations are made according to these connections. This technique become more popular as social networks emerged as they provided simple access to information about social relations of the users. Compared to collaborative filtering, instead of searching for users who rated the same items as the active user, rating of active user's friend is examined and recommendation is made for items with high ratings. This eases the situation of CF when there are not enough users with co-rated items to compute the similarity.

4.5 Hybrid Recommendation Systems

According to [Burke 2007], there are seven basic techniques of combining recommendation approaches. Not all recommendation methods can be combined in all hybridization techniques and some of them are order sensitive. For instance, a hybrid model is different when using feature augmentation on CB-CF to the one of CF-CB combination. The seven basic hybridization techniques are:

- **Weighting**: the recommendation is done for each approach separately and the results are then combined using some kind of linear weighting. This is the simplest hybrid system. Usually empirical means are used to determine the best weights.
- **Mixed**: different components of this hybrid model work side-by-side in a combined list. Instead of combining evidence, two methods are merged based on predicted rating or on recommender confidence.
- **Switching**: this technique selects a single recommender among its constituents based on the situation. Here, the switching criterion is crucial for a successful recommendation. Some researches use confidence values, others use external criteria.
- **Feature combination**: the idea is to inject features of one source into an algorithm designed to process data with a different source. In other words, the recommendation logic from another technique is borrowed.
- **Feature augmentation**: instead of borrowing features from another recommendation method, this technique generates a new feature for each item by using the recommendation logic of the contributing domain. This approach is especially beneficial when there is a well-developed strong primary recommendation component, and a desire to add additional knowledge sources to strengthen the

existing technique. Compared to the feature combination technique, this approach is more flexible.

- **Cascade**: this technique uses secondary recommender only to break ties when predicting rating or recommendation value of the primary one.
- **Meta-level**: model learned by one recommender is used as the input for another recommendation method. It may seem to be similar to the feature augmentation as the contributing recommender provides input to the actual recommender. Here, however, the contributing recommender completely replaces the original knowledge source with a learned model that the actual recommender uses in its computation.

Chapter 5. Content Based Filtering Approach

When content information is available, it can be very useful in rating prediction and it should not be omitted. Modern television provides a short program description. Moreover, on HBB TV platform, a TV set also has internet access and more content description can be found online. Combining these two sources, there is a plenty of information to base the recommendation on. In this chapter, approaches considered in this research for developing the best performing recommendation engine model for HBB TV application are described. The focus is on techniques able to work with categorical data as this information had often been overlooked in the past. All of the selected techniques use graphical model, therefore some time is spent to explain the basics of this modelling. Graphical models are chosen as the main approach. The reasons are:

- Clear interpretability of results – thanks to the graphical structure the model can be read easily. The dependences/independences between features are clear to see and understand.
- Probabilistic representation –inferences between nodes as well as the nodes itself can be expressed in terms of probabilistic representation. As mentioned earlier, most of the features used in the design are categorical and it is better to translate it into probabilities. Therefore this property of graphical modes is very advantageous.
- Insight –thanks to the graphical representation, inferences between variables are clear and easy to be managed or adjusted if needed.

As a baseline comparison, Naïve Bayes classifier is used. Further, an algorithm relaxing the independence of the Naïve Bayes while keeping its benefits is deployed. The third group of graphical approaches is based on building a user specific model. Because the space of possible graphical models increases dramatically with the number of features, attention is payed to constrained based models. The aim is also to enhance performance of graphical models. To achieve this a smoothing technique is implemented along with an algorithm estimating missing values. All of the considered graphical approaches were tested with adding these performance enhancements and are denoted in this thesis as improved versions.

5.1 Graphical models

Graphical models or probabilistic graphical models (PGM) are used to encode probability distribution of a number of random variables. These variables interact with each other and PGM captures the relationship between them. PGMs are combination of statistics and computer science using concepts from different disciplines like probability theory, graph algorithms, machine learning, and more. They have been applied to a many research areas such as image processing, speech recognition, medical diagnosis, natural language processing and others.

There are two branches of PGMs, Bayesian networks and Markov networks. Bayesian networks encode probability distribution of random variables as directed acyclic graph. Graph consists of nodes and edges where each node represents a variable and its associated probability function and an edge indicating a relationship between two variables. If two nodes are connected, there is an existing relationship between variables. If two variables are independent of each other, nodes representing these variables have no edge between them. Markov networks also consist of nodes and edges representing variables and relationship between them, but unlike Bayesian networks the edges are not directed. Causality of variables in Bayesian networks is clear and can be used as a guide to construct the graph structure. These type of networks are also easier to learn and clearly reflects relationships between variables. Markov networks are preferred if circumstances between variables need to be expressed and its application include physics and vision applications, while Bayesian networks are widely used in artificial intelligence and statistics [Koller 2009].

Compared to regression models, graphical models are able to make accurate predictions for a smaller training sample set. Another benefit is the probabilistic representation of a user preference, which allows faster predictions than methods that store the entire user history.

This work focuses on Bayesian networks only as it is a great tool to draw insights about the user preference.

5.1.1 Directed graphs

Each graph structure consists of nodes and edges connecting the nodes. In directed graphs, the edge direction between nodes is clearly given and expresses a parent-children relationship, where link originates in the parent node pa and goes to the children node ch as shown in Figure 5. Conditional probability distribution $p(ch|pa)$ is associated with each node representing the relationship strength between nodes connected via the edge.

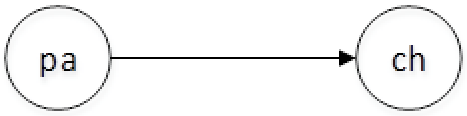

Fig. 5 Parent-children relationship in a directed graph.

Directed graphs are also called Bayesian networks, because Bayes probability theory can be applied to express inferences between feature nodes as well as the overall graphical representation. For a given graphical representation, joint distribution can be derived, which can be expressed as a product over all the nodes included in a graph:

$$p(\boldsymbol{x}) = \prod_{k=1}^{K} p(x_k|pa_k) \tag{1}$$

where K is the number of nodes in the graph and each node is denoted by x_k. Note that a probability representation is given by dependences and independences in a graph. If there is a parent-children pair, this is expressed by the conditional probability $p(x_k|pa_k)$ and nodes without parent are given by their marginal probability $p(x_k)$.

The focus is restricted to graphs with no closed loops called directed acyclic graphs (DAGs) as they are easier to work with [Thulasiraman 1992].

5.1.2 D-separation

It is important to know about independences in a graphical structure as it can affect computation of graph probabilities. Algorithms for building a graphical network are based on these principles. All the independence in every directed graph can by identified by the d-separation criterion if the graphical representation for a given probabilistic distribution forms an independence map (I-map). Directed acyclic graph (DAG) is an I-map of a probabilistic distribution if all variables in this distribution are also included in a DAG. There are three basic d-separation rules as shown in Figure 6.

- **Collider** – this is a unique case when n_1 and n_2 are two independent nodes unless they are conditioned on collider n_3, which reveals some relationship between these two nodes.
- **Fork –** nodes n_1 and n_2 are independent of each other, given node n_3, which is a common cause of nodes n_1 and n_2. This is structure is seen in Naïve Bayes network and is used to simplify computation.

- **Chain –** nodes n_1 and n_3 are independent given node n_2, which is again common cause of nodes n_1 and n_3.

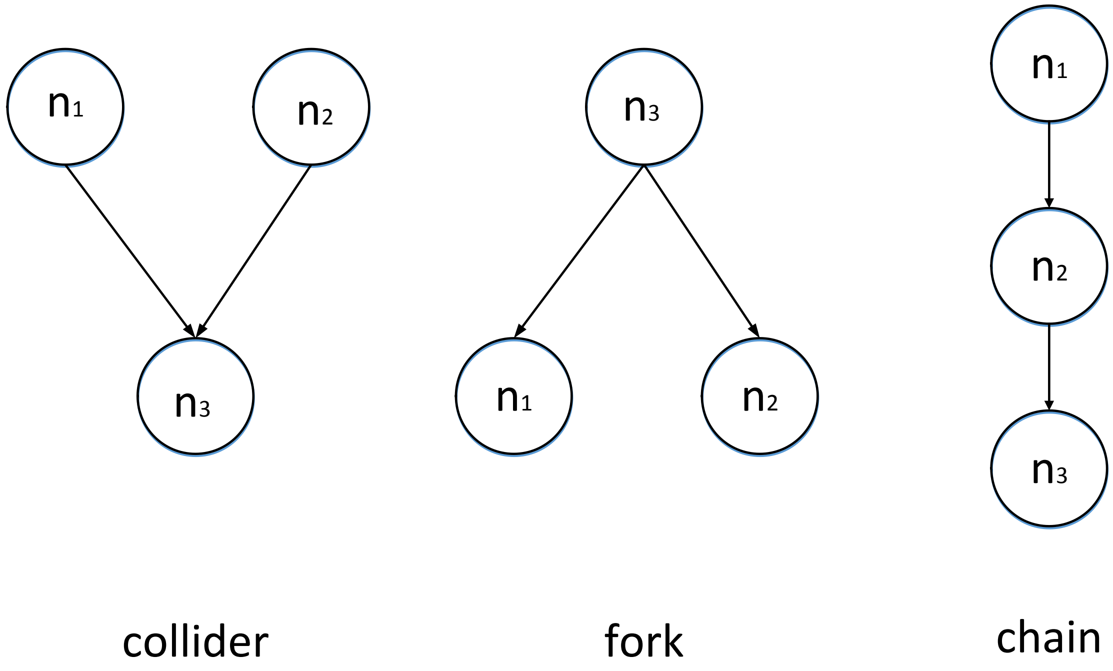

Fig. 6: D-separation criterion used in the Naïve Bayes model.

The first who described this d-separation in graphs was [Pearl 1988].

5.1.3 Smoothing

In the case of sparse feature matrices like actor, there are many values with low occurrence count. Often, a new item description has many new feature values. For unseen feature values, the conditional probability of having a certain rating is zero. When applied to equation (1), it would result in a zero rating probability, and other non-zero elements are disregarded. To overcome the sparsity issue, Laplace smoothing is applied to smooth the occurrence count data.

Laplace or additive smoothing is a technique used on categorical data to add smoothing parameter $\alpha = \{0,1\}$ to a distribution, where $\alpha = 0$ corresponds to no smoothing, to every value of the variable and then normalize it with the number of times the smoothing parameter has been added to.

$$p(r_k) = \frac{\#r_k}{\sum_k r_k} \rightarrow \frac{\#r_k + \alpha}{\sum_k r_k + \alpha * k} \qquad (2)$$

It has been argued that this smoothing parameter should be one, i.e. $\alpha = 1$ [Jurafsky 2008], [Russell 20010], in which case the term add one smoothing is also used. According to Cromwell's rule [Jackman 2007], saying that the event will never occur, thus its prior probability is 0, or the event will always occur, thus its prior probability is 1, is a statement that can never be strictly justified in physical situations and should be avoided.

When Laplace smoothing is applied to the joint probability of feature f_n having F feature values and rating r_k having k rating values, this is computed as:

$$p(f_n, r_k) = \frac{\#(f_n \cap r_k) + \alpha}{(\sum_F f_n + \sum_k r) + \alpha * F * k} \tag{3}$$

Similarly, for conditional probability, the formula is:

$$p(f_n|r_k) = \frac{\#(f_n \cap r_k) + \alpha}{(\sum_k r) + \alpha * k} \tag{4}$$

Different values of smoothing parameter $\alpha = \{0.1, 0.2, \dots, 1\}$ were considered in this thesis and it is been found that the optimal value is $\alpha = 1$.

5.1.3 Creating Feature Tables

Each movie is described by 16 features. Moreover, some features like actor can have multiple feature values. These information are encoded in feature tables. Table 1 below shows an example of movie description, where the left side lists the feature names and the right side provides information about this feature. Note that '\N' stands for missing entry. If a feature has more than one feature values, these are divided by '|'.

Table 1 Content description of move title The Lost Weekend

Feature name	**Description**
running time	330min
MPAA rating	R
release year	2003
distributor	Columbia Pictures
genres	rime/Gangster|Comedy|Action/Adventure|Thriller
directors	Michael Bay
crew types	Editor|Production Designer|Cinematographer|Composer|Producer Screenwriter|Special Effects|Composer Doane

<table>
<tr><td></td><td>Harrison|Hans Dreier|John Seitz|Miklos Rozsa|Charles Brackett|Gordon Jennings|Victor Young</td></tr>
<tr><td>crew members</td><td>Ray Milland|Jane Wyman|Philip Terry|Howard Da Silva|Doris Dowling|Frank Faylen|Mary Young|Anita Bolster</td></tr>
<tr><td>actors</td><td>\N</td></tr>
<tr><td>average critic rating</td><td>\N</td></tr>
<tr><td>number of critic ratings</td><td>5</td></tr>
<tr><td>awards won</td><td>Best Actor |Best Director |Best Screenplay |Best Picture |Best Picture |Grand Prix |Best Male Performance |10 Best Films |Best Actor (in Drama) |Best Director |Best Film |10 Best Films |Best Actor |Best Actor |Best Picture |Best Direction |Best Film |10 Best Films</td></tr>
<tr><td>awards nominated</td><td>Best Cinematography |Best Editing |Best Score |Competing Film |10 Best Films</td></tr>
<tr><td>rating from The Movie Mom</td><td>\N</td></tr>
<tr><td>global non-personalized popularity</td><td>2.45290434</td></tr>
<tr><td>number of users who rated this item</td><td>\N</td></tr>
</table>

While most content based filtering methods work with a few features like genre and actor, the data set in this thesis looks at the item from different aspects. Therefore, the algorithm needs to be able to work with a number of features and their values, which may or may not be present, while keeping the computational complexity low. This set represents the situation when information is downloaded from a content provider. Different feature descriptions are available but not always present.

From the user set, feature tables are created for each of the 16 considered features. The user watching history is then stored in these tables rather than the item and its description. This allows easier access to the information. An example of a feature table for user 99 in the training set for the feature “genre” is shown in Table 2. Every row corresponds to one entry and a “1” indicates that the feature value is present for this entry. The user 99 watched 12 movies and 6 different genres were detected in this user’s set. Every entry was described by at least one feature value from the genre feature.

Table 2 Genre feature table

Crime/Gangster	Kids/Family	Action/Adventure	Suspense/Horror	Comedy	Science Fiction
1	0	0	0	0	0
0	1	0	0	0	0
0	0	1	0	0	0
0	0	0	1	0	0
0	0	1	0	0	0
0	0	0	0	1	0
0	0	0	0	1	0
0	0	1	0	0	0
1	0	0	0	0	0
0	0	0	1	0	0
0	0	0	0	0	1
0	0	0	0	1	0

It can be assumed that the feature values are independent of each other. Therefore when computing a probability of a feature having a particular rating $p(f_n|r_k)$, this is computed as a product of conditional probabilities over all feature values f_{n_v} the feature f_n have:

$$p(f_n|r_k) = \prod_v p(f_{n_v}|r_k) \tag{5}$$

5.1.4 Missing Values

Most real world datasets are incomplete. The dataset used in this thesis has more than 66% of feature values missing. In the context of Bayes networks this becomes an issue as the algorithms for learning the Bayes model assumes complete data [Riggelsen 2006]. The simplest solution is to omit missing data, which would result in large information loss. In my approach, each user set is processed individually for missing entries, which makes the algorithm more attuned to a particular user.

To impute the missing values, first, the occurrence of every feature value within a given feature is counted. For genre feature from the example in Table 2, elements are med along the columns in the table. This provides an empirical distribution of samples. Then the distribution is sampled with a probability given by the empirical distribution function

$$\widehat{F_{n_v}}(t) = \frac{\#f_{n_v}}{m} \tag{6}$$

where $\#f_{n_v}$ is the number of occurrences of the feature value f_{n_v} and m is the number of entries in the user data set. As missing values are imputed by sampling from the distribution specified in equation 6, it is clear that feature values with more occurrences are more likely to be sampled as the missing value. In the case of user 99, action/adventure and comedy genres have the highest count, although in this case, there is no dominating genre.

Some features have more missing entries than others. For instance, information about awards the movie won or was nominated is often not present. In fact in 98% of cases this information is missing. Therefore, it would be difficult to find similar item with non-missing value. If there is no entry in the user set having this feature specified, the feature is disregarded as it can be assumed that the user does not make his/her choice based on this feature.

5.2 Naïve Bayes Classifier

Naïve Bayes is a simple but a powerful technique to construct a classifier. Every item can be described by a set of features $F = \{f_1, f_2, \dots, f_n\}$. Naïve Bayes considers each of these features to contribute independently to the class prediction whereas the class variable is always parent of all the other features nodes. Because the graphical structure of Naïve Bayes classifier is always given, the step of searching for an optimal graphical representation of the data is skipped. This allows a fast implementation of the model and keeps the computational cost low. It is because of these attributes that Naïve Bayes classifier had been widely implemented in many applications and is often used as the baseline comparison.

Naïve Bayes classifier uses Bayes theorem to explain the relationship between the class variable and evidence, which in this case is the set of features describing an item. In this application, class labels are possible item ratings ranging from 1 to 5, $k \in \{1,2,3,4,5\}$. Therefore the probability that an item will have a rating r_k given its features as the evidence can be expressed as the conditional probability $p(r_k | f_1, \dots, f_n)$. The prior probability $p(r_k)$ of a rating value is the best estimate of a rating before considering item features as the evidence. As the evidence that an item with feature F having a rating r_k is added, the posterior probability $p(r_k | f_1, \dots, f_n)$ of the rating value can be obtained.

In the case of television program, the evidence may include features like actor, director, genre, producer and others, where every feature can have a number of values. Suppose there is a history of watched programs with their content description represented by a set of features and their values. Naïve Bayes model represents a simple relationship between features and user rating as shown in Figure 7, where feature nodes are observed from data and independent of each other having rating as a parent node. This graphical representation corresponds to the fork structure depicted in Figure 6. Independent assumptions are drawn from d-separation principles about this graphical structure to simplify computation.

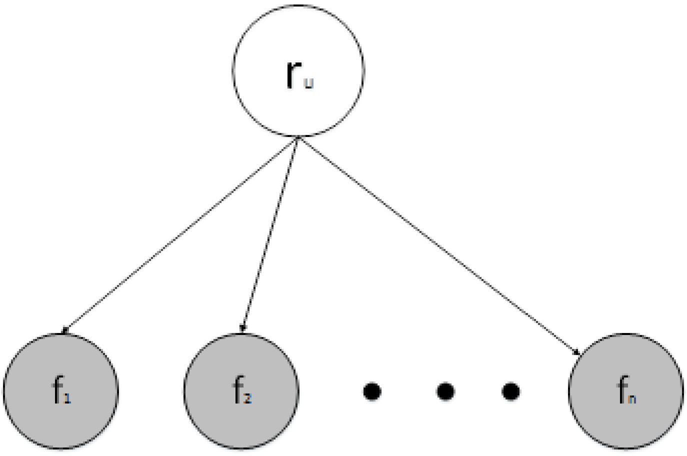

Fig. 7: Graphical representation of user rating prediction based on Naïve Bayes model.

The conditional probability of a rating giving feature nodes as the evidence can be expressed according to Bayes theorem, as the probability that the hypothesis and evidence are true, divided by the probability that the evidence is true.

$$p(r_k|f_1,\dots,f_n) = \frac{p(r_k)p(f_1,\dots,f_n|r_k)}{p(f_1,\dots,f_n)} = \frac{p(r_k,f_1,\dots,f_n)}{p(f_1,\dots,f_n)} \tag{7}$$

Although one can try to estimate the joint probability, but in case when the number of features is too high, it is better to decompose the formula. Because the denominator is not dependent on the class that is being predicted, the nominator decomposition is what matters. Applying the chain rule, the nominator is derived as follow:

$$p(r_k,f_1,\dots,f_n) = p(f_1|f_2 \dots,f_n,r_k) \dots p(f_i|f_{i+1} \dots,f_n,r_k) \dots p(f_n|r_k)p(r_k) \tag{8}$$

Now, because feature nodes are independent of each other based on the Naïve Bayes model, the equation can be simplified as:

$$p(f_i|f_{i+1} \dots,f_n,r_k) = p(f_i|r_k) \tag{9}$$

The joint probability is now given by:

$$p(r_k, f_1, \dots, f_n) = p(r_k) \prod_n p(f_n|r_k) \tag{10}$$

The denominator in equation (7) is a constant if the evidence is known. This is represented by a scaling factor Z.

$$p(r_k|f_1 \dots f_n) = \frac{1}{Z} p(r_k) \prod_n p(f_n|r_k) = \frac{p(r_k) \prod_n p(f_n|r_k)}{p(f_1, \dots, f_n)} \tag{11}$$

The aim is to predict rating for given a set of features. Therefore, the probability that a particular rating is true for a given feature setting is calculated and the rating value with the highest probability is chosen. In other words, the maximum a posteriori (MAP) probability that the rating value is true having features as evidence is selected. In classification the scaling factor can be omitted and MAP decision on the nominator is applied.

$$\hat{r} = MAP[p(r_k|f_1 \dots f_n)] = \arg\max_{r_k} \left\{ p(r_k) \prod_n p(f_n|r_k) \right\} \tag{12}$$

5.3 Averaged One-Dependence Estimator

Averaged one-dependence estimator (AODE) is method described in [Webb 2005] and was designed as a text classifier. It is a novel approach of model aggregation relaxing the independence assumption of Naïve Bayes network. Instead of searching in a space of possible models, the resulting network is an one dependence classifier. This allows one to skip the step of model selection, which is often computational costly as the number of ways to organize a graphical structure grows exponentially with the number of features. Another advantage of this method is that it avoids additional variance, which is introduced in the model selection process.

