MOVIE RECOMMENDATION SYSTEM
USING
CONTENT BASED FILTERING

A PROJECT REPORT

Submitted by

KUMAR ANKIT [18BCS2093]

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE & ENGINEERING

CHANDIGARH UNIVERSITY: PUNJAB 140413

MAY 2022

BONAFIDE CERTIFICATE

Certified that this project report "**MOVIE RECOMMENDATION SYSTEM USING CONTENT BASED FILTERING**" is the bonafide work of"**KUMAR ANKIT**" who carried out the project work under my/our supervision.

SIGNATURE	**SIGNATURE**
SONALI DASH	REEMA GOYAL
SUPERVISOR	**HEAD OF THE DEPARTMENT**
Department of Computer Science Engineering	**Department of Computer Science & Engineering**

Submitted for the project viva-voce examination held on:__________________

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We consider it our privilege to express through the pages of this report a few words of gratitude and respect to all those personalities who inspired us in the completion of this project. At the outset, we must acknowledge with gratitude the generous help that we have received from our Mentor Dr. Sonali Dash, and all other teachers who have helped us in this project. We are grateful to all the personnel for giving us chance to do this project to explore our talent in programming skills, making us capable of utilizing the object-oriented concept, and besides these their cooperation in the collection and collation of information and material required for the preparation of this project report is also appreciable. We would also like to express our gratitude to our college Chandigarh University and the Department of Computer Science & Engineering who gave us such an opportunity. Finally, it would not be impartial to acknowledge our friends who extended their helping hands to our project completion.

Kumar Ankit[18BCS2093]

ABSTRACT

Recommender System is a tool that helps users find content and conquer information overload. Predicts user interests and makes recommendations based on the user interest model. The existing content-based recommendation system is a continuous development of collaborative filtering, which does not require user testing in movies. Instead, the similarities are calculated based on the knowledge of the movies the users selected and then making recommendations accordingly. With the development of machine learning, the current content-based recommendation system can create user profiles and movies, respectively. They are creating or updating a profile based on analytics of user-friendly movies. The system can compare user and movie profiles and recommend the most similar movies. Therefore, this recommendation method that directly compares the user to the movie cannot be delivered to the co-filtering model. There are many features released in the movie; they are varied and unique and also different from other recommendation programs. Simply put, systems can suggest movies based on the person's two or more attribute.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

Chapter 1
Introduction

Because of Information overload is the difficulty in understanding an issue and effectively making decisions when one has too much information about that issue, and is generally associated with the excessive quantity of daily informationThe advent of modern information technology has been a major factor in the proliferation of information in many fields: by the quantity produced, the easy distribution, and the wide range of audiences reached. The technological aspects have long been reinforced by the proliferation of social media and the economy of attention, making it easier to steal attention. In the age of integrated digital technology, informatics, online culture (or digital culture), information overload is associated with excessive exposure, excessive viewing of information, and the input of information and data entries.. Even on YouTube, if you want to watch a video of a particular concept, there are many videos available to you.The data in such large volumes often appears to collide, and without proper knowledge and processing of this information, it is wasted. In such a case, where there is a lot of information the users should search a few times before finding what they originally wanted. To address this issue, researchers have come up with interesting ideas. Promotional programs are used on YouTube to recommend the video, Amazon and Flipkart to recommend a product, Netflix and Amazon Prime to recommend a movie, etc. Whatever you do on such platforms, an unknown structure(system) detects your search and ultimately suggests things that you are most likely to do surfing with. So, the function of the recommendation system is to raise the most important things for the user. This research paper discusses movie proposal and the background thinking of the movie proposal system,Standard movie suggestion systems, news related to standard movie suggestion programs, and the proposed Artificial Intelligence solution based on your personalized movie recommendations system. Recommender System, which helps users find information that is important to them and otherwise pushes information to specific users. This is a win-win situation for both customers and content providers.

Collaborative filtering is one of the well known and most extensive techniques in recommendation system its basic idea is to predict which items a user would be interested in based on their preferences. Recommendation systems using collaborative filtering are able to provide an accurate prediction when enough data is provided, because this technique is based on the user's preference. User-based collaborative filtering has been very successful in the past to predict the customer's behavior as the most important part of the recommendation system. However, their widespread use has revealed some real challenges, such as data sparsity and data scalability, with gradually increasing the number of users and items.

Great filtering strategies or algorithms to recommend movies are as follows:

1.Content Based Filtering

2.Collaborative Filtering

3.Hybrid Filtering

Some of these strategies can be further subdivided into sub-categories

1.1 Background

In the era of information overload, it is very difficult for users to get information that they are really interested in. And for the content provider, it is also very hard for them to make their content stand out from the crowd. That is why many re- searchers and companies develop Recommender System to solve the contradiction. The mission of Recommender System is to connect users and information, which in one way helps users to find information valuable to them and in another way push the information to specific users. This is the win-win situation for both customers and content providers.

VionLabs is a media-tech startup company. The company provides a new way on how consumers are given access to good and suitable content. The mission of VionLabs is to increase needs of its digital user base. Vionel is the movie website developed by VionLabs, which is a place for people who love movies can gather all the information about films in one place[5]. This thesis report will present a more practical recommendation method that can be used on a movie website that does not have enough users. In order to achieve the best possible compliment, many experts have been focusing on designing different recommendations for years. The interactive filtering technology is widely used in many personalized recommendations.Sang-Min Choi et al. spoke about the lack of a shared filtering system that includes a blank issue or a cold front problem. The authors have counseled an answer for the usage of phase information to avoid this hassle. The writers have proposed a film-promoting gadget based on style relationships. The authors have shown that grouping facts exist in newly created content simplest. Therefore, even though new content no longer has sufficient ratings or perspectives, it is able to still appear within the list of recommendations with the assistance of category or grouping facts. The proposed end result is not biased towards the most favored and considered new content material. Therefore, even a brand new film may be suggested by an advisory program.

1.2 Problem Statement

For building a recommender system from scratch, we face several different problems. Currently there are a lot of recommender systems based on the user information, so what should we do if the website has not gotten enough users. After that, we will solve the representation of a movie, which is how a system can understand a movie. That is the precondition for comparing similarity between two movies. Movie features such as genre, actor and director is a way that can categorize movies. But for each feature of the movie, there should be different weight for them and each of them plays a different role for recommendation. So we get these questions:

- How to recommend movies when there are no user information.
- What kind of movie features can be used for the recommender system.
- How to calculate the similarity between two movies.
- Is it possible to set weight for each feature.

1.2 Goals

The goals of this thesis project is to do the research of Recommender Systems and find a suitable way to implement it for Vionel.com. There are many kinds of Recommender Systems but not all of them are suitable for one specific problem and situation. Our goal is to find a new way to improve the classification of movies, which is the requirement of improving content-based recommender systems.

1.3 Methodology

In order to achieve the goal of the project, the first process is to do enough back- ground study, so the literature study will be conducted. The whole project is based on a big amount of movie data so that we choose quantitative research method. For philosophical assumption, positivism is selected because the project is experi- mental and testing character. The research approach is deductive approach as the improvement of our research will be tested by deducing and testing a theory. Ex post facto research is our research strategy, the movie data is already collected and we don't change the independent variables. We use experiments to collect movie data. Computational mathematics is used data analysis because the result is based on improvement of algorithm. For the quality assurance, we have a detail explana- tion of algorithm to ensure test validity. The similar results will be generated when we run the same data multiple times, which is for reliability. We ensure the same data leading to same result by different researchers.[12]

1.4 Ethics

Movie information is the only part that may have ethics problem. However, all the information we get for research is from public database such as Wikipedia and our own movie database. So there are no data confidentiality and user privacy problems.

1.5 Delimitations

In this project, we will not deep in how to collect and generate the data, which is the business of other teams in VionLabs. In the validation part, the data is divided into training and testing parts. But how to split data will not presented in detail neither.

It is important to build a real recommender system for a business company and in the end I have built a successful demo which helped our company to get the opportunity of cooperation with other companies. But in the thesis report, I will focus on the improvement of movie representation and explain why our new approach is more helpful to recommender system. I will not talk about the detail of implementation of the system because of confidentiality agreement.

1.6 Outline

In the first chapter, background of the project is introduced. After that, the lay- out of the report is as follows: Chapter 2 provides an overview of related work on recommender systems. I will introduce the mainstream approaches and some famous business recommender systems. Chapter 3 shows the basic principles of our recommender system and how we improve it. Chapter 4 I will illustrate how recommendation works by a specific movie. Chapter 5 is the validation part, I will present our test result to illustrate the improvement of our approach. In the end, Chapter 6 and 7 is the conclusion and possible future work respectively.

1.7 TASK IDENTIFICATION:

PHASE	TASK	DURATION
1	Information Gathering	10 days
2	Feasibility Study	10 days
3	Analysis	10 days
4	Design	30 days
5	Coding	40 days
6	Testing	10 days
7	Documentation	10 days
TOTAL		120 days

Chapter 2

Literature survey

Since the advent of the information age, the immense growth of the World Wide Web gives rise to the difficulty for users to quickly find what they want given a variety of applications. Recommendation systems have rigorously been used in various applications as a way to suggest items that a customer would likely be interested in by predicting customer preference. The most popular applications using recommendation systems are movies, music, news, grocery shopping, travel guides, online dating, books, restaurants, E-commerce sites and so forth.

Recommendation systems can be broadly categorized as contents-based filtering, collaborative filtering, and hybrid approach [3]. Contents-based filtering systems are used to recommend items based on a description of items the user used to like before, or corresponding with pre-defined attributes of the user, such a system having its roots in information retrieval techniques. Collaborative filtering systems recommend items to user based on the past preferences of items rated by all users. Hybrid techniques combine both these approaches. In this paper, I will deal mainly with collaborative filtering (CF).

u and an item *d*, how the user like the item is defined as the similarity between $ContentBasedProfile(u)$ and $Content(d)$:

$$p(u, d) = sim(ContentBasedProfile(u), Content(d)) \quad (2.3)$$

Using keywords to model item is an important step for many recommender systems. But extracting keywords of an item is also a difficult problem, especially in media field, because it is very hard to extract text keywords from a video. For solving this kind of problem, there are two main ways. One is letting experts tag the items and another one is letting users tag them. The representative of expert-tagged systems are Pandora for music and Jinni for movies. Let's take Jinni as an example, the researchers of Jinni defined more than 900 tags as movie gene, and they let movie experts to make tags for them. These tags belong to different categories, including movie genre, plot, time, location and cast. Figure 2.1 is from *Jinni*, which are the tags for movie Kung Fu Panda. As we can see from the figure, the tags of Kung Fu Panda are divided into ten categories totally, *Mood*, *Plot*, *Genres*, *Time*, *Place*, *Audience*, *Praise*, *Style*, *Attitudes* and *Look*. These tags contain all aspects of movie information, which can describe a movie very accurately.

