User- and system initiated approaches to content discovery
Abstract

Social networking has encouraged users to find new ways to create, post, search, collaborate and share information of various forms. Unfortunately there is a lot of data in social networks that is not well-managed, which makes the experience within these networks less than optimal. Therefore people generally need more and more time as well as advanced tools that are used for seeking relevant information. A new search paradigm is emerging, where the user perspective is completely reversed: from finding to being found. The aim of present thesis research is to evaluate two approaches of identifying content of interest: user-initiated and system-initiated. The most suitable approaches will be implemented. Various recommendation systems for system-initiated content recommendations will also be investigated, and the best suited ones implemented. The analysis that was performed demonstrated that the users have used all of the implemented approaches and have provided positive and negative comments for all of them, which reinforces the belief that the methods for the implementation were selected correctly. The results of the user testing of the methods were evaluated based on the amount of time it took the users to find the desirable content and on the correspondence of the result compared to the user expectations.

Keywords: user-initiated content discovery, system-initiated content discovery, content-based filtering, collaborative filtering, recommender systems.
# Table of Contents

1 Introduction ................................................................. 1  
  1.1 Background ........................................................... 1  
  1.2 Problem definition and research question ...................... 1  

2 Theory ............................................................................. 3  
  2.1 Introduction .............................................................. 3  
  2.2 User-initiated approaches ......................................... 3  
  2.3 System-initiated approaches ...................................... 6  
  2.4 Applicability of the approaches ................................. 8  

3 Implementation ............................................................... 10  
  3.1 Implemented content discovery methods ....................... 10  
  3.2 Data used for testing ................................................. 10  
  3.3 Database ................................................................. 11  
  3.4 Multi-level categories list approach ........................... 12  
  3.5 Recommendation based on content-based filtering approach ... 13  
  3.6 Recommendation based on collaborative filtering approach ... 14  
  3.7 Simple search approach .......................................... 17  

4 Evaluation ........................................................................ 18  
  4.1 Evaluation process .................................................... 19  
  4.2 Evaluation form ........................................................ 17  
  4.3 User feedback ........................................................... 20  
  4.4 Results of evaluation ................................................. 23  

5 Conclusion ....................................................................... 25
List of figures

Figure 2.1: A tag cloud with terms related to Web 2.0 .................................. 5
Figure 2.2: Example of the data represented as a tree-like structure ............ 6
Figure 3.1: Database scheme ................................................................. 12
Figure 3.2: Multi-level categories list approach user interface .................. 12
Figure 3.3: Channels data organized as a tree-like structure .................... 13
Figure 3.4: Recommendation based on content-based filtering approach user
interface .................................................................................................. 13
Figure 3.5: Recommendation based on collaborative filtering approach user
interface .................................................................................................. 14
Figure 3.6: Simple search approach user interface ..................................... 17
List of tables

Table 2.1: Utility matrix .......................................................................................... 8
Table 3.1: Open-source recommender engines ......................................................... 16
Table 3.2: Apache Mahout library advantages and disadvantages ......................... 17
Table 4.1: Task 1 - Find a movie to watch with your mother .................................. 22
Table 4.2: Task 2 - Find a movie you haven’t seen which will cheer you up ........... 22
Table 4.3: Task 3 - Find a movie, you haven’t rated or seen, which you think you’ll enjoy ............................................................................................................. 22
Table 4.4: Task 4 - Find a popular documentary ..................................................... 22
Table 4.5: Task 5 - Find an old horror (or any other genre) movie you think you’ll enjoy ............................................................................................................. 23
Table 4.6: Multi-level categories list evaluation ....................................................... 23
Table 4.7: Recommendation based on content-based filtering evaluation ............. 23
Table 4.8: Recommendation based on collaborative filtering evaluation ............. 23
Table 4.9: Simple search evaluation ......................................................................... 24
Table 4.10: Average results ...................................................................................... 24
1 Introduction

In this chapter the problem background, the motivation of the task and goals will be discussed. The problem background and motivation also dive into discussion. Goals are defined according to the formulation of the problem and the scope of the thesis task.

1.1 Background

In the past decade, social media platforms have grown from a pastime for teenagers into tools that pervade nearly all modern adults’ lives due to its nature that allows people to meet other people with similar interests [1]. Social networking has encouraged users to find new ways to create, post, search, collaborate and share information of various forms. Social media users are usually trying to organize themselves around specific interests, such as hobbies, jobs and other daily life categories. However, the ever-growing ocean of content produced online makes it harder and harder to identify streams and channels of interest to an individual user. Within the social media to find specific topics of interest users are usually trying to engage with like-minded people. Unfortunately there is a lot of data in social networks that is not well-managed, which makes the experience within these networks less than optimal. Therefore people generally need more and more time as well as advanced tools that go beyond those implementing the canonical search paradigm for seeking relevant information. A new search paradigm is emerging, where the user perspective is completely reversed: from finding to being found. Within the present research I aim to evaluate two approaches of identifying content of interest: user-initiated and system-initiated. The former comprises ways of organizing content to best facilitate discovery. The latter investigates ways in which the system can make suggestions based on behavioral data about users.

It has always been a top priority for entertainment providers to help consumers to use their services in the best way. Nowadays social media is transforming the way this business is carried out, because it provides many excellent platforms for content discovery. Present thesis research is conducted within the project NextHub. NextHub is a media channel for organizations seeking to increase engagement with end users and customers in a way that is not possible in today’s social media landscape. NextHub is always looking for ways to improve and facilitate the content discovery by their users and serves as an excellent research platform for these goals.

1.2 Problem definition and research question

The problem of choosing the approach that best corresponds to the application needs is emerging – due to the lack of resources only a limited amount of content discovery methods can be implemented and supported. There has been a lot of theoretical research into individual methods of content discovery, and some methods have been compared in depth (e.g. recommender systems). This thesis takes a different approach and aims to compare the practical results of using these methods. More concretely, the goal is to study and compare existing methods and tools for finding content (such as free-text search, tagging, fixed categories, navigational hierarchy, etc), implement the most suitable methods and according to the practical results evaluate their efficiency, output quality and applicability. The research question that this thesis aims to answer is “how do different methods of content discovery compare in their applicability, efficiency and output quality in practice?”
2 Theory

In this chapter the theory behind content discovery field will be discussed. I focus on techniques, technologies and algorithms that can be used by the user to identify the content of interest.