Two models that belong to the family of graphical model estimators use Naïve Bayes classifier while relaxing its independent assumption with Lazy Bayesian Rules (LBR) [Zheng 2000] and Super Parent Tree Augmented Naïve Bayes (SP-TAN) [Friedman 1997]. Both have their shortcomings where AODE seems to overcome. LBR uses lazy learning and for every feature depending on the class a set of features is selected by a simple heuristic wrapper approach minimizing the training error. It shows high precision but works well only for a small number of examples as it takes a long time to train. In comparison SP-TAN allows every feature to depend only on class and one other feature. The parent feature is selected using conditional mutual information. Then the model is selected using Minima Description Length (MDL) function to find the optimal network structure. Therefore, this approach still needs to build a graphical model and the process of parent selection is repeated potentially until every feature has a parent. This introduces model variance and is also computationally costly.

5.4.1 AODE model description

As for SP-TAN, this model consists of one feature dependent only on the class and one parent feature. In this way, an independent model is built for every feature and the resulting class prediction is computed as the average over these models. The feature is selected based on some threshold criteria. As this model was designed for text classification, the threshold in this case was the number of times the feature appeared in the user training set. If the feature was represented less than 30 times, the feature was not selected and the classifier would only consist of a selected node dependent on the class node. The number 30 is selected based on a broadly used statistical significant sample size. If the feature is selected, then the classifier consists of one dependence classifier. All selected one-dependence classifiers are then aggregated to make predictions. By applying the chain rule on joint probability of rating and set of features $p(r_k, F)$, the resulting formula for a feature f_i having parent node f_j and dependent on class r_k look as follow:

$$p(r_k, F) = p(r, f_j)p(f_i|r_k, f_j) \tag{13}$$

As equation (13) holds for every feature f_j, it also holds for the mean over any group of features:

$$p(r_k, F) = \frac{\sum_{j\epsilon\{\#f_j>T\}} p(r, f_j)p(f_i|r_k, f_j)}{|j\epsilon\{\#f_j > T\}|} \tag{14}$$

where the sum is computed over all the feature combinations holding the threshold condition. In this case, it is true for every feature value having more than 30 occurrences in the training set. For the data set used in this thesis, however, considering that the mean training set size is 27, it would be hard to find a feature appearing in the user training set 30 or more times. Because of that, instead of this hard threshold, conditional entropy (CE) $H(f_i|f_j)$ is implemented to decide whether two nodes are dependent or not.

$$H(f_i|f_j) = \sum p(f_i, f_j)\, log \frac{p(f_i, f_j)}{p(f_j)} \tag{15}$$

Conditional entropy explains how much information is needed to describe the outcome, which is in this case the dependent feature f_i. Conditional entropy $H(f_i|f_j)$ is 0 only if the dependent feature f_i is completely determined by the parent feature f_j, and is equal to entropy $H(f_i)$ if the feature f_j does not give any information about f_i and they are independent.

In [Chow 1968] mutual information $I(f_i, f_j)$ was used to make this decision and in [Friedman 1997] conditional mutual information $I(f_i, f_j|r)$ was used.

The denominator in equation (15) is constant for all class values, therefore it can be omitted. The classifier again will look for the highest probability, therefore similarly as in the equation (11), the denominator does not need to be computed. The class is then selected as:

$$\arg\max_{r_k}\left\{\sum_{j\epsilon\{H(f_i|f_j)<T\}} p(r_k, f_j)\prod_{i\epsilon n} p(f_i|r_k, f_j)\right\} \tag{16}$$

The sum in equation (16) is over all the feature combinations holding the condition that the threshold T is higher than the conditional entropy of two features $H(f_i|f_j)$, where f_j is the parent feature. If no combination of features satisfies this condition for the user training set, equation (12) for Naïve Bayes classifier is used instead.

As this approach is also affected by the data set sparsity, Laplace smoothing is also applied as in case of NBC.

5.4 Learning Casual Models – Constrain Based Learners

Sometimes the relationship between graph variables is known and the graphical structure is constructed accordingly. In case of this design, the purpose is to find patterns in user data, examine how item features interact with each other and how they affect the user rating. Therefore the focus is on algorithms that construct a graphical structure from data. First, the following assumptions needs to be made:

- **Causal sufficiency assumption** - There exist such a DAG that represents the relations of causation among the variables. There is no common unobserved variable which may explain dependences of observed variables, or lack thereof.
- **Markov condition** – any node in a Bayesian network is conditionally independent of its non-descendants, given its parents. For a given Bayesian network, there is a limited set of independence relationships between a node and its non-descendants.

- **Faithfulness** – graphical representation of Bayes network and a probability distribution are faithful to one another if all of the independences from probability distribution are entailed by the Markov condition in the given graphical structure.

There are two approaches of building a graphical model:

- **Score based** – searches a space of possible graphical structures and score each of them. The score represents the ability of a network to represent the data. The graphical structure with the highest score is then selected. The number of possible graphical structures, however, increases exponentially with the number of features making this searching procedure NP-hard [Heckerman 1995].
- **Constraint based** – conditional independence is often used as the constraint when using graphical structure. This type of constraint assumes no missing values in set.

5.4.1 PC Algorithm

In case of data set used in this research, the number of features is already too high to apply the scoring based approach to find the best graphical representation of data. Therefore constraint learning techniques are more suitable, more specifically, PC algorithm named after Peter and Clark [Spirtes 2001] is used.

PC algorithm originates in IC algorithm designed by Verma and Pearl [Pearl 1994]. This algorithm is based on two principles. The first principle uses directed edges to recover all the causal dependences. The second principle sets direction of edges between any three connected nodes for which the direction is unknown. The same principle is used for directing edges in PC. The algorithm then iterates through all undirected edges until all of the edges have set direction. This algorithm assumes knowledge of conditional independences between nodes. This is however often not the case and this knowledge need to be obtained from the data. Another issue of this algorithm is that independences between all pairs of nodes need to be examined given the full subset of variables excluding the examined pair. In other words, this subset represents all the nodes adjacent to the examined pair. Because the number of such subsets increases exponentially with the number of features, this algorithm becomes unfeasible for a large number of variables.

The first issue can be overcome by applying a statistical significance test for conditional independence. Partial correlation can be used to test this significance between two variables and a subset of dependent variables. Then the standard significance test is used to decide if the partial correlation is equal to zero, meaning that the conditional independence between a pair of variables exists. This had been applied by the causal discovery program TETRAD II [Scheines 1994]. PC algorithm goes even further. It applies the independence test and reduces the complexity of the search through the subset of variables. This makes the PC algorithm easy to implement the constraint learning method that can be used to discover the topology of a network [Korb 2010].

PC algorithm needs the assumption of a large database with no missing inputs, no errors in the statistical test, and all possible dependencies can be represented in a directed acyclic graph (DAG). Because real world data sets often contain missing inputs, this condition is not always satisfied, resulting in a poor performance of the approach. This issue is being resolved by implementing the algorithm for estimating missing entries described in subsection 5.1.4.

PC algorithm begins with a fully connected graph and removes the edge between pair of nodes if the conditional independence test is positive. During this process, the approach keeps track of nodes which were d-separated, which prevents it from testing the same pair for independence twice. Partial correlation is used to test this independence. This technique allows fixing the number of adjacent nodes. The edge is often removed for a small number of adjacent nodes and therefore much earlier in the test process compared to IC algorithm. This however depends on the true graphical representation of data and dense models need more time and higher order subsets to be tested for independence. This is however rarely the case and most of the models are sparse, resulting in increasing effectiveness and reduction of the computational cost.

This method has one shortcoming occurring when an edge is removed by accident early in the process. This error then progresses throughout the computations of partial correlation of a higher order of subset. This results in increasing number of correlations needed to be estimated and introduces further errors especially for moderately large networks. This error is more likely to occur for large models with moderately small sample sets, while models with large sample sets usually do not exhibit this issue [Dai 1997].

5.4.2 PC Algorithm Aproach Description

PC algorithm has two main steps. In the first step, an undirected graphical structure is learned from data. Independence test is used to decide whether nodes should be connected or not. In the second step, a set of rules are applied to direct edges. The main benefit of this method is that variables are conditioned only on subset of variables adjacent to them. Compared to Spirtes, Glymour and Scheines (SGS) algorithm, it does not require higher order independence relations to be tested. The independence between nodes is determined by the statistical significance test rather than d-separation. Partial correlation is used as the independence measure. Partial correlation measures degree of independence of two variables (X,Y) given a set of controlling variables S [Baba 2004]. The partial covariance matrix is then given by:

$$\Sigma_{XY.Z} = \begin{bmatrix} \sigma_{XX.S} & \sigma_{XY.S} \\ \sigma_{YX.S} & \sigma_{YY.S} \end{bmatrix} \tag{17}$$

Where $\sigma_{XY.S}$ is the partial covariance coefficient, defined as the projection of X and Y residuals on the linear space spanned by set S:

$$\sigma_{XY.S} = cov\left(X - \hat{X}(S), Y - \hat{Y}(S)\right) \tag{18}$$

where $\hat{X}(S)$ is the projection of X defined as:

$$\hat{X}(S) = E(X) - \Sigma_{XS}\Sigma_{SS}{}^{-1}\big(S - E(S)\big) \tag{19}$$

The partial correlation is then:

$$\rho_{XY.Z} = \frac{\sigma_{XY.S}}{\sqrt{\sigma_{XX.S}\sigma_{YY.S}}} \tag{20}$$

where $\rho_{XY.S}$ is the correlation coefficient. This coefficient is positive definite and therefore invertible

Feature X is independent to feature Y, given a set Z, if and only if the partial correlation coefficient is zero $\rho_{XY.Z} = 0$. Two features are completely dependent if the correlation coefficient is one $\rho_{XY.Z} = 1$. The independence property does not usually hold. Therefore p-value test is often used to determine independence. If the *p*-value is less than or equal to a selected threshold α, then the null hypothesis is rejected and, if the *p*-value is large, say more than α, then the null hypothesis holds

Because algorithms searching for a graphical structure can be computationally costly, the complexity of this approach needs to be considered. In the worst case scenario, the number of independence tests performed on a graph is given by:

$$\frac{n^2(n-1)^{k-1}}{(k-1)!} \tag{21}$$

where n is the number of edges.

Algorithm: PC algorithm

If the size of set S of adjacent variables to (X,Y) is given by k, then:

1. Begin with the fully connected skeleton model; i.e., every node is adjacent to every other node.
2. Set k=0. For all pairs of nodes X and Y, initialize the group of nodes separating X and Y to be empty (X,Y) =Ø.
3. For every adjacent pair of nodes X and Y, remove the arc between them if and only if for all subsets S of the order k containing nodes adjacent to X (excluding node Y) the statistical significance test holds.
   Record the nodes separating (X,Y) in a separation matrix SM.
4. Repeat step 3 until all pair of nodes are tested for a given k. Then, increment k and go to step 3.
5. For each triple X – Y – Z in an undirected chain (such that X and Y are connected, Y and Z are connected, but not X and Z), replace the chain with X → Y ← Z if and only if Y ∉ (X,Z).
6. Iterate through all undirected arcs Y – Z in the graph. Orient Y → Z if and only if either
   a. at a previous step Y appeared as the middle node in an undirected chain with X and Z (so, Step 5 failed to indicate Y should be in v-structure between X and Z) and now the arc between X and Y is directed as X → Y;
   b. if nodes Y ← Z are being directed, then a cycle would be introduced.
7. Continue iterating through all the undirected arcs until one such pass fails to direct any arcs.

5.4.2 Computing inferences in DAG created from PC algorithm

Once a DAG is created using the PC algorithm described in previous section, the principle of d-separation is applied and Bayes theorem is used to compute probabilities at each node. The resulting probability is the conditional probability of rating given the feature nodes $p(r|F)$, where F is the set of parent feature nodes and their resulting probabilities. The probability distribution is best explained by an example. The example of DAG created using the PC algorithm is given in Figure 8. In this case feature node f_4 has three parent nodes $\{f_1, f_2, f_3\}$

which are conditionally independent of each other given a feature node, i.e. $f_1 \perp f_2 \perp f_3 | f_4$. First, Bayes theorem is applied and because only the most probable outcome is used for recommendation, maximum posterior rule is applied. The probability of this node is then computed as:

$$p(f_4|f_1, f_2, f_3) = \frac{p(f_1, f_2, f_3|f_4)p(f_4)}{p(f_{1,}f_2, f_3)} \propto p(f_4) \prod_{n \in P(f_4)} p(f_n|f_4) \tag{22}$$

In general this equation can be written as:

$$p(ch|F) \propto p(ch) \prod_{fn \in F} p(\pi_{f_n}|ch) \tag{23}$$

where ch is simply any children node, feature or rating node, having a set of parent nodes F. Each parent node has its own probability distribution, which is given by π_{f_n}.

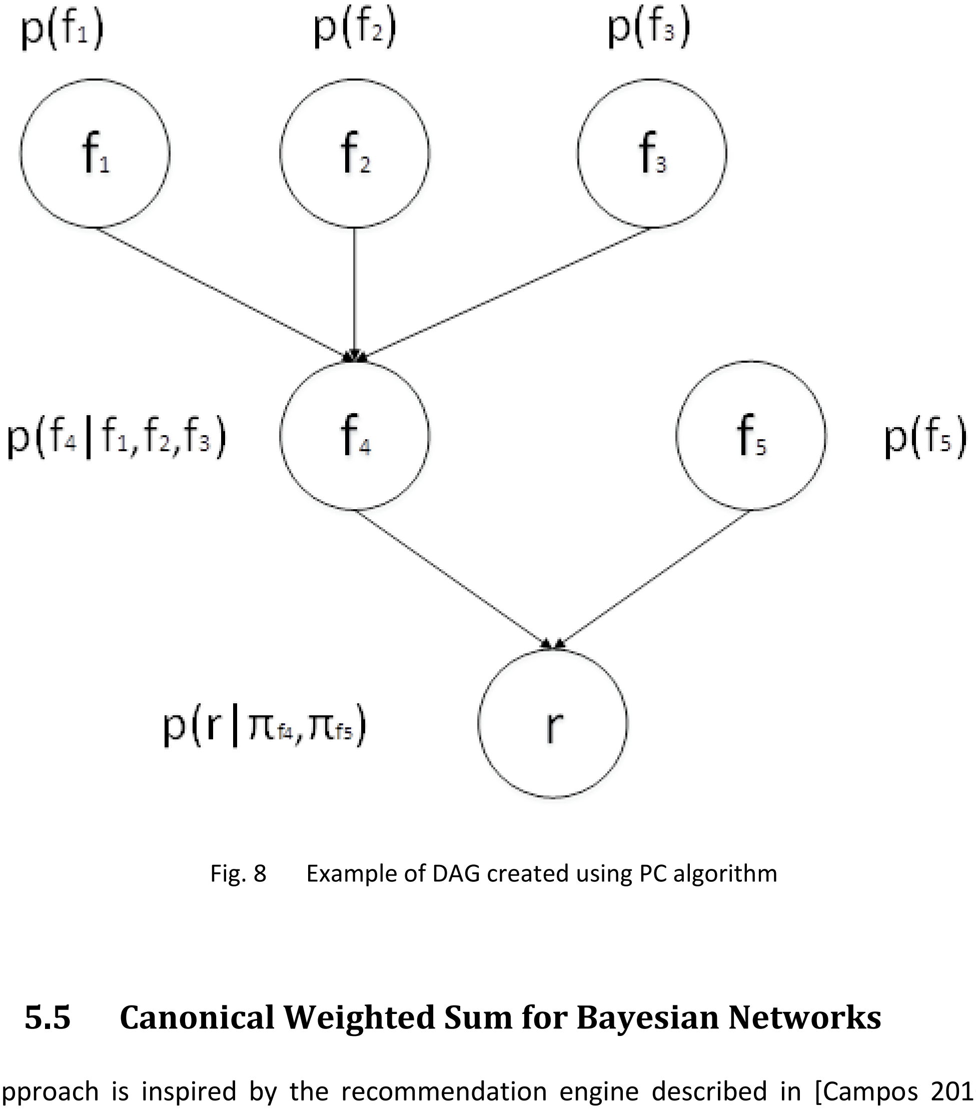

Fig. 8 Example of DAG created using PC algorithm

5.5 Canonical Weighted Sum for Bayesian Networks

This approach is inspired by the recommendation engine described in [Campos 2010]. The same methodology is applied to the Naïve Bayes network topology depicted in Figure 7. The definition of canonical weighted sum is:

”Let X_i be a node in a BN, let $Pa(X_i)$ be the parent set of X_i, and let Y_k be the kth parent of X_i in the BN. By using a canonical weighted sum, the set of conditional probability distributions stored at node X_i are then represented by means of

$$\Pr\left(x_{ij}\middle|pa(X_i)\right) = \sum_{Y_k \in Pa(X_i)} w(y_{k,l}, x_{i,j}) \tag{24}$$

where $y_{k,l}$ is the value that variable Y_k takes in the configuration $pa(X_i)$, and $w(y_{k,l}, x_{i,j})$ are weights (effects) measuring how the lth value of a variable Y_k describes the jth state of node X_i. The only restriction that we must impose is that the weights are a set of non-negative values verifying that for each configuration $pa(X_i)$” [Campos 2010].

Only content based component of this design is researched in this thesis, where the network consists of items and features describing the items. The edge between item and a feature exists only when the feature describes the item. By constructing such a network, two items become dependent if they share a common subset of features. The network finishes with connection of the item nodes to the active user node, which is a representation of ratings given by the active user to the set of items from user set. Figure 9 shows an example of such a network, where $\{f_1, f_2, \ldots, f_n\}$ are all possible features describing an item and $\{I_1, I_2, \ldots, I_m\}$ are all possible items in data set. Node AUr represents ratings of the active user given to the sets of items in the active user set.

The aim is to predict the rating of an unobserved item. This item will simply be connected to the network topology of the active user along with all the connections between the item and feature nodes describing it. Taking the topology described in Figure 9, the added item is represented by the node I_3 and all the edges related to this node are drawn in dashed lines. Then the probability that the user will assign a particular rating value to this item given the evidence ev: $\Pr(AUr = r|ev)$ is computed. To do this, the evidence and its propagation towards the AUr node is identified. The evidence comprises the features describing the item the predictions are made for.