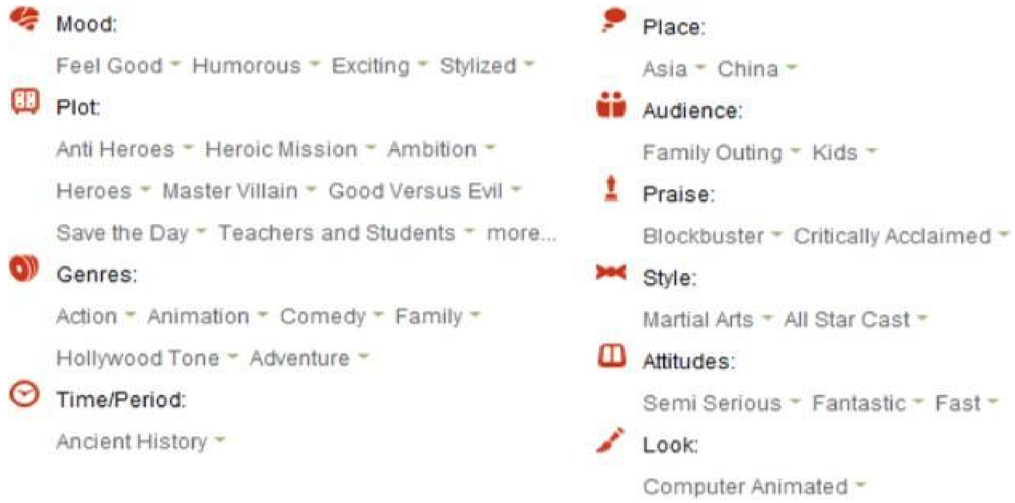


Figure 2.2. Tags for Kung Fu Panda

Compared with expert-tagged system, user-tagged system is applied more widely. The representative websites are *Delicious* and *Flickr*. The feature of user-tagged sys- tem is the tags are more diversity than that of expert-tagged system. But the weak- ness is that the tags are of lower quality, even there are a lot of wrong tags. So in the user-tagged systems, there are two main problems, one is tag recommendation[34], which means when a user tags an item, the system can recommend some relative tags for him to choose. The purpose is first to be convenience for users and second it can increase the quality of tags. Another question is how to recommend items based on tags(tag-based recommendation[33]). After items are tagged, the simplest recommendation approach is to use tags as keywords of the item, and recommend by the content-based algorithm.

2.1 Collaborative-filtering Recommendation

Collaborative-filtering recommendation is the most famous algorithm in recom- mender systems. This algorithm models user's taste according to the history of user behavior. GroupLens published the first paper[31] about collaborative filtering and the paper raised user-based collaborative filtering. In 2000, Amazon came up with item-based collaborative filtering in their paper[20]. These two algorithms are very famous in business recommender systems. Collaborative filtering takes more information from the user into consideration. It gives automatic prediction based on searching a larger group of people finding a group of people who like the particular thing. It is based on reactions by similar users. It has a wide range of applications in the area where there is a large amount of user data and interaction. Automatic recommendation for users using the information preference of many other users. Suppose if user 1 likes one set of movies and user 2 another set of movies respectively. For target user, the recommendation will be a set of movies common among the two user1 and user 2

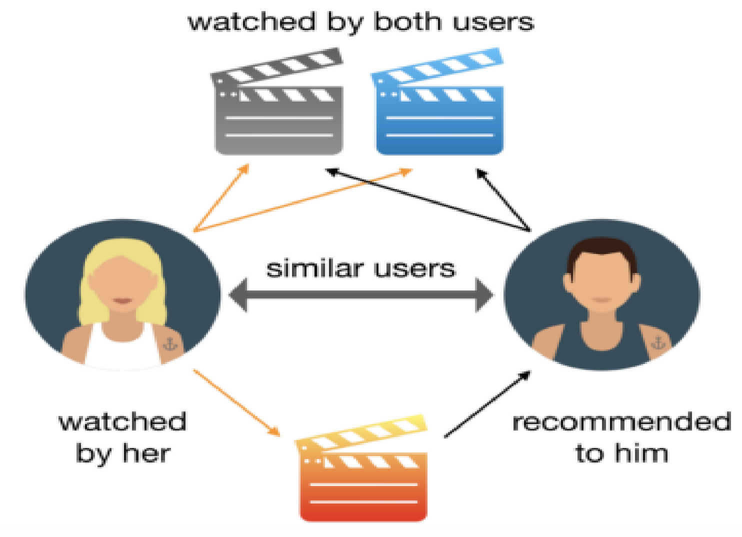
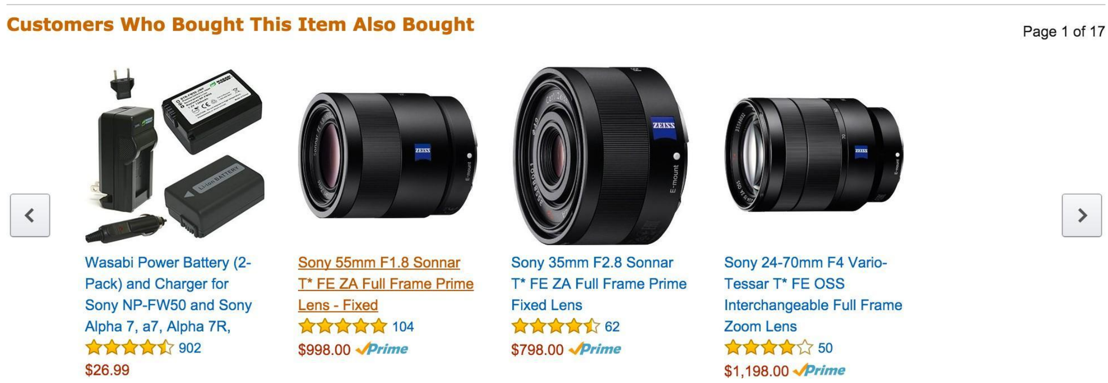


Figure 2.3. Collaborative Filtering Diagram



2.1.1 User-based collaborative-filtering

In user-based collaborative filtering, it is considered that a user will like the items that are liked by users with whom have similar taste. So the first step of user-based collaborative-filtering is to find users with similar taste. In collaborative filtering, the users are considered similar when they like similar items. Simply speaking, given user u and v, $N(u)$ and $N(v)$ are items set liked by u and v respectively. So the similarity of *u* and *v* can be simply defined as

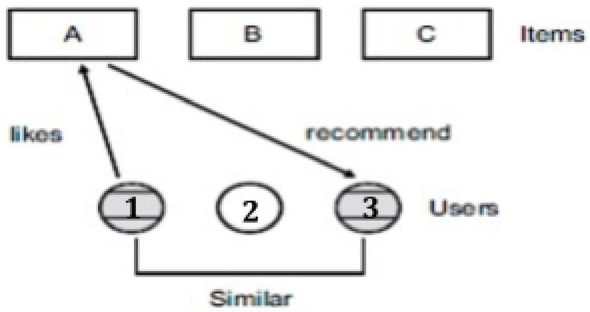


Fig 2.1

fig 2.1 is an example of User-based CF recommendation. According to the interest history of User A, only User C can be the neighbor of him, so Item D will be recommended to User A.

2.1.2 Item-based collaborative-filtering

Item-based collaborative-filtering is different, it assumes users will like items that are similar with items that the user liked before. So the first step of item-based collaborative-filtering is to find out items that are similar with what the user liked before. The core point of item-based collaborative-filtering is to calculate the simi- larity of two items. Item CF considers that items that are liked by more same users, the more similar they are. Assume $N(i)$ and $N(j)$ are user sets who like i and j respectively. So the similarity of i and j.

Table 2.2 is an example of Item-based CF recommendation. According to the interest history of all the users for Item A, people who like Item A like Item C as well, so we can conclude that Item A is similar with Item C. While User C likes Item A, so we can deduce that perhaps User C likes Item C as well.

User/Item	Item A	Item B	Item C
User A	!		!
User B	!	!	!
User C	!		recommend

Table 2.2. Item-based CF

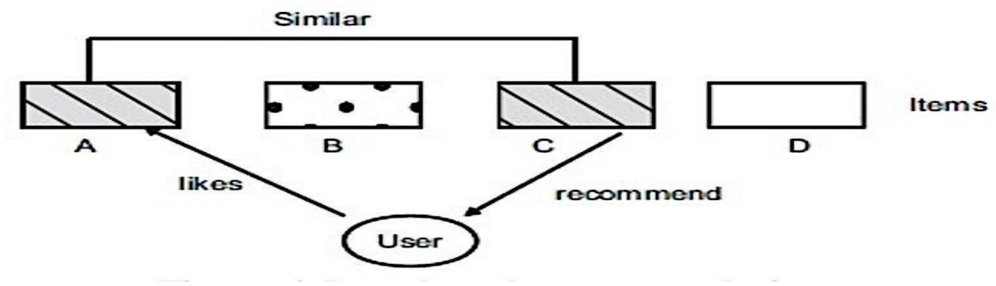


Fig 2.2

User-based and Item-based collaborative-filtering algorithms are all neighborhood- based algorithm, there are also a lot of other collaborative-filtering algorithms. Hoff- man raised Latent Class Model in this paper[13], the model connects user and item by latent class, which considers that a user will not become interested in items di- rectly. Instead, a user is interested in several categories that contain items, so the model will learn to create the categories according to user's behavior. On top of Latent Class Model, researchers came up with Matrix Decomposition Model, which is called Latent Factor Model[4] as well. There are a lot of models based on ma- trix decomposition and they mostly came from Netflix Prize Competition, such as RSVD[28], SVD++[18] and so on.

Besides Matrix Decomposition Model, Graph Model is widely applied in collaborative- filtering. Baluja introduced graph model of co-view behind the recommender algo- rithm of YouTube in [3] and also raised a broadcast algorithm on graph to measure

how much a user like an item. This literature[27] research how to increase serendip- ity of recommendation result by means of the analysis of the path between nodes

in the graph. Mirza[26] systematically studied recommendation problems based on graph model and point out the essence of the recommendation is to connect user and item. The graph is the natural method for that. [10] studies similarity algo- rithms between the nodes of the graph and compares the recommendation precision of different algorithms.

Hybrid Recommender Systems

Hybrid Recommender System is more and more popular currently. Combining col- laborative filtering and content-based filtering can be more effective by recently research. There are many ways to implement hybrid recommender systems: simply combine the result of CF and CB recommendations, add CF capability to a CB method.[1]

There are seven hybridization methods:

Weighted: Add scores from different recommender components.

Switching: Choose methods by switching in different recommender compo- nents.

Mixed: Show recommendation result from different systems.

Features Combination: Extract features from different sources and combine them as a single input.

Feature Augmentation: Calculate features by one recommender and put the result to the next step.

Cascade: Generate a rough result by a recommender technique and recom- mend on the top of the previous result.

Meta-level: Use the model generated by one recommender as the input of another recommender technique.

