Journal of Engineering and Applied Sciences 13 (6): 1506-1514, 2018
ISSN: 1816-949X

Keyword-Based Collaborative Filter Recommendation System Using Scraping

Young Jun Kim, Jeong Min Park, Sung Taek Chung and Jeong Joon Kim
Department of Computer Science and Engineering, Korea Polytechnic University,
Gyeonggi-do, 15073 Siheung-si, Korea

Abstract: CF (Collaborative Filtering) is one of the methods generally utilized in recommendation system. The goal of CF is to analyze the purchase trend of other customers similar to a target customer and recommend items that can be preferred by the customer among the items he or she has not bought. Conventional CF, however is hardly capable of predicting any new customer's purchase trend for they have no existing purchase list. To resolve the problem, it surveys customers too much or changes items into profile, causing huge expenses and difficulty. In this study, as a new method to solve such a problem, keyword-based collaborative filter recommendation system using scraping is proposed.

Key words: Collaboration filter, recommendation system, scraping, movie recommend, keyword-based, purchase

INTRODUCTION

The recent rapid information technology development caused information overload. As a result, users have become unable to easily find the information they want and information quality has also degraded. To solve this problem, a lot of studies are goingon how to provide individually customized service based on individual preference by not regarding them as crowd collectively (Resnick and Varian, 1997). Recommendation system provides items, contents service, etc., potentially preferred by an individual and is often utilized in particularly, shopping malls, video streaming and electronic commerce in order to provide useful service to users (Mulvenna *et al*., 2000). Good examples are Youtube, Netflix and Amazon. Such a recommendation can be implemented in diverse methods. Recently, collaborative filtering CF-based methods have been more researched. CF is one of the methods to recommend items that could be favored by a customer among the items the target customer has not bought by analyzing other similar customer's purchase trend. The analysis method is first, to collect direct or indirect item evaluation data such as user rate; second, to produce customer-specific profile based on the collected data, third to form the nearest neighbors with a similar purchase trend to the target customer based on the produced customer-specific profile, then to analyze the neighbors, predict the grade of items the target customer has not rated and accordingly recommend items preferable by the target customer. Conventional CF, however is limited for new customers because it requires data accumulation on evaluation and purchase trend in advance which is called cold start problem (Lika *et al*., 2014). Moreover, since, it bases on product items evaluated and experienced by users, it can hardly recommend any new item beyond the normal purchase pattern which is the first-rater problem. There are many ways to address such problems (Kumar and Bhatia, 2012). One is to build a lot of surveys to understand an initial user's disposition and make a recommendation based on it. Another one is to refer to item similarity instead of user's purchase pattern to make a preferable recommendation. It is content-based CF (Melville *et al*., 2002). However, such solutions face limitation that either users have to actively answer many questions or item profile is hardly established. To overcome these, there is a growing need for study on a new method capable of making recommendations to new customers. This study proposes keyword-based collaborative filter recommendation system using scraping as a new way to address such problems.

Literature review

Recommendation system using user-based collaborative filtering: Figure 1 shows user-based collaborative filtering process in three steps. Step 1 is to collect user's item evaluation data directly or indirectly and produce each customer-specific profile. Customer-specific profile produced in this manner is utilized in step 2 for CF process. Depending upon the resulting CF outcomes, the system recommends items potentially preferred by a customer in step 3.

Corresponding Author: Jeong Joon Kim, Department of Computer Science and Engineering, Korea Polytechnic University, Gyeonggi-do, 15073 Siheung-si, Korea

