A Smart Movie Recommendation System

Sang-Ki Ko[1], Sang-Min Choi[1], Hae-Sung Eom[1], Jeong-Won Cha[2], Hyunchul Cho[3], Laehyum Kim[3], and Yo-Sub Han[1,*]

[1] Department of Computer Science, Yonsei University, Seoul, Republic of Korea
{narame7,jerassi,haesung,emmous}@cs.yonsei.ac.kr
[2] Department of Computer Engineering, Changwon National University, Changwon, Republic of Korea
jcha@changwon.ac.kr
[3] Intelligence and Interaction Research Center, KIST, Seoul, Republic of Korea
{hccho,laehyunk}@kist.re.kr

Abstract. We propose a movie recommendation system based on genre correlations. We modify the previous algorithm; we use a list of movies as input instead of genre combinations. We implement a new recommendation algorithm as Android application with additional functions. By combining with existing web services such as Google Movie Showtimes and Open APIs, our system can recommend movies playing in cinemas currently and show the detailed information of movies. Location-based function is also implemented. We utilize GPS information of mobile device and web service provided by Google Maps for recommending suitable cinemas for users with mobile devices.

Keywords: recommendation system, movies, smartphones, Android.

1 Introduction

As information technology develops in these days, we can easily obtain the information from the Internet. Since there are massive materials on the Internet, it is difficult to use all of them efficiently. Thus we should choose which materials to use. This means that we need to know which one is useful and which is not for better recommendation.

Nowadays, smartphones become one of the most important tools for our life. Most of smartphone users tend to use their phones instead of computers when they search information, since mobile phones are more portable and smartphones has many searching applications for various objectives.

We propose a smart movie recommendation system for smartphones. We use a recommendation technique based on the genre correlations investigated by Choi and Han [1]. We implement the proposed system on Android platform.

In Section 2, we revisit the previous research with respect to recommendation systems. Then, we propose our approach that applies recommendation technique to mobile applications for movie recommendation systems in Section 3. Then we present the result of implementations on Android platform in Section 4. We conclude with future works of this research in Section 5.

* Corresponding author.

M.J. Smith, G. Salvendy (Eds.): Human Interface, Part I, HCII 2011, LNCS 6771, pp. 558–566, 2011.

2 Related Works

There are lots of recommendation techniques investigated by many researchers [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]. Recommendation system attempts to recommend information items that are likely to be of interest to the user. There are four kinds of recommendation techniques, content-based, association, demographic and collaborative method.

Content-based method uses item-to-item similarity. If a user like B, we recommend A that is similar to B. Association method also uses item-to-item similarity. In this method, we do not decide whether actually they are similar or not. If items have high correlation with each other, we decide that they are similar. Demographic method and collaborative method use people-to-people similarity both. Demographic method needs actual features of people to decide whether they are similar. Collaborative method uses correlation between users.

Basically, our system uses item-based method. The detailed explanation for our recommending algorithm will be covered later.

2.1 Collaborative Filtering

As the amount of information in the world is increasing very quickly, we need techniques to find relevant information efficiently. One of such technique is to use a recommendation system and the collaborative filtering [7, 8, 9, 10] is one of the most promising methods. Collaborative filtering is a method of making automatic predictions about the interests of a user by collecting taste information from many users. The underlying assumption of this approach is that those who agreed in the past tend to agree again in the future.

We need to build a database of preferences for items by users at first. A new user is matched against the database to discover neighbors that are users who have historically had similar taste to the new user. Items that the neighbors like are then recommended to the new user as he will probably also like them. This approach has been very successful in many recommendation systems.

2.2 Genre Correlation

General collaborative filtering approach is based on user preferences. This implies that the system should wait until it has enough input data from users. Researchers proposed several methods to avoid this problem [1, 2, 3, 4, 5, 6]. One of such approaches is to use information that is reliable and available initially. Notice that we cannot always have such information available. Thus, we choose a movie recommendation system domain since a movie has a category information (called genre) given by experts.

Recently, Choi and Han [1] proposed a movie recommendation system based on genre correlations. Their system does not require lots of user preferences. The system first calculates genre correlations based on the genre combinations of each movie. Then the system applies the genre combination of all movies and user-preferred genres to the average rating of each movie based on the genre correlations. Finally, it ranks movies according to the newly computed point.

3 Our Approach

We propose a movie recommendation system based on genre correlation. Choi and Han [1] suggested the method that users should input their favorite movie genres into the recommendation system manually and the system calculates the recommendation points. On the other hand, our recommendation system uses movie lists as input and obtains the genres of movies in lists and thus the preferences of users. (We profile movie preferences of users.) This step assumes that users would prefer genres appeared in the list to other genres.