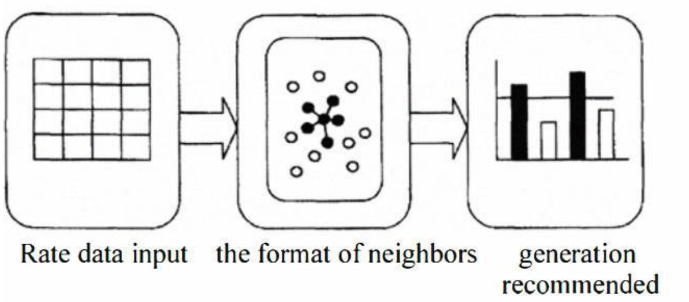
Although there are many combinations theoretically, it is not always efficacious for a specific problem. The most important principle of hybrid recommendation is to avoid or make up the weakness of every single recommender technique.[6]

2.2 Collaborative Filtering Process

In a fundamental scenario, collaborative filtering (CF) processing can be mainly divided into three steps; Step 1) collecting user ratings data matrix, Step 2) selecting similar neighbors by measuring the rating similarity, and then Step 3) generating prediction as seen diagram [Figure 1] [4, 6, 7, 8, 9].

2.2.1 User Rating Score Data Input

Generally, input data in recommendation system based on the CF technology consists of user, item, and user opinions on observed items as a matrix m × n as shown in [Table 1]. Symbol m symbolizes the total number of users and n symbolizes the total number of items. *Rm,n* is the score of item In rated by user Um.



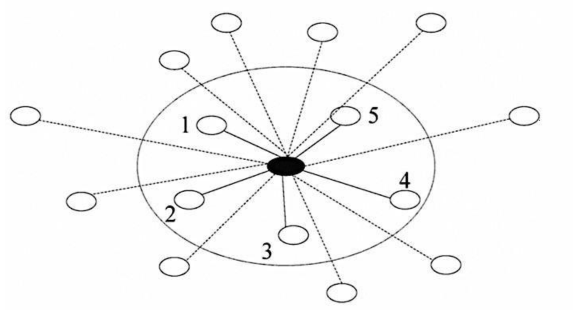
[Figure 4] The Collaborative filtering process

Item ***User***	***I1***	***I2***	***I3***	**…**	***In***
U1	$^{R}1,1$	$^{R}1,2$	$^{R}1,3$	…	$^{R}1,n$
U2	$^{R}2,1$	$^{R}2,2$	$^{R}2,3$	…	$^{R}2,n$
U3	$^{R}3,1$	$^{R}3,2$	$^{R}3,3$	…	$^{R}3,n$
…	…	…	…	…	…
Um	$^{R}m,1$	$^{R}m,2$	$^{R}m,3$	…	$^{R}m,n$

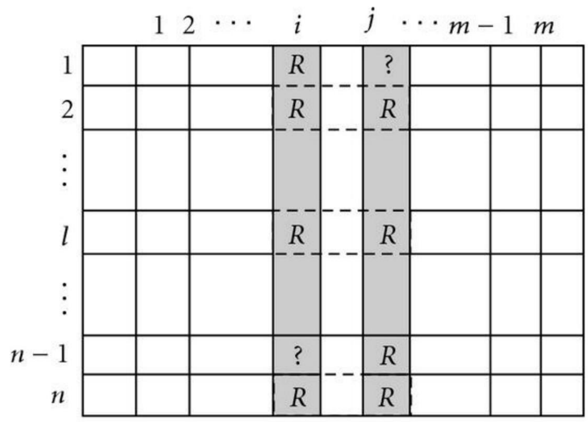
[Table 1] User-Item ratings matrix

2.1.1 The Formation of Neighbors

The CF approaches use statistical techniques to analyze the similarity between users and to form a set of users called neighbors. A set of similarity measures is a metric of relevance between two vectors [9]. User-based similarity is to compute the relevance between users as the values of two vectors. In UBCF, after the similarity is calculated, it is used in building neighborhoods of the current target user. For example, as seen in [Figure 5] [4], the distance between the target node (black node) and every other node is calculated by a similarity measure. And then, 5 users in the center are selected by k-nearest neighbor algorithm (k = 5).



[Figure 5] The neighborhood formation process



[Figure 6] Item based similarity computation

2.3 Comparison

Each approach has its advantage and disadvantage, and the effects are different as well for different dataset. The approach may not suitable for all kinds of problems because of the algorithm itself. For example, it is hard to apply automate feature extraction to media data by content-based filtering method. And the recommenda- tion result only limits to items the user ever chose, which means the diversity is not

so good. It is very hard to recommend for users who never choose anything. Collab- orative filtering method overcomes the disadvantage of mentioned before somehow. But CF based on big amount of history data, so there are problems of sparsity and cold start. In terms of cold start, as collaborative filtering is based on the similarity between the items chosen by users, there are not only new user problem[30], but also new item problem, which means it is hard to be recommended if the new item has never been recommended before[1]. The comparison is in Table 2.3[11].

2.4 Famous Recommender Systems

What is the difference between recommender system and search engine is that rec- ommender system is based on the behaviors of user. There are a lot of websites using recommender system in the world. Personalized recommender system analyzes a huge amount of user behavior data and provides personalized content to different users, which improves the click rate and conversions of the website[36]. The fields that widely use recommender system are e-commerce, movie, video, music, social network, reading, local based service, personalized email and advertisement.

2.4.1 E-Commerce

The most famous e-commerce website, Amazon, is the active application and pro- moter of recommender system. The recommender system of Amazon reaches deeper into all kinds of products[32]. Figure 2.2 is the recommendation list of Amazon.

Apart from personalized recommendation list, another important application of recommender system is relevant recommendation list. When you buy something in Amazon, the relevant goods will be shown below [15]. Amazon has two kinds of relevant recommendation, one is customers who bought this item also bought in Figure 2.3.

Another is what other items do customers buy after viewing this item in Figure 2.3. The difference between the two recommendations is the calculation of the dif- ferent user behaviors. The most important application of relevant recommendation is cross selling. When you are buying something, Amazon will tell you what other customers who bought this item also bought and let you decide whether buy it at the same time[7]. If you do, the goods will be packed and provide a certain discount.

2.4.2 Movie and Video website

Personalized recommender system is a very important application for movie and video website, which can help users to find what they really like among the vast of videos. Netflix is the most successful company in this field[24]. Amazon and it are

Recommendation algorithms	Advantages	Disadvantage
Content based	Recommendation result is intuitive and easy to interpret; No need for users? access history data; No new item problem and no sparsity problem; Supported by the mature technology of classification learning.	Limited by the features extraction methods; New user problem; The training of classifier needs massive data; Poor scalability.
Collaboration filtering	No need for professional knowledge; Performance improving as the increasing of the user number; Automatic; Easy to find user?s new interesting point; Complex unstructured item can be processed. eg. Music, Video, etc.	Sparsity problem; Poor scalability; New user and new item problem; The recommendation quality limited by the history data set.

Table 2.3 Comparison of contented-based and collaborative

Figure 2.4. Personalized recommendation of Amazon

Figure 2.5. Relevant Recommendation, Customers Who Bought This Item Also Bought

the two most representative companies in recommender systems.

Figure 2.5 is the recommendation page of Netflix. We can find that the recom- mendation result consists of the following parts.

- The title and poster of the movie.
- The feedback of the user, including Play, Rating and Not Interested.
- Recommendation reason.

Figure 2.4. Relevant Recommendation, What Other Items Do Customers Buy After Viewing This Item

It can be illustrated that the recommendation algorithm is similar with Ama- zon according to the recommendation reason of Netflix. Netflix declared that 60% of their users find movies that they are interested in by the recommender system.[19]

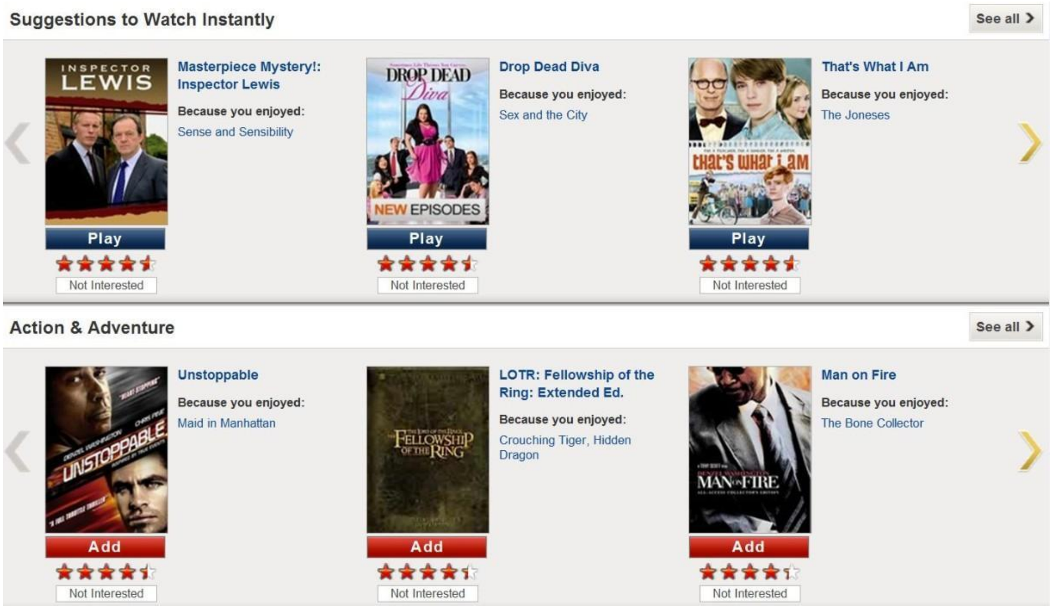


Figure 2.5. Netflix Recommender System

As the biggest video website in America, YouTube has a huge amount of videos that uploaded by users. So the information overload is a serious problem for YouTube. In the latest paper[8] of YouTube, in order to prove the availability of personalized recommender system, researchers of YouTube ever made an experi- ence comparing the click rate of personalized recommendation list and popular list.

The result showed that click rate of personalized recommendation list is twice of that of popular list.

2.4.3 Internet Radio

The successful application of personalized recommender system needs two require- ments, one is information overload because if users can find what they like easily, there is no reason to use recommender systems. The second is the user doesn't have clear requirements. Because they will use search engine directly if they do.

Under the two requirements, recommender system is very suitable for personal- ized Internet radio. First of all, people cannot listen to all the music in the world and find which one they like. Secondly, people often do not want to listen to a specific music, they wish to listen to whatever musics that match their mood at that moment.[16]

There are a lot of personalized Internet radios, such as Pandora and Last.fm.

Figure 2.6 is the main page of Last.fm.