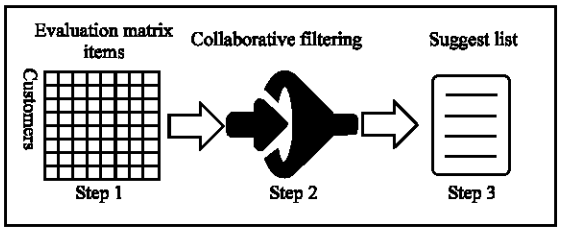

Fig. 1: User-based collaborative filtering process

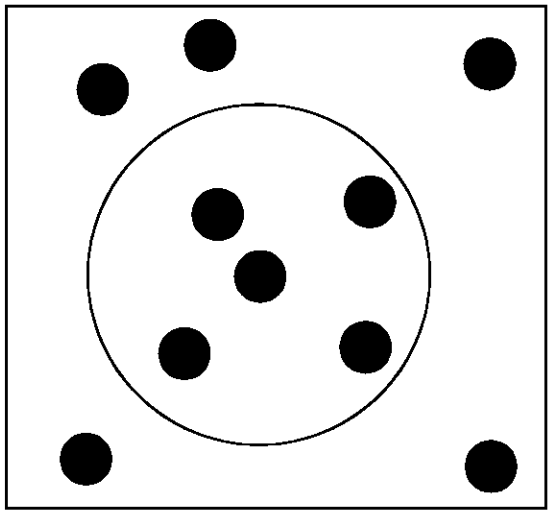

Fig. 2: Constructed nearest neighbors items

Table 1: Evaluation table matrix

Customers	Item 1	Item 2	Item 3	Item 4
1	Ø	$V_{1,2}$	Ø	$V_{1,4}$
2	$V_{2,1}$	Ø	$V_{2,3}$	$V_{2,4}$
3	$V_{3,1}$	$V_{3,2}$	Ø	$V_{3,4}$

Step 1; Evaluation matrix preparation: In step 1, each customer fills in evaluation matrix by evaluating each survey item. The lines represent customers and columns, product items.

In Table 1, $V_{i,j}$ is customer's evaluation of item jrate and Ø means that the customer has yet, to evaluate the corresponding item. For instance, $V_{2,1}$ represent customer 2's evaluation result of item 1. The evaluation matrix produced in this manner is utilized in analyzing the similarity in purchase tendency of customers in step 2.

Step 2; Collaborative filtering process: The main goal of step 2 is to analyze the similarity of purchase tendency among customers based on the customer-specific profile obtained in step 1 and just in Fig. 2 to form the nearest neighbors. Here, the similarity is as in Eq. 1:

$$\text{Similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \cdot B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \cdot \sqrt{\sum_{i=1}^{n} (B_i)^2}} \quad (1)$$

Generally, cosine is utilized as a measurement indication to assess similarity as in Eq. 1. In Eq. 1, A is a set of common evaluation vectors of a target customer with other customers whereas B, a set of common evaluation vectors of other customers with the target customer. In Table 1, for instance, when A is the common evaluation of customer 2 and 3 and B that of customer 3 and 2; A = $\{V_{2,1}, V_{2,4}\}$, B = $\{V_{3,1}, V_{3,4}\}$. Then, to utilize the (Eq. 1) coordinate system is mapped like Fig. 2, similarity is calculated and clustering is performed to form the nearest neighbors. Frequently utilized clustering methods include threshold-based select clustering and K-Nearest Neighbors (K-NN). Threshold-based select clustering is to conduct clustering on items exceeding threshold in other similar items to deal with a new item. K-NN is to predict a new item category based on the most similar item among the existing ones (Zhang *et al*., 1996; Cover and Hart, 1967).

Step 3; Show suggest list: Step 3 seeks to make a prediction about items not evaluated by a customer based on the nearest neighbor's evaluation established using Eq. 2:

$$v_{i,j} = \bar{v}_i + \frac{\sum_{k=1}^{n} w(i,k)\left(v_{k,j} - \bar{v}_k\right)}{\sum_{k=1}^{n} |w(i,k)|} \quad (2)$$

In Eq. 2, $V_{i,j}$ means items not evaluated by a user and it has to be predicted to make a suggestion list. That is in Table 1, it means that customeri has not evaluated item j (Ø). To predict $V_{i,j}$, the average of all items evaluated by customeriand pure evaluation value are aggregated. This is to compensate the deviation from the absolute size of scores between the target customer and other customers. For instance, if customer A has the average evaluation score of 5 for each item and customer B, 8; the deviation between the averages needs to be compensated for. Therefore, as in Eq. 2, V_i mean sand the nearest neighbor's pure evaluation values are utilized to add compensation value to the obtained deviation. Here, w(i, k) is pearson correlation coefficient, representing the linear relationship between customeri and customer k. The range is expressed -1<w(i, k)<1. The closer the resultant value moves to 1, a positive correlation is indicated whereas the closer to -1, a negative correlation. By

multiplying this by the value of customer k's evaluation of item j less customer k's item evaluation, prediction can be made about a new item by weighing high the kind of items highly evaluated by customers with similar tendency while low the kind of items highly evaluated by customers with different tendency. Based on this prediction of $V_{i,j}$, items that has yet, to be evaluated by a target customer can be rated and a suggestion list can be established.