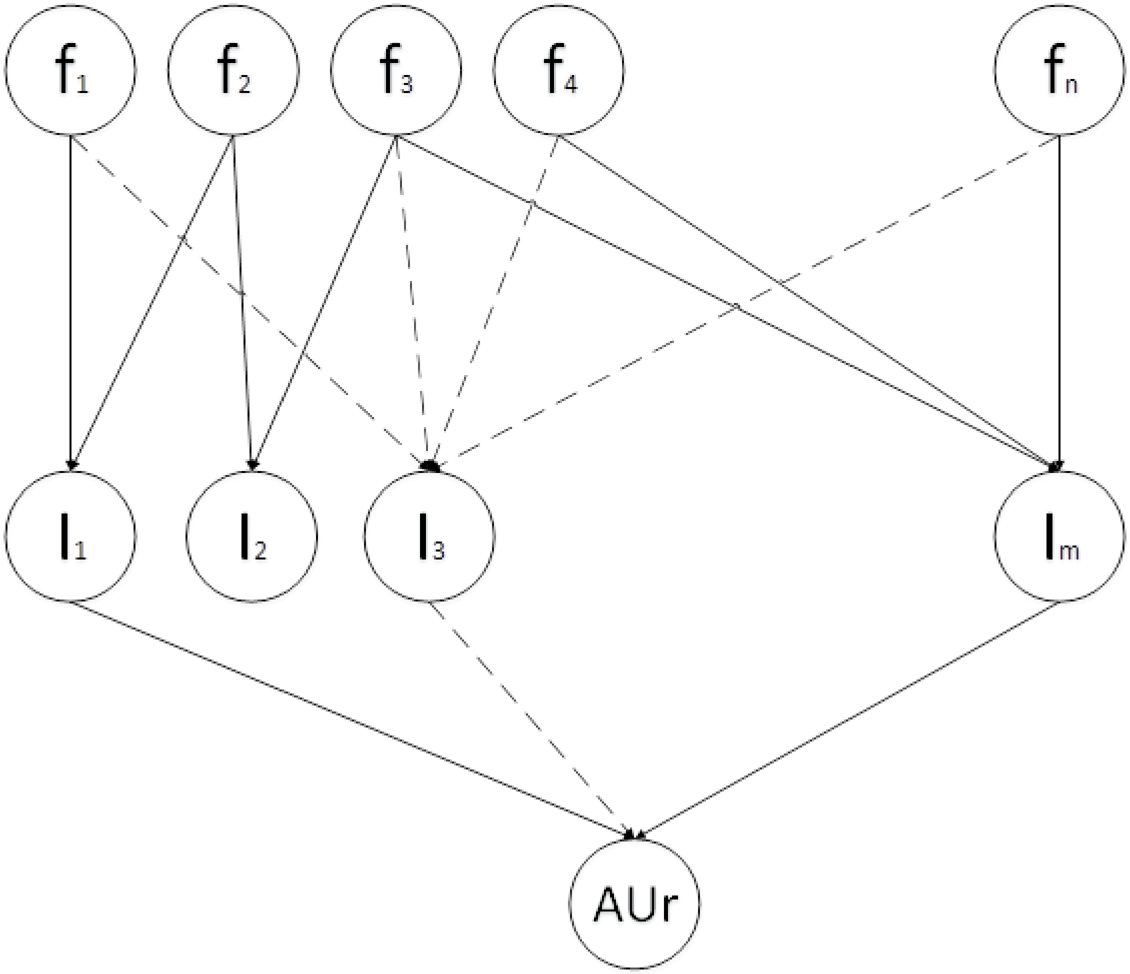

Fig. 9 Example of Bayesian network for canonical weighted sum approach.

5.5.1 Propagation of the evidence

Because the evidence is propagated in a Bayesian network, every node is independent of its parents if the parent is an observed variable. The conditional probabilities are then computed as the canonical weighted sum based on equation (24). Finally, the posterior probability distribution can be efficiently computed as a top-down inference mechanism. The distributions of one layer are then obtained from the posterior probabilities of the previous layer. Each node collects the evidence from its parents. This evidence is not further distributed to the node descendants. The following theorem from [Campos 2010] explains how the computation is done.

Theorem 1 [Campos 2010]**.** Let X_a be a node in a BN network, let m_{X_a} be the number of parents of X_a, Y_j be a node in $Pa(X_a)$, and l_{Y_j} the number of states taken by Y_j. If the conditional probability distributions can be expressed under the conditions given by the equation (24) and the evidence is only on the ancestors of X_a, then the exact posterior probabilities can be computed using the following formula:

$$\Pr(x_{a,s}|ev) = \sum_{j=1}^{m_{X_a}} \sum_{k=1}^{l_{Y_j}} w(y_{j,k}, x_{a,s}) * \Pr(y_{j,k}|ev) \tag{25}$$

In this case the evidence is represented by the set of features describing an item. Feature nodes have no parents. Therefore for these nodes only the prior probability distributions needs to be computed. The relative frequency to estimate the feature probability is employed:

$$\Pr(f_{k,1}) = \frac{n_k + 0.5}{m + 1} \tag{26}$$

where n_k is the number of times feature f_k has been used to describe an item and m is the number of items. The probability for the feature not being present for a particular item is then given by $\Pr(f_{k,0}) = 1 - \Pr(f_{k,1})$.

For item variables equation (25) is used as it is the probability of an item being described by a set of features, which are parent nodes to the item and serves as the evidence $\Pr(i_{j,1}|pa(I_j))$.

To weigh the importance of features describing an item, invert document frequency is used. This concept is commonly used in information retrieval [Salton 1983]. The idea is to give higher importance to those features, which are less frequently used to describe an item as they carry more information about the item than a feature which describes many items in a data set. The weights are computed as:

$$w(f_{k,1}, i_{j,1}) = \frac{1}{M(I_j)} \log\left(\left(\frac{m}{n_k}\right) + 1\right) \tag{27}$$

where $M(I_j)$ is a normalizing factor computed as:

$$M(I_j) = \sum_{F_k \in Pa(I_j)} \log\left(\left(\frac{m}{n_k}\right) + 1\right) \tag{28}$$

In case the feature does not describe an item, it importance is zero $w(f_{k,0}, i_{j,1}) = 0$.

Now the prediction node is defined as the rating node of the active user AU_r. The influence of every item the user rated is considered and weights are assigned. The AU_r node has several states s representing all the possible rating values. As the rating ranges from 1 to 5, there are 5 possible states for this node. For any item I_k belonging to the user set, the state of the rating node is known and all the probability mass is assigned to this value s. It is assumed that all the items are equally important for predicting a rating by the active user for an unseen item. Because of that, the weights are computed as follow:

$$w\left(i_{k,1}, u_{au,s}\right) = \frac{1}{I(U_{au})} \tag{29}$$

where $u_{au,s}$ is the active user rating an item $i_{k,1}$ with the rating value s. For all other rating values, the weight is equal to zero, i.e. $w\left(i_{k,1}, u_{au,t}\right) = 0 \; for \; t \neq s \, ; t \in r$.

The rating for an unseen item is then chosen as the maximum posterior probability:

$$\hat{r} = MAP\left[\Pr\left(u_{au,r} \middle| ev\right)\right] \tag{30}$$

In the paper [Campos 2010], the evaluation of an algorithm performance is done using MAE as the accuracy measure. In this research, this measure is used together with weighted rating error, which will be described in the evaluation section.

Chapter 6. Transfer Learning

It has been observed that a human brain can use knowledge acquired in a previous task to learn a new task faster. Just as when someone learns one foreign language, to learn the second foreign language is much easier. It is the ability to transfer knowledge from one learning task, called the source domain, to another, the target domain. The new task usually takes less time to learn and the accuracy of acquired knowledge is higher.

Common machine learning techniques learn tasks in isolation. Merging machine learning algorithms with the ability to transfer knowledge between learned tasks bring the machine learning closer to the efficiency of human learning. Transfer learning techniques are highly dependent on the machine learning algorithms. They can be therefore considered as an extension of those algorithms. There are basically two ways transfer learning adds to the machine learning techniques. One is through the inductive learning used for classification and inference tasks. Another is the reinforcement learning used for Q-learning and policy search [Torrey 2009].

The aim of transfer learning is to improve learning of the target task. The learning of target task can be improved in two different ways:

- **Learning the task faster** – the task can be learning significantly faster using transfer learning compared to learning from scratch. For instance, deep learning networks take a long time to be trained. Once they are trained they perform with high accuracy. Transfer learning can decrease the learning time by re-learning only the last few layers of an existing deep learning network.
- **Higher performance** – this is usually a desirable outcome when transfer learning is added to a machine learning algorithm. The aim is for the algorithm to perform with a higher accuracy.

Figure 10 shows benefits which can be observed during the learning process when transfer learning is implemented to the original algorithm. It compares an algorithm using transfer learning to the original algorithm without the transfer learning implemented.

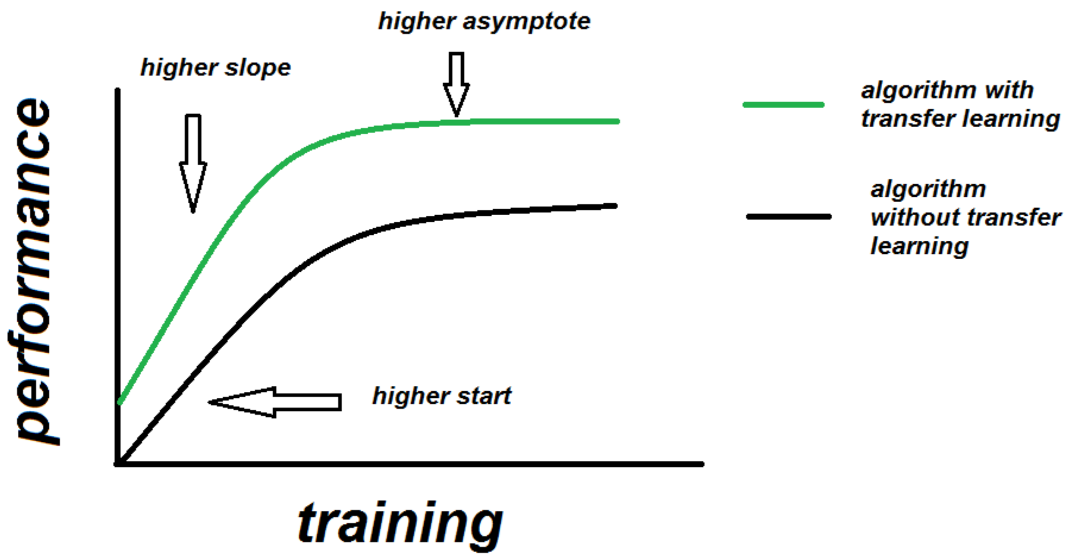

Fig. 10 Comparison of algorithm with and without transfer learning in training process from performance point of view [Torrey 2009]

Some machine learning techniques, such as neural networks, deep learning, Bayesian networks and others, require long training time. Although the models can perform with high accuracy after training, a new model needs to be trained for another task or when the data set changes. Transfer learning reduces this second training to a fraction of the original time, while assuring the same or higher accuracy. Another common problem is insufficient amount of data needed to train the model. Knowledge from another similar domain with extensive data set can be transferred. These are the main reasons why this topic is of ongoing interest in the machine learning community.

It is rare that transfer learning decreases the performance of the original method. If this happens, it is called a negative transfer. This usually happens when tasks are very weakly related. There are a couple of ways to approach this issue. It is always important to map the characteristics of the original task to the new task well. In this process, the correlation between the two tasks is identified and the transfer learning is adjusted so that the highest possible benefit of adding transfer learning is achieved.

6.1 Transfer Learning Notation

So far only the case of transferring the knowledge from the original, also called source task, to the new or target task has being described. In other words it is a transfer between two knowledge domains. Domain is specified by a feature space $X = \{x_1, x_2, \dots, x_n\}$ and a marginal probability distribution $P(X)$: $D\{X, P(X)\}$ [Pan 2010]. The task is specified by a class or a label space $Y = \{y_1, y_2, \dots, y_l\}$ and a predictive function $f(.)$: $T\{Y, f(.)\}$.The predictive function is used to predict the label for unobserved entries and it is trained on a labelled data set. In my work, the predictive function is a probability function $P(Y|X)$. The source domain D_S is then given by the pair of features and labels as $D_S = \left\{(x_{S_1}, y_{S_1}), (x_{S_2}, y_{S_2}), \dots, \left(x_{S_{n_S}}, y_{S_{n_S}}\right)\right\}$ and similarly the target domain is defined by: $D_T = \left\{(x_{T_1}, y_{T_1}), (x_{T_2}, y_{T_2}), \dots, \left(x_{T_{n_T}}, y_{T_{n_T}}\right)\right\}$. Typically, the number of feature-label pairs for source domain is higher than the number of pairs for the target domain $0 \leq n_T \ll n_S$.

Note that there can be more than one source domain as well as more than one target domain in the process of transfer learning. The focus of this research is on the case when there is only one source and one target domain.

Transfer learning can be done for different domains as well as for different tasks. In case the learning is performed for different domains $D_S \neq D_T$, either the feature space is different $X_S \neq X_T$ or the marginal probabilities are different $P_S(X) \neq P_T(X)$. An example of this is the text classification concept. The feature space is different when learning different languages, where words are features, while it is assumed that the marginal probability distribution of words is the same for both languages. The example of different marginal probabilities describes the case when the learning is performed for the same language but different documents. Therefore words are the same, i.e. same feature space, but their probability distribution differs in each document. If the learning is performed for different tasks $T_S \neq T_T$, then either the labels are different $Y_S \neq Y_T$ or the probability functions are different $P_S(Y|X) \neq P_T(Y|X)$. An example can again be a text classification concept. The labels are different when two documents use different number of class labels. The probability functions are different when two documents have a very unbalanced user defined classes.

Two domains or tasks are related if there exists some kind of relationship between them. This relationship needs to be identified in order to make a successful transfer between domains or tasks.

6.2 Transferring Knowledge in Inductive Learning

Inductive learning is typically used for classification tasks, which is the case of the recommendation system built to predict rating and then recommend an item with the highest rating to a user. Inductive learning is commonly used in artificial neural networks, rule-based learners and graphical models like Bayesian and Markov networks [Richardson 2006].

For Bayes and Markov networks, the learning algorithm searches in a space of possible models, this is called the hypothesis space. Inductive bias is used to search in this hypothesis space to choose the model with the lowest error rate. Inductive bias is defined as a set of assumptions about the true distribution of the training data [Mitchell 1997]. Naïve Bayes classifier does not need to search for the best model as it uses independent assumptions which define the model. Inductive bias is also used for rule-based learners to search in a space of rules and determine the order in which hypothesis are considered.

Transfer learning is used in inductive learning tasks to affect the inductive bias in the process of searching in hypothesis space for the best model. The knowledge of source task affects the inductive bias when the knowledge is transferred to the target task. There are basically two approaches for applying transfer in inductive learning based on what the desired outcome is for the target task, which can be:

- Finding the model faster – It is often desired to speed up the process of searching for the best model. This can be done by either narrowing the hypothesis space or removing some search steps from consideration.
- Increasing model complexity - The model of the resulting task can be more general by adding new search steps in a process of searching the hypothesis space or by broadening this space.

The example of finding the model faster by adjusting the hypothesis space is illustrated in Figure 11. On the left hand side, the search for the best model is done in a considered hypothesis space which is represented by the circle and it is selected by an inductive learning method from the space of all possible hypotheses. On the right hand side, the same process takes less steps when the transfer learning is added to the original inductive learning method by narrowing the space of considered hypotheses, where the dashed line represents the original space and the ellipse is the considered hypothesis space selected by the transfer inductive learning.

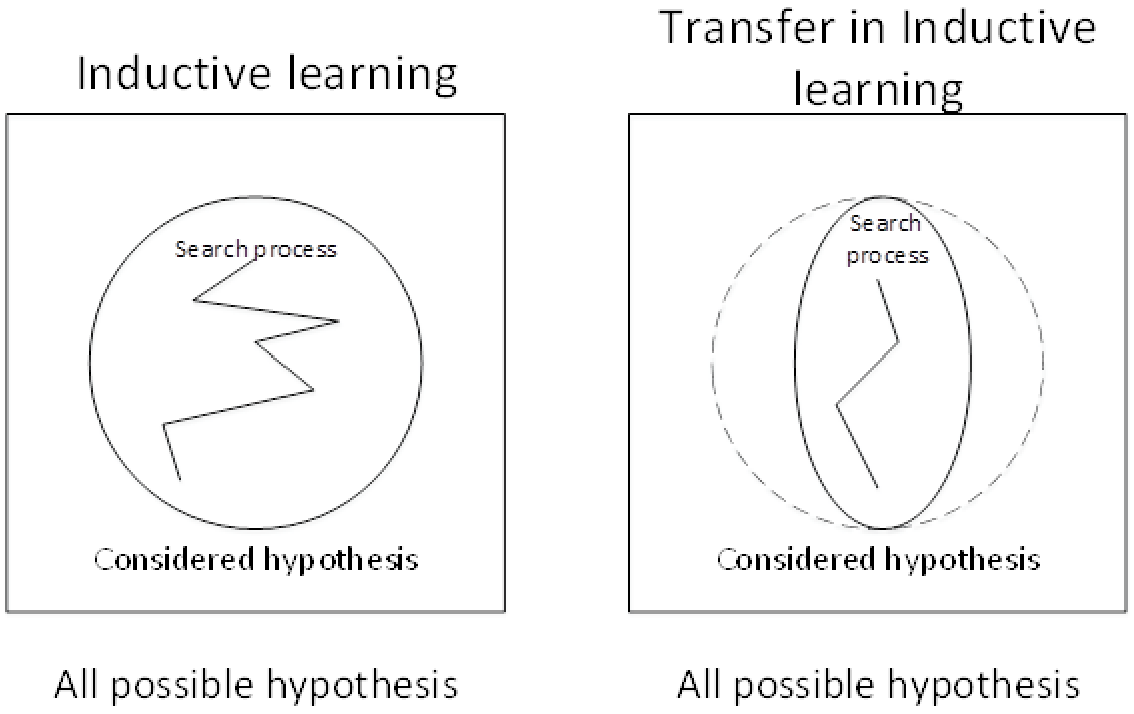

Fig. 11 Comparison of the process of searching in a hypothesis space for inductive learning and when transfer learning is involved [Torrey 2009].

The application can be found in [Baxter 2000]. The paper deals with the task of narrowing the hypothesis space by solving a set of related source tasks in each hypothesis space and choses the one with the lowest error in the target task. It describes the process of learning inductive bias to improve the generalization capability of a target task based on the knowledge derived from a number of source tasks.

The considered hypothesis space for the target task can be better specified by adjusting or removing hypothesis that are too general or too specific for the source domain. It is another way of narrowing the hypothesis space. This approach is used to learn Markov logic networks in [Mihalkova 2007].

Learned model often needs to be updated over a period of time. This requires the change of its parameters. This task can be viewed as learning a target task from a source task. The time needed for model updating can be decreased using transfer inductive learning. This approach is used in [Thrun 1995] applied to neural networks to update the network over time by encountering a collection of related problems. Instead of searching in a hypothesis space, the approach is based on adjusting the gradient-descent with learned slope information.

The inductive transfer can be divided into three basic methodologies:

- **Bayesian transfer** – prior knowledge about model is transferred to the target learning task. Close attention is payed to this setting in the next subsection.
- **Hierarchical transfer** – knowledge is extracted from multiple simple tasks and combined to learn a more complex task.
- **Unsupervised or semi-supervised transfer** – some literatures described this type as a separate category of transfer learning [Jialin 2010]. In this case, data do not have labels in both source and target domain. This application includes typical unsupervised classification tasks such as clustering, dimensionality reduction and density estimation.

6.2.1 Bayesian Transfer

Bayesian networks are the primary approach chosen in this study to design recommendation engine for HBB TV. Transfer learning for Bayesian networks is a specific area of inductive transfer. When learning the network, prior distribution is often determined before seeing any data. This can be seen as knowledge about the source task, which makes Bayesian networks perfect for applying transfer learning. Prior distribution is then combined with the knowledge about data to define a posterior distribution. This is learning the target task.

The knowledge can also be transferred between domains using different data sets. The model is then trained on one set of data, which is in this case the source domain, and transferred to another set of data, which shares the same features but the probability distribution is different. This is often used when there is more data in the source domain than in the target domain.

Typical application is using inductive transfer learning for Naïve Bayes classifier for text classification [Nigam 2000], [Dai 2007]. EM algorithm is used to transfer knowledge from a source data to a target data set. The relationship between probability distribution of source and target data sets are determined by Kullback-Leibler divergence.

Bayesian transfer has been applied not only to learn Bayes networks but also to logistic regression tasks [Marx 2005], [Raina 2006]. Gaussian distribution is used as a prior and by applying inductive transfer, the mean and variance of the distribution is averaged over several source tasks.

6.3 Transfer Learning for Naïve Bayes Classifier

Transfer learning approach described in this section belongs to the family of inductive Bayesian transfer. As users usually have only a few items in their training set, there is insufficient data associated with a user to make accurate predictions. Transfer learning technique is chosen as a tool to extend the information that can be acquired about a user by creating group of users with similar interests. The aim is to group users according to their similarity, using different similarity measures, train Naïve Bayes classifier for this group and then transfer the knowledge to the individual user to predict the rating of unseen items. As Naïve Bayes classifier is used, there is no need to search in a space of possible graphical models. Only model parameters need to be derived.

In this approach, first, a correlation network of users based on data in their training set is created. From the correlation network, users are grouped using different settings of threshold to find the one which produces the least prediction error. Once the groups are created, training sets of users within the group are combined and a Naïve Bayes classifier using this set is trained. This knowledge is then transferred using Expectation-Maximization (EM) algorithm and Kullback–Leibler (KL) divergence. It is the type of transfer where feature sets are the same but the distribution is different in the source and target domain: $P_S(X) \neq P_T(X)$. This way of knowledge transfer was inspired by the work of [Dai 2007], where it was applied to transferring knowledge in text classification from training set to predict items from test set supposing that these two have slightly different distribution. To my knowledge transfer learning has not been applied to TV recommendation.