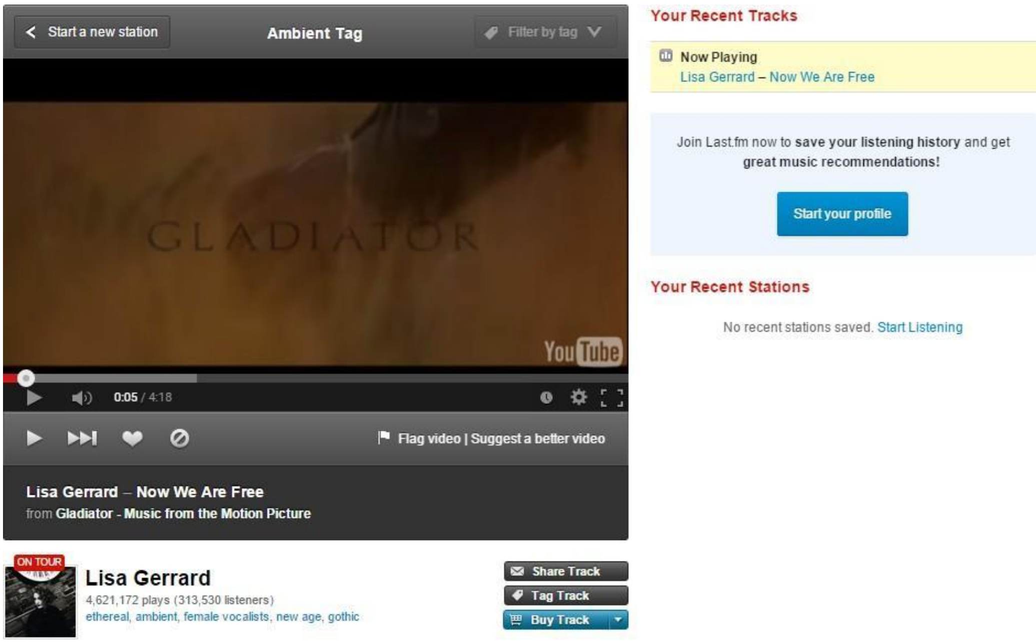


Figure 2.6. Last.fm

Chapter 3

System Requirement and Analysis

3.1 Item Based Collaborative Filtering Applying Dimension Reduction

UBCF is easy to implement and good to scale correlated items [2]. However, as stated previously above, it comes up against a couple of problems: data sparsity and data scalability. Data sparsity problem could lead to a skewed prediction and low reliability of predictions. Besides, data scalability requires low operation time and high memory feature to scale with all users and items in the database.

To address these issues in UBCF, this paper proposes IBCF approach applying dimension reduction [6].

3.1 IBCF Applying Dimension Reduction

Enormous users and products have been added at E-commerce domains. A typical example is Amazon. Amazon added 30 million new customers in 2013 and had had over 244 million active customers as Geekwire reported in 2014 [18]. Also, Amazon had sold over 200 million products as ReportX reported in 2013 [19]. Currently in 2015, it is expected that Amazon would have more than these numbers of users and products. If the recommendation system using UBCF at

Amazon should look into all datasets similar to a 244 million × 200 million matrix, it will encounter data scalability and data sparsity issues. In UBCF, more the number of users and items increase, more the number of matrix dimensions increase and runtime takes long to find nearest neighbor of users. Therefore, it is assumed that using denser data having much more preference information given by users with IBCF effectively addresses data scalability and data sparsity problems. To focus on active items assuming that they have many ratings given by users, matrix is required to reduce dimension in IBCF without regard to passive items.

3.1 Datasets

The datasets in this project are from Vionel database and public place such as Wikipedia. The data is in JSON format and each JSON object represents one movie information. We store the whole data in MongoDB, which is a cross-platform document-oriented database.

Each item of the data has a IMDb id as a unique identifier, which in the future will be replaced by our vionel id. There are now 8 features that are used in the project. They are *director*, *actor*, *genre*, *keyword*, *theme*, *scene* and *location*. Note that not all movies have complete features, some of the movies may miss some feature information, and this fact has been considered in the algorithm.

3.2 Feature Extraction

There are a lot of benefits to represent document as vector space model, for example we can measure cosine similarity and even extent to Clustering and Classification. However, vector space model does not have the ability to solve these two typical problems: a word has multiple meaning and a meaning has multiple words[22].

We ever consider to build a thesaurus, but it is too much time consuming. There are two mainstream approach, one is lexical co-occurrence and another is using shallow parsing to analyze the grammatical relation or syntax dependence between vocabulary[22]. Generally speaking, lexical co-occurrence is more robust and grammatical relation is more accurate.

In the previous vector space, we only pay attention on the frequency of a single word. But the word co-occurrence is a very important information as well, which based on a fact that the appearance of two or more co-occurrence words in the document is not occasional. Latent Semantic Indexing is an approach to explore the inner semantic relations[9][23]. The latent semantic indexing is to map the co-occurrence words to the same dimensional space. The co-occurrence words are considered semantically related.

Compared with our previous vector space, latent semantic space has less dimen- sions. That is because many words are mapped to the same dimension. Therefore, latent semantic indexing is a kind of dimensionality reduction method. The process of dimensionality reduction is mapping objects in high-dimensional space to low-dimensional space.

We can transform previous vector to term-document matrix, which is a M N matrix C consists of M terms and N documents. Each row of the matrix represents a term and each column of the matrix represents a document.

3.2.1 Singular Value Decomposition

Singular Value Decomposition is an important matrix decomposition in linear alge- bra. Let us see a theorem first.

Assume r is the *rank* of M N matrix C, so the SVD(Singular Value Decom- position) of matrix C is:

$$C = USV^T \tag{3.1}$$

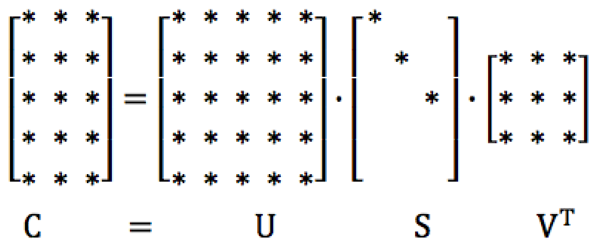


Figure 3.1. Singular Value Decomposition

Thereinto, the columns of U are orthonormal and so are columns of V . S is a diagonal matrix and elements on diagonal are singular value of C.

3.2.2 Matrix Low Rank Approximation

This is the process to solve matrix low rank approximation problem:

1. Given C, construct SVD by Equation 3.1 and decompose it: $C = USV^T$
2. Set $r - k$ minimum singular value on the diagonal of S zero, then get S_k
3. Calculate $C_k = US_kV^T$ as the approximation of C

3.2.3 Application of LSI

Matrix $S_k V^T$ is k N , k is the rank after using LSI, N is the total number of doc- uments. Each column of matrix is the new coordinate of corresponding document on dimension-reduced space.

SVD is first raised by Deerwester[9] in Information Retrieval. LSI represent the documents on a new dimension-reduced space, which is actually the linear combi- nation of each dimension on old space. It can be considered as a soft clustering.

Feature extraction is not the main task of the thesis, so I just introduced the basic principle simply, which is with the help of other research team in VionLabs.

3.3 Feature Representation

Vector Space Model

Before being deep into feature representation, it is very important to get the idea of document representation. Assume there are several documents and each document consists of one single sentence. Then we can represent the document as a model in Figure 3.2, which is called Vector Space Model. Consider each feature of a movie as a term, then a feature can represented by this model. It is obviously that some features are more important than others and the importance is the weight in the similarity calculation.

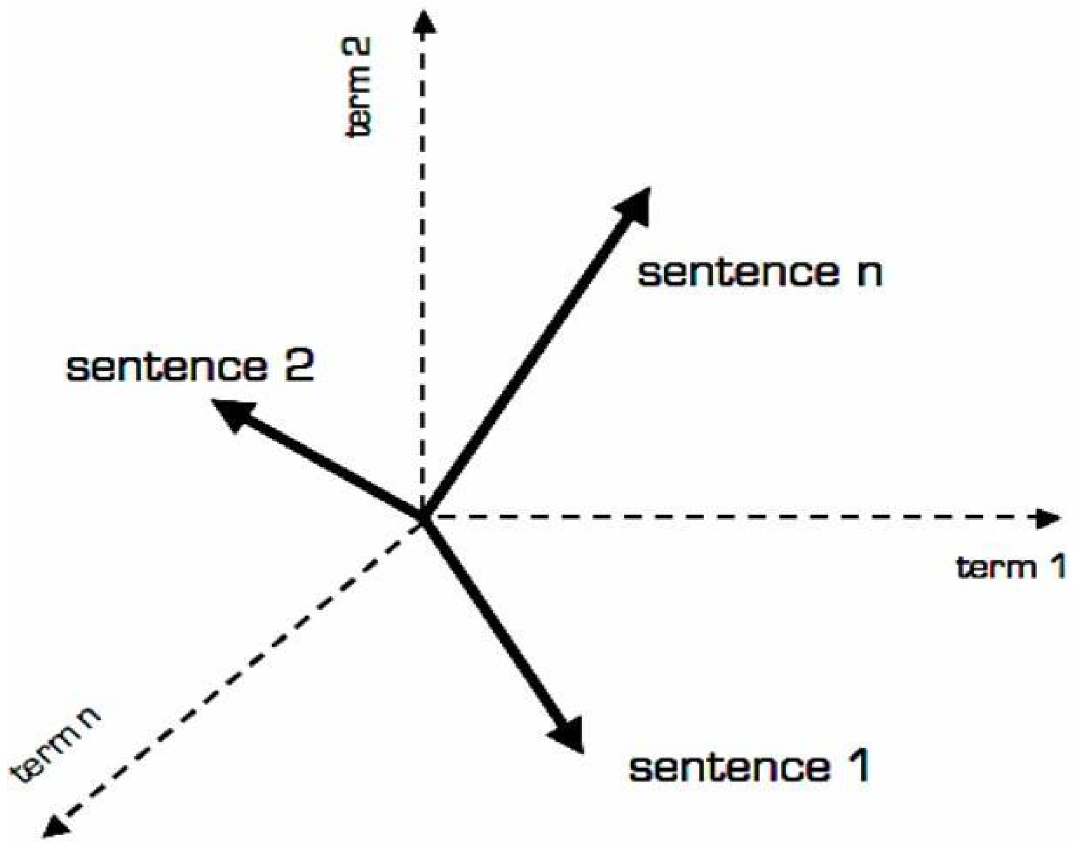


Figure 3.2. Vector Space Model of documents

3.4 TF-IDF

TF-IDF is short for term frequency-inverse document frequency that is a common weighting technique for information retrieval and text mining, which reflects how important a word is for a document[29]. The importance of a word increases pro- portionally to the times the word appears in the document, but it also decreases inversely proportional to the frequency the word appears in the whole corpus.

TF-IDF consists of TF and IDF, which are Term Frequency and Inverse Doc- ument Frequency respectively. TF represents the frequency a word appears in the document. The main idea of IDF is: if a term appears more in other documents, the term will be less important.

3.4.1 Term Frequency

Term frequency is about how many times a term t_i appears in document d_j, which can be represented by $TF\ (t_{ij})$. In the condition of removing stop-words, the more t_i appears in the document, the more important term t_i is for the document. It can be defined as:

$$TF\ (t_{ij}) = \frac{N\ (t_i, d_j)}{N\ (d_j)} \tag{3.2}$$

$N(t_i, d_j)$ is the number of times t_i appears in d_j and $N(d_j)$ is the total number of terms in document d_j

3.4.2 Inverse Document Frequency

In order to understand inverse document frequency, let us see what document fre- quency is. Document frequency is how many times term t_i appears in all documents C, which is represented by $N(t_i, C)$. The more term t_i appears in all documents C, the weaker term t_i can represent document d_j.

Inverse document frequency means that the represent ability of term t_i for doc- ument d_j and its amount in all documents $N(t_i, C)$ is inverse proportion, which is represented by $IDF(t_i)$:

$$IDF(t_i) = \log \frac{N(C)}{N(t_i, C)} \tag{3.3}$$

$N(C)$ is the total amount of documents, $IDF(t_i)$ decrease with the increase of $N(t_i, C)$. The less $N(t_i, C)$ is, the more representative t_i is for d_j.