Recommendation system using content-based collaborative filtering: Content-based CF is to measure item similarity to suggest preferable items to a user. It has the same process as the user-based collaborative filtering in Fig. 1 but is different from it in not utilizing user's evaluation similarity but suggest item similarity as evaluation matrix. Content-based CF makes item information into vectors and maps them on coordinate system for clustering to form the nearest neighbors. Since, the method does not require user information, it causes no cold start but is limited in requiring much efforts and cost to vectorize the items (Balabanovic and Shoham, 1997).

Technology trends: Recently, to enhance the accuracy of recommendation system, content-based CF and user-based CF are combined to make the final suggestions to users (Claypool *et al*., 1999). Another different way in ongoing study is to improve the performance and accuracy of content-based CF, a CF technique weighing on new parameters including genre (Bobadilla *et al*., 2010). In this present study, keyword-based CF recommendation system using scrapingis proposed as a new method to overcome the problem of cold start in conventional CF and limitations of content-based CF.

MATERIALS AND METHODS

Keyword-based CF recommendation system suggestion using scraping: This study introduces keyword-based CF through movie. Figure 3 shows the overall flow of proposed CF which is similar to conventional CF process. In step 1, web scarping is employed to collect meta data and through the pre-processing, they are processed into information on movie then saved in the HDFS of Hadoop eco system. Step 2 vectorizes movie information and forms the nearest neighbors with similar movies through clustering. Step 3 analyzes item correlations through the nearest neighbors then, produces and recommends a suggestion list regarding items similar to those evaluated by user.

Step 1; Data set building: Step 1 aims at collecting and saving Item data. In this study, Python 3 was employed to collect user's movie reviews and ratings in data through web scraping in the Korean portal site of Naver and the collected data were saved in the HDFS of Hadoop eco system using Hive.

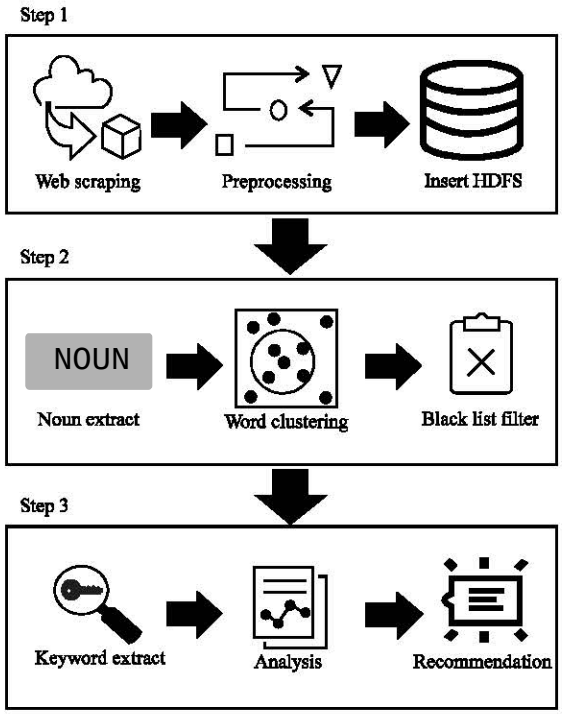

Fig. 3: Overall flow of keyword-based CF

Web scraping and preprocessing: On the portal site of Naver as in algorithm, movie is accessible via. URL code variable. Then, the code part is solely extracted to get movie comments, meta tags, etc. are removed and only the comments are saved.

Alogrithm 1; Web scraping:

```
url1 = 'http: //movie. naver. com/movie/bi/mi/point Write Form List.nhn?code='url2 = '&type = after and is Actual Point Write Execute = false and is Mileage Subscription Already = falseand is Mileage Subscription Reject = false and page = 'URL = 'http: //movie.naver.com/movie/bi/mi/basic. nhn?code = '+str (input ('code : '))
code = getCode(URL)
getReviewResult(code)
defgetCode(URL):
code_st = re.search('code=[0-9]+',URL).group()
    code = re.search('[0-9]+',code_st).group()
    return code
defgetReviewResult(CODE):
    path = str(raw_input("input path : "))+".txt"
    file = open(path,'w')
    page = int(1)
    count = int(input('Page Number: '))

  while count:
      print "Parse Page"+str(page)
      URL = url1+CODE+url2+str(page)
openRef = urlopen(URL)
```

```
        html = openRef.read().decode('utf-8')
        soup = BeautifulSoup(html,'lxml')
score_result = soup.find('div',class_= 'score_result')
lis = score_result.find_all('li')

        for li in lis:
            page = int(page)
reple = li.find('div',class_= 'score_reple').find('p').get_text()
            score = li.find('div',class_= 'star_score').find('em').get_text()
            children = li.find('dt')
            child = children.findChildren()
            time = str(child[3])
time_s = time[4:17]
file.write(reple.encode('utf-8')+'n')
           count -=1
           if not count:
               break
           page+=1
file.close()
```