6.3.1 User groupping

For every pair of user, a similarity $im(u_i, u_j)$ is computed, where $u_i, u_j \in U = \{u_1, u_2, \dots, u_U\}$ and a correlation matrix $CR(i,j) = sim(u_i, u_j)$ is created. Different similarity metric were tested to measure user correlation. Based on this similarity metric, correlation network is created, where every link represents the correlation strength between users. Experimentally, the threshold for correlation strength is set and links below this threshold are removed to create user groups.

Considering an example of 10 users, it can be demonstrated how user groups are formed. Illustrative example of a correlation network is depicted in Figure 12. For better readability of the network, the similarity is not marked on every link, but note that each link has a similarity associated with it.

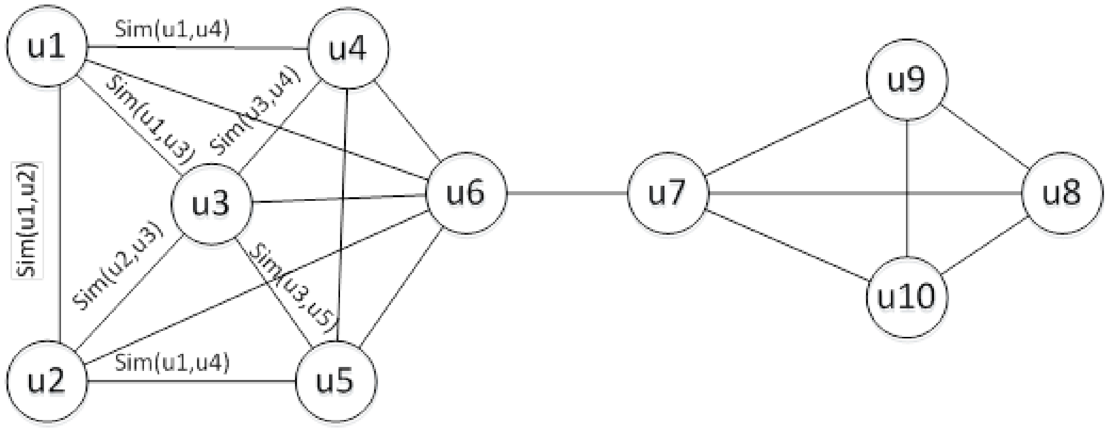

Fig. 12 Example of correlation network

Groups are the formed by looking at neighboring nodes, where it is set that the minimum group size is 2 members. From the example in Figure 12, 8 groups are formed as seen below. The groups contain lot of overlaps and if these overlaps contain more than half of the other group, then they are merged together to form one group. This way, 8 groups are merged into 2 groups named g_1 and g_7.

$$\left.\begin{array}{c} g_1\{u_1, u_2, u_3, u_4, u_6\} \\ g_2\{u_1, u_2, u_3, u_5, u_6\} \\ g_3\{u_1, u_2, u_3, u_4, u_5, u_6\} \\ g_4\{u_1, u_3, u_4, u_5, u_6\} \\ g_5\{u_2, u_3, u_4, u_5, u_6\} \\ g_6\{u_1, u_2, u_3, u_4, u_5, u_6, u_7\} \end{array}\right\} \quad g_1\{u_1, u_2, u_3, u_4, u_5, u_6\}$$

$$\left.\begin{array}{c} g_7\{u_6, u_7, u_8, u_9, u_{10}\} \\ g_8\{u_7, u_8, u_9, u_{10}\} \end{array}\right\} \quad g_7\{u_7, u_8, u_9, u_{10}\}$$

Users were groupped using three similarity measures, namely cosine similarity, Ochiai coefficient and Jaccard distance. Cosine similarity and Jaccard distance are described in the following section. Ochiai coefficient is similar to cosine similarity. It uses the same principles as cosine similarity and it is applied if sets can be presented as binary vectors.

Algorithm: ***User groupping***

```
For $u_i, u_j \in U = \{u_1, u_2, \dots, u_U\}$
    $CR(i,j) = sim(u_i, u_j)$
    If $CR(i,j)$ < threshold $\Rightarrow$ $CR(i,j) = 0$
end
graph($CR$)
For $i \in \{1,2,\dots,U\}$
    $g_i = neighbors(CR, u_i)$
end
For $g_i \in G = \{g_1, \dots g_G\}$
    If sum(ismember($g_i$, $G$)) > size($g_i$) $\Rightarrow$ join groups
end
```

6.3.2 Expectation Maximization algorithm

EM algorithm is an iterative method using maximum likelihood to find the model parameters of a statistical model. It has two steps. In E-step, it creates an expectation of the log-likelihood of the current parameter estimate and in M-step, parameters maximizing the log-likelihood are computed. This algorithm was proposed by Dempster, Laird and Rubin in [Dempster 1977]. It has been successfully applied in the area of statistical estimation to solve problem of incomplete data, mixture estimation, multi-frame super-resolution restoration methods and others [Hardie 1997], [McLachlan 1996].

For a set of independent data entries $\{x_1, x_2, \dots, x_m\}$, to make estimations for unseen entries, first the model $p(x,z)$ described by model's parameters θ needs to be fitted, where z are latent random variables. The likelihood function for model parameters estimation is given as:

$$l(\theta) = \sum_{m} \log p(x_m; \theta) = \sum_{m} \log \sum_{z} p(x_m, z; \theta) \tag{31}$$

Maximizing the likelihood function $l(\theta)$ in order to estimate model parameters can be done as an iterative process of estimating the lower bound of $l(\theta)$ (E-step) followed by its optimization (M-step).

The EM algorithm benefits from the Jensen's inequality theorem about convex functions. It has been proven that a convex transformation of a mean is less than or equal to the mean applied after convex transformation [Jensen 1906].

$$E[f(X)] \geq f(E(X)) \tag{32}$$

Where $f(X)$ is a convex function of a random variable X. The equality holds if and only if the random variable is constant, meaning that $X = E(X)$.

Suppose the latent variables have distribution $Q(z)$: $\sum_z Q(z) = 1\,; Q(z) \geq 0$, then using Jensen' inequality the equation (31) can be re-written as [Ng 2012]:

$$\sum_m \log \sum_z p(x_m, z; \theta) = \sum_m \log \sum_z Q(z) \frac{p(x_m, z; \theta)}{Q(z)} \geq \sum_m \sum_z Q(z) log \frac{p(x_m, z; \theta)}{Q(z)} \tag{33}$$

Equation (33) then returns a lower-bound on $l(\theta)$ for any set of distributions $Q(z)$. Suppose there is some prior knowledge about the model and some model parameters to start with can be initialized. This starting point then creates a lower-bound tight at these values of model parameters and it increases monotonically as the algorithm progresses through the steps of EM algorithm. The bound is tight for a particular setting of θ when the Jensen's inequality holds. This is achieved when the expectation is taken over a constant:

$$\frac{p(x_m, z; \theta)}{Q(z)} = c \;\rightarrow\; Q(z) \propto p(x_m, z; \theta) \tag{34}$$

Because $Q(z)$ is a distribution over latent variables summing to one, this leads to a further simplification resulting in:

$$Q(z) = \frac{p(x_m, z; \theta)}{\sum_z p(x_m, z; \theta)} = \frac{p(x_m, z; \theta)}{p(x_m; \theta)} = p(z|x_m; \theta) \tag{35}$$

where latent variables are given by the observed data entries x_m and model parameters θ. This is the E-step of EM algorithm, where the lower bound is specified for a given setting of model parameters. M-step then maximizes the equation (33) resulting in a new model parameter setting.

$$\theta = \arg max \sum_m \sum_z Q(z) log \frac{p(x_m, z; \theta)}{Q(z)} \tag{36}$$

6.3.2 Expectation Maximization algorithm application

Once the user groups are formed, user data sets within one group are concatenated and parameters of the Naïve Bayes classifier are trained using this group set. The improved Naïve Bayes classifier as described in the Section 5.2 with Laplace smoothing and missing vales estimation is implemented to this model setting. Expectation Maximization (EM) algorithm is used to train Naïve Bayes classifier using the group information while being able to make personalized predictions for a particular user.

EM algorithm together with Naïve Bayes has been used in text classification to transfer knowledge from unlabeled to labeled data [Nigam 2000], [Dai 2007]. In this thesis, this approach of transfer learning is used to transfer knowledge from a group set of users D_g to an individual user D_u.

Maximum likelihood estimate of an unknown parameter denoted by h is determined by the marginal likelihood of known data, which is in this case a group set D_g:

$$p(D_g|h) = \sum_{D_S} p_{D_u}(D_g, D_u|h) \tag{37}$$

The sum is performed over both set $D_s = \{D_g, D_u\}$.

The algorithm looks for a local optimum of the following Maximum a posteriori hypothesis under the user probability distribution $p_{D_u}(.)$:

$$h = \arg\max p_{D_u}(h)\, p_{D_u}(D_g, D_u|h) \tag{38}$$

However, instead of maximizing the formula in equation (38), the log likelihood of $l(h|D_g, D_u)$ is maximized as follow:

$$l(h|D_g, D_u) \propto \sum_i log \sum_r p_{D_u}(i|r,h) p_{D_u}(r|h) \tag{39}$$

where i are items in the considered data set and r are rating values.

It is now time to break down the steps of EM algorithm in the way it is implemented. In E-step, class for item i is predicted. As the prediction is done for Naïve Bayes classifier, and features are independent of each other, the conditional probability of a feature f_n having a rating r_k is the product of probabilities of features f_n describing an item i having a rating r_k and the prior class probability computed in the previous iteration. For simplicity, from now on, a feature $f_n \in F = \{f_1, f_2, \dots, f_n\}$ will be denoted as f and a rating r_k having k possible values as r.

E-step:

$$p_{D_u}(r|i) \propto p_{D_u}(r) \prod_{f \in i} p_{D_u}(f|r) \tag{40}$$

Result of equation (40) is then used to update parameters in M-step. Both probabilities are computed in the M-step across both data sets, $D_s \in \{D_u, D_g\}$. Note that the user training set is a subset of the group set $D_{u_{trainings}} \subset D_g$, but the user test set does not belong there. Because of that some features may not be represented in the group set. The chance, however, of a test feature not being present in a training set is much less in case of the group set than when the model is trained using only the user training set.

M-step:

Rating probability is computed for both data set distributions and $p_{D_u}(D_s)$ indicates how these two sets differ. To compute this, Kullback–Leibler divergence is used, which is explained later in this section.

$$p_{D_u}(r) \propto \sum_{s \in \{g,u\}} p_{D_u}(D_s)\, p_{D_u}(r|D_s) \tag{41}$$

where the probability of rating for a given data set is denoted as the sum over all items belonging to the data set of conditional probability of an item i having a rating r and a probability that the item belongs to the data set.

$$p_{D_u}(r|D_s) = \sum_{i \in D_s} p_{D_u}(r|i).\, p_{D_u}(i|D_s) \tag{42}$$

The probability of a feature f having a rating r is extended by space the samples are drawn from, which are in this case both data sets D_s. This probability is then computed by applying the law of total probability:

$$p_{D_u}(f|r) \propto \sum_{s \in \{g,u\}} p_{D_u}(D_s)\, p_{D_u}(r|D_s) p_{D_u}(f|r, D_s) \tag{ 43}$$

Laplace smoothing is applied to conditional probability $p_{D_u}(f|r, D_s)$, which can be estimated as

$$p_{D_u}(f|r, D_s) = \frac{1 + n_{D_u}(f, r, D_s)}{n_{D_u}(r, D_s) + 1 * F * k} \tag{44}$$

The $n_{D_u}(.)$ is the count function of the data set, which can be decomposed using chain rule:

$$n_{D_u}(f, r, D_s) = n_{D_u}(f|r, D_s) n_{D_u}(r|D_s) n_{D_u}(D_s) \tag{45}$$

Using Bayes rule and the independence assumption that feature and rating are independent given the data set, the conditional probability $n_{D_u}(f|r, D_i)$ can be simplified as:

$$n_{D_u}(f|r, D_s) = \frac{n_{D_u}(f, r|D_s)}{n_{D_u}(r|D_s)} \propto \frac{n_{D_u}(f|D_s) n_{D_u}(r|D_s)}{n_{D_u}(r|D_s)} = n_{D_u}(f|D_s) \tag{46}$$

The reason this assumption can be made is that users are grouped based on their similar preferences. Therefore, it can be assumed that a feature will have the same rating in user set as well as in the group set. The count functions are then computed as probabilities for every item i belonging to data set D_S:

$$n_{D_u}(f, r, D_s) = \sum_{i \in D_s} \#i * p_{D_u}(f|i) p_{D_u}(r|i) \tag{47}$$

$$n_{D_u}(r, D_s) = \sum_{i \in D_s} \#i * p_{D_u}(r|i) \tag{48}$$

6.3.3 Kullback–Leibler Divergence

To measure the difference between an user set and a group set, Kullback–Leibler (KL) divergence is used, which is a measure of the non-symmetric difference between two probability distributions over the same variable [Kullback 1987]. This measure originates in probability and information theory and has been commonly used in the data mining research field. It is closely related to relative entropy, information divergence and information for discrimination. KL divergence measures the amount of information lost when probability distribution $q(x)$ is used to approximate the probability distribution of $p(x)$, which is generally considered to be the true distribution, where x is the common variable. Both distributions sum up to 1 and are non-negative for all the values of x.

These are some properties of the KL divergence:

- **non-symmetrical** - the measure is non-symmetrical because it is not a typical metric measure. The non-symmetrical property means that the KL divergence from $q(x)$ to $p(x)$ is different from the KL divergence from $p(x)$ to $q(x)$: $KL(p(x)||q(x)) \neq KL(q(x)||p(x))$.
- **triangular inequality –** KL divergence does not need to satisfy triangular inequality
- **non-negativity** – KL divergence is a non-negative measure, $KL(p(x)||q(x)) \geq 0$. It is equal to 0 if and only if the probability distributions are exactly the same: $KL(p(x)\big|\big|q(x)\big) = 0$ if $p(x) = q(x)$.

The general formula for KL divergence is:

$$KL(p(x)\big|\big|q(x)\big) = \sum_{x \in X} p(x) * ln \frac{p(x)}{q(x)} \tag{49}$$

The case when one of the probability distributions contains zero element will now be more elaborate. If the original probability distribution is zero for some x, then the limit of equation (49) is:

$$\lim_{p(x)\to 0} p(x) * ln\frac{p(x)}{q(x)} = 0 \tag{50}$$

Unless all the elements of the probability distribution $p(x)$ are zero, results can still be obtained for KL divergence. If however one of the elements of the approximate distribution is zero, the same limit goes to infinity, causing the whole KL sum to go to infinity:

$$\lim_{q(x)\to \infty} p(x) * ln\frac{p(x)}{q(x)} = \infty \tag{51}$$

This means that if the original distribution predicts that a variable x is possible with some probability and the approximating probability distribution estimates that it is not, these distributions are absolutely different and their KL divergence is therefore zero. Because the distribution is derived from observations, it is wise to take into account the unseen events and use some form of a smoothing function to give a small probability to these events. In this way the probability distribution never contains zero elements.

KL divergence is used to identify the difference between the probability distribution of features within a group of users, where the active user belongs to, and the distribution of features in user set:

$$KL(D_g||D_u) = \sum_f P(f|D_g) * \log_2 \frac{P(f|D_g)}{P(f|D_u)} \tag{52}$$

Then the group probability distribution $p_{D_u}(D_g)$ equals the $KL(D_g||D_u)$ and the user probability distribution is $p_{D_u}(D_u) = 1 - p_{D_u}(D_g)$.

Because the user set is much smaller than the group set, it is more likely that a feature is not observed in the user set. Rather than stating that the two distributions are completely different, smoothing function described in Section 5.1.3 is applied.

Algorithm: Transfer learning for Naïve Bayes

```
Training phase:
Initialize parameters for t=0 using Naïve Bayes Classifier algorithm
        ${p_{D_u}}^0(r)$, ${p_{D_u}}^0(f|r)$
For t=1:1:T
        E-step:
        For $r_k \in k$ and $i \in D_u$
                calculate ${p_{D_u}}^t(r|i)$ using ${p_{D_u}}^{t-1}(r)$, ${p_{D_u}}^{t-1}(f|r)$ according to equation (25)
        end

        M-step:
        For $r_k \in k$
                Calculate ${p_{D_u}}^t(r)$ using ${p_{D_u}}^t(r|i)$ according to equation (26)
        end
        For $f \in F$
                Calculate ${p_{D_u}}^t(f|r)$ using ${p_{D_u}}^t(r|i)$ according to equation (28)
        end
end

Prediction phase:
For $i$ in user test set $D_s$
        Find features describing item $i$
        If $f \cap D_g = \emptyset$
                Exclude $p_{D_s}(f|r)$ from further computation
        end
        Compute $p_{D_s}(r|i)$ according to equation (25)
        Compute $\mathrm{r} = \max_{r \in k}\{p_{D_s}(r|i)\}$
end
```

Chapter 7. Similarity Measures

Similarity measure is a function measuring difference, or distance in case of vector based measures, between two variables. Three basic similarity measures were studied and tested, namely, cosine similarity, Jaccard distance, and Hamming distance. All of them were implemented as standalone content based recommendation engines as well as measures to group users based on similarity of their data sets.

Because the data set used in this research is categorical, similarity measure has to be able to work with this kind of data. Every feature and every feature value is assigned a dimension in vector space model. The value of the vector in this dimension corresponds to the number of times a feature value occurs in the data set. If item similarity is being measured, the count of how many times these feature values appear in content description of item A and item B is used. If user similarity is being measured, user profiles are used to count the number of times feature values appear in user profiles A and B, which are the profiles the comparison is made for. This approach is commonly used in information retrieval and text mining.

When considering application of this approach to TV recommender, similarities between test items need to be computed every time user asks for recommendation. Although the similarity computation is easy to understand, it is costly because it compares the feature values of an item. This is usually a large number. In my work, an item is described by 20 features, where every feature can acquire a number of values. In general the computation cost is $(N * I)^M$, where I is the number of feature values for feature f_n, N is the number of features and M is the set size. If the average number of items in the user training set is 27, the computational cost for this method would be 20^27 = 1.3*10^35 if every feature would have only one value. In reality this number is much higher as some features have more than 10 values. This issue might be addressed using clustering as described in [Krauss 2013]. The issue with data storage however remains. User data needs to be stored along with its original item descriptions. This imposes the requirement for a user device to have sufficient memory and because the data can be accessible in this form, user privacy is another issue to deal with.

If however, in the algorithm similarity is used for user grouping, this can be done upfront and stored as a correlation matrix. The recommendation time is not affected by it.

7.1 Cosine similarity

Cosine similarity is one of the popular approaches for user or item similarity measurement. It measures the similarity of two nonzero vectors A and B by the cosine angle between them, where *d* is the vector dimension. It acquires any number between -1 and 1, where:

- $\cos(0) = 1 \ \rightarrow$ same
- $\cos\left(\frac{\pi}{2}\right) = 0 \ \rightarrow$ dissimilar
- $\cos(\pi) = -1 \rightarrow$ opposite

$$\cos(A, B) = \frac{\sum_d A_d * B_d}{\sqrt{\sum_d A_d^2}\sqrt{\sum_d B_d^2}} \tag{53}$$

If sets are represented as binary vectors, Ochiai coefficient can be obtained [Jackson 1989]. Ochiai coefficient is a powerful similarity measurement and in certain applications outperforms other approaches [Abreu 2008]. This coefficient computes the number of common features to the total number of overall features from both items:

$$OC = \frac{n(A \cap B)}{\sqrt{n(A) * n(B)}} \tag{54}$$

Cosine similarity measure is implemented as content based filtering method to measure similarity of items. Similarity between a test item and a set of items from a user data set is computed. This can be represented as a similarity vector $s = \{s_1, \dots, s_m\}$ where s_m represents how similar the m^{th} item is to the test item. If a user data set has M items, each item has a rating r_k, the rating vector of all items in the user data set is $r = \{r_{k1}, \dots, r_{km}\}$. To estimate the rating, dot product is applied to similarity vector and rating vector divided by the sum over similarity vector:

$$\hat{r} = \frac{\boldsymbol{s}\, x\, \boldsymbol{r}}{\sum_M s_m} = \sum_M \frac{s_m * r_{km}}{\sum_M s_m} = \sum_M W_m * r_{km} \tag{55}$$

where $W_m = \frac{s_m}{\sum_M s_m}$ is the similarity contribution of an item m to a test item.

