Chapter 10

Theoretical Foundations for Recommender Systems

Mirza Zaeem Baig Hasina Khatoon[1] *Syeda Saleha Raza*[2]
and
Muhammad Qasim Pasta[3]

1.1 Introduction

Recommender system (RS) is a software that provides suggestions to a user in decision making process. The decision making may be for commercial purpose (What things to buy?), for personalized applications (What movie to watch and which music to listen to?), or simple information retrieval (What are the most relevant papers for the topic?). These systems have become an important component of almost all applications that relate to some form of information retrieval and processing by using various search techniques. This chapter provides the background to the theoretical foundations of recommender systems. We start with a traditional approach and discuss some commonly used definitions of recommender systems. After justifying the need for such systems in the current scenario of information overload, application areas where recommender systems can be useful and productive are discussed. It includes the discussion on both existing and possible future areas of applications. The list presented is not exhaustive, as many areas have opted to add value to their applications by integrating with some form of recommender system. The next section gives a brief overview of the phases through which recommender system passes in order to perform its function. The types of recommender systems are discussed giving their advantages and disadvantages, and content-based recommenders, collaborative filtering based recommenders, hybrid recommenders, image based recommenders, and graph database based recommenders are discussed in detail. After a brief overview of the problems identified with the current recommender systems, some datasets are listed that may be used for research and evaluation of various recommender system. The chapter ends with the cited references.

[1] National University of Computer & Emerging Sciences (FAST-NU), Karachi, Pakistan. Emails: k143054@nu.edu.pk, hasina.khatoon@nu.edu.pk
[2] Habib University, Karachi, Pakistan. Email: saleha.raza@sse.habib.edu.pk
[3] Usman Institute of Technology, Karachi, Pakistan. Email: qasim.pasta@ieee.org

1.1.1 Definitions of recommender systems

There are many definitions of recommender systems found in literature. Recommender systems are defined as a decision-making strategy in the presence of a large, complex set of information [1]. It allows for listing of personalized contents and services with prioritization and personalization. In the current scenario of information overload from the Internet, it makes possible the timely access to important information. It is also referred to as information filtering system that deals with the problem of information overload [2]. The purpose of recommender systems is to provide support to users in terms of additional knowledge in the form of recommendations and to provide hints about products and systems.

A good definition from [3] states that the Recommender Systems are systems that, based on information about the past patterns and consumption patterns in general, recommend new items to the user. It is a subclass of information filtering system that seeks to predict the 'rating' or 'preference' that a user would give to an item [4]. According to Seroussi [5], discovery assistance is a better term for such systems and he argues that the recommender systems are systems that help users discover items they may like.

1.1.2 The need for a recommender system

In this era of web marketing, these systems play an important role. The goal of recommender system is to generate meaningful recommendations to users for items or products that might interest them. It fetches more revenue for product providers and adds value to products and services.

The rising trend towards the use of internet for an ever-growing set of applications has forced users to use a number of search engines. A search engine is helpful in finding out about an object and results in the form of listing of all possible sources of information from all over the world. For every query made to a search engine, there are thousands of links suggested to the user. The amount of information obtained is very large and a user can find it difficult to filter and narrow down the search to explore further. Moreover, a large amount of irrelevant information is also obtained as a result of the search. This is because the search engine relies on pattern matching and finding similarities. This problem of information overload from the Internet can be alleviated with the use of recommender systems. Using various filtering techniques, a recommendation system narrows down the information on the basis of the behaviour and inclination of the user.

1.2 Applications of recommender systems

There are a number of areas where recommender systems are applied and used. Some of these have been in use since the arrival of the Internet. There are a number of other areas where the deployment of recommender systems can be useful as this is also expected to increase the efficiency of the underlying applications. The following subsections cover these areas. The presented applications may not be an exhaustive list, because many more areas of applications are adding recommender systems to explore their benefits and usefulness.

1.2.1 Current use of recommender systems

1.2.1.1 Product recommendation (e-commerce)

These RSs are meant mostly for commercial applications. For example, if we show the intention to buy a book having a title/author/area, the result would be accompanied with recommendations for similar titles or other books from the same author or other books from the same area. In addition, information is also provided about other buyers who bought the same book and also some of the additional book suggestions in the same area or companion books which others purchased. Online marketing of apparels and clothes also increase their sales by adding recommender systems.

1.2.1.2 Movie or music recommendation (entertainment)

These RSs are personalized according to the user's likes and previous accesses. Such systems should have a profile of the user or similar users to give a better and more accurate recommendation. It may have a commercial aspect from the sellers/distributors of the genre of movie or music.

1.2.1.3 Scholarly search and news articles

These RSs ease the search process by profiling the user and other users who search from the same area or are involved in the same area of research. Political inclinations and the regional environment are also considered for personalized newsletters and news services.

1.2.1.4 Services

These RSs are built to provide recommendations for travelling, consultation, most suitable shopping malls, hospitals, etc.