This study collected only the comments and ratings as data in such a manner yet, additionally, film synopses and other articles on movie can be scraped from Facebook, Twitter, Wiki, etc. In the process of scraping, the research may be stopped for a security reason. In such a case, distributed type web scraper could be implemented or AWS server resion function can be utilized to set each system server area differently to resolve the problem. Table 2 is the result of translating user comments and ratings scraped from Naver into English using Google translation API.

Saving in HDFS: The extracted word list is loaded to memory using Hive, create a table and save them. To identify the data, tables were created with movie titles and corresponding comments then saved in HDFS. Alogrithm is hive code to read the scraped data from text file form a table and save it.

Alogrithm 2; HDFS table creation and deletion:

```
drop table if exists scraping_data
create table scraping_data(text string)
load data local inpath '/input/' into table scraping _data
create table result as SELECT INPUT__FILE__NAME as name, text from
scraping _data
INSERT OVERWRITE DIRECTORY '/user/hive/data' select*from result
drop table if exists word_list
create table word_list(text string)
load data local inpath '/input/WordList' into table word_list
create table list as SELECT INPUT__FILE__NAME as name, text from
word_list
INSERT OVERWRITE DIRECTORY '/user/hive/crawling' select*from
list
```

Step 2; Keyword creation and nearest neighbors formation: In step 2, the key is to vectorize item information and form the nearest neighbors. In this study to compare item similarity, movie keywords were employed. Keywords were extracted from the data collected through scraping in step 1. The process of keyword creation is first to extract nouns on movie comments which are the corresponding movie's theme or frequent material candidates, second, to perform word clustering. Even the same words can be differently expressed such as love, loves, to love and loving; thus, a representative word, love should be set up and its set clustering is necessary. Third is to filter out meaningless words appearing highly frequently through a pre-determined word blacklist for the clustered word set. Among such filtered words, highly frequent ones are set as movie keywords. The keywords are utilized to assess movie similarity and the details are explained.

Table 2: Example of Python 3 web scraping's result

Index	Review comments
1	Best 55 min ago corner for best last 5 min
2	Because of the last five minutes, I still take this movie and take it out occasionally
3	My favorite movie goes into 10 fingers
4	Any adult man who has a first love will agree with Takagi and he will fall into a big aftereffect for a while
5	BEST At the end of Wanna, Takaki is waiting until the train passes by
6	And the real scene is not a blur, but the chest is so sore and tears.
7	The real best movie aftereffect is really bad
8	I have been in contact for some time but the way of life that I have to walk with is so, different and far away
9	I have been out of contact for two years but for over a decade, my old memories are still clear in my chest room
10	Will she be good? The last ost really remembers
11	I feel different because the ending of the movie that makes me think about love is so realistic
12	It is a masterpiece because it deals with the love of reality rather than a thin love story and expresses it more closely
13	It is bitter, coarse and frustrating
14	I caught sight of it

Noun extract: In the noun extraction phase, nouns on movie comments are extracted. The nouns should be a movie's theme or its frequent material candidates. In this study, data were collected from the Korean portal site of Naver and the collected data were in Korean. For this reason, to extract nouns from the data, Alogrithm 3 and the open source program for Korean noun extraction of Han Nanum-analyzer were employed. However, to extract nouns in other language, other method should be used.