Similarly, when similarity is applied to users, W_u represents the similarity contribution of a user u to an active user and the rating is estimated as the sum over all users U who rated the test item as r_{ku}:

$$\hat{r} = \sum_U W_u * r_{ku} \tag{56}$$

7.2 Jaccard Distance

It is a statistical measure to determine dissimilarity of two sample sets. Sample set in this work is either an item when predicting rating, or a user profile when measuring user similarity. It is complementary to Jaccard coefficient. It is computed as the ratio between the size of the intersection divided by the size of union.

$$d_j(A,B) = \frac{|A \cup B| - |A \cap B|}{|A \cup B|} \tag{57}$$

Jaccard distance is any number between 0 and 1, where:

- $d_j(A,B) = 1$ → same
- $d_j(A,B) = 0$ → dissimilar

Rating is predicted in the same way as for cosine similarity using equation (56).

7.3 Hamming Distance

It measures the distance between two strings of the same size as the number of different symbols at the same positions of these two strings. In this work, vectors are used as described at the beginning of this section. Therefore, it is the number of positions at which the two vectors have different values. The coefficient is then normalized by the size of the vector.

$$H(A,B) = \frac{|A \cap B|}{size(A)} \tag{58}$$

It is any number between 0 and 1, where:

- $H(A,B) = 1$ → dissimilar
- $H(A,B) = 0$ → same

As for Jaccard distance, rating is then predicted according to equation (56).

Chapter 8. Collaborative Filtering Approach

Two CF methods are employed, namely Slope One and Pearson correlation, to compare user similarity based approaches with content based approaches. These two methods were chosen because they are commonly used CF-based approaches [Ekstrand 2010], [Lemire 2005]. Their popularity is due to the simplicity of implementation and their cost effectiveness.

Note that collaborative filtering recommends items based on user similarity. The term 'active user' is used in this thesis for the user predictions are being made for and the term 'related user' for a user who shares item or items with the active user.

8.1 Slope One and Item Similarity

This approach is widely implemented in recommendation engines and is frequently combined with other CF or CB methods [Krauss 2013], [Gao 2011], [Mi 2012]. The rating of an unseen item is predicted by finding users who rated this item. Then ratings of items which both active and related user rated are compared to determine the similarity between users. Difference in rating behaviour between the two users is computed as the deviation $dev(i_1, i_2)$ between two items rated by an active user and a set of users who rated similar items $r_{u,i1}$ and $r_{u,i2}$, over all the users who rated both items U_{i1i2}.

$$dev(i_1, i_2) = \frac{\sum_{u \in U_{i1i2}} (r_{u,i1} - r_{u,i2})}{|U_{i1i2}|} \tag{59}$$

This difference is then used to predict the rating of the test item. Slope one coefficient of user u and test item j denoted as $SO(u, j)$ is computed as the difference between active user rating $r_{u,i}$ of items from the user set and the rating deviation computed as in equation (59), divided by the number of compared items.

$$SO(u, j) = \frac{\sum_{i \in Ij} \left(r_{u,i} - dev(i, j) \right)}{|I_j|} \tag{60}$$

The resulting Slope one coefficient is then rounded to the closest rating. This approach suffers from data sparsity of user-item matrix. Especially, when training set for one user is small, it is difficult to find enough commonly rated items with other users to make an informative prediction. Because of that, it is beneficial to add content information about the item. Cosine similarity and Ochiai coefficient can be combined with Slope One, which adds weighting to the rating deviation as:

$$dev(i_1, i_2) = \frac{\sum_{u \in U_{i1i2}} \left(r_{u,i_1} - r_{u,i_2}\right)}{\left|U_{i_1 i_2}\right|} * sim(\vec{\imath_1}, \vec{\imath_2}) \tag{61}$$

where $sim(i_1, i_2)$ is the similarity of the two items computed according to equations (53). When two items are the same, the resulting similarity is 1, which does not change the rating deviation. Every other value is smaller than one and reflects the degree of correlation between items. According to [Burke 2007], this method of combining two recommendation methods, slope one as collaborative filtering and cosine similarity as content based method, belongs to the category of mixed hybridization techniques, where cosine similarity serves as a measure of recommendation confidence.

8.2 Pearson Correlation

Pearson correlation is a measure of the linear correlation between two users' ratings. Its value is between +1 and −1. It has a value of 1 when two users have an identical rating pattern, 0 when there is no correlation between users, and −1 if users rate items completely different. The difference between an active user rating for an item from user set $r_{a,i}$ and the average active user rating $\overline{r_a}$ is multiplied with the rating difference of related user $r_{u,i} - \overline{r_u}$. Differences are summed up over the entire related user's set and normalized by the denominator to obtain the Pearson correlation between active and related user P_{au}

$$P_{au} = \frac{\sum_{i \in I}(r_{a,i} - \overline{r_a}) \times (r_{u,i} - \overline{r_u})}{\sqrt{\sum_{i \in I}(r_{a,i} - \overline{r_a})^2 \times \sum_{i \in I}(r_{u,i} - \overline{r_u})^2}} \tag{62}$$

Then the predicted rating for an unseen item p_{aj} is computed as the weighted average of deviations from the related user's mean over all related users:

$$p_{aj} = \overline{r_a} + \frac{\sum_{u \in U}\left(r_{u,i} - \overline{r_u}\right) \times P_{au}}{\sum_{u \in U} P_{au}} \tag{63}$$

Similar to Slope one, Pearson correlation also suffers from the issue of a sparse user-item matrix. Because of that, existing approaches set a threshold for the number of co-rated items and either implement some kind of scaling factor [Herlocker 2002] or switch to another recommendation method [Lekakos 2008]. The typical threshold for co-rated items is 30 or higher. In the data set I used, the average number of rated items per user is 27, therefore applying the threshold would switch to other recommendation approach in most of the cases. However, when content information is added to the computation, better results can be achieved compared with pure Pearson correlation. Cosine item similarity and Ochiai coefficient are implemented as weighting to enrich the rating comparison with the information about an item.

$$P_{au} = \frac{\sum_{i \in I}(r_{a,i} - \overline{r_a}) \times (r_{u,i} - \overline{r_u}) \times \cos(pi, i)}{\sqrt{\sum_{i \in I}(r_{a,i} - \overline{r_a})^2 \times \sum_{i \in I}(r_{u,i} - \overline{r_u})^2}} \tag{64}$$

Computation cost of both methods increase with increasing number of users who rated the item. On the other hand, when not having enough users to compare with the active user, the predictions are less accurate. All these methods fail to make a recommendation when no user has seen and rated the test item, which is the cold start problem typical for CF methods.

Chapter 9. Experiments

The dataset in my experiment contains missing feature values for some items. Moreover, a user set is usually very sparse as many features can acquire a large number of values and many of these appear only once per training set. An algorithm therefore needs to be able to cope with these issues. For the application to HBB TV, a recommendation needs to be done in real time. As user profile is compared to hundreds of items at the time of doing recommendation, the algorithm needs to be computationally efficient. Algorithms comparing item features are not as efficient as algorithms working with model parameters. Also, it is safer and more efficient if instead of a storing raw history of watched items with all their description, the model stores feature probabilities, joint probability tables, or other model parameters.

9.1 Data Set Description and Evaluation Metrics

In my experiments, the rating of an unseen item is predicted. Tests were conducted on the Yahoo Labs movie dataset [Yahoo 2014]. It is a movie database containing 11,915 movies/items with detail descriptions. The training set consists of 211,231 ratings from 7,642 users, where the average number of ratings per user is 27.64 (training and test set combined) and the average number of ratings per item is 17.73. In this research, the set was divided into training and test set in the ratio of 2/3 for training and 1/3 for testing purposes. The items are rated from 1 to 5, where 1 is low preference/dislike and 5 is the highest preference/strong like. Some recommendation systems return recommendation value instead of predicting rating [Ikawa 2010, Uluyagmur], which can be any number in the considered scale. Then, the item with the highest recommendation value is presented to the user. However, as my system uses the rating range from 1 to 5, rating values are normalized to fit into this scale.

The accuracy of recommendation is measured using mean absolute error (MAE), as this measure is commonly implemented for recommendation system based on rating prediction. Another method of measuring rating accuracy by penalizing more severely recommendation errors for items with high rating is also proposed. In program recommendation application, if a rating prediction error is made for items with low rating, the viewer can always decide not to watch the recommended items. On the other hand, if rating prediction error is made for items with high rating, it is not possible for a viewer to decide to watch the items since the items will not be recommended to the viewer in the first place. Undoubtedly this will have a larger negative impact on viewing experience. In other words, the impact of making prediction error is not symmetric for items with low and high rating. As making mistake for highly rated items is more costly than for low rated items, the weighted recommendation error (WRE) is proposed as a new rating accuracy measure to evaluate the performance of rating based recommendation systems:

$$WRE = (r - \hat{r})^{rv} \tag{65}$$

WRE is the rating difference between the true rating r given to an item by the active user and the rating predicted by the recommendation algorithm $\hat{r}$, exponentially penalized by the item's rating value rv.

Several state-of-the-art content based and collaborative filtering based approaches were tested, and collaborative models were enriched with content information to create hybrid recommendation systems and compare their performances with existing algorithms.

9.2 Evaluation

A number of state of the art content and collaborative filtering techniques were examined with focus on methods that can work with categorical values. Some content-based algorithms described in Chapter 5 have been previously implemented as text classifiers, recommendation engines in TV environment or as video on demand recommenders. These techniques are frequency count based techniques such as the algorithms described in in [Ikawa 2010] and [Uluyagmur 2012], cosine similarity applied in [Krauss 2013], canonical weighted sum as an alternative to probabilities when using Naïve Bayes graphical structure described in [Campos 2010], and commonly applied collaborative filtering methods such as slope one and Pearson correlation. Naïve Bayes algorithm is often used as baseline comparison.

More research had been done in the area of text classification compared to TV recommendation. Therefore some techniques from this research area were applied to this research and implemented as a TV recommender. Methods belonging to this group are AODE and transfer learning algorithm using Naïve Bayes classifier which in this thesis was applied on groups of users and the knowledge was transferred to individual user. To my knowledge, PC algorithm had not been previously applied to any of these areas. This application is novel and provides all the benefits required for a recommendation engine in TV environment. Because it is a probabilistic algorithm, it can work with categorical data and can provide insight into user preferences and how different aspects of user profile influences user decision making.

Graphical model based approaches were enhanced with smoothing and missing entries were filled using algorithms described in Chapter 5. These enhancements improved the performance greatly. These methods are denoted as improved approach compared to the original approach.

Table 3 MAE and WRE comparison of state-of-the-art recommendation system approaches.

	Method name	**MAE**	**WRE**	**Average time (second)**
Content based filtering methods	**Improved Naïve Bayes classifier**	1.2995	79.8826	0.0444
	Naïve Bayes classifier	3.5975	861.3422	0.0141
	Canonical Weighted Sum	0.8172	52.6489	0.0080
	Ikawa [Ikawa 2010]	3	556.3143	7.4190e+04
	Uluyagmur [Uluyagmur 2012]	2.7810	684.4952	7.9545e+03
	Item cosine similarity (ICS)	0.9464	61.3272	1.3471
	Jaccard distance	0.9248	56.2325	1.352
	Hamming distance	0.9279	56.2588	1.354
	AODE	1.3091	124.4908	57.67
	Improved AODE	1.1200	88.0168	57.69
	PC algorithm	0.8568	40.0977	24.4733
	Improved PC algorithm	**0.8276**	**32.2557**	25.633
Collaborative filtering methods	**Slope One (SO)**	1.3897	3.8420e+03	0.0138
	Pearson Correlation	1.8804	182.2746	1.3491
Hybrid methods	**SO with ICS**	1.0557	139.3750	1.3609

	Transfer learning baseline comparison	1.0951	60.7923	0.0141
	Transfer learning cosine similarity	11.9522	0.2376	0.0141
	Transfer learning Ochiai coefficient	17.1674	0.3365	0.0141
	Transfer learning Jaccard distance	12.7869	0.2287	0.0141

Table 3 summarizes the performance of several content, as well as collaborative filtering based methods, and their hybrid combinations, using MAE and weighted recommendation error (WRE) as the performance metrics. Because application to TV environment is considered, a recommendation needs to be made within seconds and algorithm needs to run through hundreds of items. Therefore the time aspect is very crucial in this application. While some methods run very fast, others exhibit high latency and are computationally costly. Methods that involve counting the number of occurrences of a term or a feature in the whole dataset require a lot of memory which slows down the whole recommendation process. This is the case for methods proposed in [Ikawa 2010] and [Uluyagmur 2012]. These methods require original information about watching history to be stored in the user profile. This is not a preferred format of storing data about a user. Due to user privacy, it is preferred to store user information as parameters. Similarity measure techniques also require to store complete user data. These methods, however, run much faster than methods based on frequency count with significantly lower error rates.

Graphical model based methods generally have low error rates. Approaches based on Naïve Bayes, including transfer learning, are the fastest methods and make rating prediction in just a couple of milliseconds. Transfer learning algorithm takes time to group users and transfer knowledge from a group to individual user. Once this is done, the prediction time is the same as for Naïve Bayes classifier. AODE makes predictions in about 57 seconds and PC algorithm in about 25 seconds. It can be noticed that the improvements made to these algorithms does not contribute to the computational time while significantly reducing the error rates. The most significant improvement is noted by Naïve Bayes classifier reducing its WRE by almost 800 points. AODE reduced its WRE by 34 points while PC algorithm by about 8 points. PC algorithm is the best performing algorithm even without any improvements. Note that because WRE is a weighted measure, even small decrease of the error rate reflects a noticeable improvement of the methods. The decrease by 8 points not only means higher accuracy of the rating prediction power of PC algorithm but it also makes this algorithm perform better than the second best performing algorithm, which is canonical weighted sum, by 20 points. This is a significant improvement in performance compared to known state-of-the-art approaches.

Transfer learning approach outperforms all of the other considered methods. PC algorithm is the best performing CB method. The WRE of transfer learning for cosine similarity is lower by 62.5% compared to improved PC algorithm. Compared to the baseline comparison when all users are in one group, this method brings an improvement in WRE by 80% for cosine similarity.

Collaborative filtering methods return results very fast, but their performance is poor. These methods often fail to make a recommendation due to the sparse user matrix. There is simply not enough evidence for these methods to find a user with enough commonly rated items as an active user. Because of this, slope one method is the worst performing algorithm. Adding content information to slope one method and so creating a hybrid recommendation engine significantly improved performance of this method but it still could not compare to the performance of content-based filtering methods.

The failure rate of each of the tested methods was further examined because failure to make a recommendation reflects poor performance of an algorithm. Both similarity measure methods and collaborative filtering methods have issues with lack of evidence. While for collaborative filtering methods, it is not enough users with the same rated content in their sets. For similarity measure methods, it is lack of content information. This results in the failure to make prediction ranging from 1 to 4% of cases. Naïve Bayes classifier was originally the algorithm with the highest number of failures in almost half of the cases. Adding smoothing and missing value estimation removed this issue completely and its improved version has now zero failure rate. The other two methods that are always able to make predictions are AODE and transfer learning algorithm. Canonical weighted sum algorithm has a small percentage of failure to make predictions, below 1%. Although the failure rate is very small, compared to the best performing algorithm, which is the improved PC algorithm, it is 100 times higher. The failure rate of PC algorithm is negligible, keeping this algorithm in the top. Algorithm described in

[Uluyagmur 2012] has the highest failure rate, over 14% which is almost every 6th item. Another algorithm counting frequency of a feature occurrence is [Ikawa 2010]. Although these two algorithms perform with similar WRE and MAE, Ikawa algorithm has zero failure rate. On the other hand it has many cases when the difference between true and estimated rating is 4 resulting in high error rates.

Table 4 Percentage of failure to make recommendation

Method name	**Failure rate**
Improved Naïve Bayes	0%
Naïve Bayes classifier	44.6893%
Canonical Weighted Sum	0.8320%
Ikawa [Ikawa 2010]	0%
Uluyagmur [Uluyagmur 2012]	14.2857%
Item cosine similarity (ICS)	2.8033%
Jaccard distance	2.7572%
Hamming distance	2.7572%
PC algorithm	0.0096%
Improved PC algorithm	0.0096%
AODE	0%
Improved AODE	0%
Slope One (SO)	2.9122%
Pearson Correlation	1.3728%
SO with ICS	3.8565%
Transfer learning	0%

The best performing methods will now be further examined. For this, algorithms performing with WRE below 100 are selected to closer elaborate on their performance. These methods are Improved Naïve Bayes classifier, cannonical weighted sum, cosine similarity, Jaccard distance, Hamming distance, improved AODE and improved PC algorithm. The same set of tests on methods performing with WRE above the 100 threshold will also be discussed. These methods are Ikawa, Uluyagmur, and collaborative filtering methods slope one and Pearson correlation.

9.2.1 Best performing methods comparison

First, the rating predictions and difference from the true rating is measured. These results are concluded in Figure 13. Rating difference is 0 when the predicted rating equals the true rating and 5 when the algorithm fails to make predictions. The focus is on the first two columns where the difference between the true and predicted rating is 0 and 1, meaning an algorithm makes very accurate predictions. Methods like canonical weighted sum and PC algorithm have the highest number of accurate predictions resulting in zero difference. Algorithms based on similarity measure, cosine similarity, Jaccard distance and Hamming distance, have the most predictions where the difference between an estimation and user rating is 1. Methods assuming an independence in graph like Naïve Bayes classifier and AODE have the lowest number of accurate predictions, while having significantly higher number of 3 and 4 differences between the true and predicted rating.

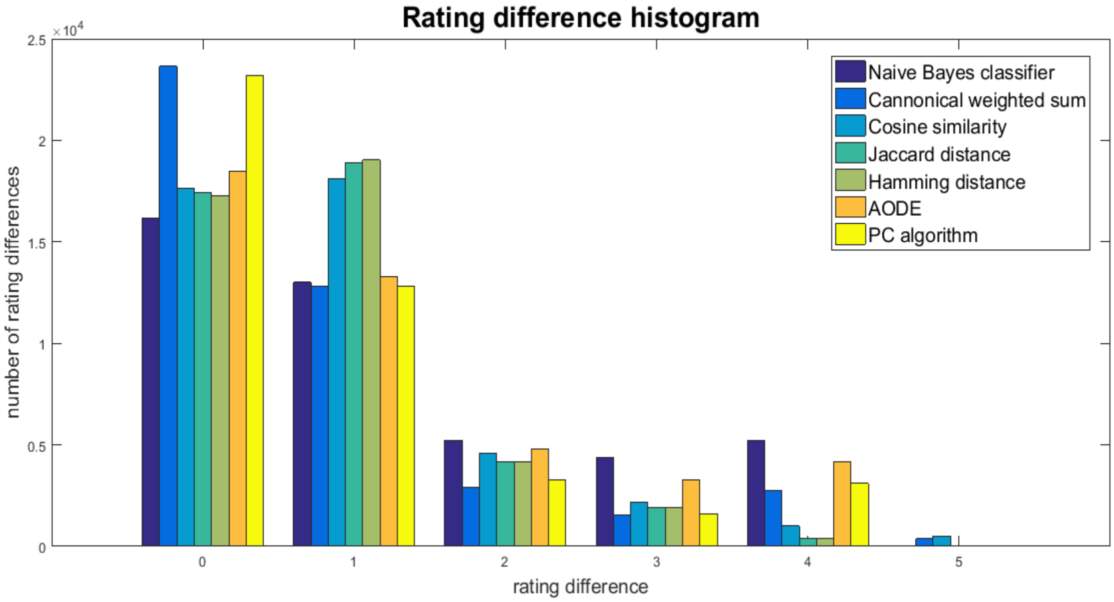

Fig. 13 Rating difference comparison for methods with WRE below 100.

Secondly, accuracy per rating was measured. This is an important measure, because items with the highest rating are recommended. Therefore making mistakes for highly rated items is more costly than for low rated items, as was pointed out earlier. Figure 14 shows the cumulative rating difference for particular rating value. In this graph, the algorithm performs better when the bar shows a low error for rating values 5 and 4. Algorithms making the most mistakes for high rating values are improved Naïve Bayes and AODE algorithms. Similarity measure based algorithms all performs very similarly having slightly more errors for the rating value 5 than canonical weighted sum. Improved PC algorithm performs significantly better than the other tested methods reflecting as a very low count of errors for high rating values.