1.2.2 More areas for recommender systems

Most of the time, we associate recommender systems with online retail trade, but there are many emerging areas where recommender systems have found their use. Designers of various applications find value addition to their products by adding a recommender system. Decision making is therefore made easier in these applications. Some of the emerging areas are discussed below:

1.2.2.1 Recommend courses for students

On an online registration system for students of a University, this system would return a list of courses (based on the past courses and grades) where the student has the potential to do well. This would work as the advisory service that would include the past pattern for the same academic program.

1.2.2.2 Perform career counselling

Based on the academic background and the social status of the user, proper career counselling can be performed by the system. In addition, the current trends towards a particular field gathered by analysing users would help in the decision-making process.

Besides the above, many non-traditional areas have started using recommender systems. Some of these areas are: tourist guides, financial investments, match-making, acquiring of property, etc.

The next section of the chapter describes algorithms and types of recommender systems. It discusses about the phases of recommender system and then discusses in detail about different ways recommender systems could be implemented and used.

1.3 Algorithms and theoretical foundations of recommender systems

Work in the domain of RSs has been continuing since the mid-1990s [6]. Since then, various useful studies on recommender systems have been done. Some have surveyed the field of recommender systems [6] [7] [8] [9], some have described different approaches to recommendations [6] [10] [11] [12] [13] [14], some have thrown light on the advantages, disadvantages, and problems related to different approaches [6] [8] [15], while others have tried to overcome these issues. In this section, we will provide comprehensive detail of recommender systems, along with the description of the broad categories that encompasses the recommender systems.

Recommender system can be defined as the system that predict and recommend most suitable items to the users based on their characteristics. In order to do so, the recommender system goes through a set of phases. These phases are described in the next subsection.

1.3.1 Phases of recommender systems

RSs go through the set of following phases to provide recommendations:

1.3.1.1 Information collection

In this phase, information related to user is collected. Using this information, a user profile is constructed. This profile is essential for the RSs in order to provide efficient predictions. There are different ways of collecting such information including implicit, explicit, and hybrid feedback from the user. Implicit feedback doesn't need user's input, rather the RS implicitly collects information through different techniques like user's behaviour on the system, pages visited, products clicked, etc. Explicit feedback requires user's input to collect information and construct the profile. Generally, this is done through developing a user profile by letting the user fill different web forms that collect the information and stores it. Hybrid feedback combines both implicit and explicit information collection techniques. [16]

1.3.1.2 Learning phase

In this phase, user's features are filtered out and the relevant ones are extracted which could be used in the prediction phase.

1.3.1.3 Prediction/Recommendation phase

In this phase, the prediction or recommendation is made as to which items the user may prefer. The prediction could be made solely by using the user's profile, or it may use other techniques to provide recommendations like item similarity measures, machine learning techniques, Bayesian techniques, decision trees etc.

1.3.2 Types of recommender systems

Although RSs can be categorized into various categories, we have classified them into five broader ones based on the approach they follow to provide recommendations to the users. These categories are described in detail below:

1.3.2.1 Content Based Recommenders

Introduction

The type of recommender systems that provide recommendations of items to a user based on the description of item and user's interest. For example, if the user has liked a particular type of clothing item – which has a feature or description like 'cotton jeans' – in the past, the system would recommend to this user, the items that have the similar features or description.

Discussion

In content based systems, an item profile and user profile is constructed. Item profile is a collection of records that represent different characteristics of the item. For example, particular clothing item would have its size, colour, price, fabric etc. as the specific records of its item profile. Similarly, a user profile contains certain information of the user that is useful for the RS to produce recommendations. User profile can be constructed through explicit information collection from the user or implicit methods for information collection about the user as discussed above. RSs that produce user profiles implicitly are more preferred over the ones that require users to provide explicit information, because most of the time, users do not prefer to provide information, or the information they provide is not authentic and/or accurate [6] [17].

Once the item and user profiles are constructed, the RS then combines this information to generate recommendations for a particular user. A recommendation of an item ‘T_i’ for a user ‘U_i’ is generated by considering ratings ‘R (U_i, t)’ given by user ‘U_i’ to the items ‘t’, where $t \in T$(T is the set of items) and are similar to item ‘T_i’ based on the item profile. Similarity among the items can be computed by using any similarity measures such as cosine distance and/or Jaccard distance [17].

Recommendations in such cases could also be generated through classification algorithms, where the probability that a user would like an item could be estimated. These algorithms also help in generating top N list of recommendations for a particular user. Other methods for producing content based recommendations are Bayesian classifiers and machine learning techniques like clustering, K-nearest neighbours, decision trees, and neural networks [6].

Content based recommender systems have several advantages, some of which are described by Poonam et. al. in [8]:

i. Explicit rating system in this approach provides users an independence and way to develop their own user profile in the system.
ii. They can still recommend items that are not yet seen by any user. This is advantageous in the case when a new user joins the system.
iii. They provide transparency to the user by giving explanation about how this recommender system works.
iv. They do not need a large amount of memory to store data and compute recommendations.