Alogrithm 3; Noun extraction JAVA code:

```
public class NounExtractor implements PosProcessor {
        /** the buffer for noun morphemes */
        private LinkedList<String> Morphemes = null
        /** the buffer for tags of the morphemes */
        private LinkedList<String> Tags = null
        @Override
        public void initialize(String baseDir, String configFile) throws
        Exception {
                Morphemes = new LinkedList<String>()
                Tags = new LinkedList<String>()
        }
        @Override
```

```
        public Sentence doProcess(Sentence st) {
                Eojeol[] eojeols = st.getEojeols()
                for (inti = 0; i<eojeols.length; i++) {
                        String[] morphemes = eojeols[i]. get
                        Morphemes()
                        String[] tags = eojeols[i]. getTags()
                        Morphemes.clear()
                        Tags.clear()
                        for (int j = 0; j <tags.length; j++)
                                {char c = tags[j]
                                charAt(0)
                                if (c == 'n') {
                                Morphemes.add
                                (morphemes[j])
                                Tags.add (tags [j])
                        }
        else if (c == 'f') {
                                Morphemes.add
                                (morphemes[j])
                                Tags.add("ncn")
                        }
                }
                eojeols[i].setMorphemes(Morphemes.to
                Array(newString[0]))
                eojeols[i].setTags(Tags.toArray(new String[0]))
        }
        st.setEojeols(eojeols)
        return st
    }
}
```

Following table is the result screen of noun extraction via. Han Nanum-analyzer. It is the English translation by Google translation API.

Word clustering using Levenshtein distance: In this study, Levenshtein distance is employed to perform word clustering. As in table the extracted nouns are not solely the noun part but in different words, though the same words, if they have additional particles. Or they can be expressed in different words due to misprint or typo. To address the problem, a representative word should be created for similar words and similar words should be clustered to create word sets. For instance, love, love, loves, loved and their typoloce can be represented by love. Levenshtein distance is an algorithm assessing word similarity. In this study, it is utilized to assess word similarity and create sets through clustering technique. Let's say, for example, string A is 'delegate' and string B, 'delete'. If the 5th g and 5th a of string A are deleted, it is the same as string B. The frequency of actions taken to make the same word is called edition distance. In the case of the example, the edition distance is 2. The sum of string A and B lengths is 14, so, the two string's similarity is approximately 0.857 using the 1-(edition distance/string length aggregation) equation. Here, one thing to note is not to cluster if the first letters are different in order to prevent clustering words such as there and where together which are two different words having the same letters excluding the initial one. This bases on the aspect that when people recognize a word, they heavily rely on both ends of it or that is the starting and ending parts of a word under the word superiority effect. Word clustering is elaborated as.

Table 3 is the matrix of distance coefficients on word similarity generated based on Levenshtein distance. The groups with the highest similarity are combined to make a new group and obtain new similarity. A group with the second highest similarity is added to a new group. This is repeated until gaining a group of similar words. This process of combining highly similar two groups is called group average connection. There are inter-group average and intra-group average techniques. Inter-group average method, unlike intra-group average, does not factor in the distance between words in the same group in its average calculation. Since, the data is a typical without a set frame, it is difficult to choose which clustering model to use. So, inter-group average technique is utilized to perform bottom up clustering as it can deliver the most

Be extracted noun:

My life	Boring	Really	^^^o	Ending	Frustrated	^^^,
Emotion	It was dry	Love	Ending	Translate	Last	Ending
Lyrics	Movie	High Lyte	Masterpiece	Below level	Teenager	Man
Favorite Jin	Once	Chewy	Your name is	Feeling	Degree	No
Considered	Background	Scene	Even if	Characters	Background	It does
Not melt.	Configurationside	Regret	Too long	Boring	Ending	Making
Elegy	Nothingness	Immersion	^,,	Is it so young?	Impression	Fun
Movie	Ending	I'll kill you	Middle	Boring	I saw the ending	Movie
I'm making fun of you	^,^,^,	The	Mood	Astronaut	Frustrated	To
Movie	Growth	Process	Who	Once	First love	Story

Table 3: The distance coefficient matrix of literature

Variables	A	B	C	D	E
A	-	-	-	-	-
B	1.0	-	-	-	-
C	4.0	2.0	-	-	-
D	9.0	9.0	5.0	-	-
E	8.0	8.0	6.0	2.5	-

Table 4: Cluster pair and integrate state example

States	Clustering pairs
1	A, B, C, D, E
2	{A, B}, C, D, E
3	{A, B}, C, {D, E}
4	{A, B, C}, {D, E}

average performance. The followings are the example of bottom up clustering in inter-group average technique using Table 3 and 4.