3.5. WEAKNESS OF TF-IDF

3.4.3 Normalization

In order to reduce the inhibition of stop words, we will normalize each variable. After normalization, the calculation of $TF - IDF$

Equation 3.4 base on the principle: The term that is more representative for a document is the word that appears in the document more often and less often in other documents.

3.5 Weakness of TF-IDF

The weakness of TF-IDF is mainly because of the fact that TF-IDF does not con- sider the same category of documents.

1. The concept of inverse document frequency is that the more number of times term t_i appears in document, the more representative term t_i is for the docu- ment, which does not consider the situation that the terms in same category. If the number of term t_i increases in the same category, it means that this t_i can represent the category very well. So the term should be given higher weight. But in Equation 3.3, the IDF will decrease when the number of term t_i in category C_l $N(t_i, C_l)$ increases. So we conclude: the weight of terms appearing frequently in same category should be strengthened.

2. For term t_1 and t_2, the appearance of t_1 is more average than t_2 in documents, so t_1 is more representative than t_2. If t_2 only appears in one or two documents of the category and almost does not appear in others, it can be considered that t_2 is of lower importance for the category. This kind of terms is not representative and they should be given lower weight. TF-IDF cannot solve
this situation neither.

3.6 Improvement of TF-IDF

We researched related papers and found that the improvement approach is limited according to our analysis above.

Paper [25] introduces a new parameter to represent the in-category character- istic, which give us ideas about how to handle situation of terms in same category. TF-IDF focus on the ability to differentiate different documents, which ignores the fact that the term appears in the documents belonging to the same category. So in-category term frequency is something important for the improvement. In paper [35], we know that term weight is positive correlated to their frequency. But it also points out that the higher frequency term maybe distributed in a part of the document. Such terms are not so representative for the whole document. The paper comes up the idea to solve the situation.

According to the analysis of Section 3.5 and research of related topic, we will improve TF-IDF base on the following two rules.

1. The number of term appearing in same category increases, the higher weight the term should be.
2. If the appearance of term in documents of a category is more even, the higher weight the term should be.

We came up with an improved feature weight algorithm $TF - IIDF - DC$: $weight_{TF-IIDF-DC}(t_{ij}) = TF \times IIDF \times DC$ (3.5) TF is term frequency, $IIDF$ is the improved inverse term frequency and DC is distribution coefficient.

3.6.1 Improved IDF

Consider the first rule mentioned above, we make the transformation:

$$IDF = \log \frac{N(C)}{N(t_i, C_l) + N(t_i, \bar{C}_l)} \tag{3.6}$$

$N(t_i, C_l)$ represents how many times term t_i appears in category C_l, $N(t_i, \bar{C}_l)$ represents how many times term t_i appears in other categories.

If a term mostly appears in a same category and less in others, the weight of the term for the category should be higher. So we introduce a coefficient λ for concentration, which is the in-category term frequency mentioned before:

$$\lambda = \frac{N(t_i, C_l)}{N(C_l)} \tag{3.7}$$

$N(C_l)$ is the total number of documents in category C_l, it is obvious that $\lambda \leq 1$. After plugging in concentrative coefficient λ, the IIDF is:

$$IIDF = \log\left[\frac{N(C)}{N(t_i, C_l) + N(t_i, \bar{C}_l)} \times \lambda\right] \tag{3.8}$$

$$= \log\left[\frac{N(C)}{N(t_i, C_l)} - \times \right]$$

$$N(t_i, C_l) + N(t_i, C_l)\, N(C_l)$$

3.6. IMPROVEMENT OF TF-IDF

After a simple transformation:

$$IIDF = \log\left[\frac{1}{1 + \frac{N(t_i, \bar{C}_l)}{N(t_i, C_l)}} \times \frac{N(C)}{N(C_l)}\right] \tag{3.9}$$

cofiCostFunc.m | fmincg.m | loadMovieList.m | normalizeRatings.m | movie_recommender.m

```
% ============= Part 1: Entering ratings for a new user ==============
%  Before we will train the collaborative filtering model, we will first
%  add ratings that correspond to a new user that we just observed. This
%  part of the code will also allow you to put in your own ratings for the
%  movies in our dataset!
%
movieList = loadMovieList();

%  Initialize my ratings
my_ratings = zeros(1682, 1);

% Check the file movie_idx.txt for id of each movie in our dataset
% For example, Toy Story (1995) has ID 1, so to rate it "4", you can set
my_ratings(1) = 4;

% Or suppose did not enjoy Silence of the Lambs (1991), you can set
my_ratings(98) = 2;

% We have selected a few movies we liked / did not like and the ratings we
% gave are as follows:
my_ratings(7) = 3;
my_ratings(12)= 5;
my_ratings(54) = 4;
my_ratings(64)= 5;
my_ratings(66)= 3;
my_ratings(69) = 5;
```

$N(C)$ and $N(C_l)$ are constants, so we can see that when the number of term t_i appearing in category C_l increases, the $N(t_i, C_l)$ increases so that IIDF will in- crease. When $N(t_i, \bar{C}l)$ which is the number of term t_i appearing in other categories

increases, IIDF will decrease. $N(t_i, \bar{C}l)$ is 0 when t_i only appear in category C_l, then

IIDF gets the maximum value $\log \frac{N(C)}{}$. So t_i is the most representative for cate-

gory C_l at this moment. When t $N(C_l)$ $_i$ appears in every document of category C_l, then

$N(t_i, C_l) = N(C_l)$ and $N(t_i, C_l) + N(t_i, \bar{C}l) = N(C)$, IIDF is 0 now. t_i is not representative for category C_l in this situation.

3.6.2 Distribution Coefficient

If the term t_i is distributed evenly in every document of a category, it is obvious that t_i is representative for the category. We define a distributed coefficient:

$$N(C_i)$$

l i l l

)

$$\sum_{j=1} [TF(t_{ij}) - TF(t_i, C_l)]^2$$

$$DC = 1 - N(C) \times TF(t, C)^2 \qquad (3.10)$$

$TF(t_i, C_l)$ is the average number of term t_i appearing in documents of category C_l.

$$\overline{TF}(t_i, C_l) = \frac{1}{N(C_l)} \sum_{j=1}^{N(C_l)} TF(t_{ij}) \qquad (3.11)$$

When t_i appears in every document of C_l, it is obvious that $TF(t_{ij}) = \overline{TF}(t_i, C_l)$. Then DC is 1 which is the maximum value at the time. That means t_i is the most representative for category C_l. When term t_i only appears in one document of category C_l, we can easily know:

3.6.3 Normalization

Finally, we got the equation for calculating weight of movie features after the nor- malization.

$$weight(t_{ij}) = \frac{TF \times IIDF \times DC}{\sqrt{\sum^{n} \ldots}} \qquad (3.14)$$

3.7 Similarity

Calculate the similarity between movies is the objective of content-based recom- mender systems. The content can be anything such as text, video and image. In our project, each movie is represented by a feature vector. I will introduce the cosine similarity algorithm then.

Cosine Similarity is the most popular measurement for document similarity. In order to calculate the similarity between two features, we can calculate the cosine of the angle between the feature vector .

3.7. SIMILARITY

The range of the similarity is between -1 and 1. -1 means that the direction of the two vectors are totally opposite and 1 means they are in the same direction. The cosine similarity is 0 if the two vectors have no relationship. For text matching, it is obviously that the weights are non-negative, so the range should be 0 to 1 in our case. Here is an example for illustrating cosine similarity.[14]

Given two sentences:

- A: I like watching TV, but I don’t like watching films.
- B: I don’t like watching TV and films.

How can we calculate the similarity between the two sentences? The basic idea is: the more similar the words used by the two sentences are, the more similar the sentences are.

First step: Word segmentation.

- A: I / like / watching / TV, but / I / don’t / like / watching / films.
- B: I / don’t / like / watching / TV / and / films. Second step: List all the words.
- I, like, watching, TV, but, don’t, films, and Third step: Word frequency calculation.
- A: I 2, like 2, watching 2, TV 1, but 1, don’t 1, films 1, and 0.
- B: I 1, like 1, watching 1, TV 1, but 0, don’t 1, films 1, and 1. Forth step: Get word frequency vector.

- A: [2, 2, 2, 1, 1, 1, 1, 0]
- B: [1, 1, 1, 1, 0, 1, 1, 1]

Then we can calculate the cosine of the two vectors by Equation 3.15.

$$\cos(\theta) = \frac{2 \times 1 + 2 \times 1 + 2 \times 1 + 1 \times 1 + 1 \times 0 + 1 \times 1 + 1 \times 1 + 0 \times 1}{\sqrt{2^2 + 2^2 + 2^2 + 1^2 + 1^2 + 1^2 + 1^2 + 0^2} \times \sqrt{1^2 + 1^2 + 1^2 + 1^2 + 0^2 + 1^2 + 1^2 + 1^2}}$$

The value is 0.85 so that the two sentences are much similar

Chapter 4

Experiment

In this chapter, we will introduce how to implement the content-based recommender system based on the principle mentioned. After that, we will test the system and give out the result to prove the improvement of our system.

4.1 Dataset

All the movie data we used is from IMDb, Wikipedia and our own database in VionLabs. In the end we get 178356 movies and related information. For the perspective of recommender system, a movie can be described by a collection of features, which can be genres, actors, directors and so on.

- *Director* : The director is from IMDb, most of movies only have one director, but some of them have two or more.
- *Actor* : A movie normally has a lot of actors, but most of them is useless for recommender system and bring disadvantageous effects. So we only get three main actors for one movie. They are from IMDb as well.
- *Keyword*: We use LSI to extract keywords from Wikipedia plot, which is under the help of the colleagues in VionLabs.

- *Release Year*: This is when the movie is released and data is from IMDb.
- *Vionel Theme*: Theme is a kind of keyword that describes movies in a different perspective, such as *Time Travel* and *Comic Book*. They are defined by VionLabs.
- *Language*: Language is from IMDb, which is the language that occurs in the movie.
- *Location*: Location is from IMDb, which is where the movie happens. *Vionel Scene*: Scene of the movie is analyzed by other research in our team. We will recognize the background of every frame in the movie by machine learning. For example, bar, hall room, store are what we recognized.

4.2 Category

In the scenario of movie, we will divide the movies into 23 categories by the normal genres. Table 4.1 shows the categories we used. Each movie in the case is a doc- ument, which is represented by the eight features described in Section 4.1. As we said before, the movie is represented by Vector Space Model, each feature for the movie is a term in the document.