3.1 Calculating Genre Correlation Based on Movie List

Since we have a list or several lists of movies for recommendation, we should figure out the number of appearances of genres in the list. If there are two movies with the first movie having comedy and drama as a genre combination and the second movie having romance and drama as a genre combination, the number of appearances of drama is two. In the same way, comedy is one and romance is also one. These numbers are used for weighting each genre when we calculate the recommendation points.

3.2 Movie Information Retrieval

Our application retrieves the movie information from the famous portal sites. When a user tries to see the information of a movie, this application sends the title of movie to

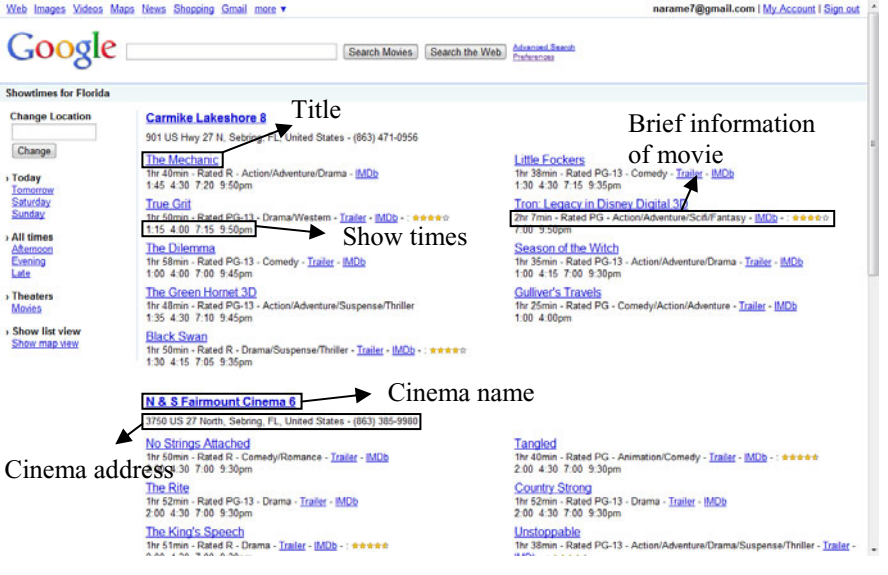

Fig. 1. This page shows the show times of movies in Florida, United States of America. Google Search provides this service via web pages. You can enter the location where you are living instead of Florida. The address of this page is http://www.google.com/movies?near=Florida

Open API services. Then they return the information of movie in XML format. In our application, since we develop this in South Korea, we use Korean portal sites offering Open API services for movie information.

3.3 Location-Based Cinema Recommendation

Google Search offers Movie Showtimes [11] service that provides the show times of movies to users. Users can know at which cinema a certain movie is playing near them and show times of movies and even the address of cinemas. We use this service for our application. Detailed explanation of how we used this web service for our system is described in Section 4.3.

4 Implementation

We implement our recommendation system on Android OS. Testing device is HTC Desire with Android 2.2 version. Since Android platform uses Java programming language, we use Java for developing this application. PHP language is also used for server programming. The overall structure of our system is depicted in Fig. 2.

There exist three objects in our system: server, mobile device and web services. We use a server here because the size of movie database is too large to handle on mobile device. When a mobile phone recommends movies, we need to calculate recommendation points of all movies according to genre correlation matrix and the calculation consumes too much time and resource when it is computed on a small device such as smartphone. Therefore, a mobile device sends the list of movies and the server calculates recommendation points of all movies stored in server and sort them. Finally, the server returns the resulting list, that is, the list of recommended movies to the mobile phone.

In Fig. 2, there are three web services used in our recommendation system. We use Google Movie Showtimes service and Google Maps service. As we briefly cover in Section 3.3., Google Movie Showtimes service provides regional cinema information and movie information. They provide these two kinds of information in http://www.google.com/movies page. We can obtain regional cinema information by putting the address as a GET parameter.

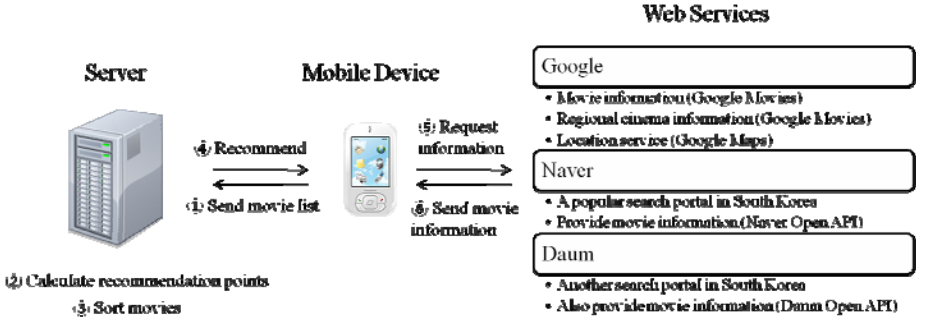

Fig. 2. This diagram describes the procedural steps of our recommendation system. Mobile device generally means a smartphone or a tablet PC using Android OS with GPS information.