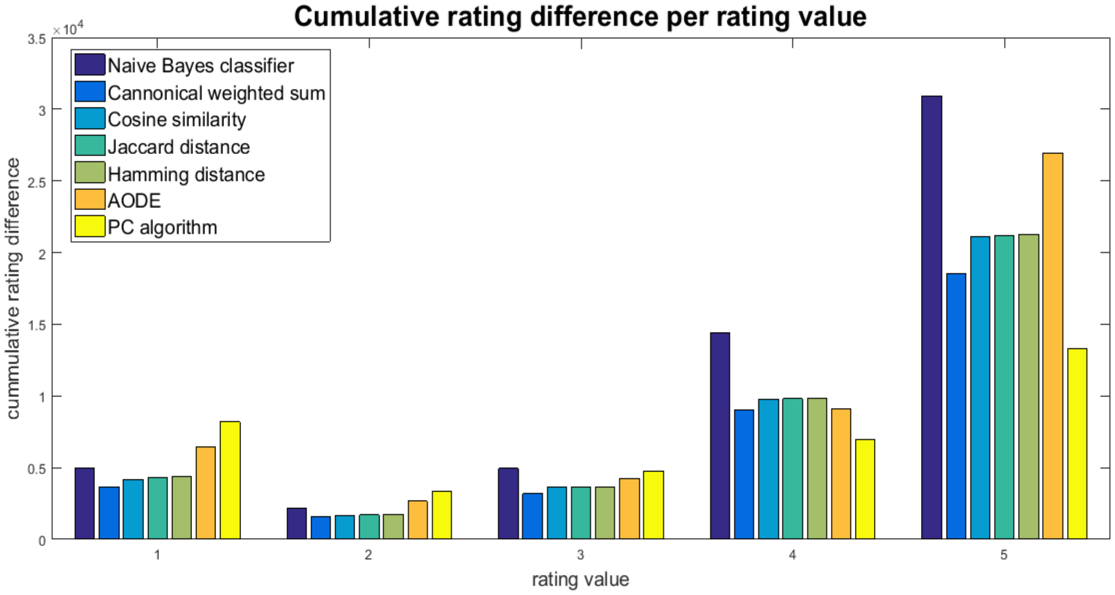

Fig. 14 Cumulative rating difference comparison for methods with WRE below 100.

The last aspect of accuracy measure is the weighted rating error. It is similar to the previous measure with added penalties to rating errors according to the true rating value. Because the focus is again on high rating values, higher penalties are given to these ratings. Figure 15 uses the same formula as WRE and provides more insight to a particular algorithm's performance by breaking results down to rating values. As in the graph before, columns with rating values 5 and 4 are columns of interest. The algorithms with the lowest accumulated error for these rating values are the best performing ones. The figure shows that AODE together with improved Naïve Bayes classifier accumulates the highest errors for all rating values. Cosine similarity in this test results is the second best performing algorithm with lower error than the other similarity measures. Canonical weighted sum has about the same accumulated error as Jaccard and Hamming distance. Again, PC algorithm has the lowest accumulated error among all other considered approaches.

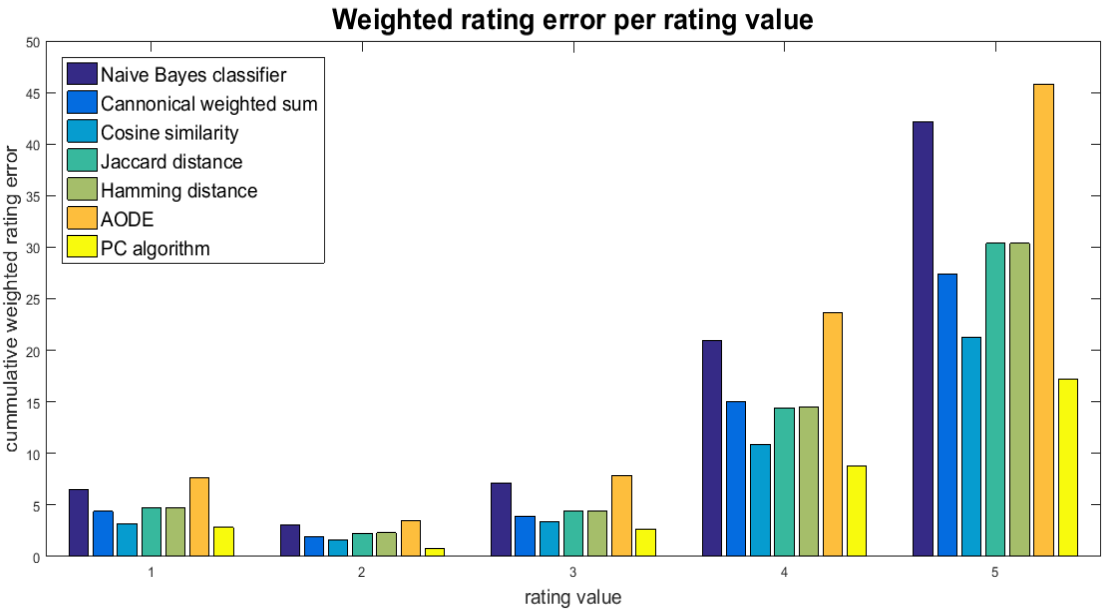

Fig. 15 Cumulative weighted error per rating value comparison for methods with WRE below 100.

9.2.2 Worst performing methods comparison

The same sets of test were applied to algorithms with WRE above 100. In this section, it is demonstrated why developing WRE was so important for this research and how other tests would be insufficient to draw conclusions about an algorithm performance.

In Figure 16 the number of items with rating error ranging from 0 to 5 is observed as explained in previous section. Here Slope one method with item cosine similarity is the best performing method with the highest number of items with rating difference 0 and 1. The second best method appears to be slope one method and other methods such as Pearson correlation, Ikawa and Uluyagmur fall far behind these two. This test corresponds to the results of MAE and as it will be shown in further graphical illustrations, this measure is not sufficient.

This figure also shows failure to make predictions and explains the performance of Ikawa algorithm. As shown in the Table 4, Uluyagmur algorithm has the highest rate of failures whereas Ikawa has never failed to make a recommendation. This is because this method uses ranking of items. This ranking needs to be further mapped to the range of rating values. Because of this, Ikawa algorithm always produces a recommendation but as shown in the Figure 16, the rating estimation is often far from the true user rating resulting in many items having the rating difference 4.

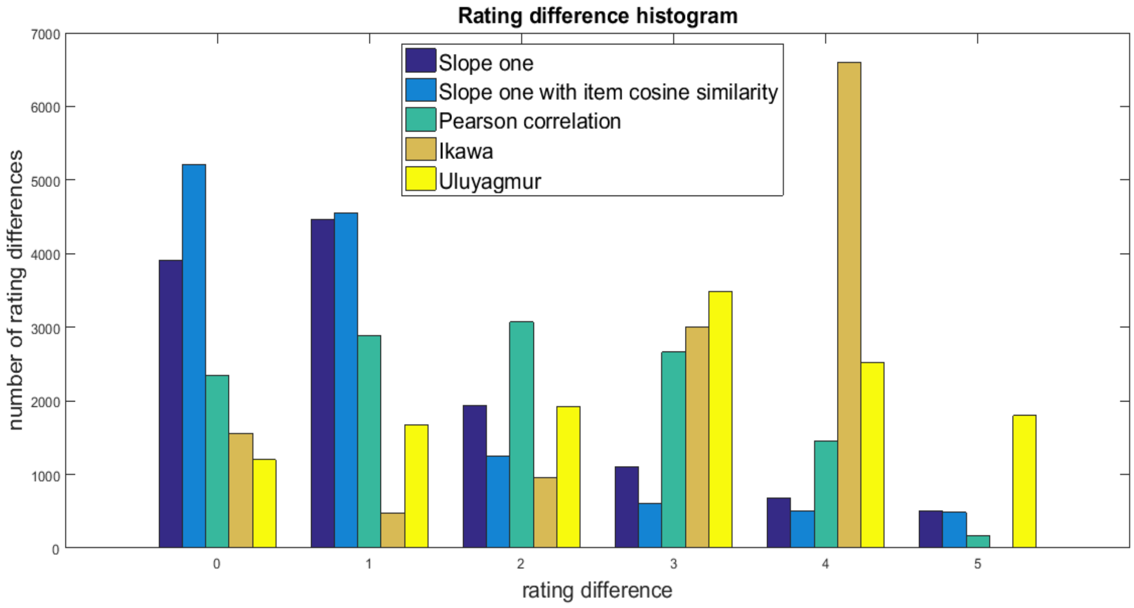

Fig. 16 Rating difference comparison for methods with WRE above 100.

The second test breaks the rating differences down and shows the accumulated error for particular rating value in Figure 17. The aim is to have the lowest accumulated error for high rating values. When it comes to the accuracy comparison, results are similar to the previous figure showing slope one with item cosine similarity as the best performing method and simple slope one as the second best with the rest of the methods having significantly higher accumulated rating differences for the rating value 5.

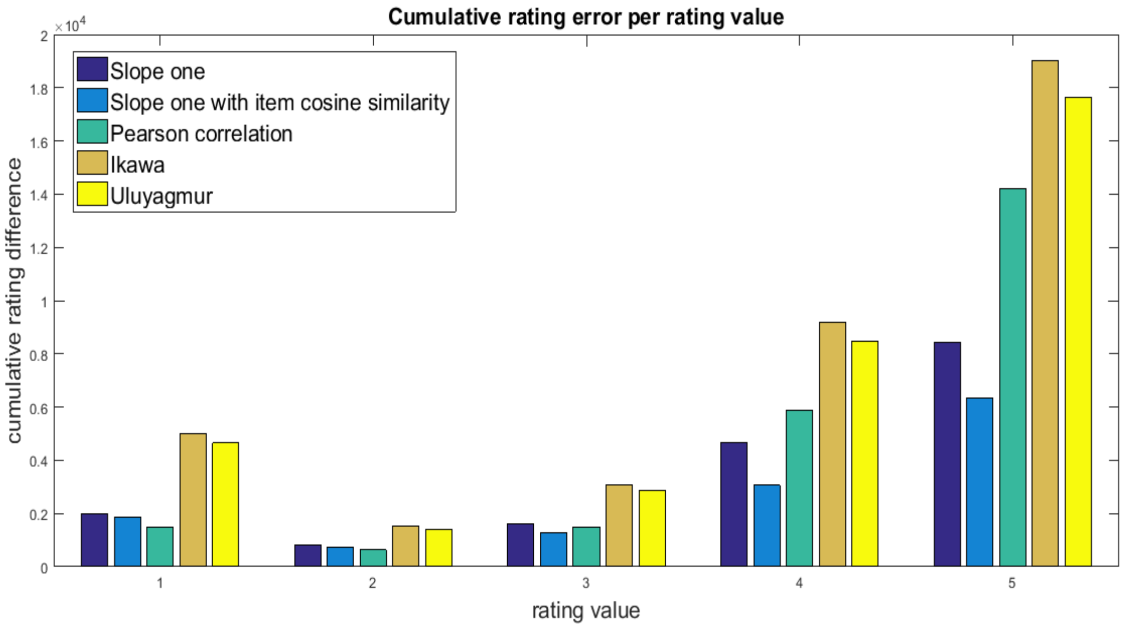

Fig. 17 Cumulative rating difference comparison for methods with WRE above 100.

The last test shown in Figure 18 demonstrates why the previous tests were insufficient to make conclusions about an algorithm's performance. While in previous figures, slope one algorithm seemed to be the second best performing method, this graph shows its poor performance when it comes to predict rating for high rating values. This algorithm often makes mistakes or fails to make a recommendation for the items where user rated highly. This reflects as high WRE in Table 3. Adding item cosine similarity to this method significantly improves its prediction accuracy and this hybrid combination is the best performing method among methods considered in this section. Pearson correlation is the second best performing method and this is because this method also considers how user rates items on average and rating of other users is added as weighted average of deviations from the related user's mean according to equation (64).

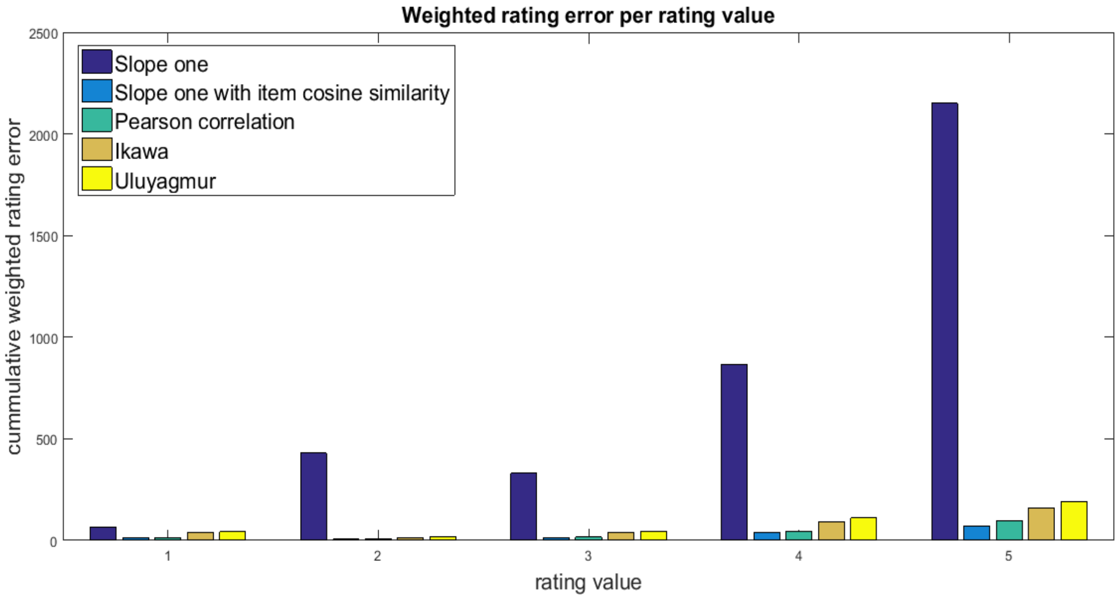

Fig. 18 Cumulative weighted error per rating value comparison for methods with WRE below 100.

Note that the recommendation engine design makes recommendations based on the highest predicted rating. Therefore MAE measure might be a sufficient performance measure for other applications, but in this case WRE provides the necessary insight into the algorithm ability to predict rating for high rating values by penalizing the rating error according to the rating value.

9.2.3 Comparison of original algorithms and their improved version

In this section, improvements done to Naïve Bayes classifier, AODE and PC algorithm are closely examined and their impact on method performances is demonstrated.

Naïve Bayes classifier was improved with missing value estimation and smoothing. The smoothing algorithm made the biggest difference in this case because lots of predictions were lost due to zero conditional probabilities entering equation (12). AODE algorithm described in [Webb 2005] had smoothing already implemented therefore the improvement for this method is purely from missing value estimation. PC algorithm does not involve smoothing and only benefits from missing value estimation. Smoothing can however be added and this can be a task for further research.

The first test shows the number of rating differences regardless of the true rating value. Figure 19 demonstrates the massive improvement in Naïve Bayes classifier when smoothing is added. Originally this method had high rate of failures to make a recommendation due to zero conditional probabilities entering the algorithm. For the classifier to fail to make predictions, only one zero conditional probability is needed as the whole rating conditional probability for the given set of features would then be zero. Considering there are 16 features describing an item, the chance that one of them has zero conditional probability is high. Therefore adding smoothing algorithm was crucial for this method.

AODE algorithm did not exhibit failures in making rating predictions so adding missing value imputation made no difference. PC algorithm had a small percentage of failure and this change did not have a positive impact (see Table 3) because this happened only for a small number of items this can be seen only when the graph is enlarged.

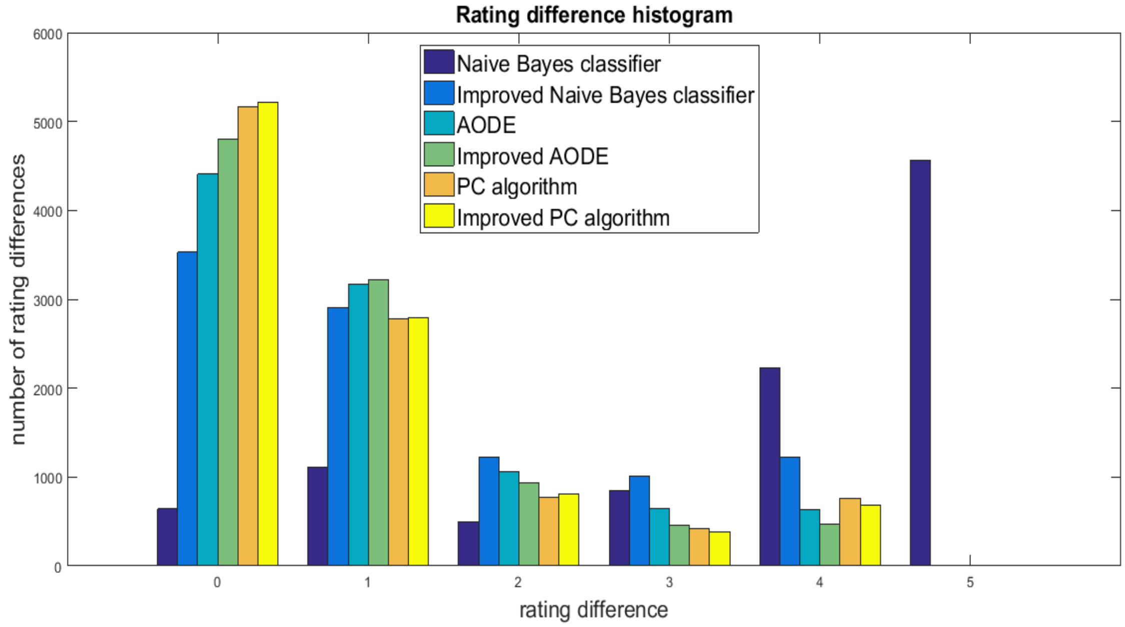

Fig. 19 Rating difference comparison for original and improved algorithms.

Looking at graph showing accumulated error for individual rating values in Figure 20, the results indicate that Naïve Bayes classifier made a massive improvement in rating prediction accuracy for all rating values and especially for high rating values. The AODE method also decreased in accumulated error for rating values 5 and 4. The original PC algorithm was already performing very well and exhibits the least decrease in accumulated error.

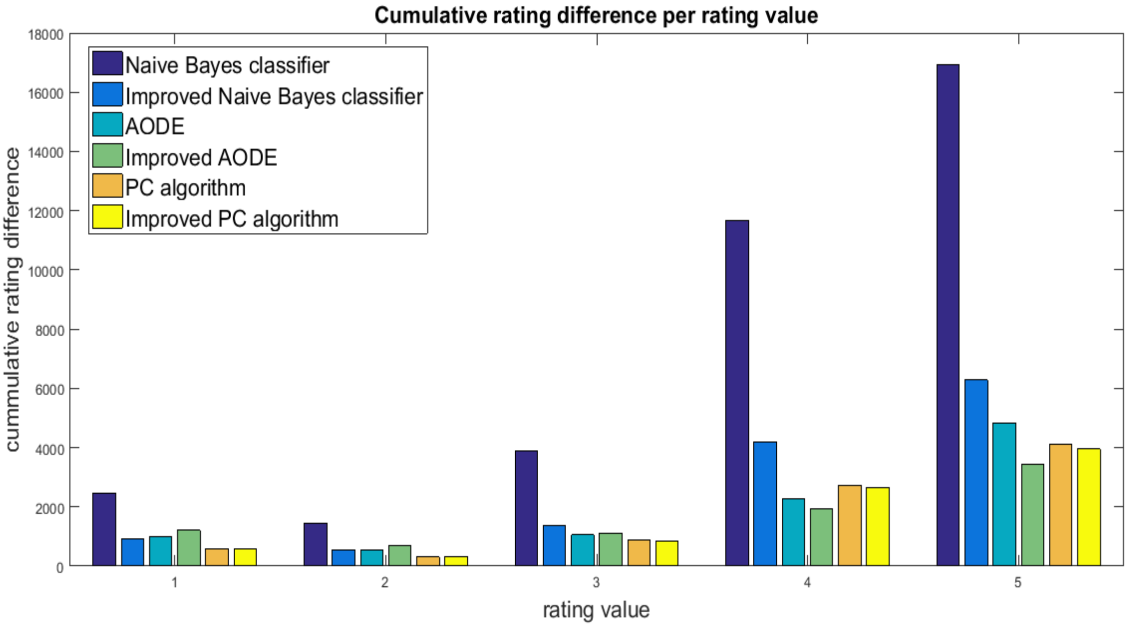

Fig. 20 Cumulative rating difference comparison for original and improved algorithms.