In Table 2 and 4, find the closest cluster pair and integrate. Since, D (A, B) is 1.0, the smallest, A and B are integrated. Here, D(A,B) is the distance between A and B or that is similarity gap. The wider the similarity gap, the larger the value and the less the similarity they have. As in state 1, find the closest cluster pair and integrate. In D({A, B},C), D(A, C) = 4 and D(B, C) = 2, thus, the distance is determined as the average of the two, 3. Here, D(D, E) distance is 2.5. Since, the distance of D (D, E) is shorter than that of D({A, B}, C), D and E are combined. D({A, B},C) = 3 and D(C, D) = 5. Since, D (C, E) is 6, D(C{D, E}) becomes 5.5, the average of 5 and 6. As the distance of D({A, B}C) is the shortest {A, B} and C are combined. As D (A, D) = 9, D (A, E) = 8, D (B, D) = 9, D (B, E) = 8, D (C, D) = 5 and D (C, E) = 6; {A, B, C} and {D, E} are combined. However, if clustering in the method above, time complexity is O(n3), causing performance problem. If there are a T number of groups in total, T*(T-1)/2 times of exploration is necessary. A pair with the highest similarity is combined to reduce the number of groups by 1. If this process is repeated, time complexity is as in Eq. 3:

$$\begin{aligned}&\sum_{i=1}^{rT}\frac{(T\text{-}i+1)(T\text{-}i)}{2}\\&=\frac{1}{2}\sum_{i=1}^{rT}\left\{(T\text{-}i)^2+(T\text{-}i)\right\}\\&=\frac{1}{2}\sum_{i=1}^{rT}\left\{i^2\text{-}(2T+1)i+T^2+T\right\}\\&=\frac{r^2\text{-}3r^2+3r}{6}T^3+\ldots\end{aligned}\tag{3}$$

Such a problem can be resolved with divide and conquer. If it is certain which words are not to be grouped together, they can be separated into different tasks and each can be under parallel transaction. In this case when the total size is N, the overall time complexity of clustering task is $F(N) = O(n^3)$. But if this is bisected, $F(N) = F(N/2) + F(N/2) = O(n^3/2^3)+O(n^3/2^3) = O(n^3/2^3)/4 = F(N)/4$. In other words, if the whole word is divided in two and each part is clustered, the entire time taken would be 1/4 and the more it is divided, the faster the velocity would be exponentially. However, this is possible only for words that we can be sure not to be grouped together. This study, based on the word superiority effect introduced in 2, regarded words with different first letters as different words. Under the assumption there was no misprint or typo in the first letters, this study classified words with each different first letters into different buckets and ran clustering within the buckets.

Table 5: Result of word clustering using Levenshtein distance

Words	N	Similar forms
Love	68	Love, lonely, love, lose, loves, loved
Movie	66	Movie, movies, movie
The	61	The, this
Like	57	Like, life, liked, live, lives, line, lived, like, likes
See	46	See, seems, seen, seemed, seeing, seem, seems, seeped
First	42	First, first
Beautiful	34	Beautiful, beauty, beautiful
Good	33	Good, good
Ending	32	Ending, end, ending, ends
Story	32	Story, stories
Feel	31	Feel, feeling, feels, feeling, feelings, feelings
Really	28	Really, really
Can	26	Can
Boring	25	Boring, bored, barely
Time	24	Time, tired, times, time, tie

Black list filtering: In this study, clustered words as exhibited in Table 5 were utilized as movie keywords. The left end column has representative words and right end has similar words contained in each group. The numbers in the middle show the total number of set elements. These clustered representative words of each set are utilized as keyword candidates. However, it is likely that there exist on omatopoeia, meaningless words or repeated words in casual comments. For example, words such as articles like 'the' and repeated words like 'movie', given the nature of comment. It is necessary to remove such words to extract movie keywords, hence, blacklist filtering.

Of the words in Table 5, 'ㅋ' in Korean is an equivalent to 'lol' and' 'LMAO' in English. Moreover, since, they are movie comments, they have many repeated words including movie and cinema as well as articles like the and conjunctions which are highly likely to be unrelated with movie keywords. Such words were registered with database in advance and filtered out. In addition, words like 'series', 'first film' and 'second film' were replaced with the title of target movie. Here, one thing to note is not to filter out word such as like because, though they are likely to be irrelevant to the movie, they can show the audience's preference over the movie. The

representatives of word sets produced as a result of the step 2 filtering were viewed as movie keywords and then, based on them, analysis and recommendation are to be made.

RESULTS AND DISCUSSION

Step 3; Analysis and recommendation: In step 3, inter-movie similarity is calculated through extracted movie keywords. Based on the similarity, the nearest neighbors of each movie is formed and when a user chooses a certain film, it is aimed at to recommend similar films.