Poonam et. al. [8] also discussed the disadvantages that a content based recommender system could have. They are as follows:

i. It is difficult to extract item features in certain domains.
ii. They work on the item similarity approach so it suffers from overspecialization problem (later discussed in section 4 of the chapter).
iii. It is hard to acquire feedback from the user about a particular recommendation. Therefore, it is not possible to determine if the recommendation made is correct or not.
iv. It is harder to construct user profiles, as users do not provide information about them explicitly or easily.

1.3.2.2 Collaborative Filtering Recommenders

Introduction

Collaborative filtering (CF) recommenders are the most popular among other recommendation approaches [8]. They recommend items to the user based on the ratings given to the items by certain similar users. For example, let us consider User A is similar to User B. So, if User B has liked item A then it is likely that User A would also like item A. In this way, collaborative filtering recommenders construct a utility matrix of users and items, which represents which user likes what items. This utility matrix is then used to compute similarity among users and predict ratings of unrated items by a certain user. Table 1.1 shows an example utility matrix. In this matrix, Users A-D have rated products P1 – P7 on a rating scale of 1-5. Notice that, from this utility matrix, we can infer the similarity of users by seeing what ratings they have given to certain products. As an illustration, see User A and User B, they seem to like similar products like 'P1'. So, they could be said similar to each other. Now, see User A and User C, they seem to give opposite ratings to products P4 and P5. So, they could be said dissimilar to each other.

Table 1.1 Example Utility Matrix

	P1	P2	P3	P4	P5	P6	P7
User A	4			5	1		
User B	5	5	4				
User C				2	4	5	
User D		3					3

However, we don't usually infer the ratings directly by looking at the table. The CF recommenders use different similarity measures to compute similarity among users and items, which we would discuss in the upcoming subsections.

Discussion

Collaborative filtering process works through the utility matrix as in Table 1.1. The focus here is to predict the ratings for the unrated items based on the available data in the matrix, and producing recommendations later on. The following process is followed [6]:

1. Each entry say 'T_{ij}' represents a rating given by user 'i' to the item 'j' in the matrix. The rating is numeric, and it can also be 0, which means that the item 'j' is not yet rated by the user 'i'.
2. The missing ratings are predicted by first computing the similarity between items and users. So, the most similar 'N' users to the user 'i' are found and similarly most similar 'N' items to the item 'j' are found.
3. Using the similarities calculated in step 2, the output is generated. The output consists of the **prediction** of the value 'T_{ij}' and a list of top N recommended items for the user.

Collaborative filtering could be done through two major techniques:

1. **Memory based:**
   In memory based techniques, the entire or a sample of data from the utility matrix is used. In this approach, unknown ratings are predicted and

similarities among users and items are calculated using the rating information in the utility matrix, and a list of 'N' items to be recommended to a particular user is generated.

The memory based approach is classified into two types [6]:

i. **User based:** In the user based approach, similarity between users is computed and then the rating of user 'i' for an item 'j' is predicted based on what average rating has been given to this item by the similar 'N' users.
ii. **Item based:** In item based approach similarity between items is computed and then the rating of user 'i' for an item 'j' is predicted based on what average rating the user 'i' has given to the 'N' similar items.

Similarity Measures: There are different similarity measures that could be used to compute the similarity between users and items. Some commonly used are [6]:

i. Cosine Similarity
ii. Correlation based Similarity
iii. Pearson Correlation
iv. Cosine Vector Similarity
v. Adjust Cosine Vector Similarity
vi. Spearman Correlation
vii. Gaussian Kernel

2. **Model based:**
In model based approaches, recommendations are provided by constructing statistical models for predicting the ratings. Not all of the data and information is used in this approach. Some of the model based approaches for estimating the probability of ratings are [6]:

i. Cluster based Models
ii. Bayesian Networks
iii. Latent Factor & Matrix Factorization

CF scheme has its own advantages, which are explained by Poonam et. al. in [8]:

i. Implementation of recommender systems is made easy through memory based CF technique.
ii. Data addition is made easy and incremental in memory based CF technique.
iii. Prediction performance and accuracy is improved through Model based CF technique.

CF scheme has its disadvantages as well. Some researchers have discussed them as following [8] [18]:

i. CF recommenders require huge amount of data to make recommendations.
ii. CF recommenders require a huge amount of computation power to compute the results.
iii. In CF recommenders, the utility matrix often gets sparse which affects the quality of recommendations.

1.3.2.3 Hybrid Recommenders

Introduction

The type of recommenders where multiple recommendation techniques are combined together to produce a single unified recommendation system is called hybrid recommenders [6]. The basic idea behind this approach is to overcome the drawbacks of individual recommendation techniques by combining the advantageous features of different techniques together, and hence producing a unified recommender [8]. Hybrid recommender systems can be implemented in various ways, some of them are discussed by [6] & [8] as following:

- Implement different recommendation techniques individually and combine their predictions & recommendations.
- Consolidate some features of content based technique into the other one.
- Consolidate some features of collaborative filtering technique into the other one.
- Develop a unified recommender system combining multiple techniques together.