2.1 Introduction

Nowadays the amount of information in the global network grows exponentially. A huge number of new web sites appear; new articles, books, movies and other types of data is uploaded daily. The volume of unfiltered and even unwanted content has exploded, so it’s becoming harder and harder for an average user to handle tons of new information. Therefore the main task of each service provider is to present only relevant content to readers and to simplify the user’s operation with it. More and more new technologies and tools for achieving this goal appear every day. A new search paradigm is emerging, where the user perspective is completely reversed: from finding to being found.

There are two basic classes of approaches for content discovery: user-initiated and system-initiated. User-initiated approach is based on the active form of needed information finding. It may include free-text or boolean search engines, tagging systems, navigating data hierarchy etc. (all the methods will be discussed later in more details). System-initiated approach is the opposite of the first method – user is not actively involved in the content discovery – the system does everything for him using content-based, collaborative filtering and other suggestion techniques.

2.2 User-initiated approaches

Active content discovery strategies entail direct interaction between communicator (user) and target (the content) during which different tactics are enacted to elicit desired information. One of the advantages of these approaches is that there are never any ethical concerns inherent to them – all data is prepared by the content provider and there is no risk in exposing any sensitive or private data to the users who should not see it or have access to it.

Information search is the process of finding material of an unstructured nature that satisfies information needs from within large collections of data. It should be noted that when talking about unstructured data, we are still dealing with data that has some hidden structure characteristic of natural languages. For example, most of the texts have headings, paragraphs, and foot notes [2]. There are several models for the implementation of information discovery today: the Boolean model, the vector model, probabilistic model, a model based on the Bayes approach and so on. Of course, each model has its advantages and disadvantages. Due to the fact that information discovery is not a trivial task the choice of a particular model should be carried out on the basis of the goals of retrieval.

Ellis and Haugan propose and elaborate a general model of information seeking behaviors based on studies of the information seeking patterns of social scientists, research physicists and chemists, and engineers and research scientists in an industrial firm. One version of the model describes six categories of information seeking activities as generic: starting, chaining, browsing, differentiating, monitoring, and extracting.

Starting comprises those activities that form the initial search for information - identifying sources of interest that could serve as starting points of the search. Identified sources often include familiar sources that have been used before as well as less familiar sources that are expected to provide relevant information. While searching the initial sources, these sources are likely to point to, suggest, or recommend additional sources or references [8].
Following up on these new leads from an initial source is the activity of Chaining. Chaining can be backward or forward. Backward chaining takes place when pointers or references from an initial source are followed, and is a well-established routine of information seeking among scientists and researchers. In the reverse direction, forward chaining identifies and follows up on other sources that refer to an initial source or document. Although it can be an effective way of broadening a search, forward chaining is much less commonly used [9].

Having located sources and documents, Browsing is the activity of semi-directed search in areas of potential search. The individual often simplifies browsing by looking through tables of contents, lists of titles, subject headings, names of organizations or persons, abstracts and summaries, and so on. Browsing takes place in many situations in which related information has been grouped together according to subject affinity, as when the user views displays at an exhibition, or scans books on a shelf.

During Differentiating, the individual filters and selects from among the sources scanned by noticing differences between the nature and quality of the information offered. For example, social scientists were found to prioritize sources and types of sources according to three main criteria: by substantive topic; by approach or perspective; and by level, quality, or type of treatment. The differentiation process is likely to depend on the individual's prior or initial experiences with the sources, word-of-mouth recommendations from personal contacts, or reviews in published sources.

Monitoring is the activity of making sure that all the facts about development are recent in an area by regularly following particular sources. The individual monitors by concentrating on a small number of what is perceived to be core sources. Core sources vary between professional groups, but usually include both key personal contacts and publications.

Extracting is the activity of systematically working through a particular source or sources in order to identify material of interest. As a form of retrospective searching, extracting may be achieved by directly consulting the source, or by indirectly looking through bibliographies, indexes, or online databases. Retrospective searching tends to be labor intensive, and is more likely when there is a need for comprehensive or historical information on a topic [8][9][10].

Following the categories of information seeking activities listed above, the following types of information retrieval can be distinguished:

- Free-text search;
- Boolean search;
- Vector space search;
- Retrieval via tags and labels;
- Navigating hierarchy;
- and others.

Free-text search is a good example of active information seeking. It is one of the most common and simple ways of content discovery. In a free-text search, a search engine examines all of the words in every stored document as it tries to match search criteria (text specified by a user). Such search techniques became common in online bibliographic databases in the 1990s.

When dealing with a small number of documents, it is possible for the search engine to directly scan the contents of the documents with each query, a strategy called serial scanning. However, when the number of documents to search is potentially large, or the quantity of search queries to perform is substantial, the problem of full-text search is often divided into two tasks: indexing and searching. The indexing stage will scan the text of all the documents and build a list of search terms (which is usually called an
In the search stage, when performing a specific query, only the index is referenced, rather than the text of the original documents.

The indexer will make an entry in the index for each term or word found in a document, and possibly note its relative position within the document. Usually the indexer will ignore stop words (such as "the" and "and") that are both common and insufficiently meaningful to be useful in searching.

A more advanced version of the simple search is a Boolean retrieval model that was originated in the 1950s.

Boolean retrieval is based on Boolean algebra, named after the English mathematician George Boole (1815-1864). In Boolean retrieval words, strings or other symbols are organized in sets, which are combined with the logical "AND", "OR" or "NOT" (the Boolean operators) [3].