Sci-Fi	Crime	Romance	Animation	Music
Comedy	War	Horror	Adventure	News
Biography	Thriller	Western	Mystery	Short
Drama	Action	Documentary	Musical	History
Family	Fantasy	Sport		

Table 4.1. Categories

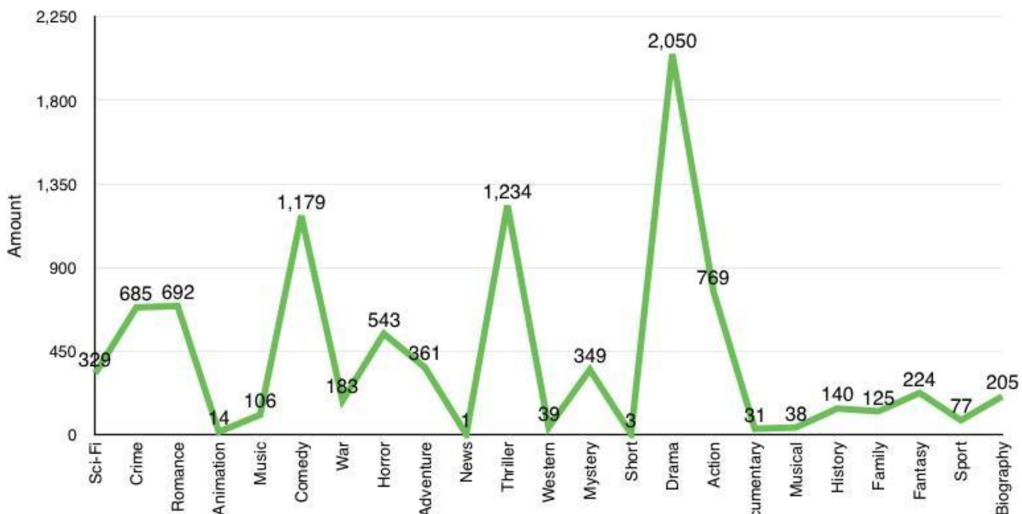
In many other content-based recommender systems, the genre is used as vector to calculate similarity. But this is only one aspect of the movie and there are a lot of other features of movie such as background, actor, etc. So we add more features and some of them are very unique because they are extracted by our own research.

But we didn't simply add features together to calculate TF-IDF, we have dis- cussed the reasons in Section 3.5. The genre is the natural feature that we can use as category. Figure 4.1 shows the distribution of genres in the movie database. Compared with the principle mentioned before, each genre is a category for doc- uments. The number of document in each category is shown in the figure. Each document contains many terms which are features in our case, they are described in Section 4.1.

4.3 Document

As we discussed before, the document in our case is the movie which contains many features. The movie will be represented by vector space model in the experiment. In Section 4.1, we introduced the features that are used to model the movie. The vector space model is like this kind of format:

$$MovieModel = [Directors, Actors, Keywords, ReleaseYear, VionelThemes, Languages, Locations, VionelScenes] \tag{4.1}$$



4.3. DOCUMENT

A movie can have multiple directors and actors, so the vector is pretty long generally. Here we use movie *The Dark Knight* to illustrate the model.

Directors	Christopher Nolan
Actors	Christian Bale Heath Ledger Aaron Eckhart
Keywords	Keywords
Release Year	2008
Vionel Themes	Business and finance Comic book Bromance Cars and racing Mafia & Gangsters Revenge
Language	English, Mandarin
Location	USA, UK

Vionel Scene	Basement, Bar

Table 4.2. Information of The Dark Knight

Table 4.2 is the basic information of movie *The Dark Knight*. I don’t list key- words in the table because there are 63 keywords for the movie, which is difficult to show them in the table.

We can get a very long vector after the calculation according to Equation 3.14, which is the model for the movie. Each weight represents the importance of a fea- ture for the movie. In order to show it intuitively, I present it in Figure 4.2.

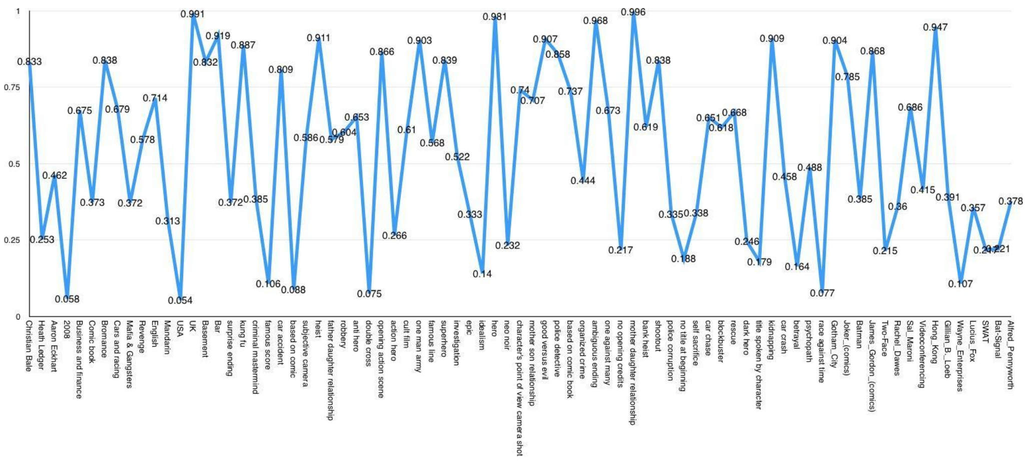


Figure 4.2. Feature Weight of The Dark Knight

There are 80 features totally for movie *The Dark Knight*, the number is cal- culated by our TF-IIDF-DC, which shows the importance of each feature. From this perspective, we can see that if features of a movie have similar distribution, it means that the two movies are similar.

Actually, one feature is one dimension for the model, I list all features in one dimension in the figure just because multiple-dimension is hard to show by figure.

4.4 Result

In our case, feature to movie is term to document. We can easily convert movie to vector space model which can be used to calculate the similarity. After previous calculation, every movie in the database can be represented by a vector. Then we use cosine similarity discussed in Section 3.7 to calculate similarity for each movie. Figure 4.3 shows the final recommendation of The Dark Knight, which is a screenshot of our current demo system.

4.4. RESULT

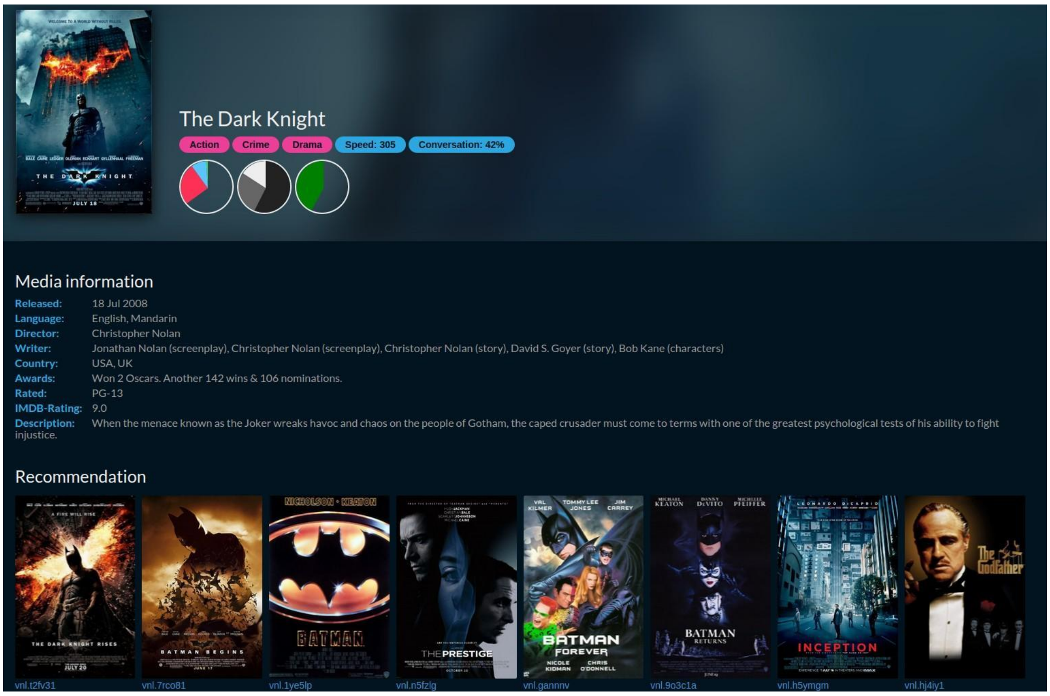


Figure 4.3. Recommendation of The Dark Knight

Command Window

```
Top recommendations for you:
Predicting rating 5.0 for movie Someone Else's America (1995)
Predicting rating 5.0 for movie Santa with Muscles (1996)
Predicting rating 5.0 for movie Star Kid (1997)
Predicting rating 5.0 for movie Saint of Fort Washington, The (1993)
Predicting rating 5.0 for movie They Made Me a Criminal (1939)
Predicting rating 5.0 for movie Entertaining Angels: The Dorothy Day Story (1996)
Predicting rating 5.0 for movie Marlene Dietrich: Shadow and Light (1996)
Predicting rating 5.0 for movie Great Day in Harlem, A (1994)
Predicting rating 5.0 for movie Aiqing wansui (1994)
Predicting rating 5.0 for movie Prefontaine (1997)


Original ratings provided:
Rated 4 for Toy Story (1995)
Rated 3 for Twelve Monkeys (1995)
Rated 5 for Usual Suspects, The (1995)
Rated 4 for Outbreak (1995)
Rated 5 for Shawshank Redemption, The (1994)
Rated 3 for While You Were Sleeping (1995)
Rated 5 for Forrest Gump (1994)
Rated 2 for Silence of the Lambs, The (1991)
Rated 4 for Alien (1979)
Rated 5 for Die Hard 2 (1990)
Rated 5 for Sphere (1998)
```

Chapter 5

Validation

In order to illustrate the improvement of our recommender system, we will firstly prove our improved TF-IDF can generate better weight for features of the movie. In this chapter, we introduce a classification approach to validate the effect of movie model that is generated by different weight.

5.1 k-NN

***k-Nearest Neighbors* is a classical algorithm in Machine Learning. *k* represents its nearest *k* items.**

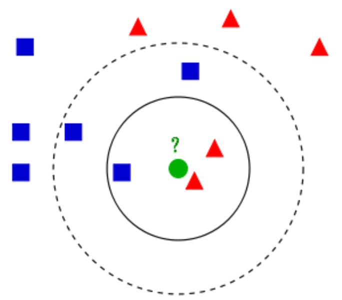


Figure 5.1. k-NN

Figure 5.1 is from Wikipedia of k-NN, we can see there are two types of items, one is blue rectangle and another is red triangle. The green cycle is the item to be classified.

In this case, if k = 3, the nearest items to the green cycle are two red triangles and one blue rectangle. This three items vote and the green cycle belongs to red triangle category. If k = 5, the nearest items to the green cycle are two red triangles and three blue rectangles. Then the five items vote and the green cycle should belong to the blue rectangle category.

For our movie case, we split all the movie data into two sets. One is used as un- classified set which is test set and another is classified set which is validation set.

For each movie in test set, we will calculate the similarity between it with all the movies in validation set. Then we can find the most k similar movies for each movie in the test set. So the movie to be classified should belong to the category that appears most in the k movies.

The k is set to 15 in our project which is decided by a lot of experience. We cannot make sure 15 is the best for our project but it is enough for the validation.

After that, every movie in test set is classified, which means every movie in test set is classified with a genre in our case. We have known the right genre of the movie, so we will find out the right and wrong classifications and measure the improvement by metrics in following section.