Keyword extract and analysis: The keywords extracted in the study are movie information, playing a crucial role in assessing movie similarity. They are equivalent to the vectorized items in content-based CF introduced in a relevant study. The nearest neighbors are formed based on them. Algorithm 4 is to set other movies including TOP N as the candidates of the nearest neighbors and form the nearest neighbors through threshold-based select clustering.

Algorithm 4; Saving and similarity analysis:

```
moviesNameTop10
mvtxt<- sapply(1:10, function(x)(paste("./ext/", moviesNameTop10[x], ".txt", sep = "")))
readmv<-sapply(1:10, function(x)(readLines(mvtxt[x], encoding ="euc-kr")))
mvnum<- sapply(1:10, function(x)(paste("movie",x)))
view <- mvnum
df<- data.frame(readmv, view, stringsAsFactors = FALSE)
corpus <- Corpus(VectorSource(df$readmv))
tdm<-TermDocumentMatrix(corpus, control=list( wordLengths= c(2, Inf) ))
tdm<- as.matrix(tdm)
cosine_dist_mat<- as.matrix(dist(t(tdm), method = "cosine"))
```

This study sets other movies containing the keywords extracted with Top N via. TermDocumentMatrix as the candidatesforthe nearest neighbors. The corresponding keywords in the relevant documents are counted and notified. Then, through Cosine, the similarity between the nearest neighborsis calculated and expressed in a table form as in Table 6 and 7.

In Table 6, the table values are expressed as $V_{i,j}$, that represents the cosine similarity difference between items i and j. For instance, just as $V_{1, 1}$, $V_{2, 2}$, $V_{3, 3}$, … and $V_{i, j}$, the cosine values in the diagonal in a matrix with the samei and j are 0. Here, 1 less the cosine value is inter-item similarity.

Recommendation: In knowledge extract and analysis cosine similarity matrix was established forthe nearest neighbors using term document matrix. Based on this similarity difference matrix, number of movies were extracted. The extracted movies are clustered based on threshold-based select clustering. Exhibits a suggestion candidate list extracted as a result of clustering.

The recommendation results of series films. When films were recommended in relation to Harry Potter and the Chamber of Secrets, Harry Potter and the Chamber of Secrets has 100% similarity, thus, the cosine value of 0. It is also found that other films in the Harry Potter series ranked higher in terms of similarity. Hearty Paws in the 6th place is a comic adventurous movie for family. It has a similarity in terms of the move genre to Harry Potter and the Chamber of Secrets which is a fantasy, adventure, action film for family. On the other hand, Herat Isis a soap opera for family and has less similarity. Though not in the same series, Howl's Moving Castle and I'm Legend contain fantasy and dramatic elements and have a relatively high similarity.

Figure 4 shows recommendation results when the threshold is 0.5 for Table 7. The top line shows the title of movie a user selected. Then, the higher the similarity, the higher the movie is placed on the list in bigger letters.

Figure 4 recommends films in relation to One Piece Film Gold, 2016. The target movie genre is animation, adventure, fantasy, action and pirate. Suggested movies are One Piece Filmz, 2012 in the same series of One Piece along with The Pirates, 2014 and Pirates of the Caribbean-Dead Men Tell No Tales, 2017 that used the same materials such as adventure, action and pirate. Others include Naruto, the Last, 2014, an action fantasy animation. Figure 4 and 5 indicate that movies in similar genre and materials were suggested.

Table 6: Movie similarity difference matrix

Movie ID	1	2	3	4	5	6	7	8	9	10
1	0.0000000	0.3247799	0.4562266	0.3526073	0.3042807	0.8579302	0.5034659	0.7110097	0.5779408	0.5605412
2	0.3247799	0.0000000	0.3055765	0.4370986	0.2815227	0.8123955	0.1321232	0.5716761	0.3956702	0.2546360
3	0.4562266	0.3055765	0.0000000	0.4611621	0.4459305	0.8032123	0.4793299	0.5207804	0.5576307	0.4053690
4	0.3526073	0.4370986	0.4611621	0.0000000	0.4033711	0.8088247	0.6894685	0.7743255	0.6218236	0.6961880
5	0.3042807	0.2815227	0.4459305	0.4033711	0.0000000	0.7891989	0.4171982	0.6320989	0.4510799	0.4781832
6	0.8579302	0.8123955	0.8032123	0.8088247	0.7891989	0.0000000	0.7611495	0.8026213	0.7545427	0.7861715
7	0.5034659	0.1321232	0.4793299	0.6894685	0.4171982	0.7611495	0.0000000	0.5992000	0.4494239	0.2722340
8	0.7110097	0.7516761	0.5207804	0.7743255	0.6320989	0.8026213	0.5992000	0.0000000	0.6361389	0.5448859
9	0.5779408	0.3956702	0.5576307	0.6218236	0.4510799	0.7545427	0.4494239	0.6361389	0.0000000	0.4681971
10	0.5605412	0.2546360	0.4053690	0.6961880	0.4781832	0.7861715	0.2722340	0.5448859	0.4681971	0.0000000