Clean formalism, implementation simplicity and intuitive concept are advantages of the Boolean retrieval model. However, the model has a number of following disadvantages: difficulties in ranking output, equal weight of all terms and complexity of translating a query into a Boolean expression.

Another method of user-based retrieval is vector space search. It involves converting documents into vectors. Each dimension within the vectors represents a term. If a document contains that term then the value within the vector is greater than zero. Such technology has both advantages and disadvantages. The former are the following: retrieval is ranked; terms are weighted by importance, etc. The fact that assumed terms are independent and weighting is intuitive but not very formal represents the latter.

Retrieval via labels and tags has become very popular recently. Labeling and tagging is carried out to perform functions such as aiding in classification, marking ownership, noting boundaries, and indicating online identity. They may take the form of words, images, or other identifying marks.

Tagging has gained wide popularity due to the growth of social networking, photography sharing and bookmarking sites. These sites allow users to create and manage labels (or “tags”) that categorize content using simple keywords. Keywords have been used as a part of identification and classification systems long before these systems were automated with the help of computers. In the early days of the web keywords meta tags were used by web page designers to tell search engines what the web page was about. Today's tagging takes the meta keywords concept and re-uses it. The users add the tags, which are clearly visible and act as links to other items that share that keyword tag [4].

To represent the data by tags, a tag cloud can be used. It is a visual representation for text data, typically used to depict keyword metadata (tags) on websites, or to visualize free form text. Tags are usually single words, and the importance of each tag is shown with font size or color [7]. The example of a cloud tag is shown on the figure 2.1.
The benefits of tagging are the following:
- tags allow more flexible content organizing;
- tags facilitate the annotation process because less or even no knowledge about the system is required;
- tags facilitate content retrieval and discovery.

Apart from these advantages of tagging, research also identifies a number of disadvantages. According to Golder & Huberman[5] and Mathes[6] there are also disadvantages that require serious consideration:
- ambiguity (polysemy and homonymy) – a term or concept that have multiple meanings;
- synonymy – multiple terms which describe the same items or actions. This includes misspellings, spelling variations, conjugated verbs and mixed uses of singular and plural forms;
- level of term specificity (hyponymy and hyperonomy) – e.g. “Siamese”, “cat”, or “animal”;
- tag syntax – multiple words, spaces, symbols etc.

Another method of user-initiated information retrieval is navigating through data hierarchy. To facilitate content discovery and make data finding more convenient a hierarchical data model can be used. All the data is organized and represented as a tree-like structure (see figure 2.2). Each second level category can have many records, but each record only has one parent.
The most common user-initiated approaches have been discussed in this section. List of methods is not finished, it can be endless. New techniques and tools that can help user to find required information are appearing right along.

2.3 System-initiated approaches
Passive strategies involve acquiring information about a target through unobtrusive observation. There is an extensive class of applications that involve predicting user responses to options. Such a facility is called a recommendation system. This approach does not require active participation of the users, because when User #1 is explicitly collaborating with User #2, an algorithmic mediation engine can push some of User #2’s activity on to User #1 without requiring User #1 to make additional effort. The goal of a recommender system is to generate meaningful recommendations to a collection of users for items that might interest them. Suggestions for books on Amazon or movies on Netflix are real-world examples of industry-strength recommender systems.

Unlike user-initiated approaches, system-initiated approaches can have issues with data sensitivity. Since these approaches are based on analyzing data and providing suggestions to the users automatically, they may use the data confidential to other users in their decision processes. This may be undesirable by these users and it is a responsibility of the content platform to make sure that users are aware of this possibility (if it is used) and approve gathering of their private data. The other concern is that these systems may provide suggestions to the users, which they may find offensive or inappropriate. This is harder to deal with and is often handled by using various rating systems (e.g. PEGI), which can be used to block content based on user’s own preferences and settings, as well as legal requirements. Ultimately, it is the responsibility of the content provider to make sure that the recommender (and other similar) systems they use are compliant with ethical norms and legal regulations.

There may be other problems with recommender systems, which are not common for the user-based content discovery approaches. The users need to be able to trust the recommended results, and when the results often seem random or surprising (i.e. the correspondence of the recommendation to the user expectation is low), they may lose
their belief in the ability of the recommender system to deliver the content that they are looking for or may be interested in. There is also an issue of fraud, “flash-mob” kind of attacks on the recommender systems, which can easily bring down the confidence of the users in the system and even the service in general.

Recommender systems differ in the way they analyze data sources to provide suggestions to the user. There are two basic approaches that are used to recommend the item to the user:

- collaborative filtering;
- content-based filtering.

Collaborative filtering methods are based on collecting and analyzing a large amount of information on users’ behaviors, activities or preferences and predicting what users will like based on their similarity to other users. A key advantage of the collaborative filtering approach is that it does not rely on machine analyzable content and therefore it is capable of accurately recommending complex items such as movies without requiring an “understanding” of the item itself. Many algorithms have been used in measuring user similarity or item similarity in recommender systems. For example, the k-nearest neighbors (k-NN) approach [11] and the Pearson Correlation.

Collaborative filtering has two meanings, a narrow one and a more general one [13]. In general, collaborative filtering is the process of forecasting the point of interest for the user using the known preferences (evaluations) of the user group in the recommender systems. For example, with the help of the method music applications are able to predict what kind of music the user likes according to his preferences. The forecasts are prepared individually for each user, but information collected from many participants is used.

In the newer, narrower case, collaborative filtering is a method which can produce predictions about the user’s points of interests by collecting preferences or some other information from many users that can be used to make predictions (collaborating). The basic meaning of the collaborative filtering approach is that if a person A has the same opinion as a person B on an issue, A is more likely to have B's opinion on a different issue x than to have the opinion on x of a person chosen randomly. These predictions are specific to the user, but use information gathered from many users.

Examples of systems that use collaborative filtering include Amazon, iTunes, Netflix, LastFM, StumbleUpon, and Delicious. Thanks to collaborative filtering, a site like Amazon can say, "People who bought books A and B also bought book C."