The graph in Figure 21 shows even greater impact of added improvements to all the methods. Especially the Naïve Bayes classifier exhibits decrease in accumulated weighted error resulting in a significant decrease of WRE in Table 3. The error of AODE algorithm for high rating values 4 and 5 decreased to half compared to the original algorithm.

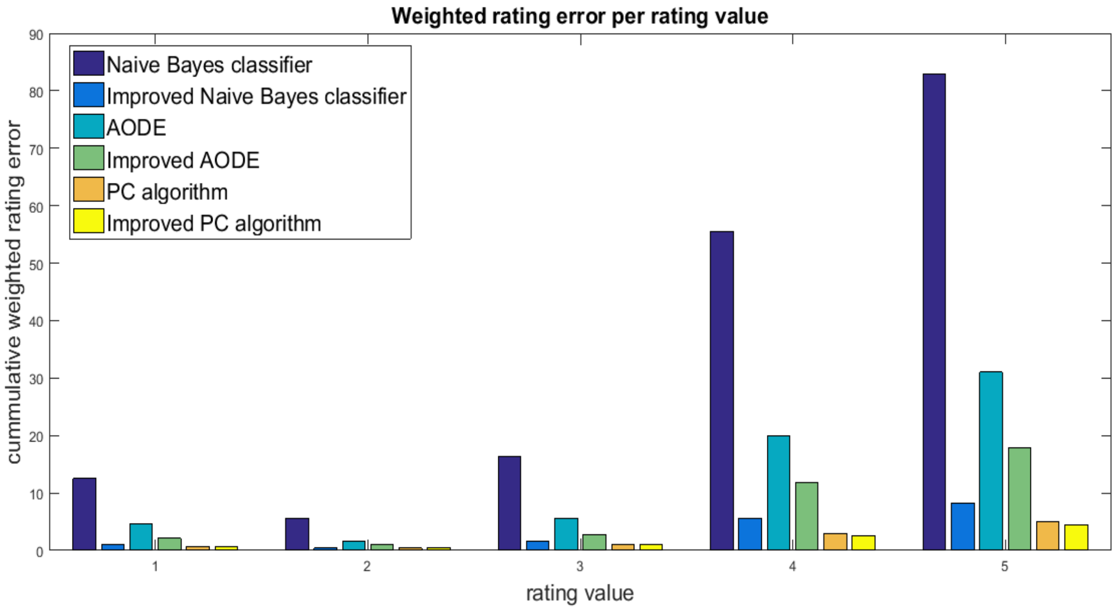

Fig. 21 Cumulative weighted error per rating value comparison for original and improved algorithms.

9.2.4 Transfer learning similarity matric comparison

Transfer learning algorithm was applied to Naïve Bayes model as described in section 6.3 and this algorithm was tested using three similarity measure techniques for grouping users. These techniques are cosine similarity, Ochiai coefficient and Jaccard distance. This approach never fails to make predictions for any of the similarity measures as previously stated in Table 4.

The baseline comparison when all users were grouped together is shown in Table 3 resulting in 60.7923 WRE and 1.0951 MAE. Rating errors for different settings of threshold will now be compared. The higher the threshold is, the finer is the user grouping resulting in higher number of groups with smaller number of users. The WRE and MAE results stated in the tables below are averaged through all user groups created with the respective threshold. Tests were performed on 831 users with equally distributed data set sizes.

Table 5 Threshold comparison Cosine similarity

Threshold	Number of groups	WRE	MAE
0.8	2	60.0037	1.0590
0.99	8	56.8322	1.0671
0.996	18	47.6217	1.0152
0.999	51	18.8903	0.4233
0.9999	148	11.9522	0.2376

Table 6 Threshold comparison Ochiai coefficient

Threshold	Number of groups	WRE	MAE
0.7	2	60.4439	1.0786
0.9	6	45.5051	0.9444
0.95	16	43.8499	0.8857
0.99	66	24.0685	0.5069
0.995	111	17.1674	0.3365

Table 7 Threshold comparison Jaccard distance

Threshold	Number of groups	WRE	MAE
0.7	2	54.3053	1.0837
0.2	9	37.6771	0.8116
0.15	18	36.7286	0.6437
0.09	97	20.7219	0.3761
0.05	136	12.7869	0.2287

The error rates are decreasing for each of the considered similarity measure as the user grouping is refined. The best results are achieved for number of user groups 100 and higher.

This does not necessary mean that users were distributed in equally sized groups. For instance the Cosine similarity for threshold 0.9999 contains many singular groups and groups of different sizes ranging from 22 to 4 users.

In the graphs below, the threshold that yields the lowest WRE was selected for each technique. For the cosine similarity this threshold is 0.9999 with WRE of 11.9522, Ochiai coefficient with threshold of 0.995 and WRE 17.1674, and Jaccard distance with threshold of 0.05 and WRE 12.7869.

In Figure 22, Cosine similarity and Jaccard distance appear to perform similar. Further diagrams however provide more insights in their performance and a distinction between the best performing method will be more apparent. This figure shows that there are no failures to make a recommendation in this approach as there are no rating differences of value 5.

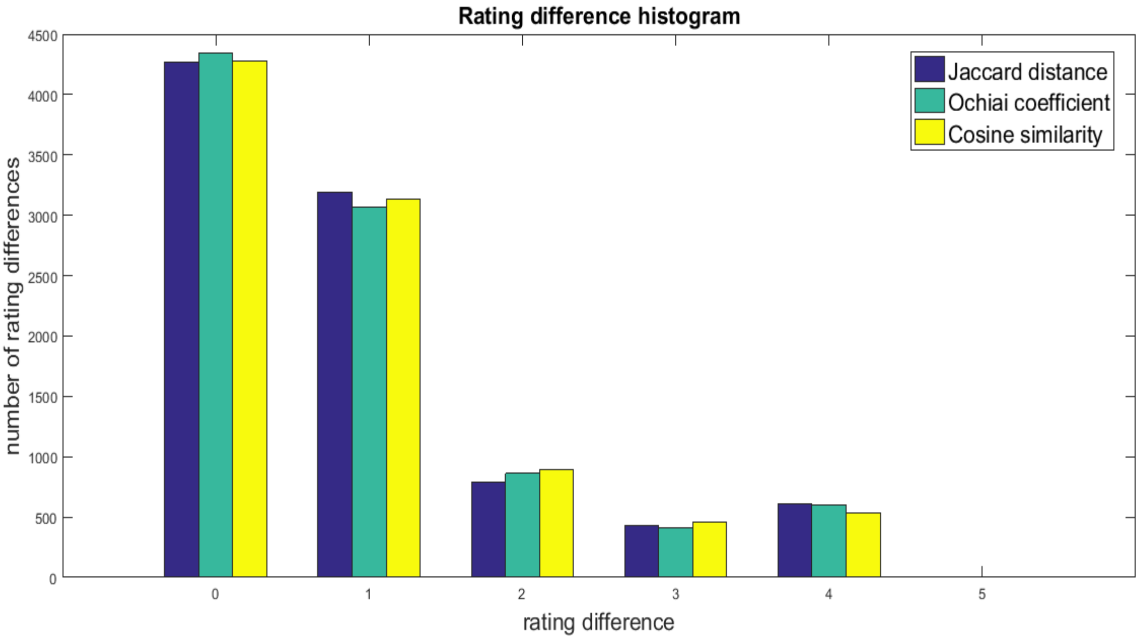

Fig. 22 Rating difference comparison for the best performing transfer learning methods.

Figure 23 shows the cumulative sum of rating difference for each rating value. Here Ochiai coefficient appears to be the best performing method among the three approaches with the lowest bar for the rating value 5 and second lowest bar for the rating value 4. Cosine similarity appears to be second and Jaccard distance third in performance. This graph however does not take into account the rating difference in relation to rating value.

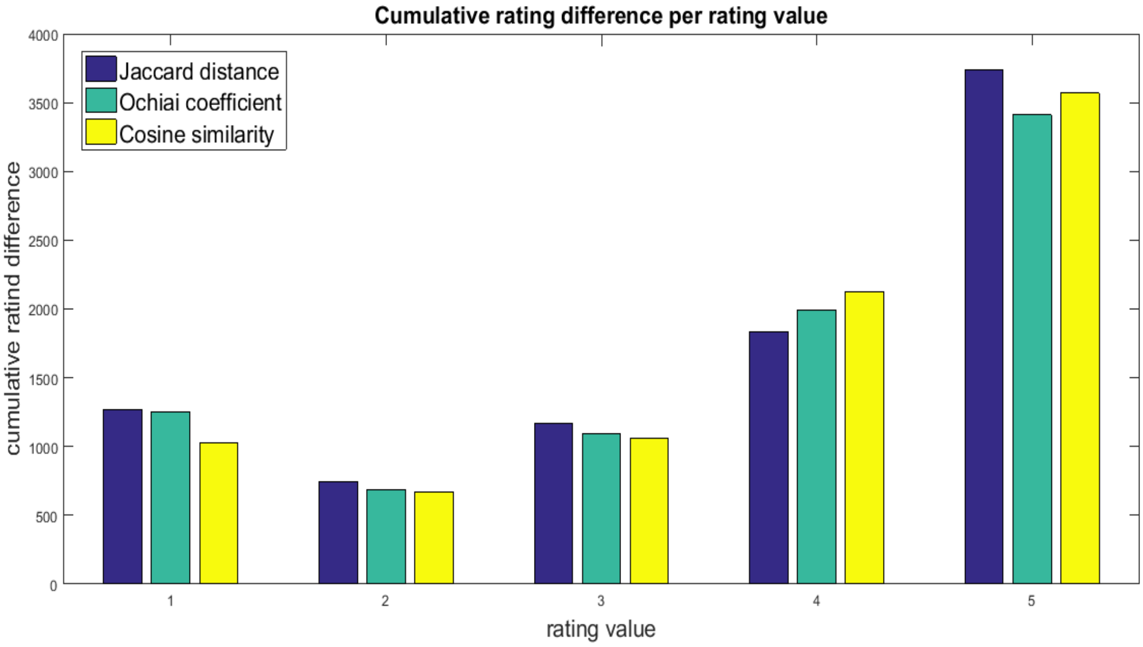

Fig. 23 Cumulative rating difference comparison for the best performing transfer learning methods.

In Figure 24, it is observed that WRE and cosine similarity has clearly the lowest cumulative rating error for the highest rating value, making this method the best performing among the similarity measures for transfer learning. Here, Ochiai coefficient and Jaccard distance perform about the same for rating value 5. For rating value 4, Ochiai has the lowest cumulative weighted rating error while Jaccard distance and cosine similarity result in similar error rates.

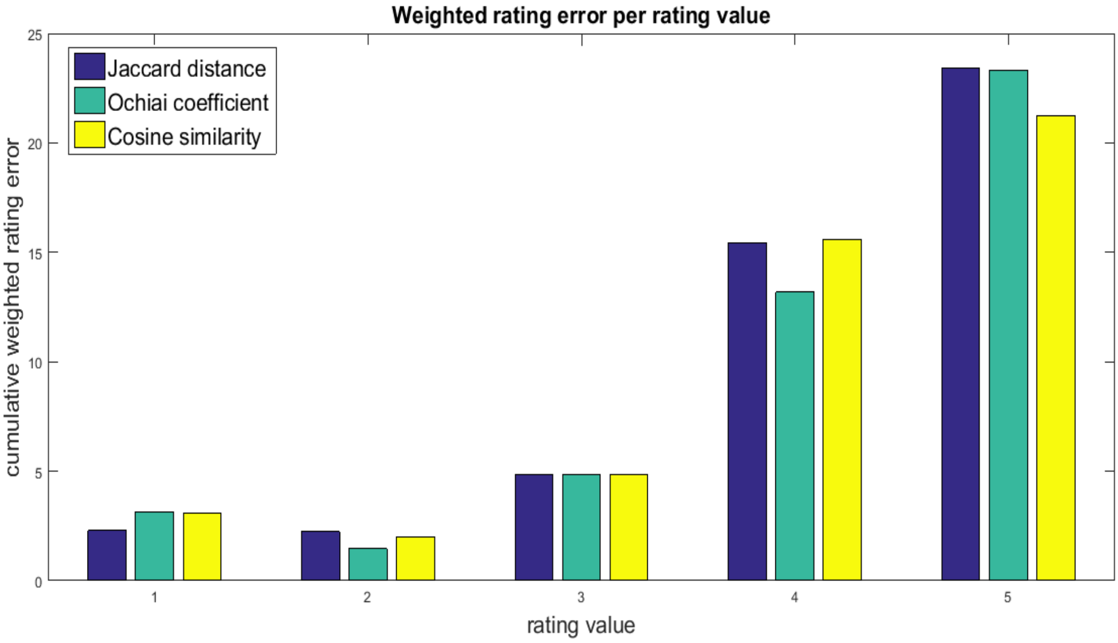

Fig. 24 Cumulative weighted error per rating value for the best performing transfer learning methods.

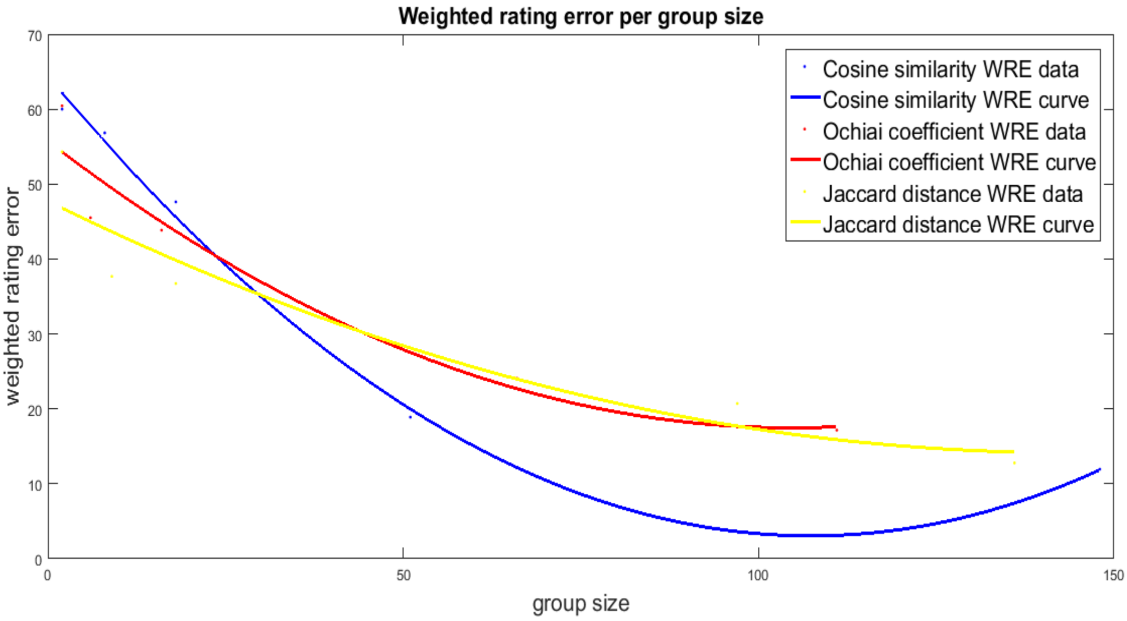

Fig. 25 Second degree polynomial fit of the weighted rating error per group size for three similarity measures

According to Tables 5 to 7, the WRE appears to decrease as the number of groups is higher. Second degree polynomial curves are used to capture the evolution of WRE for each similarity measure. Figure 25 shows that cosine similarity has the steepest decline and reaches low WRE quicker than other measures.

Chapter 10. Conclusion

This thesis provides an overview of recommendation engine techniques applied in TV environment. TV stream is a source of rich information about the transmitted content and this thesis focuses on using this information to draw conclusions about user preferences and make recommendations accordingly. Research gaps in designing recommendation system for TV environment using content information and making recommendation according to user rating were identified. The research focuses on model design using categorical data describing TV content. This data can be extracted from a TV service provider. HBB TV standard provides an excellent platform for implementing applications such as recommendation engine. The environment of HBB TV was described as well as the implementation of an recommendation engine running as an application on HBB terminals. This thesis also elaborates on the issue of data collection and unification from different data sources.

Basic approaches of recommendation systems were described with the focus on content-based filtering techniques, more specifically on graphical models. Graphical models were chosen due to their ability to handle categorical data, which is the data type extracted from TV service provider. Graphical models provide deep insight into user data and the relationships between item’s features and user rating. The thesis elaborates on methods such as Naïve Bayes classifier, AODE algorithm and PC algorithm in detail. Because the data extracted from a TV service provider are typically sparse and contain missing entries, a smoothing algorithm and missing value imputation algorithm was implemented. These additions improved the accuracy of the original methods.

As a comparison, state-of-the-art algorithms were chosen such as canonical weighted sum, similarity measures, and approaches designed for TV recommenders described in [Ikawa 2010] and [Uluyagmur 2012], and two collaborative filtering methods, namely slope on and Pearson correlation.

Novel approach of transfer learning was also studied and the method of inductive transfer approach was described in detail. Transfer learning for Naïve Bayes has previously been implemented in text classification to transfer knowledge from labelled documents to unlabelled datasets. The thesis builds on this model and creates a model grouping users with similar interests as determined by their training data sets using a similarity measure. This approach was tested for three similarity measures, i.e. cosine similarity, Ochiai coefficient and Jaccard distance. The best threshold for each similarity measure to create users groups yielding the lowest error rates was found. EM algorithm was applied to transfer knowledge once the user groups were formed. The similarity between a user set and a group set was determined by Kullback-Leiber divergence.

The recommendation is based on the highest estimated rating. Therefore in evaluation phase the focus is on the difference between the true user rating and the estimated rating. A new measure called weighted rating error (WRE) was developed to determine the accuracy of tested algorithms as commonly used MAE measure was shown to be insufficient. Approaches using content information, except methods described in [Ikawa 2010] and [Uluyagmur 2012], outperformed collaborative filtering methods of slope one and Pearson correlation. The best results were achieved with graphical models. They showed the best ability to estimate a user rating and make recommendation based on the highest estimated rating. They also have low computation time, and usually do not fail to make a recommendation and do not suffer from the sparse user-item or the item-feature matrix problem. According to my research, PC algorithm is the best method among the considered content and collaborative filtering methods to build a recommendation engine in TV environment. It has high accuracy, low computation time and rarely fails to make recommendations.

Special attention was payed to transfer learning methods and multiple tests for different threshold settings were performed. Transfer learning approach significantly outperformed all the other approaches for all considered similarity measures. Compared to the best performing content-based filtering method, transfer learning reduced the weighted rating error by 62.5%. This approach has many other advantages. It never fails to make a recommendation and once the users are grouped, the computational time is very low. This approach is therefore the best choice and is suitable for application in TV environment. It also has the potential to overcome the drawback of typical content-based filtering method. As users are grouped according to similar interests, the training data set of user is enriched by more items. This overcomes the issue of lack of novelty and brings serendipity in recommendation. It also proved to be robust to the missing data issue and the user-item as well as the feature-item sparsity problem.

In summary, this thesis provides a detailed comparative study of state-of-the-art recommendation methods in TV environment. Graphical model based approaches became the centre of attention as they are the most suitable for this application. These approaches were extended making them robust to missing data and data sparsity. A novel approach using the technique of transfer learning and combined similarity measures and graphical models to created powerful prediction algorithm was developed. My experimental results show a significant decrease in error and prove that the design is suitable for application in TV environment.

References

[Abreu 2008] R. Abreu, P. Zoeteweij, A.J.C. van Gemund: An Observation-based Model for Fault Localization. In Proceedings of the 6th Workshop on Dynamic Analysis (WODA'08), colocated with the International Symposium on Software Testing and Analysis (ISSTA'08), Seattle, WA, USA, July 2008, pages 64-70. ACM Press, 2008.

[Amolochitis 2014] E. Amolochitis, I. T. Christou, Z. H. Tan: Implementing a Commercial-Strength Parallel Hybrid Movie Recommendation Engine. In IEEE Intelligent Systems, Vol. 29, Issue 2, Mar.-Apr. 2014, pp 92 – 96, DOI: 10.1109/MIS.2014.23.

[Baba 2004] K. Baba, R. Shibata, M. Sibuya: Partial correlation and conditional correlation as measures of conditional independence. In Australian and New Zealand Journal of Statistics, Vol. 46, No. 4, 2004, pp. 657–664, DOI:10.1111/j.1467-842X.2004.00360.x.