Naver[1] is a popular search portal in South Korea, with a market share of over 70%, compared to 2% of Google. They provide Internet services including a news service, an e-mail service an academic thesis search service and so on. They also provide Open API service that we can get movie information by using the title of movie. If we request the movie information with the title of movie, they return the information of movie as XML form. The information includes the English title and Koran title of movie and directors, actors, user rating, related links and so on.

Daum[2] is also a popular web portal in South Korea. Daum offers many Internet services to web users, including a popular free web-based e-mail, messaging service, forums, shopping and news. They also provide Open API service relating to movies.

4.1 Development Environment

We work on Windows 7 64bit OS. Tool used for our development is Eclipse Helios version and JDK version is 1.6.0. We also use a server for server programming. Our server is using CentOS release 5.5 as an operating system and kernel version is 2.6.18. The version of PHP language installed on our server is 5.2.10 and the version of MySQL is 5.0.76.

4.2 GroupLens Database on Our Server

We use an open movie database called *GroupLens* database[3]. The GroupLens database has three sub-databases: movie database, user database and rating database. The movie database has information of 10681 movies. We create a MySQL database on server for storing this database.

4.3 Recommendation Result Based on Movie List

Our system has two kinds of movie recommendation methods. The first method is recommending the movie in the database. Since we use the GroupLens database that has 10681 movies, the recommendation result always consists of the movies in the GroupLens database. The second method is recommending the movie that is now playing in cinemas.

The right picture of Fig. 3 shows the result of recommending the movies that are now playing in cinemas. The English title of the first movie in the list is 'The Private Lives Of Pippa Lee' and the second movie is 'Gulliver's Travels'. The genre of these movies is both comedies. The genres of other movies in the list are all comedy or animation. Note that the genre combinations of three movies in the left list contain comedy and animation. We can confirm that the result of recommendation is quite reasonable.

[1] http://www.naver.com
[2] http://www.daum.net
[3] http://www.grouplens.org/

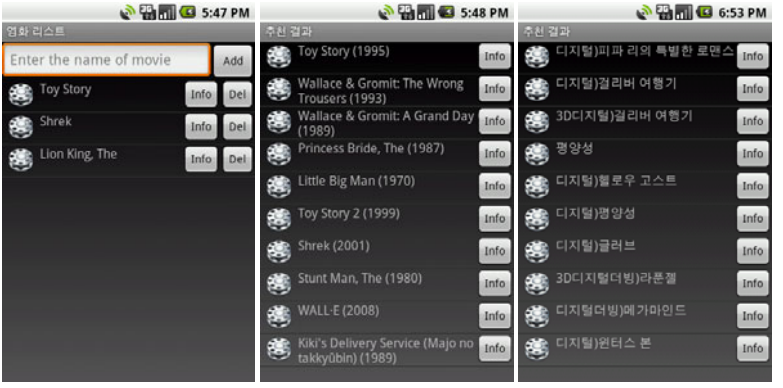

Fig. 3. The left shows the list of movies and the center shows the result of recommendation when we apply our method to the list. The right shows the result of recommending the movies that are now playing in cinemas.

4.4 Open API Services

When our system displays the result of recommendation as a list of movies, users can see the detailed information of the movies. This information retrieved is from Open API services of Naver and Daum [13, 14], famous South Korean portal sites. The process of retrieving information is quite simple. When users want to see the detailed information of the movie, our system sends the name of movie to Open API services. Since sometimes there is no available information of certain movie, we call two Open API services one after the other. That means if there is available information in first Open API service, we do not need to call the other. The retrieved information is displayed on mobile device with Android layout as illustrated in Fig. 4.

Fig. 4. These pictures show the detailed information of two movies 'Toy Story 3' and 'Love Actually'. We capture in Android SDK Virtual Device. Since the retrieved information is from South Korean portal sites, the contents are in Korean. The contents consist of poster, Korean title, English title, directors, actors, user rating, and summary of two movies.

Since the return forms of Open API services are in XML format, we implement Java-based XML parser to display this information on Android layout.

4.5 Location-Based Cinema Recommendation

Google Movie Showtimes [11] service provides the movies which are now playing in cinemas and regional cinema information. We can know the cinema information of certain region by inputting the address or GPS information as a GET parameter.

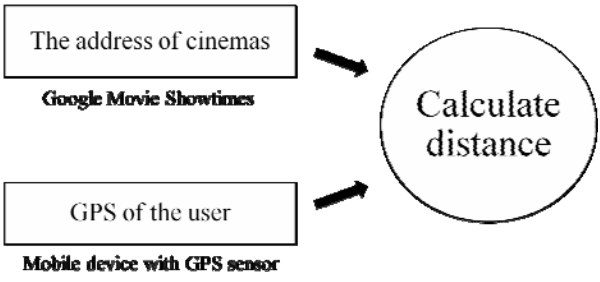

Fig. 5. The process of obtaining the distance between the user with a mobile device and the cinema. GPS of the mobile device can be obtained by GPS sensor in the mobile device.