Content-based filtering is another recommendation approach that is not inferior to the efficiency and accuracy of the previous one. Content-based filtering methods are based on a description of the item and a profile of the user’s preference [12]. In a content-based recommender system, keywords are used to describe the items and a user profile is built to indicate the type of item this user likes. In other words, these algorithms try to recommend items that are similar to those that a user liked in the past (or is examining in the present). In particular, various candidate items are compared with items previously rated by the user and the best-matching items are recommended. This approach has its roots in information retrieval and information filtering research.

In recommender system applications there are two classes of entities, users and items. Users have preferences for certain items depending on their interests, needs etc. Items can be represented by books, movies, groups of interest and other types of data depending on the application domain. The data in recommendation systems itself is represented as a utility matrix which displays users’ ratings on items. For example, utility matrix, representing users’ ratings of movies on a 1 to 5 scale. Blanks represent the situation where the user has not rated the movie (see table 2.1).
The goal of a recommender system is to predict the blanks in the utility matrix depending on past behavior of the user. However, some applications have a slightly different goal – it is not necessary to predict every blank entry in the matrix. Rather, it is only necessary to discover some entities in each row that are likely to be high. In most applications, the recommender system does not offer users a ranking of all items, but rather suggests a few that the user should value highly.

### 2.4 Applicability of the approaches

Basic approaches for content discovery were discussed in this section. Certainly, each approach has both advantages and disadvantages. Some of methods using each approach are difficult and costly to implement, some take a lot of processing time to generate results, some consume a lot of resources either for storing data or analyzing it. Which variant is optimal depends on the specific problem faced by the system.

Even though both approaches server the same goal of helping users to discover and identify content of interest for them, the methods of different approaches should not be compared to each other directly. Each approach is applicable in different situations and works on different kinds of data and for different types of users. When there is a lot of user-generated content available, system-based approaches work best, as it is very difficult to make a hierarchical data model, and the users are less likely to look for a particular item rather than just anything that would suit their needs. On the other hand, when users have precise and specific targets in mind, user-based approaches, which take less liberty in interpreting the user needs, may be more suitable. For example, for social networks system-based approaches are by far more useful to let the users efficiently communicate, while for public libraries which provide textbooks and academic articles, users will know what they are looking for and precision of search is far more important.

User-based approaches excel for simpler, static systems and services which are mainly interesting to the users who know what they are looking for, and the content provider has limited resources or smaller audience, with which it makes no sense to implement the heavier and more complex system-based approach. For small systems with limited content discovery requirements (smaller services, enterprise systems) simple implementations of some of the user-based methods can often be enough. That does not mean though that user-based approaches are only good for small systems – some of the largest services available in the modern world (like Google, Microsoft Bing and many other) use basic searches as a main part of their functionality. This approach works best when there is not enough data available to the system to make a recommendation to the user at the right moment, which will fulfill their goals well enough. This can be illustrated by the backlash that Google has faced when they implemented some history-based suggestions (which is a system-based approach) into their search results (which is a user-based approach). User-based approaches also have no inherent issues with ethics, as they are not based on the private or sensitive data of other users.
System-based approaches are well suited for large services, which provide a lot of non-homogenous content for a wide and diverse range of users. For example, social networks, which work with mainly user-generated content, can greatly increase user activity by supplying them with recommendations of content that similar users or users related to them in some way have generated or consumed before. System-based approaches can be used anywhere, but produce best results when sufficient data is available (compared to the amount of users and their diversity) for their engines to efficiently cluster, categorize and otherwise aggregate and analyze that data. System-based approaches usually face privacy problems and often raise ethical concerns, as they usually automatically parse and analyze content, which is often prepared by users themselves. This content may be sensitive or the results of the recommendation may be offensive to the users. These issues may be very difficult to handle for the service providers, and the responsibility for managing them can be a negative argument for using system-based approaches.

Performance is another consideration when making a choice between different content discovery methods and approaches. Search-based approaches usually can use efficient pre-processed data sets (like indices), which don’t require much storage or processing power. Recommender systems can use both real-time and preprocessed data, but the amount of data they use and store grows much faster, as they need to distinguish correlations between different content and users.
3 Implementation

This chapter describes the process of implementing the application prototype, tools and technologies that were used in the development process.

3.1 Implemented content discovery methods

Following a study of existing methods and tools for finding content, I have implemented the most suitable approaches. Also various recommendation systems for system-initiated content recommendations have been investigated, and best suited ones were implemented.

Four different methods have been chosen for comparison. These methods are very commonly used in modern applications. Most users are already well familiar with them, which makes them good targets for the study, as the users do not need any special learning to use these methods and their skills with using these methods will not skew the experimental results. Moreover, for most service and content providers, the selection of content discovery methods is usually made between these methods, which make the results of this research most useful for the real-world applications. The selected methods are as follows:

- Content-based filtering. Content-based filtering methods are based on an item profile (i.e. a set of discrete attributes and features) characterizing the item within the system.
- Collaborative filtering. Methods are based on collecting and analyzing a large amount of information on users’ behaviors, activities or preferences and predicting what users will like based on their similarity to other users (Apache Mahout library is used).
- Simple search. Method allows finding the occurrence of the search terms in the database by one or more keywords.
- Hierarchical data model. The data is organized into a tree-like structure. The structure allows representing information using parent/child relationships: each parent can have many children, but each child only has one parent.

Present research is conducted within the project NextHub which is a media channel for organizations seeking to increase engagement with end users and customers in a way that is not possible in today’s social media landscape. Therefore in order to compare and evaluate the considered approaches an application prototype has been built. The prototype takes into account all the specific features and requirements of NextHub. It has been implemented as a Java desktop application and it uses flat comma-separated values (csv) files as a database. The application is a single-user system and it does not assume user separation by the role.