Command Window

```
New user ratings:
Rated 4 for Toy Story (1995)
Rated 3 for Twelve Monkeys (1995)
Rated 5 for Usual Suspects, The (1995)
Rated 4 for Outbreak (1995)
Rated 5 for Shawshank Redemption, The (1994)
Rated 3 for While You Were Sleeping (1995)
Rated 5 for Forrest Gump (1994)
Rated 2 for Silence of the Lambs, The (1991)
Rated 4 for Alien (1979)
Rated 5 for Die Hard 2 (1990)
Rated 5 for Sphere (1998)
```

Command Window

```
Training collaborative filtering...
Iteration     1 | Cost: 3.281339e+05
Iteration     2 | Cost: 1.246920e+05
Iteration     3 | Cost: 1.057921e+05
Iteration     4 | Cost: 8.016496e+04
Iteration     5 | Cost: 6.207777e+04
Iteration     6 | Cost: 5.318401e+04
Iteration     7 | Cost: 4.671600e+04
Iteration     8 | Cost: 4.476383e+04
Iteration     9 | Cost: 4.315379e+04
Iteration    10 | Cost: 4.201271e+04
Iteration    11 | Cost: 4.122893e+04
Iteration    12 | Cost: 4.072081e+04
Iteration    13 | Cost: 4.050861e+04
Iteration    14 | Cost: 4.023356e+04
Iteration    15 | Cost: 3.999350e+04
Iteration    16 | Cost: 3.987475e+04
Iteration    17 | Cost: 3.981650e+04
Iteration    18 | Cost: 3.961572e+04
Iteration    19 | Cost: 3.955722e+04
Iteration    20 | Cost: 3.950201e+04
Iteration    21 | Cost: 3.938140e+04
Iteration    22 | Cost: 3.930738e+04
Iteration    23 | Cost: 3.924423e+04
Iteration    24 | Cost: 3.920240e+04
Iteration    25 | Cost: 3.917654e+04
```

Command Window

```
Top recommendations for you:
Predicting rating 5.0 for movie Someone Else's America (1995)
Predicting rating 5.0 for movie Santa with Muscles (1996)
Predicting rating 5.0 for movie Star Kid (1997)
Predicting rating 5.0 for movie Saint of Fort Washington, The (1993)
Predicting rating 5.0 for movie They Made Me a Criminal (1939)
Predicting rating 5.0 for movie Entertaining Angels: The Dorothy Day Story (1996)
Predicting rating 5.0 for movie Marlene Dietrich: Shadow and Light (1996)
Predicting rating 5.0 for movie Great Day in Harlem, A (1994)
Predicting rating 5.0 for movie Aiqing wansui (1994)
Predicting rating 5.0 for movie Prefontaine (1997)


Original ratings provided:
Rated 4 for Toy Story (1995)
Rated 3 for Twelve Monkeys (1995)
Rated 5 for Usual Suspects, The (1995)
Rated 4 for Outbreak (1995)
Rated 5 for Shawshank Redemption, The (1994)
Rated 3 for While You Were Sleeping (1995)
Rated 5 for Forrest Gump (1994)
Rated 2 for Silence of the Lambs, The (1991)
Rated 4 for Alien (1979)
Rated 5 for Die Hard 2 (1990)
Rated 5 for Sphere (1998)
```

5.3 Evaluation Metrics

Precision and Recall are two measurements for statistics, which are used to evalu- ate the quality of statistic result. Precision is used to calculate the ratio of related documents with selected documents. Recall is used to calculate the ratio of related documents with all related documents in selected documents. Below is the defini- tion of the precision and recall under the context of our movie case.

Assume TP_i represents the number of test documents belonging to C_i and they are classified to C_i as well. FP_i represents the number of test documents that do not belong to C_i are classified to C_i. FN_i represents the number of test documents belonging to C_i are classified to other categories. So the Precision and Recall in category C_i is defined by:

$$P_i = \frac{TP_i}{TP_i + FP_i} \tag{5.1}$$

$$R_i = \frac{TP_i}{TP_i + FN_i} \tag{5.2}$$

Generally we should comprehensively consider precision and recall, then we in- troduce F-Measure.

5.4 Analysis

In order to show the improvement of our TF-IIDF-DC, we split the data into 80% as training data and 20% as testing data. Both TF-IDF and TF-IIDF-DC are cal- culated and evaluated by Equations above. The result is in Table 5.1.

	TF-IDF			**TF-IIDF-DC**		
Category	**P**	**R**	**F**	**P**	**R**	**F**
Sci-Fi	86.64	83.56	85.07	89.32	85.51	87.37
Crime	78.57	75.98	77.25	85.54	81.19	83.31
Romance	70.65	67.87	69.23	76.76	69.34	72.86
Animation	78.78	77.32	78.04	85.34	81.49	83.37
Music	70.45	67.29	68.83	77.45	72.83	75.07
Comedy	80.67	75.19	77.83	88.76	82.73	85.64
War	77.87	72.87	75.29	85.91	77.34	81.40

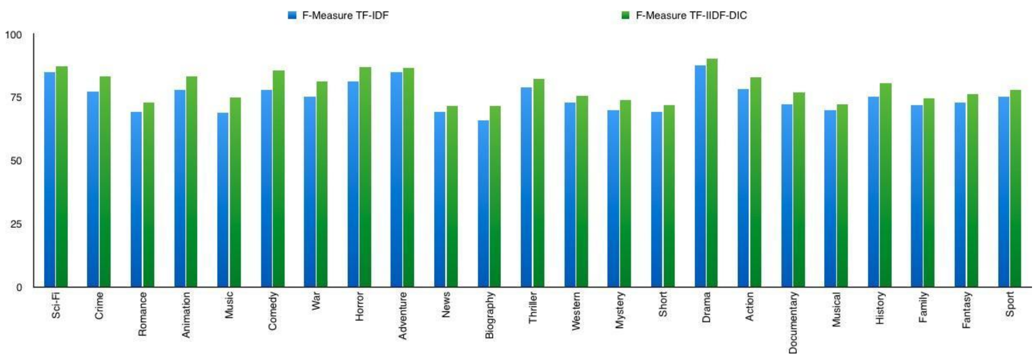
Horror	82.34	80.17	81.24	89.21	84.65	86.87
Adventure	86.32	83.76	85.02	88.75	84.35	86.49
News	70.44	67.93	69.16	73.79	69.72	71.70
Biography	69.98	62.48	66.02	74.39	68.93	71.56
Thriller	79.89	78.28	79.08	84.18	80.38	82.24
Western	75.45	70.32	72.79	77.65	73.76	75.66
Mystery	71.22	68.95	70.07	76.85	71.45	74.05
Short	70.39	67.87	69.11	74.28	69.89	72.02
Drama	89.87	85.69	87.73	92.48	88.41	90.40
Action	80.43	76.21	78.26	84.56	81.44	82.97
Documentary	73.93	70.48	72.16	78.58	75.65	77.09
Musical	70.22	69.28	69.75	73.23	71.28	72.24
History	76.89	73.47	75.14	81.74	79.34	80.52

Family	73.49	70.34	71.88	76.43	72.67	74.50
Fantasy	73.68	71.94	72.80	77.98	74.87	76.39
Sport	77.23	73.45	75.29	80.64	75.34	77.90

Table 5.1. Comparison of TF-IDF and TF-IIDF-DC(%)

As we can see in Table 5.1 and Figure 5.2, the Precision, Recall and F-Measure by TF-IIDF-DC are all higher than that of TF-IDF. From the experiment, TF- IIDF-DC strengthens the weight of representative terms and weaken terms of no use in the category, which is so-called noise.

The most important factor for content-based recommender systems is feature. How to describe a movie is the most important task because the more accurate a movie is described, the better results recommender system generates. So from this perspective, we have proved the improvement of our approach in content-based recommender system.



Chapter 6

Conclusion

Recommender system has become more and more important because of the infor- mation overload. For content-based recommender system specifically, we attempt to find a new way to improve the accuracy of the representative of the movie.

For the problems we mentioned at beginning, firstly, we use content-based rec- ommender algorithm which means there is no cold start problem. In Section 4.1, we list all the features in our recommender system. Some of them are from other research team in the company, so the features are diversity and more accurate than others. Then we introduced the cosine similarity which is commonly used in in- dustry. For the weight of features, we introduced TF-IIDF-DC which improve the representative of the movie.

This master thesis introduces a content-based recommender system for the movie website of VionLabs. The features used in the system are extracted from various aspects of the movie, which are diversity and unique. We introduce a new approach for setting weight for these features, the movie can be represented more accurately by TF-IIDF-DC which is the key point of our research.

In the end of the project, we use k-NN and various metrics to evaluate the im- provement of the new approach. It is illustrated that the new approach contributes positively according to the evaluation.

Chapter 7

Future Work

Recommender system has developed for many years, which ever entered a low point. In the past few years, the development of machine learning, large-scale network and high performance computing is promoting new development in this field. We will consider the following aspects in future work.

- Use collaborative filtering recommendation.
  After getting enough user data, collaborative filtering recommendation will be introduced. As we discussed in Section 2.2, collaborative filtering is based on the social information of users, which will be analyzed in the future research.[37]
- Introduce more precise and proper features of movie.[1]
  Typical collaborative filtering recommendation use the rating instead of object features. In the future we should extract features such as color and subtitle from movie which can provide a more accurate description for movie.
- Introduce user dislike movie list.
  The user data is always useful in recommender systems. In the future we will collect more user data and add user dislike movie list. We will input dislike movie list into the recommender system as well and generate scores that will be added to previous result. By this

way we can improve the result of recommender system.

- Introduce machine learning.

  For future study, dynamic parameters will be introduced into recommender
  system, we will use machine learning to adjust the weight of each feature automatically and find the most suitable weights.

- Make the recommender system as an internal service.

  In the future, the recommender system is no longer a external website that
  will be just for testing.

Bibliography

[1] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *Knowledge and Data Engineering, IEEE Transactions on*, 17(6):734–749, 2005.

[2] Ricardo Baeza-Yates, Berthier Ribeiro-Neto, et al. *Modern information re- trieval*, volume 463. ACM press New York, 1999.

[3] Shumeet Baluja, Rohan Seth, D Sivakumar, Yushi Jing, Jay Yagnik, Shankar Kumar, Deepak Ravichandran, and Mohamed Aly. Video suggestion and dis- covery for youtube: taking random walks through the view graph. In *Proceed- ings of the 17th international conference on World Wide Web*, pages 895–904. ACM, 2008.

[4] Robert Bell, Yehuda Koren, and Chris Volinsky. Modeling relationships at mul- tiple scales to improve accuracy of large recommender systems. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 95–104. ACM, 2007.

[5] Suvir Bhargav. Efficient features for movie recommendation systems. 2014.

[6] Robin Burke. Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction*, 12(4):331–370, 2002.

[7] Robin Burke. Hybrid web recommender systems. In *The adaptive web*, pages 377–408. Springer, 2007.

[8] James Davidson, Benjamin Liebald, Junning Liu, Palash Nandy, Taylor Van Vleet, Ullas Gargi, Sujoy Gupta, Yu He, Mike Lambert, Blake Livingston, et al. The youtube video recommendation system. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 293–296. ACM, 2010.