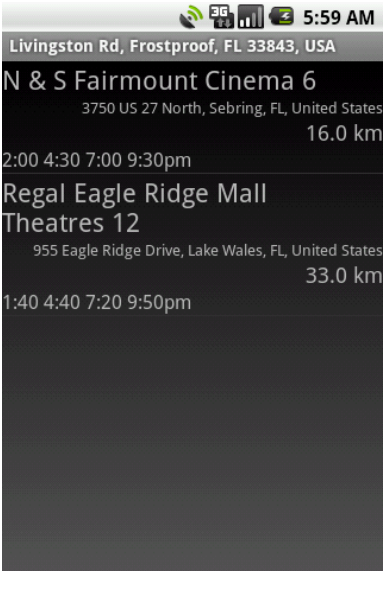

Fig. 6. Our application shows a list of cinemas where certain movie is playing. This picture shows the list of cinemas with some information such as name of cinemas, address of cinemas, the distances between the mobile device and the cinemas and the show times of the movie at each cinema.

Google Maps [12] offers the service that converts GPS information to an address or an address to GPS information. Our system provides the distance between the user and the cinema where the recommended movie is playing. Fig. 5 presents the process of obtaining the distance using Google web services.

Fig. 6 shows a list of cinemas where one of recommended movies is playing. We can find name, address, distance from the user, show times of the movie of each cinema.

As seen in Section 3.3, Google Movie Showtimes service provides their information in HTML web pages. To parse the page and retrieve useful contents for us, we implement HTML parsing program with PHP on our server. When our application on Android device request the cinema information with the title of movie and the current location, the PHP program returns the cinema where the movie is playing and close to the current location of Android device.

5 Conclusions

We propose a movie recommendation system based on genre correlations. In this paper, a list of movies is used for calculating recommendation points instead of genres. We implement the proposed algorithm as a mobile application on Android OS. By combining with existing web services such as Google Movie Showtimes, our application provides a list of recommended movies that are now playing in cinemas and the regional information of cinemas.

We use genre information here for recommending movies in this paper. Since there are more features in movies such as directors, actors and so on, we can utilize those features in our future works.

In future, we can connect our service with ticket reservation system of cinemas. Then, the users can be recommended and can reserve the tickets of cinemas around the users at a time by using our application.

Acknowledgement. This research was supported by the IT R&D program of KME/IITA 2008-S-024-01. Han is supported by the Basic Science Research Program through NRF funded by MEST (2010-0009168).

References

1. Choi, S.-M., Han, Y.-S.: A content recommendation system based on category correlations. In: The Fifth International Multi-Conference on Computing in the Global Information Technology, pp. 1257–1260 (2010)
2. Huang, Z., Chen, H., Zeng, D.D.: Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. ACM Trans. Inf. Syst. 22(1), 116–142 (2004)
3. Ishikawa, M., Géczy, P., Izumi, N., Morita, T., Yamaguchi, T.: Information diffusion approach to cold-start problem. In: Web Intelligence/IAT Workshops, pp. 129–132 (2007)
4. Popescul, A., Ungar, L.H., Pennock, D.M., Lawrence, S.: Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments. In: Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence, pp. 437–444 (2001)

5. Schein, A.I., Popescul, A., Ungar, L.H., Pennock, D.M.: Methods and metrics for cold-start recommendations. In: The 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 253–260 (2002)
6. Wilson, D.C., Smyth, B., O'Sullivan, D.: Sparsity reduction in collaborative recommendation: A case-based approach. IJPRAI 17(5), 863–884 (2003)
7. Hill, W., Stead, L., Rosenstein, M., Furnas, G.: Recommending and Evaluating Choices in a Virtual Community of Use. In: Proceedings of CHI 1995 (1995)
8. Konstan, J., Miller, B., Maltz, D., Herlocker, J., Gordon, L., Riedl, J.: GroupLens: Applying Collaborative Filtering to Usenet News. Communications of the ACM 40(3), 77–87 (1997)
9. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J.: GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In: Proceedings of CSCW 1994. Chapel Hill, NC (1994)
10. Shardanand, U., Maes, P.: Social Information Filtering: Algorithms for Automating 'Word of Mouth'. In: Proceedings of CHI 1995, Denver, CO (1995)
11. Google Movie Showtimes, `http://www.google.com/movies`
12. Google Maps, `http://maps.google.com/`
13. Naver Open API, `http://dev.naver.com/openapi`
14. Daum Open API, `http://apis.daum.net`