Basic functionality allows the user to try in action four different content discovery approaches: two user-initiated and two system-initiated. All the NextHub data is organized through channels. Channels enable public figures, businesses, organizations and other entities to create an authentic and public presence on NextHub. Each user can connect to these channels by becoming a subscriber and then receive automatic updates of their personal news feeds and interact with them and other users through instant messaging. NextHub can also be used for small group communication and for people to share their common interests and express their opinions. It allows people to come together around a common cause, issue or activity to organize and express objectives, discuss issues, post photos and share related content.

3.2 Data used for testing

The implemented prototype is arranged according to the same analogy: test data is organized in the form of channels. A beta-testing version of the application is based on
the movies dataset, as it is the most popular domain among the end users that will be involved in the research. The entire process used a collection of movies provided by MovieLens. The MovieLens datasets were collected by the GroupLens Research Project at the University of Minnesota. The dataset is covering a wide range of dimensions (different genres, styles, locations etc).

This dataset consists of:

- 100,000 ratings on a scale from 1 to 5 from 943 users on 1682 movies.
- Each user has rated at least 20 movies.
- Simple demographic info for the users (age, gender, occupation, zip code).

The data was collected through the MovieLens website (movielens.umn.edu) during the seven-month period from September 19th, 1997 through April 22nd, 1998. The users who had less than 20 ratings or did not have complete demographic information were removed from this data set [14]. The dataset is openly available and all the data is anonymous, without sufficient information, which will allow to identify any real person who provided it, so there are no privacy concerns in using it.

The dataset is organized in a number of data files that consist of different kinds of information. Detailed descriptions of the data files that were used during prototype implementation are:

- u.data - The full data set, 100000 ratings by 943 users on 1682 items. Each user has rated at least 20 movies. Users and items are numbered consecutively from one. The data is ordered randomly. The data structure of the file is: user id | item id | rating | timestamp.
- u.info - The number of users, items, and ratings in the u data set.
- u.item - Information about the items (movies); this is a tab separated list of movie id | movie title | release date | video release date | IMDb URL | unknown | Action | Adventure | Animation | Children's | Comedy | Crime | Documentary | Drama | Fantasy | Film-Noir | Horror | Musical | Mystery | Romance | Sci-Fi | Thriller | War | Western |. The last 19 fields are the genres, a one indicates the movie is of that genre, a zero indicates it is not. The movie ids are the ones used in the u.data data set.
- u.genre - A list of the genres.
- u.user - Demographic information about the users; this is a tab separated list of user id | age | gender | occupation | zip code. The user ids are the ones used in the u.data data set.

3.3 Database

During the application development process some of the files have been modified to provide easier handling (i.e. data unused in this research was removed). For example, file u.item contains 19 fields that represent the movies genres, where value ‘1’ indicates the movie is of specific genre, a ‘0’ indicates it is not. The information in these columns has been merged into one column that represents the movie genre by its id. The database structure is shown in the figure 3.1.
To find a channel of interest the user can use multi-level categories list method, recommendation system (with two different approaches) or a simple search method.

3.4 Multi-level categories list approach

To facilitate content discovery and make finding channels more convenient a hierarchical data model has been used. All the channels (movies dataset in beta-version) are organized and represented as a tree-like structure. Each second level category can have many records, but each record only has one parent. The user interface implemented for using this method is shown on the figure 3.2.
Each channel data stores the link to the second level of hierarchy, and top level of hierarchy is stored as a reference from the second level. The movies data, which is used in prototype, is organized according to this principle: genres function as top and second level of hierarchy. An example of the hierarchy is shown on the figure 3.3.

**3.5 Recommendation based on content-based filtering approach**

Content-based filtering methods are based on a description of the item and a profile of the user’s preference. In a content-based recommender system, keywords are used to describe items. A user profile is built to indicate the type of item this user likes. However, currently the prototype was designed in a way that it has no possibility to work with user profiles. Furthermore, the movies dataset is strictly limited and therefore the recommendation is based only on item attributes such as genre, release year and others.
To get the recommendation using content-based filtering, the user should open a page with the movie he is interested in and press the button “Similar” (see figure 3.4). It produces a list of movies similar to the chosen one. At the moment no algorithms or previously designed solutions were used for the reason mentioned above.

### 3.6 Recommendation based on collaborative filtering approach

![Figure 3.5: Recommendation based on collaborative filtering approach user interface](image)

Collaborative filtering methods are based on collecting and analyzing a large amount of information on users’ behaviors, activities or preferences and predicting what users will like based on their similarity to other users.

As already mentioned the prototype currently was designed in a way that it has no possibility to work with user profile and it is used for test and research purposes. Therefore to get the recommendations the user has to provide a list of movie preferences based on MovieLens dataset.

When the user provides a list of preferences and the data is added to the database, the user can simply press the button “Recommendation” and get a list of suggested movies. The graphical user interface for this feature is shown on the figure 3.5.

Nowadays there is a wide variety of systems that developers can use to provide the potential application users with suggestions. All of them use different methods and algorithms; some are open-source, some are distributed with a license fee.

Basic algorithms that are used in recommender systems are listed below:

- Pearson correlation. Present algorithm is based on calculation of similarity between two users and their attributes, such as items rating. It selects k users that have the highest similarity with active user and compute the prediction from weighted combination of the selected neighbors’ ratings using following formulas:

  \[
  w_{a,u} = \frac{\sum_{i \in I} (r_{a,i} - \bar{r}_a) \cdot (r_{u,i} - \bar{r}_u)}{\sqrt{\sum_{i \in I} (r_{a,i} - \bar{r}_a)^2 \cdot \sum_{i \in I} (r_{u,i} - \bar{r}_u)^2}}
  \]

  where

  - \( w_{a,u} \) – similarity between user \( a \) and \( u \),
  - \( I \) – set of items rated by both users,
\[ r_{u,i} \] – rating given to item \( i \) by user \( u \),
\[ \hat{r}_u \] – rating given by user \( u \)

\[ p_{a,i} = \hat{r}_u + \frac{\sum_{u \in k}(r_{u,i} - \hat{r}_u) \cdot w_{a,u}}{\sum_{u \in k} w_{a,u}} \]

where

\[ p_{a,i} \] – prediction for the active user for item \( i \)

The result is always between -1 and 1. The closer the value of \( r \) gets to zero, the greater the variation the data points are around the line of best fit.