Discussion

The hybrid recommender systems are classified into seven classes that are explained by [8] & [15]:

i. **Weighted:** Scores from multiple techniques are combined together to produce a single recommendation.
ii. **Switching:** Based on the context, the recommender system switches between the available recommendation techniques to produce a recommendation.
iii. **Mixed:** Recommendations from a combination of different recommendation techniques are combined and produced at the same time.
iv. **Feature Combination:** Features from several different recommendation data sources are combined together in a single recommendation algorithm.
v. **Cascade:** One recommender with higher priority refines the results produced by the lower priority recommenders and produces the final recommendation.
vi. **Feature Augmentation:** One recommenders output is used as an input to the other recommender.
vii. **Meta Level:** One recommender’s model learned is used as an input to the other recommender.

1.3.2.4 Image Based Recommenders

Introduction

The type of recommender systems that provide recommendations to the users based on the similarity between the images of the products. The similarity is computed by focusing on different features of the product like style, colour, size, shape, texture etc.

Discussion

There are different approaches that are being followed to develop image based recommender systems. These approaches are commonly based on different machine learning techniques encompassing supervised and unsupervised learning approaches. Some of the used techniques involve simple feed forward neural networks, convolutional neural networks, and SVMs etc. As an illustration, Table 1.2 provides an overview of different approaches being used in different scenarios for image based recommendations.

Table 1.2 An overview of Image Based Recommendation Systems

Research Paper	Scenario	Approach
Dey et. al. [19]	Smartphone based image recommendations	SVMs
Saxena et. al. [20]	Shoe Recommendations	Developed own filtering based algorithm focusing on unsupervised learning.
Wang et. al. [21]	Handbag Recommendations	Joint learning of attribute project, and SVM
McAuley et. al. [22]	Complimentary Objects Recommendation	Convolutional Neural Networks along with some other approaches
Bell & Bala [23]	Interior Design and Product Recommendations	Convolutional Neural Networks

1.3.2.5 Recommender Systems using Graph Databases

The goal of recommendation systems is to generate meaningful recommendations to enhance the user experience in the context of choosing a possibility. This is usually done by predicting the interest of a user based on various type of information [24] [25] [26]. The usage of graphs in recommendation systems isn't something new and the people working in domain of recommendation systems have been using graphs in different ways since early days of recommendation systems [25] [27] [28]. Aggarwal et al. used an approach based on graph algorithms for content based filtering [27].

Silva et al. used graph to represent the social network in order to recommend friends in a social network based on the topology of the network graphs [29]. Huang et al. used a two layer graph approach for a digital library recommendation system [25]. An attempt to produce a hybrid collaborative filtering and content filtering based online newspaper recommender using graphs is discussed in [28].

There has been tremendous work done in last decade to develop new approaches for recommendation systems. The performance of a recommendation system is heavily dependent on the processing of a lot of users' historical data [30]. Graph databases (GDBs) have introduced new perspective to deal with interconnected data in huge sizes. Despite being fairly new, GDBs provide opportunity to improve the performance of applications dealing with huge amount of interconnected data and recommendation systems are one such example.

Introduction

A graph (data structure to hold connected data), is a collection of edges and vertices. These vertices, also known as nodes, are used to represent entities in a domain such as a person, an actor, a place or an item. The edges represents the relationship among vertices such as friendship relationship between two persons, or purchase relationship between an item and a customer [31]. The vertices and edges may also have properties to show further details about them. In case of vertices, examples of such properties could be age of a person or price of an item whereas in case of edges, it could be date of purchase of an item by a customer [32].

Graph databases, usually categorized as NoSQL databases, are aiming to resolve everything using graphs. They focus on entities (vertices or nodes) and the relationships (edges) between them. A graph database stores data in the graph format in which vertices (nodes) and edges (relationships) are the building blocks. GDBs are optimized for highly connected data and can become an effective tool for modelling interconnected data. Due to diversification in structure, the graph database has been adapted in various domains such as social networks, geographic information systems [33], e-commerce [34] [35] [36], and recommendation systems [37] [38] [39].

Since graph databases are optimized for highly connected data, they give performance advantage over relational databases, as relational databases provide limited capability to capture the semantics. In today's era of big data, where we have huge data to process and analyse, the handling of such complex data of interactions becomes next to impossible in conventional databases. In schema based databases, such as relational databases, the schema itself puts a limitation on how the information could be stored, and change in the schema requires a manual process in order to adapt new data. This is quite infeasible when we are exploring new kind of data on daily basis [26]. Therefore, it has been considered that relational databases are not suitable for storing relationship based data [24]. It is also important to mention that all traversals in graph databases are localized and not required to explore unrelated data which is a problem with SQL [40]. Graph databases, such as, Neo4j, InfoGrid, CosmosDB, and Infinite are popular in the market these days.