[9] Scott C. Deerwester, Susan T Dumais, Thomas K. Landauer, George W. Fur- nas, and Richard A. Harshman. Indexing by latent semantic analysis. *JAsIs*, 41(6):391–407, 1990.

Francois Fouss, Alain Pirotte, Jean-Michel Renders, and Marco Saerens. Random-walk computation of similarities between nodes of a graph with ap- plication to collaborative recommendation. *Knowledge and data engineering, ieee transactions on*, 19(3):355–369, 2007.

[10] Xu Hailing, Wu xiao, Li Xiaodong, and Yan Baoping. Comparison study of internet recommendation system. *Journal of Software*, 20(2):350–362, 2009.

[11] Anne Håkansson. Portal of research methods and methodologies for research projects and degree projects. In *The 2013 World Congress in Computer Sci- ence, Computer Engineering, and Applied Computing WORLDCOMP 2013; Las Vegas, Nevada, USA, 22-25 July*, pages 67–73. CSREA Press USA, 2013.

[12] Thomas Hofmann and Jan Puzicha. Latent class models for collaborative fil- tering. In *IJCAI*, volume 99, pages 688–693, 1999.

[13] Anna Huang. Similarity measures for text document clustering. In *Proceedings of the sixth new zealand computer science research student conference (NZC- SRSC2008), Christchurch, New Zealand*, pages 49–56, 2008.

[14] Dietmar Jannach, Markus Zanker, Alexander Felfernig, and Gerhard Friedrich.
Recommender systems: an introduction. Cambridge University Press, 2010.

[15] Robert Jäschke, Leandro Marinho, Andreas Hotho, Lars Schmidt-Thieme, and Gerd Stumme. Tag recommendations in folksonomies. In *Knowledge Discovery in Databases: PKDD 2007*, pages 506–514. Springer, 2007.

[16] Joseph A Konstan. Introduction to recommender systems: Algorithms and evaluation. *ACM Transactions on Information Systems (TOIS)*, 22(1):1–4, 2004.

[17] Yehuda Koren. Factorization meets the neighborhood: a multifaceted collabo- rative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 426–434. ACM, 2008.

[18] Daniel Z Lieberman, Suena H Massey, Vilmaris Quiñones Cardona, and Ken- neth P Williams. Predicting content preference: Applying lessons learned from the commercial web to

therapeutic software. *Cyberpsychology*, 2(2), 2008.

[19] Greg Linden, Brent Smith, and Jeremy York. Amazon. com recommendations: Item-to-item collaborative filtering. *Internet Computing, IEEE*, 7(1):76–80, 2003.

[20] Pasquale Lops, Marco De Gemmis, and Giovanni Semeraro. Content-based recommender systems: State of the art and trends. In *Recommender systems handbook*, pages 73–105. Springer, 2011.

[21] Christopher D Manning, Prabhakar Raghavan, Hinrich Schütze, et al. *Introduc- tion to information retrieval*, volume 1. Cambridge university press Cambridge, 2008.

[22] Christopher D Manning and Hinrich Schütze. *Foundations of statistical natural language processing*. MIT press, 1999.

[23] Bradley N Miller, Istvan Albert, Shyong K Lam, Joseph A Konstan, and John Riedl. Movielens unplugged: experiences with an occasionally connected rec- ommender system. In *Proceedings of the 8th international conference on Intel- ligent user interfaces*, pages 263–266. ACM, 2003.

[24] Jiangang Yang Mingyong Liu. An improvement of tfidf weighting in text cat- egorization. 2012.

[25] Batul J Mirza, Benjamin J Keller, and Naren Ramakrishnan. Studying rec- ommendation algorithms by graph analysis. *Journal of Intelligent Information Systems*, 20(2):131–160, 2003.

[26] Kensuke Onuma, Hanghang Tong, and Christos Faloutsos. Tangent: a novel,'surprise me', recommendation algorithm. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data min- ing*, pages 657–666. ACM, 2009.

[27] Arkadiusz Paterek. Improving regularized singular value decomposition for collaborative filtering. In *Proceedings of KDD cup and workshop*, volume 2007, pages 5–8, 2007.

[28] Juan Ramos. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, 2003.

[29] Al Mamunur Rashid, Istvan Albert, Dan Cosley, Shyong K Lam, Sean M Mc- Nee, Joseph A Konstan, and John Riedl. Getting to know you: learning new user preferences in recommender systems. In *Proceedings of the 7th interna- tional conference on Intelligent user interfaces*, pages 127–134. ACM, 2002.

[30] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 175–186. ACM, 1994.

[31] J Ben Schafer, Joseph Konstan, and John Riedl. Recommender systems in e- commerce. In *Proceedings of the 1st ACM conference on Electronic commerce*, pages 158–166. ACM, 1999.

[32] Shilad Sen, Jesse Vig, and John Riedl. Tagommenders: connecting users to items through tags. In *Proceedings of the 18th international conference on World wide web*, pages 671–680. ACM, 2009.

[33] Yang Song, Ziming Zhuang, Huajing Li, Qiankun Zhao, Jia Li, Wang-Chien Lee, and C Lee Giles. Real-time automatic tag recommendation. In *Proceed- ings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 515–522. ACM, 2008

[34] Tian Xia and Yanmei Chai. An improvement to tf-idf: term distribution based term weight algorithm. *Journal of Software*, 6(3):413–420, 2011.

[35] Osmar R Zaïane, Jia Li, and Robert Hayward. Mission-based navigational behaviour modeling for web recommender systems. In *Advances in Web Mining and Web Usage Analysis*, pages 37–55. Springer, 2006.

[36] Zhi-Dan Zhao and Ming-Sheng Shang. User-based collaborative-filtering rec- ommendation algorithms on hadoop. In *Knowledge Discovery and Data Mining, 2010. WKDD'10. Third International Conference on*, pages 478–481. IEEE, 2010.

plication to collaborative recommendation. *Knowledge and data engineering, ieee transactions on*, 19(3):355–369, 2007.

[37] Xu Hailing, Wu xiao, Li Xiaodong, and Yan Baoping. Comparison study of internet recommendation system. *Journal of Software*, 20(2):350–362, 2009.

[38] Anne Håkansson. Portal of research methods and methodologies for research projects and degree projects. In *The 2013 World Congress in Computer Sci- ence, Computer Engineering, and Applied Computing WORLDCOMP 2013; Las Vegas, Nevada, USA, 22-25 July*, pages 67–73. CSREA Press USA, 2013.

[39] Thomas Hofmann and Jan Puzicha. Latent class models for collaborative fil- tering. In *IJCAI*, volume 99, pages 688–693, 1999.

[40] Anna Huang. Similarity measures for text document clustering. In *Proceedings of the sixth new zealand computer science research student conference (NZC- SRSC2008), Christchurch, New Zealand*, pages 49–56, 2008.

[41] Dietmar Jannach, Markus Zanker, Alexander Felfernig, and Gerhard Friedrich.
Recommender systems: an introduction. Cambridge University Press, 2010.

[42] Robert Jäschke, Leandro Marinho, Andreas Hotho, Lars Schmidt-Thieme, and Gerd Stumme. Tag recommendations in folksonomies. In *Knowledge Discovery in Databases: PKDD*

2007, pages 506–514. Springer, 2007.

[43] Joseph A Konstan. Introduction to recommender systems: Algorithms and evaluation. *ACM Transactions on Information Systems (TOIS)*, 22(1):1–4, 2004.

[44] Yehuda Koren. Factorization meets the neighborhood: a multifaceted collabo- rative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 426–434. ACM, 2008.

[45] Daniel Z Lieberman, Suena H Massey, Vilmaris Quiñones Cardona, and Ken- neth P Williams. Predicting content preference: Applying lessons learned from the commercial web to therapeutic software. *Cyberpsychology*, 2(2), 2008.

[46] Greg Linden, Brent Smith, and Jeremy York. Amazon. com recommendations: Item-to-item collaborative filtering. *Internet Computing, IEEE*, 7(1):76–80, 2003.

[47] Pasquale Lops, Marco De Gemmis, and Giovanni Semeraro. Content-based recommender systems: State of the art and trends. In *Recommender systems handbook*, pages 73–105. Springer, 2011.

[48] Christopher D Manning, Prabhakar Raghavan, Hinrich Schütze, et al. *Introduc- tion to information retrieval*, volume 1. Cambridge university press Cambridge, 2008.

[49] Christopher D Manning and Hinrich Schütze. *Foundations of statistical natural language processing*. MIT press, 1999.

[50] Bradley N Miller, Istvan Albert, Shyong K Lam, Joseph A Konstan, and John Riedl. Movielens unplugged: experiences with an occasionally connected rec- ommender system. In *Proceedings of the 8th international conference on Intel- ligent user interfaces*, pages 263–266. ACM, 2003.

[51] Jiangang Yang Mingyong Liu. An improvement of tfidf weighting in text cat- egorization. 2012.

[52] Batul J Mirza, Benjamin J Keller, and Naren Ramakrishnan. Studying rec- ommendation algorithms by graph analysis. *Journal of Intelligent Information Systems*, 20(2):131–160, 2003.

[53] Kensuke Onuma, Hanghang Tong, and Christos Faloutsos. Tangent: a novel,'surprise me', recommendation algorithm. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data min- ing*, pages 657–666. ACM, 2009.

[54] Arkadiusz Paterek. Improving regularized singular value decomposition for collaborative filtering. In *Proceedings of KDD cup and workshop*, volume 2007, pages 5–8, 2007.

[55] Juan Ramos. Using tf-idf to determine word relevance in document queries. In
Proceedings of the first instructional conference on machine learning, 2003.

[56] Al Mamunur Rashid, Istvan Albert, Dan Cosley, Shyong K Lam, Sean M Mc- Nee, Joseph A Konstan, and John Riedl. Getting to know you: learning new user preferences in

recommender systems. In *Proceedings of the 7th interna- tional conference on Intelligent user interfaces*, pages 127–134. ACM, 2002.

[57] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 175–186. ACM, 1994.

[58] J Ben Schafer, Joseph Konstan, and John Riedl. Recommender systems in e- commerce. In *Proceedings of the 1st ACM conference on Electronic commerce*, pages 158–166. ACM, 1999.

[59] Shilad Sen, Jesse Vig, and John Riedl. Tagommenders: connecting users to items through tags. In *Proceedings of the 18th international conference on World wide web*, pages 671–680. ACM, 2009.

[60] Yang Song, Ziming Zhuang, Huajing Li, Qiankun Zhao, Jia Li, Wang-Chien Lee, and C Lee Giles. Real-time automatic tag recommendation. In *Proceed- ings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 515–522. ACM, 2008

[61] Tian Xia and Yanmei Chai. An improvement to tf-idf: term distribution based term weight algorithm. *Journal of Software*, 6(3):413–420, 2011.

[62] Osmar R Zaïane, Jia Li, and Robert Hayward. Mission-based navigational behaviour modeling for web recommender systems.

In *Advances in Web Mining and Web Usage Analysis*, pages 37–55. Springer, 2006.

[63] Zhi-Dan Zhao and Ming-Sheng Shang. User-based collaborative-filtering rec- ommendation algorithms on hadoop. In *Knowledge Discovery and Data Mining, 2010. WKDD'10. Third International Conference on*, pages 478–481. IEEE, 2010.