The Pearson correlation, which is widely used, is a popular algorithm for collaborative filtering. However it has some limitations, e.g. bad predictions due to few neighbors (co-rated items).

- Clustering algorithms. Item clustering techniques work by identifying groups of items who appear to have similar ratings. The simplest clustering algorithm is k-means, which partitions items into k clusters. There are three phases in this algorithm:
  - User clustering;
  - Similarity computation;
  - Items recommendation.

  Initially, the items are randomly placed into clusters. Then, a centroid (or center) is calculated for each cluster as a function of its members. Each item's distance from the centroids is then checked. If an item is found to be closer to another cluster, it is moved to that cluster. Centroids are recalculated each time all item distances are checked. When stability is reached (that is, when no items move during an iteration), the set is properly clustered, and the algorithm ends. Other clustering variants include the Adaptive Resonance Theory (ART) family, Fuzzy C-means, Expectation-Maximization (probabilistic clustering) etc [15].

- Some other algorithms include: Bayesian Belief Nets, Markov chains and Rocchio classification.

The basic and most popular open-source engines for providing users with recommendations were chosen for comparison. Their descriptions are provided in the table 3.1.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Programming language</th>
</tr>
</thead>
<tbody>
<tr>
<td>LensKit</td>
<td>Open source toolkit for building, researching and studying recommender systems. LensKit is currently under continual development.</td>
<td>Java</td>
</tr>
<tr>
<td>Apache Mahout</td>
<td>Implementations of machine learning algorithms focused primarily in the areas of collaborative filtering, clustering and classifications.</td>
<td>Java</td>
</tr>
<tr>
<td>Vogoo PHP</td>
<td>Powerful collaborative filtering engine presented as library. There are 4 types of input data in the Vogoo LIB: members IDs, products IDs, ratings, categories. The latest version was released on 2008-03-29.</td>
<td>PHP</td>
</tr>
<tr>
<td>CoFE</td>
<td>Collaborative Filtering engine runs as a server to generate recommendations for individual items, top-N recommendations over all items, or top-N recommendations limited to one item type. User data is</td>
<td>Java</td>
</tr>
</tbody>
</table>
SWAMI | A framework for running collaborative filtering algorithms and evaluating the effectiveness of those algorithms. It uses the EachMovie dataset, provided by Compaq. SWAMI consists of three components: a prediction engine, an evaluation system, and a visualization component. | Java

Crab | A framework for building recommender engines integrated with the world of scientific Python packages. The engine aims to provide a rich set of components from which developer can construct a customized recommender system from a set of algorithms. Crab implements user- and item-based collaborative filtering. | Python

MyMediaLite | Lightweight, multi-purpose library of several recommender system algorithms. MyMediaLite implements rating prediction (e.g. on a scale of 1 to 5 stars), and item prediction based on user-item rating feedback (e.g. from clicks, likes, or purchase actions). | C#

| Table 3.1: Open-source recommender engines |

Apache Mahout framework was chosen as a recommender engine for the implementation of the prototype. Mahout is an open source machine learning Java library from Apache. Mahout uses triple data model such as userID, itemID, value as input (userID and itemID refer to a particular user and a particular item, and value denotes the strength of the interaction). For memory efficiency only numeric identifiers are allowed. The dataset holding all interactions is represented by the DataModel Class.

\[ DataModel \text{ model} = \textbf{newFileDataModel(new File("...")); } \]

Then, for finding similar users, the system has to compare their interactions. There are several methods for doing this. The most popular method is to compute the correlation coefficient between their interactions.

\[ UserSimilarity \text{ similarity} = \textbf{newPearsonCorrelationSimilarity(\text{model})}; \]

The next step to be done is to define which similar users we want to leverage for the recommender. For the sake of simplicity, all those have a similarity greater than 0.1 will be used. This is implemented via a ThresholdUserNeighborhood:

\[ UserNeighborhood \text{ neighborhood} = \textbf{newThresholdUserNeighborhood(0.1, similarity, model)}; \]

At this point the data is prepared and organized for the recommender:

\[ UserBasedRecommender \text{ recommender} = \textbf{newGenericUserBasedRecommender(\text{model, neighborhood, similarity})}; \]

Now to get the list of recommendations the method recommend can be used. Input for the method is userID(1 in the example) and the number of items recommended (5 in the example) [17]:

---

[16] Store in MySQL. Project is no longer maintained.

[17] For memory efficiency only numeric identifiers are allowed. The dataset holding all interactions is represented by the DataModel Class.
List<RecommendedItem> recommendations = recommender.recommend(1, 5);

Just like all the other implementations, Apache Mahout library has both advantages and disadvantages, which are shown in the table 3.2:

<table>
<thead>
<tr>
<th>+</th>
<th>-</th>
</tr>
</thead>
<tbody>
<tr>
<td>Well documented, tested, packaged and supported product</td>
<td>Performance issues on large datasets</td>
</tr>
<tr>
<td>Open source solution</td>
<td>Low precision with small amounts of data</td>
</tr>
<tr>
<td>Easy to use implementation</td>
<td></td>
</tr>
<tr>
<td>Highly scalable library</td>
<td></td>
</tr>
</tbody>
</table>

Table 3.2: Apache Mahout library advantages and disadvantages

3.7 Simple search approach

The search procedure is one of the central procedures in any kind of applications whether it is a web or a desktop application. Simple search by one or more keywords can also be used for channels discovery. It allows finding movies by title and by description. Movies with corresponding title are displayed on the top of the list, while movies with corresponding description are listed next. The example of the graphical user interface built for the prototype application is shown on the figure 3.6. Wide variety of search engines that can perform such functions exists nowadays. Most of them use indexing technology that helps to improve the speed of data retrieval and makes sorting the database much faster. The decision was made not to use any of the search engines because the movies database doesn’t contain a lot of values and a simple query is enough to get the search results and to represent them in the list.
4 Evaluation

This chapter describes the evaluation process and tests that have been conducted in order to form a complete vision regarding the content discovery approaches that have been implemented. The four implemented methods have been compared in a practical experiment and the results produced by the users have been analyzed.