In graph databases we can model almost anything by defining entities, as vertices, and the relationship among these entities, as edges. Different graph databases support different type of structure that can be modelled, but most common type supported by most of the graph databases is Property Graph which is a multi-graph consisting of attributed, labelled, and directed vertices and edges. Here, attributed means that an entity or relationship may have different attributes associated with them such as age, price etc. Labelling allows to group similar entities together.

Discussion

Broadly we can divide recommendation systems into two basic parts: Pattern Discovery and Pattern Application. As the name suggests, in Pattern Discovery, the system discovers the potential patterns that can be helpful in recommendation for a user. Such patterns can be discovered through an algorithmic approach in which a dataset is processed using machine learning algorithms to identify the patterns which were not identified earlier. This is the area which got primary focus in the last decade [41] [42] [43]. Another mechanism which can help to identify the patterns in the data is visual analytics, in which domain experts identify the hidden patterns by using graph visualization [44] [45] [46]. Yet another way to learn the patterns is through business experts who understand and know the business in detail, because of their experience in the domain [47].

Once a pattern has been found either by using machine learning algorithms or through business experts, the next step is to implement these patterns in business applications in order to generate recommendations for users. The success of a recommendation is not only dependent on how effectively it can identify the patterns, but also dependent on the ability to apply identified patterns in the business application. We may have to apply these patterns offline when recommendations are not time critical, for example, in case of generating recommendations for an e-mail campaign or identifying potentials friends for a person. Such jobs can be performed in a batch-mode and extensive algorithms can be used, in which it may be required to perform multiple iterations on the data.

However, there are applications in which recommendations need to be generated between a web request and a web response. Recommending another restaurant while a user is viewing any specific restaurant or recommending an item while user requests to view a specific item are examples of such cases. For such cases, we need systems that can generate recommendations within milliseconds which may include execution of complex queries on a huge data in order to match certain patterns. Graph databases are appropriate candidates for such cases, as they are optimized to deal with interrelated data, and hence find patterns quite efficiently as compared to other database models.

Content based filtering and Collaborative filtering are naturally supported by graph databases. We can calculate recommendation using graph in a similar fashion as we do in other recommendation systems. We can define a similarity function to compare set of nodes. This similarity function can be used to find most similar nodes which can be returned to a user as recommendations. Property Graph model allows us to store pre-calculated similarity as a property on the edges. We can use numerous already available methods to calculate similarity between two nodes such as Cosine similarity, Pearson correlation or any other method which calculates items' frequency [24] [48].

The fundamental operation to retrieve information from the graph is traversal, which is walking along the elements of a graph [31]. In graph, traversal is a localized operation unlike relational databases. It means that in order to travel from one node to another node, we don't need any global information and each node and edge act as mini-index in the graph. The absence of need of global indexes for traversal means that graph databases have no performance impact on the size of the graph which is quite a contrast from the case of SQL [26].

Unfortunately, no standardization has been made for traversal based languages which results in different implementation of languages and frameworks for this purpose. The most common implementations are Cypher Query Language for Neo4j Graph database[49], GraphQL developed by Facebook [50], Gremlin which can work with different graph databases including Neo4j [51], and SPARQL which is used for RDF based databases [52].

The next subsection gives an overview of popular datasets that have been used for evaluating & comparing recommender systems.

1.3.3 Datasets for recommendations

This section describes some publicly available datasets to evaluate recommendation systems. Some of these datasets contain only user-item rating data and, therefore, are suitable for collaborative filtering techniques, while others also contain product catalog information which can be used to make content based or hybrid recommendations. A brief description of these datasets is given below:

1.3.3.1 Movielens

Movielens dataset [53] is one of the most popular datasets used for recommendations. This dataset is compiled and made available by grouplens, a research lab in Department of Computer Science and Engineering at the University of Minnesota. The dataset was gathered through the Movielens website where users specify their preferences for movies in the form of 5-star ratings. The data is in a typical user-item rating format and contains 20 million ratings of 27,000 movies by 138,000 users. In this dataset, each user is represented by an id and no demographic information of the user is available. However, the published data set contains only those users that have rated at least 20 movies. For movies, in addition to IDs, their titles and genre are also available. This dataset is most suitable for collaborative filtering technique.

1.3.3.2 Jester

Jester [54] is a research project of UC Berkeley laboratory for Automation, Science and Engineering, to study social information filtering. The data is collected from Jester online joke recommendation website (http://eigentaste.berkeley.edu) and contains continuous ratings of jokes by anonymous users on a scale of -10 to +10. Multiple datasets have been collected over different time spans and are available on Jester's website. The largest among them is the size of 4.1 million ratings of 100 jokes by 73,421 users. This dataset is also well-suited for collaborative filtering in scenarios where large number of users have rated a small number of items.