[Barragas-Martinez 2010] A. B. Barragas-Martinez, E. Costa-Montenegroa, J. C. Burguilloa, M. Rey-Lópezb, F. A. Mikic-Fontea, A. Peleteiro: A Hybrid Content-based and Item-based Collaborative Filtering Approach to Recommend TV Programs Enhanced with Singular Value Decomposition. In Information Sciences, Elsevier Inc., 2010, doi: 10.1016/j.ins.2010.07.024.

[Baxter 2000] J. Baxter: A model of inductive bias learning. In Journal of Artificial Intelligence Research, Vol 12, 2000, pp. 149–198, February 2000, DOI: 10.1613/jair.731.

[Bell 2007] R. M. Bell, Y. Koren, and C. Volinsky: Modeling Relationships at Multiple Scales to improve Accuracy of Large Recommender Systems. In Proceedings of the 13th ACM Int. Conference on Knowledge Discovery and Data Mining (KDD’07), ACM press, 2007, pp. 95-104.

[Bellekens 2011] P. Bellekens, K. Sluijs, G. J. Houben, L. Aroyo: On-the-fly Data Integration for Personalized Television Recommender Systems. In Web Engineering, 2008. ICWE '08, 14-18 July 2008, DOI: 10.1109/ICWE.2008.20.

[Bobadilla 2012] J. Bobadilla, F. Ortega, A. Hernando, J. Bernal: A collaborative filtering approach to mitigate the new user cold start problem. In Knowledge-Based Systems, Elsevier, Vol 26, February 2012, pp. 225–238, DOI:10.1016/j.knosys.2011.07.021.

[Burke 2007] R. Burke: Hybrid Web Recommender Systems. In The Adaptive Web, LNCS 4321, Springer-Verlag Berlin, 2007, pp.377-408, DOI: 10.1007/978-3-540-72079-9_12.

[Campos 2010] L. M. de Campos, J. M. Fernández-Luna, J. F. Huete, M. A. Rueda-Morales: Combining Content-base and Collaborative Recommendations: A Hybrid Approach Base on Bayesian Networks. In International Journal of Approximate Reasoning, Volume 51, Issue 7, Elsevier, September 2010, pp. 785–799, DOI:10.1016/j.ijar.2010.04.001.

[Dai 1997] H. Dai, K. B. Korb, C. S. Wallace, X. Wu: A study of casual discovery with weak links and small samples. In IJCAI97 – Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence, August 1997, pp. 1304–1309, ISBN: 1-555860-480-4.

[Dai 2007] W. Dai, G. R. Xue, Q. Yang, Y. Yu: Transferring Naïve Bayes Classifiers for Text Classification. In AAAI'07 Proceedings of the 22nd national conference on Artificial intelligence, Vol. 1, pp. 540-545, ISBN: 978-1-57735-323-2.

[Dempster 1977] P. Dempster, N. M. Laird D. B. Rubin: Maximum likelihood from incomplete data via the EM algorithm. In Journal of the Royal Statistical Society, Series B, Vol. 39 No. 1, November 1977, pp. 1–38.

[Dobrowsky 2013] M. Dabrowski, J. Gromada, H. Moustafa, J.Forestier: A context-aware architecture for IPTV services personalization. In Journal of Internet Services and Information Security (JISIS), Vol. 3, No. 1/2, pp. 49-70.

[Ekstrand 2010] M. D. Ekstrand, J. T. Riedl, J. A. Konstan: Collaborative Filtering Recommender Systems. In Human–Computer Interaction, Vol. 4, No. 2, 2010, pp. 81–173, DOI: 10.1561/1100000009.

[ETSI TS 2008] ETSI TS 182 028, Telecommunications and Internet Converged Services and Protocols for Advanced Networking (TISPAN); NGN Integrated IPTV Subsystem Architecture, 2008.

[ETSI TS 2010] ETSI TS 102 809 (V1.1.1), Digital Vieo Broadcasting (DVB); Signalling and carriage of interactive application and services in Hybrid Broadcast/Broadband environments, 2010.

[ETSI TS 2012] ETSI TS 102 796 V1.2.1 (2012-11), Technical Specification, Hybrid broadcast broadband TV, November 2012.

[Friedman 1997] N. Friedman, D. Geiger, M. Goldszmidt: Bayesian Network Classifiers. In Machine Learning, Vol. 29, Issue 2, 1997, pp. 131–163, DOI:10.1023/A:1007465528199.

[Gao 2011] M. Gao, Z. Wu , F. Jiang: Userrank for item-based collaborative filtering recommendation. In Information Processing Letters, Elsevier, 2011, pp. 440–446, DOI:10.1016/j.ipl.2011.02.003.

[Hardie 1997] R. C. Hardie, K. J. Barnard, E. E. Armstrong: Joint MAP registration and high-resolution image estimation using a sequence of undersampled images. In IEEE Transactions on Image Processing, Vol. 6 No. 12, December 1997, pp. 1621–1633, DOI: 10.1109/83.650116.

[HbbTV 2016] Hybrid broadcast broadband TV (HbbTV). Available online: https://www.hbbtv.org/overview/. Day accessed 11th of April 2016.

[HbbTV Association 2016] HbbTV Association: HbbTV 2.0.1 Specification. Available online: https://www.hbbtv.org/wp-content/uploads/2015/07/HbbTV-SPEC20-00023-002-HbbTV_2.0.1_specification_for_publication_clean.pdf.

[HbbTV 2015] Hybrid broadcast broadband TV (HbbTV): Information about and History of HbbTV 2.0 Specification. Available online: https://www.hbbtv.org/wp-content/uploads/2015/07/Information-about-and-History-of-HbbTV-2.0-Specification.pdf.

[Heckerman 1995] D. Heckerman, D. Geiger, D. M. Chickering: Learning Bayesian networks: The combination of knowledge and statistical data. In Machine learning, Vol. 20, No. 3, 1995, pp. 197-243, DOI: 10.1023/A:1022623210503.

[Herlocker 2002] J. Herlocker, J. A. Konstan, J. Riedl: An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. In Information Retrieval, Vol. 5, No. 4, 2002, pp. 287–310, 2002, DOI: 10.1023/A:1020443909834.

[Ikawa 2010] K. Ikawa, T. Fukuhara, H. Fujii, H. Takeda: Takeda: Evaluation of a TV Programs Recommendation using the EPG and Viewer's Log Data. In Adjunct Proceedings EuroITV 2010, Tampere University of Technology, Finland, 2010, pp. 182–185.

[Jackson 1989] D.A. Jackson, K.M. Somers H.H. Harve: Similarity coefficients: measures of co-occurrence and association or simply measures of occurrence. In American Naturalist 133, 1989, pp. 436-453, DOI: 10.1086/284927.

[Jensen 1906] J. L. W. V. Jensen: Sur les fonctions convexes et les inégalités entre les valeurs moyennes. In Acta Mathematica, Vol. 30 No. 1, pp. 175–193, DOI:10.1007/BF02418571.

[Jirka 2014] V. Jirka, M. Féder, J. Pavlovičová, M. Oravec: Face Recognition System with Automatic Training Samples Selection Using Self-organizing Map. In 56th International Symposium ELMAR-2014, September 2014, Zadar, Croatia, pp. 23-26, ISBN 978-953-184-199-3.

[Jurafsky 2008] D. Jurafsky, J. H. Martin: Speech and Language Processing (2nd ed.). In Prentice Hall, p. 132, ISBN 978-0-13-187321-6.

[Jackman 2009] S. Jackman: Bayesian Analysis for the Social Sciences. In Wiley, 2009, ISBN 978-0-470-01154-6.

[Kacur 2014] J. Kacur, V. Chudy:Topological invariants as speech features for automatic speech recognition. In International Journal of Signal and Imaging Systems Engineering, Vol. 7., No. 4, 2014, DOI: 10.1504/IJSISE.2014.066601.

[Korb 2010] K. B. Korb, Ann E. Nicholson: Bayesian Artificial Intelligence, Second Edition. In CRC Press, December 2010, ISBN 9781439815915.

[Kim 2011] E. Kim, S., Pyo, S. Park, M. Kim: An Automatic Recommendation Scheme of TV Program Contents for (IP)TV Personalization. In IEEE Transactions on Broadcasting, Volume: 57, Issue: 3, September 2011, pp. 674 – 684, DOI: 10.1109/TBC.2011.2161409.

[Krauss 2013] Ch. Krauss, L. George, S. Arbanowski: TV predictor: personalized program recommendations to be displayed on SmartTVs. In BigMine'13, ACM, August 2013, DOI: 10.1145/2501221.2501230.

[Kudiri 2012] K.M. Kudiri, A.M. Said, M.Y. Nayan: Emotion detection using sub-image based features through human facial expressions. In Computer & Information Science (ICCIS), Vol.1, June 2012, pp. 332-335, ISBN: 978-1-4673-1937-9.

[Kullback 1987] S. Kullback: Letter to the Editor: The Kullback–Leibler distance. In The American Statistician, Vol. 41, 1987, pp. 340–341. DOI:10.1080/00031305.1987.10475510.

[Kuzmanovic 2012] N. Kuzmanovic, V. Mihic, T. Maruna, M. Vidakovic, N. Teslic: Hybrid broadcast broadband TV implementation in java based applications on digital TV devices. In IEEE Transactions on Consumer Electronics, Vol. 58, Issue 3, August 2012, DOI: 10.1109/TCE.2012.6311356.

[Lee 2014] H. Lee, J. Kwon: Personalized TV Contents Recommender System Using Collaborative Context tagging-based User's Preference Prediction Technique. In International

Journal of Multimedia and Ubiquitous Engineering, Vol. 9, No. 5, pp. 231-240, May 2014, DOI: 10.14257/ijmue.2014.9.5.23.

[Lekakos 2008] G. Lekakos, P. Caravelas: A hybrid approach for movie recommendation. In Multimed Tools Appl 2008, pp. 36:55–70, DOI 10.1007/s11042-006-0082-7.

[Lemire 2005] D. Lemire, A. Maclachlan: Slope One Predictors for Online Rating-Based Collaborative Filtering. In SIAM Data Mining (SDM'05), April 2005, pp. 471-475, ISBN: 978-0-89871-593-4.

[Lops 2011] P. Lops, M. Gemmis, G. Semeraro: Content-based Recommender Systems: State of the Art and Trends. In Recommender Systems Handbook, Springer Science+Business Media, LLC 2011, pp. 73-105, ISBN: 978-0-387-85819-7.

[Lovinger 2007] R. Lovinger: RDF & OWL, A simple overview of the building blocks of the Semantic Web. In Semantic Web Affinity Group, December 2007, available online http://www.slideshare.net/rlovinger/rdf-and-owl

[Luoh 2010] L. Luoh, Ch-CH. Huang, H-Y. Liu: Image processing based emotion recognition. In System Science and Engineering (ICSSE), July 2010, pp. 491-494, ISBN: 9781424464722.

[Melville 2002] P. Melville, R. J. Mooney, R. Nagarajan: Content-Boosted Collaborative Filtering for Improved Recommendations. In Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI-2002), July 2002, pp. 187-192, ISBN 978-0-262-51129-2.

[Marx 2005] Z. Marx, M. Rosenstein, L. Kaelbling, T. Dietterich: Transfer learning with an ensemble of background tasks. In Proceeding of the NIPS Workshop on Transfer Learning, 2005.

[McLachlan 1996] G. McLachlan, T. Krishnan: The EM Algorithm and Extensions. In John Wiley & Sons, New York, 1996, ISBN: 978-0-471-20170-0.

[Mi 2012] Z. Mi, C. Xu: A Recommendation Algorithm Combining Clustering Method and Slope One Scheme. In Bio-Inspired Computing and Applications, Volume 6840 of the series Lecture Notes in Computer Science, August 2011, pp. 160-167, DOI: 10.1007/978-3-642-24553-4_23.

[Mihalkova 2007] L. Mihalkova, T. Huynh, R. J. Mooney: Mapping and revising Markov logic networks for transfer learning. In AAAI'07 Proceedings of the 22nd national conference on Artificial intelligence, Vol. 1, July 2007, pp. 608-614, ISBN: 978-1-57735-323-2.

[Mitchell 1997] T. Mitchell: Machine Learning. In McGraw-Hill, March 1997, ISBN: 978-0070428072.

[Ng 2012] A. Ng: Part IX: The EM algorithm. In CS229 Lecture notes, Stanford University, November 2012.

[Nigam 2000] K. Nigam, A. K. McCallum, S. Thrun, T. Mitchell: Text classification from labeled and unlabeled documents using EM. In Machine Learning, Vol. 39, pp. 103-134, DOI: 10.1023/A:1007692713085.

[Pagano 2015] Pagano, R., et al: Prediction of TV ratings with dynamic models. In RecSysTV 2015 2nd, Workshop on Recommendation Systems for TV and Online Video, September 2015, DOI: 10.1145/2792838.2798717.

[Pan 2010] S. J. Pan, Q. Yang: A Survey on Transfer Learning. In IEEE Transactions on Knowledge and Data Engineering, Vol. 22, No. 10, October 2010, pp. 1345 – 1359, DOI: 10.1109/TKDE.2009.191.

[Pearl 1988] J. Pearl: Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Representation and Reasoning Series (2nd printing ed.), ISBN: 0-934613-73-7.

[Pearl 1994] J. Pearl, T. S. Verma: A Theory of Inferred Causation. In Logic Methodology and Philosophy of Science IX, Elsevier Science, 1994, ISBN: 9780080544953.

[Pearson 2013] Multivariate Data Analysis: Pearson New International Edition, Canonical Correlation, A supplement to Multivariate Data Analysis, Pearson Higher Ed USA, ISBN 9781292021904.

[Pera 2013] M. S. Pera, Y. K. Ng: A group recommender for movies based on content similarity and popularity. In Information Processing and Management, Vol. 49, Issue 3, 2013, pp. 673–687, DOI: 10.1016/j.ipm.2012.07.007.

[Raina 2006] R. Raina, A. Ng, and D. Koller: Constructing informative priors using transfer learning. In International Conference on Machine Learning, Jun 2006, pp. 713-720, DOI: 10.1145/1143844.1143934.

[Ricci 2011] F. Ricci, L. Rokach, B. Shapira: Introduction to Recommender Systems Handbook. In Recommender Systems Handbook, Springer Science+Business Media, LLC 2011, pp. 1-35, ISBN: 978-0-387-85819-7.

[Richardson 2006] M. Richardson, P. Domingos: Markov logic networks. In Machine Learning, Vol. 62, 2006, January 2006, pp. 107–136, DOI: 10.1007/s10994-006-5833-1.

[Russell 2010] S. Russell, P. Norvig: Artificial Intelligence: A Modern Approach (2nd ed.). Pearson Education, Inc., p. 863, ISBN-10: 0136042597.

[Vanco 2013] M. Vanco, I. Minárik, G. Rozinaj: Dynamic gesture recognition for next generation home multimedia. In ELMAR, 2013 55th International Symposium, 2013, ISSN: 1334-2630.

[Salton 1983] G. Salton, M.J. McGill: Introduction to Modern Information Retrieval. In McGraw-Hill, Inc., 1983, ISBN: 0070544840.

[Scheines 1994] R. Scheines, P. Spirtes, C. Glymour, Ch. Meek: TETRAD II:Tools for Discovery. In Lawrence Erlbaum Associates, Hillsdale, NJ, 1994.

[Smyth 2001] B. Smyth, P. Cotter: Personalized Electronic Program Guides for Digital TV. In AI Magazine, Vol. 22, No. 2, 2001.

[Sotelo 2012] R. Sotelo, J. Joskowicz, A. G. Solla: An affordable and inclusive system to provide interesting contents to DTV using Recommender Systems. In Broadband Multimedia Systems and Broadcasting (BMSB), June 2012, DOI: 10.1109/BMSB.2012.6264262.

[Spirtes 2001] P. Spirtes, C. Glymour, R. Scheines: Causation, Prediction, and Search, 2nd ed. In Adaptive Computation and Machine Learning, January 2001, ISBN: 9780262194402.

[Uluyagmur 2012] M. Uluyagmur, Z. Cataltepe, E. Tayfur: Content-Based Movie Recommendation Using Different Feature Sets. In Proceedings of the World Congress on Engineering and Computer Science 2012 Vol 1, October 2012, ISBN: 978-988-19251-6-9.

[Sedhain 2014] S. Sedhain, S. Sanner, D. Braziunas, L. Xie, J. Christensen: Social Collaborative Filtering for Cold-start Recommendations. In ACM RecSys'14, October 2014, DOI: 10.1145/2645710.2645772.

[Schilling 2013] D.R. Schilling: Knowledge Doubling Every 12 Months, Soon to be Every 12 Hours. In http://www.industrytap.com, 19. April.2013, day accessed 9.January.2017, available online: http://www.industrytap.com/knowledge-doubling-every-12-months-soon-to-be-every-12-hours/3950

[Song 2012] S. Song , H. Moustafa, H. Afifi: Advanced IPTV Services Personalization Through Context-Aware Content Recommendation. In IEEE Transactions on Multimedia, Vol. 14, Issue 6, December 2012, pp. 1528 – 1537, DOI: 10.1109/TMM.2012.2217118.

[Thrun 1995] S. Thrun and T. Mitchell: Learning one more thing. In International Joint Conference on Artificial Intelligence, August 1995, pp. 1217-1223, ISBN: 1-55860-363-8.

[Thulasiraman 1992] K. Thulasiraman, M. N. S. Swamy: Acyclic Directed Graphs. In Graphs: Theory and Algorithms, John Wiley and Son, 1992, p. 118, ISBN 978-0-471-51356-8.

[Torrey 2009] L. Torrey, J. Shavlik: Handbook of Research on Machine Learning Applications: Transfer Learning. In IGI Global, August 2009, pp. 242 - 264, DOI: 10.4018/978-1-60566-766-9.

[W3C 2012] W3C Semantic Web: Introduction to SKOS, January 2012, available online: http://www.w3.org/2004/02/skos/intro

[Webb 2005] G. I. Webb, J. R. Boughton, Z. Wang: Not so naive Bayes: Aggregating one-dependence estimators. In Machine Learning, January 2005, Volume 58, Issue 1, pp. 5–24, DOI: 10.1007/s10994-005-4258-6.

[Xu 2006] J. A. Xu, K. Araki: A SVM-based personal recommendation system for TV programs. In Multi-Media Modelling Conference Proceedings, 2006 12th International, January 2006, DOI: 10.1109/MMMC.2006.1651358.

[Yahoo 2014] Yahoo! Webscope: Dataset: ydata-ymovies-user-movie-ratings-content-v1_0, available online: http://research.yahoo.com/Academic_Relations. Date accessed 25.9.2014

[Zhang 2005] H. Zhang, S. Zheng: Personalized TV program recommendation based on TV-anytime metadata. In Consumer Electronics, 2005. (ISCE 2005), June 2005, DOI: 10.1109/ISCE.2005.1502378.

[Zheng 2000] Z. Zheng, G. I. Webb: Lazy learning of Bayesian Rules. In Machine Learning, 2000, pp. 53-84, DOI: 10.1023/A:1007613203719.

List of Published Peer Reviewed Papers during PhD Candidature

Posoldova, A., Liew, A.W.C.: Content and collaborative filtering recommendation engine for HBB TV. Submitted for journal publication.

Rollan, M. P., Posoldova, A., Rybárová, R.: Recommendation Engine Design Using Bayesian Network for Feature Inference. In In Redzur, 10th International Workshop on Multimedia and Signal Processing, May 2016, Smolenice, Slovakia, pp. 57-61, ISBN: 978-80-227-4560-4

Posoldova, A., Liew, A.W.C.: Recommendation System for HBB TV: Model Design and Implementation. In 16th edition of IEEE Region 8 EuroCon, September 2015, Salamanca, Spain. ISBN 978-1-4799-8569-2, DOI: 10.1109/EUROCON.2015.7313744

Posoldova, A., Liew, A.W.C.: Content Based Recommendation for HBB TV based on Bayes Conditional Probability for Multiple Variables Approach. In 2015 International Symposium on INnovations in Intelligent SysTems and Applications, September, 2015, Madrid, Spain. DOI: 10.1109/INISTA.2015.7276720

Posoldova, A., Liew, A.W.C., Rybarova, R.: Content Based Rating Prediction Recommendation System Designed for HBB TV as Graphical Model. In Redzur, 9th International Workshop on Multimedia and Signal Processing, April 2015, Smolenice, Slovakia, ISBN 978-80-227-4346-4

Posoldova, A., Oravec, M., Rozinaj, G., Liew, A.W.C.: User Experience for Recommendation System for Smart TV. In Redzur, 8th International Workshop on Multimedia and Signal Processing, May 2014, Dubrovnik, Coratia, ISBN 978-80-227-4162-0