4.1 Evaluation process

The main purpose of the current research was to implement, compare and test selected approaches of information retrieval. In order to compare the approaches and test the application, a test group of 10 persons consisting mostly of university students has been selected. The test group consists of people of various social, ethnical, professional and cultural backgrounds as well as different ages and genders, which makes the selection representative. University students are also the largest target group of most social networks and similar services (like NextHub), which makes them a perfect target group for this research. Ten people were deemed enough to cover a large enough variety of different backgrounds so exclude systematic bias. The entire process used a collection of movies provided by MovieLens. The dataset is covering a wide range of dimensions (different genres, styles, locations etc) and consists of 100 000 ratings on 1682 movies. Primarily the users were asked to rate a number of movies they have watched (at least ten to make the recommendation results more precise). To make movie search more convenient for the user there was possibility to sort, group and filter the movies by any parameter: name, release date, genre etc. Selected movies were evaluated on a scale from one to five (using integer numbers) and the data was sent to the evaluator and stored in the database.

Once the data was stored in the database, the users were provided with the application prototype with modifications made for each user individually. The database for each user was prepared so that only the ratings of that particular user (and the original MovieLens ratings) were included, and no test user was influenced by the results or ratings of other users. The purpose was to explore the prototype and after that to complete the tasks. Users had to try to use as many approaches as possible to complete each task. After that the participants answered a questionnaire and gave general feedback about the prototype and the content discovery approaches. As already mentioned, four approaches were chosen for comparison:

1. Multi-level categories list approach
2. Recommendation based on content-based filtering approach
3. Recommendation based on collaborative filtering approach
4. Simple search approach

In order to evaluate all the methods, the tasks were arranged in a way that most of them could be used for evaluation and be abstract at the same time. It helped the users to use as many approaches as possible to make the overall picture clear. Five tasks listed below were generated for each user.

1. Find a movie to watch with your mother.
2. Find a movie you haven’t seen which will cheer you up.
3. Find a movie, you haven’t rated or seen, which you think you’ll enjoy.
4. Find a popular documentary.
5. Find an old horror (or any other genre) movie you think you’ll enjoy.

During the testing process each user spent approximately 1-1.5 hour of which about 30 minutes were spent on the explanation of the process and reading documentation.
minutes were spent on rating the movies. Finally, the remaining time was spent on completing the tasks, evaluation of the prototype and the content discovery methods as well as writing feedback.

4.2 Evaluation form
In order to obtain the feedback and receive the comments, the following evaluation form was created:
1. Which approach/approaches did you use to complete each task:

<table>
<thead>
<tr>
<th>Task/Approach</th>
<th>Category</th>
<th>Content-based filtering</th>
<th>Collaborative filtering</th>
<th>Simple search</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2. Rate your satisfaction using each approach:
• Multi-level categories list approach

<table>
<thead>
<tr>
<th>Criteria/Task</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time taken (1-least, 5-most)</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Quality of outcome (1-worst, 5-best)</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

How could your experience have been improved?
______________________________________________________________________
______________________________________________________________________
Give the general feedback about using categories list approach
______________________________________________________________________
______________________________________________________________________

• Recommendation based on content-based filtering approach

<table>
<thead>
<tr>
<th>Criteria/Task</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time taken (1-least, 5-most)</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Quality of outcome (1-worst, 5-best)</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

How could your experience have been improved?
______________________________________________________________________
Give the general feedback about using content-based filtering approach
• Recommendation based on collaborative filtering approach

<table>
<thead>
<tr>
<th>Criteria/Task</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time taken (1-least, 5-most)</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Quality of outcome (1-worst, 5-best)</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

How could your experience have been improved?

Give the general feedback about using collaborative filtering approach

• Simple search approach

<table>
<thead>
<tr>
<th>Criteria/Task</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time taken (1-least, 5-most)</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Quality of outcome (1-worst, 5-best)</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

How could your experience have been improved?

Give the general feedback about using simple search approach

All the answers and comments received were processed and analyzed.

4.3 User feedback
The conducted research seems to indicate that on average almost all the approaches often end up complementing each other. Firstly, users were asked to complete each task using as many approaches as possible and check the approach they think is the best option for finding the answer for each task. Tables 4.1-4.5 represent the result of the first phase of testing.

<table>
<thead>
<tr>
<th>User</th>
<th>Category</th>
<th>Content-based filtering</th>
<th>Collaborative filtering</th>
<th>Simple search</th>
</tr>
</thead>
<tbody>
<tr>
<td>AA</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>AK</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>EI</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>GC</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>RD</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>VT</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 4.1: Task 1 - Find a movie to watch with your mother

<table>
<thead>
<tr>
<th>User</th>
<th>Category</th>
<th>Content-based filtering</th>
<th>Collaborative filtering</th>
<th>Simple search</th>
</tr>
</thead>
<tbody>
<tr>
<td>AA</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>AK</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EI</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GC</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RD</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VT</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>YK</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>LP</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>YH</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HK</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.2: Task 2 - Find a movie you haven’t seen which will cheer you up

<table>
<thead>
<tr>
<th>User</th>
<th>Category</th>
<th>Content-based filtering</th>
<th>Collaborative filtering</th>
<th>Simple search</th>
</tr>
</thead>
<tbody>
<tr>
<td>AA</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>AK</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>EI</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GC</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>RD</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VT</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>YK</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>LP</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>YH</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>HK</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.3: Task 3 - Find a movie, you haven’t rated or seen, which you think you’ll enjoy