1.3.3.3 BookCrossing

BookCrossing dataset [55] dataset has been gathered through a 4-week crawl from BookCrossing website (http://www.bookcrossing.com). The dataset contains 1,157,112 ratings of 271,379 distinct ISBNs, rated by 278,858 members of BookCrossing. The ratings are both implicit and explicit with explicit ratings expressed on a scale of 1-10 and implicit ratings represented by zero. The dataset also contains some demographic information of users in the form of their Age and Location. Moreover, some content based information of books, including Book title, author, year of publication, and publisher, is gathered from Amazon website and is incorporated into this dataset. This dataset can, therefore, be used for both content-based and collaborative filtering techniques of recommendation.

1.3.3.4 Amazon Product Data

Amazon product data [56] is an extensive dataset gathered from Amazon website. The dataset contains 143.7 million product reviews and ratings, and metadata of products in the form of product name, categories, price, brand, and image feature. Moreover, this dataset also contains information of other items purchased and other links viewed by the same user. Some visual features of products have also been extracted from product images using deep convolutional networks and have been made available in the dataset.

1.3.3.5 Yahoo Webscope Datasets

Yahoo Webscope [57] is a library of several datasets that have been made available by Yahoo for noncommercial purposes. Some of these datasets specifically address the areas of recommendations and classifications. A brief description of some such datasets is given below:

- Yahoo! Music rating
  This dataset contains preferences of Yahoo music community and contains over 717 million ratings of 136,000 songs given by 1.8 million users. The details of songs including artist, album, and genre are also available in the dataset.

- Yahoo Delicious
  Delicious is a bookmarking website for storing online bookmarks. Yahoo Delicious dataset contains 100,000 URLs that were bookmarked on Delicious website by its users. Each URL has been bookmarked at least 100 times.

- Yahoo! Movies rating
  This dataset contains movie preferences of Yahoo users, rated on a scale of A+ to F. Users in this dataset are anonymous. However, detailed information is available about movies that include cast, crew, synopsis, genre, average ratings, and awards. This dataset is a good candidate for both collaborative and content-based filtering.

The next section ends the chapter with a brief overview about the problems and challenges that the recommender systems face nowadays.

1.4 Problems related to recommender systems

Although recommender systems are extremely powerful tools on the web these days, yet they have some problems and challenges, which they face every now and then. These challenges are discussed in detail by [8], [6], [58], & [59] and have been summarized below:

1.4.1 Data Sparsity Problem

Data sparsity problem arises in recommender systems as time progresses. This problem is caused by a drastic increase in number of users and items. As not many of the users are rating the items very frequently, the utility matrix becomes sparse which results in degradation of recommendation quality.

1.4.2 Cold Start Problem

When a new user or item is entered into the system, the cold start problem arises. In this case, it is not easy to provide recommendations as there is not much information available regarding the new user or new item. Therefore, useful recommendations are not produced. There are three kinds of cold start problems: new user problem, new item problem, and new system problem.

1.4.3 Scalability

As the number of users and the size of data increases rapidly recommender systems suffer from scalability problem and thus, could produce inaccurate or inefficient results. In fact, the users demand timely recommendations, and to overcome this challenge, recommender systems require high computational resources.

1.4.4 Overspecialization or Diversity Problem

This problem restricts the user to get already known recommendations or recommendations that are very specific to their profile, which doesn't allow them to discover newly added items into the system or other available options.

1.4.5 Vulnerable to Attacks

Many hackers try to promote certain items on the web by hacking the recommender systems, which makes them vulnerable to attacks. This makes it one of the major challenges faced by the developer of any recommender system.

References

1 A.M. Rashid, I. Albert, D. Cosley, S.K. Law, S.M. McNe, J.A. Konstan, et al, Getting to Know You: learning new User preferences in Recommender Systems, Proceedings of International Conference on Intelligent User Interfaces, 2002, pp 127 – 134

2 J.A.Konstan, J. Reidel, Recommendation Systems: From Algorithms to User Experience. User Model User-Adapt Interact 2012, 22:101-123

3 F.O.Isinkaye, Y.O. Folajimi, B.A. Ojoko, Recommendation Systems: Principles, methods and evaluation, Egyptian Informatics Journal (2015)16, 261-273.

4 F. Ricci, L. Rokach, B. Shapira, Introduction to Recommender System Handbook, ,

5 Y. Seroussi, The Wonderful World of Recommender Systems, Article at Yasirseroussi.com 2015

6 M. Sharma, S. Mann, "A survey of recommender systems: approaches and limitations", International Journal of Innovations in Engineering and Technology, vol. 2, no. 2, pp. 8-14, 2013.

7 F. Ricci, L. Rokach, B. Shapira and P. B. Kantor, Recommender Systems Handbook, 2011, Springer.

8 P. B.Thorat, R. M. Goudar, and S. Barve, "Survey on collaborative filtering, content-based filtering and hybrid recommendation system," International Journal of Computer Applications, vol. 110, no. 4, pp. 31–36, Jan. 2015.

9 Elahi M, Ricci F, Rubens N. A survey of active learning in collaborative filtering recommender systems. Computer Science Review. 2016 Jun 2.

10 G. Linden, B. Smith and J. York, "Amazon.com Recommendations Item-to-item collaborative filtering", IEEE Internet Computing, vol. 7, no. 1, pp. 76-80, Jan. 2003.

11 Y. Koren. The Bellkor solution to the Netflix grand prize. Netflix prize documentation, 81, 2009.

12 R.J. Mooney and L. Roy, "Content-Based Book Recommending Using Learning for Text Categorization", Proc. ACM SIGIR '99 Workshop Recommender Systems: Algorithms and Evaluation, 1999.

13 G.D. Linden, J.A. Jacobi and E.A. Benson, Collaborative Recommendations Using Item-to-Item Similarity Mappings, 2001.

14 L. M. de Campos, J. M. Fernández-Luna, J. F. Huete and M. A. Rueda-Morales, "Combining content-based and collaborative recommendations: A hybrid approach based on bayesian networks", Int. J. Approx. Reason., vol. 51, no. 7, pp. 785-799, 2010.

15 N. Pereira and S. K. Varma, "Survey on Content Based Recommendation System," International Journal of Computer Science and Information Technologies, vol. 7, no. 1, pp. 281–284, 2016.

16 F.O. Isinkaye, Y.O. Folajimi, B.A. Ojokoh, "Recommendation systems: Principles methods and Evaluation", Egyptian Informatics Journal, vol. 16, pp. 261-273, 2015.

17 J. Leskovec, A. Rajaraman, and J. D. Ullman, Mining of massive Datasets. Cambridge, United Kingdom: Cambridge University Press, 2014, ch. 9.

18 H.-N. Kim, A. El-Saddik, and G.-S. Jo, "Collaborative error-reflected models for cold-start recommender systems," Decision Support Systems, vol. 51, no. 3, pp. 519–531, Jun. 2011.

19 S. Dey, S. Sonwane, and D. Muneshwar, "A Survey Paper on Image Recommender System for Smartphone," International Journal of Advance

Foundation and Research in Science & Engineering, vol. 1, no. 5, Oct. 2014.

20 Saxena A, Khosla N, Venkataraman V, Khosla N, Venkataraman V. Building an Image-Based Shoe Recommendation System.

21 Wang, Yan, Sheng Li, and Alex C. Kot. "Joint learning for image-based handbag recommendation." 2015 IEEE International Conference on Multimedia and Expo (ICME). IEEE, 2015.

22 J. McAuley, C. Targett, J. Shi and A. Van Den Hengel, "Image-based recommendations on styles and substitutes", SIGIR'15, 2015.

23 S. Bell and K. Bala, "Learning visual similarity for product design with convolutional neural networks," ACM Transactions on Graphics, vol. 34, no. 4, p. 98:1–98:10, Jul. 2015.

24 J. Skrasek. "Social Network Recommendation using Graph Databases," PhD thesis, Masaryk University Faculty of Informatics, 2015.

25 Z. Huang, W. Chung, T. Ong, and H. Chen, "A Graph-Based Recommender System for Digital Library, " in Proceedings of the 2 ACM/IEEE-CS Joint Conference on Digital Libraries, Portland, Oregon, USA: 65-73, 2002.

26 J. J. Miller, "Graph database applications and concepts with neo4j", Proceedings of the Southern Association for Information Systems Conference Atlanta GA USA, vol. 2324, 2013.

27 C.C. Aggarwal, J.L. Wolf, K-L. Wu, P.S. Yu, "Horting Hatches an Egg: A New Graph-Theoretic Approach to Collaborative Filtering", Proc. Fifth ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining, Aug. 1999.

28 M. Claypool, A. Gokhale, T. Miranda, P. Murnikov, D. Netes, M. Sartin, "Combining Content-Based and Collaborative Filters in an Online Newspaper", *Proc. ACM SIGIR '99 Workshop Recommender Systems: Algorithms and Evaluation*, Aug. 1999.

29 N. B. Silva, I.-R. Tsang, G. D. C. Cavalcanti, and I.-J. Tsang, "A Graph-based Friend Recommendation System Using Genetic Algorithm," Proceedings of the 2010 IEEE Congress on Evolutionary Computation, July 2010, pp. 1-7.

30 H. Lee, J. Kwon, "Efficient Recommender System based on Graph Data for Multimedia Application", *Int. J. of Multimedia & Ubiquitous Engineering*, vol. 8, no. 4, 2013.

31 M. A. Rodriguez and P. Neubauer, "Constructions from dots and lines," Bulletin of the American Society for Information Science and Technology, vol. 36, no. 6, pp. 35-41, August 2010.

32 I. Robinson, J. Webber, and E. Eifrem. Graph Databases. O'Reilly Media, Incorporated, 2013.

33 J. Daltio, C. B. Medeiros, "HydroGraph: Exploring Geographic Data in Graph Databases", *Proc XVI GEOINFO*, pp. 44-55, 2015.