<table>
<thead>
<tr>
<th>User</th>
<th>Category</th>
<th>Content-based filtering</th>
<th>Collaborative filtering</th>
<th>Simple search</th>
</tr>
</thead>
<tbody>
<tr>
<td>AA</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>AK</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>EI</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GC</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RD</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VT</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>YK</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LP</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>YH</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>HK</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 4.4: Task 4 - Find a popular documentary

<table>
<thead>
<tr>
<th>User</th>
<th>Category</th>
<th>Content-based filtering</th>
<th>Collaborative filtering</th>
<th>Simple search</th>
</tr>
</thead>
<tbody>
<tr>
<td>AA</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>AK</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EI</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>GC</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RD</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VT</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>YK</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LP</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>YH</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>HK</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table 4.5: Task 5 - Find an old horror (or any other genre) movie you think you’ll enjoy

<table>
<thead>
<tr>
<th>User</th>
<th>Time taken</th>
<th>Quality of outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td>AA</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>AK</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>EI</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>GC</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>RD</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>VT</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>YK</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>LP</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>YH</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>HK</td>
<td>5</td>
<td>3</td>
</tr>
</tbody>
</table>

### Table 4.6: Multi-level categories list evaluation

<table>
<thead>
<tr>
<th>User</th>
<th>Time taken</th>
<th>Quality of outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td>AA</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>AK</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>EI</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>GC</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>RD</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>VT</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>YK</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>LP</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>YH</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>HK</td>
<td>2</td>
<td>4</td>
</tr>
</tbody>
</table>

### Table 4.7: Recommendation based on content-based filtering evaluation
Also, after beta-testing users evaluated the time they spent to complete each task and rated general satisfaction using each approach (see table 4.6-4.9).

### 4.4 Results of evaluation

<table>
<thead>
<tr>
<th>Method</th>
<th>Time taken</th>
<th>Quality of outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multi-level categories list</td>
<td>4</td>
<td>3,2</td>
</tr>
<tr>
<td>Recommendation based on content-based filtering</td>
<td>3,1</td>
<td>3,8</td>
</tr>
<tr>
<td>Recommendation based on collaborative filtering</td>
<td>2,2</td>
<td>2,6</td>
</tr>
<tr>
<td>Simple search</td>
<td>2,5</td>
<td>3,4</td>
</tr>
</tbody>
</table>

#### Table 4.10: Average results

In the theoretical section it has been shown that we can’t draw meaningful conclusions from comparing methods between the two content discovery approaches and the selection of approach is usually based on the nature of the service and which kinds of data are available to the service provider.

On the other hand, methods within each approach can be compared with each other, as they are based on roughly the same kinds of data and use similar means to provide results to the user.
For the user-based approach, the methods of simple search and multi-level categories list were evaluated. The results of the evaluation (see table 4.10) show that multi-level categories list method is more time-consuming and produces less accurate results. However, according to the users’ feedback, multi-level categories list method is still useful as it is helpful when special search criteria is not set and the users do not want to expose any of their preferences to the system (e.g. for privacy reasons). The simple search method produces the results very quickly and is quite accurate when the user knows how to use the search engine and has a specific goal in mind.

For the system-based approach, the methods of recommendation based on content-based filtering and collaborative filtering were evaluated. The results of the evaluation (see table 4.10) show that there was no clear winner between the two methods – collaborative filtering is the quickest to use, but produces the most imprecise results, while content-based filtering is slower but provides far more relevant recommendations.

This shows that both methods have their uses, and depending on the available resources and preferences for the service provider either of them can be used successfully.

Small bugs were found during the beta-testing by users that later were fixed. These bugs were not relevant to the method implementations themselves and therefore did not affect the experimental results. The developed prototype can be further improved – more items can be added to the database, some features affecting usability can be improved or replaced. However, in general the beta-testing procedure showed that the experimental results have corresponded with the expectations: all used approaches have different strengths and can be used depending on the nature of the service and the data and implementation constraints.
5 Conclusion
In this paper several different content discovery approaches were briefly described. The best choice among these ultimately depends on the purpose of use. During the thesis project various recommendation systems for system-initiated content recommendations were investigated. Additionally, two methods such as free-text search and fixed categories list were selected for comparison and implementation. The prototype to perform the beta testing and practical evaluation of all the selected methods was created.

The work on the project demonstrated that content discover area is very extensive and more in-depth research should be conducted. Nowadays hundreds of different approaches exist to help the end-user to reach and discover the desired information. Moreover, new recommender systems and algorithms appear all the time.

Beta-testing that was performed on ten students that are potential users of the application. The testing has shown that the developed prototype can be further improved – more items can be added to the database, some features affecting usability can be improved or replaced. Better results might be obtained by improving the data model and expanding the amount of available data.

The analysis demonstrated that the users have used all of the implemented approaches and have provided positive and negative comments for all of them, which reinforces the belief that the methods for the implementation were selected correctly. The results of the user testing of the methods were evaluated based on the amount of time it took the users to find the desirable content and on the correspondence of the result compared to the user expectations.

The results of the evaluations show that it can be difficult to answer the research question. However, some basic conclusion can be made - multi-level categories list method is more time-consuming and produces less accurate results. The simple search method produces the results very quickly and is quite accurate when the user knows how to use the search engine and has a specific goal in mind. For the system-based approach collaborative filtering is the quickest to use, but produces the most imprecise results, while content-based filtering is slower but provides far more relevant recommendations.

The implemented methods can be seen as complementing each other; so it is impossible to choose only one that will perfectly work in the application. The best choice is to implement all of these methods, blended together in a way that works well within the particular service or application. For example, simple search can be used when search criteria are strictly defined; otherwise recommender systems can be used. If the resources are limited to build all the described approaches, deeper investigation should be performed to make a choice depending on the number of factors: the application purpose, target audience, set of data etc.

In current research the most common approaches for content discovery were reviewed and implemented; however there are a lot of other methods that are worth to review. A lot of new technologies and algorithms are appearing everyday therefore it makes it difficult to predict which content discovery approach meets the application needs in the best way.
References
[14] http://grouplens.org/datasets/movielens/ (last access 24-12-2014)