34 E. Zimeo, G. Oliva, F. Baldi, and A. Caracciolo. Designing a scalable social e-commerce application. Scalable Computing: Practice and Experience, 14(2):131–141, 2013.

35 A. Anthony, Y-K. Shih, R. Jin, and Y. Xiang. Leveraging a graph-powered, real-time recommendation engine to create rapid business value. In Proceedings of the 10th ACM Conference on Recommender Systems, pages 385–386. ACM, 2016.

36 Z. Fu, Z. Wu, H. Li, Y. Li, M. Wu, X. Chen, X. Ye, B. Yu, and X. Hu. Geabase: A high-performance distributed graph database for industry-scale applications. In Advanced Cloud and Big Data (CBD), 2017 Fifth International Conference on, pages 170–175. IEEE, 2017.

37 F. Zarrinkamal, M. Kahani, S. Paydhar, "Using Graph Database for for file Recommendation in PAD Social Network", *7th International Symposium on Telecommunications (ST)*, pp. 470-475, 2014.

38 M. Fr¨ohlich, “Case study of a graph database system that supports ideation and writing process.”

39 J. Cordeiro, B. Antunes, P. Gomes, "Context-based recommendation to support problem solving in software development", *Proc. 3rd Workshop Recommendation Syst. Soft. Eng.*, pp. 85-89, 2012.

40 M. A. Rodriguez and P. Neubauer, "The graph traversal pattern," CoRR, vol. abs/1004.1001, 2010.

41 S. Xiaoyuan, T. M. Khoshgoftaar, "A survey of collaborative filtering techniques", *Adv. Artif. Intell.*, vol. 2009, 2009.

42 C. Desrosiers and G. Karypis, "A comprehensive survey of neighborhood-based recommendation methods, " Recommender Systems Handbook, pp. 107-144, 2011. [Online]. Available: http://www.springerlink.com/index/N3JQ77686228781N.pdf.

43 J. Liu and C. Wu, “Deep learning based recommendation: A survey,” In International Conference on Information Science and Applications, pages 451–458, Springer, 2017.

44 S. Chen et al., "Interactive visual discovering of movement patterns from sparsely sampled geo-tagged social media data", *IEEE Trans. Vis. Comput. Graph.*, vol. 22, no. 1, pp. 270-279, Jan. 2016.

45 H. Zhao, H. Zhang, Y. Liu, Y. Zhang, and X. L. Zhang, “Pattern discovery: A progressive visual analytic design to support categorical data analysis,” *Journal of Visual Languages & Computing*, vol. 43, pp. 42–49, 2017.

46 T. H. Lee, K. Levinski, W. S. Tang, and L. Zhu, “Visualization of graphical representations of log files,” US Patent 9,684,707, June 20, 2017.

47 R. Van Bruggen, Learning Neo4j, Packt Publishing Ltd., 2014.

48 S. Sawant, “Collaborative filtering using weighted bipartite graph projection: a recommendation system for yelp,” in *Proceedings of the*

CS224W: Social and Information Network Analysis Conference, December 2013.

49 N. Team, *Cypher query language*, 2013.

50 H. He and A. Singh. GraphQL: Query language and access methods for graph databases. Technical report, Department of Computer Science at University of California, Santa Barbara, 2007.

51 M. A. Rodriguez, "The Gremlin graph traversal machine and language (invited talk)", *Proc. 15th Symp. Database Program. Languages*, pp. 1-10, 2015.

52 E. Prud'hommeaux, A. Seaborne, "SPARQL Query Language for RDF", 2006.

53 F. M. Harper and J. A. Konstan, “The MovieLens Datasets: History and Context,” ACM Transaction on Interactive Intelligent Systems., vol. 5, no. 4, p. 19:1–19:19, Dec. 2015.

54 K. Goldberg, T. Roeder, D. Gupta, and C. Perkins, “Eigentaste: A Constant Time Collaborative Filtering Algorithm,” Information Retrieval, vol. 4, no. 2, pp. 133–151, Jul. 2001.

55 C.-N. Ziegler, S. M. McNee, J. A. Konstan, and G. Lausen, “Improving Recommendation Lists Through Topic Diversification,” in Proceedings of the 14th International Conference on World Wide Web, New York, NY, USA, 2005, pp. 22–32.

56 “Amazon review data.” [Online]. Available: http://jmcauley.ucsd.edu/data/amazon/links.html. [Accessed: 31-Aug-2017].

57 “Webscope | Yahoo Labs.” [Online]. Available: https://webscope.sandbox.yahoo.com/catalog.php?datatype=r. [Accessed: 31-Aug-2017].

58 L. Lü, M. Medo, C. H. Yeung, Y.-C. Zhang, Z.-K. Zhang, and T. Zhou, "Recommender systems," Physics Reports, vol. 519, no. 1, pp. 1–49, Oct. 2012.

59 Y. Chen, C. Wu, M. Xie, and X. Guo, "Solving the Sparsity problem in Recommender systems using association retrieval," Journal of Computers, vol. 6, no. 9, Aug. 2011.