Table 7: Suggestion candidate list

ID	Names	Cosine
1	Harry Potter and the Chamber of Secrets, 2002	0.0000000
5	Harry Potter and The Sorcerer's Stone, 2001	0.3042807
2	Harry Potter and The Deathly Hallows_ Part 1, 2010	0.3247799
4	Harry Potter and The Pirsoner of Azkaban, 2004	0.3526073
3	Harry Potter and The Deathly Hallows_ Part 2, 2011	0.4562266
7	Hearty Paws 2, 2010	0.5034659
10	I am Legend, 2007	0.5605412
9	Howl's Moving Castle, 2004	0.5779408
8	Heaven's Soldiers, 2005	0.7110097
6	Heart Is, 2006	0.8579302

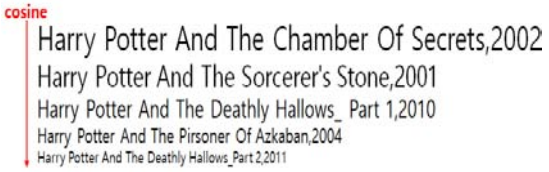

Fig. 4: Example of movie recommendation results screen 1

Fig. 5: Example of movie recommendation results screen 2

CONCLUSION

As a solution to the problem of information overload caused by the recent rapid information technology advancement, CF (Collaborative Filtering) has been researched. However, user-based CF has the issue of cold start problem indicating that because of the necessity to have data accumulation on product rating and purchase tendency in advance, recommendation suggestions can be hardly made for any new users. To address the problem, content-based CF emerged to rely on item similarity instead of other user's purchasing patterns to make item suggestions. However, this method is also limited in having difficulty building item profile. The keyword-based collaborative filtering proposed in this present study employs scraping to overcome the limitations of such content-based CF in building item profiles. This process has a total of 9 steps. First, collect meta data such as html through web scraping. Second, preprocess the collected meta data into meaningful data. Third, save the data in HDFS. Forth, read the saved data and extract keyword candidates. Fifth, set up representative words to form groups of keyword candidates and form sets of relevant words. Sixth, eliminate unnecessary data from relevant data sets through blacklist filtering and identify keywords. Seventh, form the nearest neighbors of a selected movie through keywords and create a similarity difference matrix. Eight, analyze the clustered movie based on the similarity difference matrix and produce a suggestion list. Ninth, visualize and present the data on the suggestion movie list through top N.

This present study utilized only the movie ratings and comments as data. But other additional data can be utilized such as subtitles, synopses, user evaluation on SNS (Social Network Services), etc. Moreover, item information is easily accessible on portal sites or SNS pages to accumulate data. However, as for a new item, if the public is not much interested in it, data accumulation would be difficult. The method is also difficult to apply to classic materials with insufficient amount of available information. They indicate that the cold start problem has yet, to be fully overcome. When clustering words, in addition, this method relies solely on word forms.

Therefore, future study will need to explore a method combinable with the existing collaborative filtering and perform a qualitative assessment in comparison with the conventional system. Furthermore, when clustering words in addition to the form-based clustering, semantic clustering should also be researched.

REFERENCES

Balabanovic, M. and Y. Shoham, 1997. Fab: Content-based, collaborative recommendation. Commun. ACM., 40: 66-72.

Bobadilla, J., F. Serradilla and J. Bernal, 2010. A new collaborative filtering metric that improves the behavior of recommender systems. Knowledge Based Syst., 23: 520-528.

Claypool, M., A. Gokhale, T. Miranda, P. Murnikov and D. Netes *et al*., 1999. Combing content-based and collaborative filters in an online newspaper. MCS Thesis, Department of Computer Science, University of Worcester, Worcester, England, UK.

Cover, T. and P. Hart, 1967. Nearest neighbor pattern classification. IEEE Trans. Inform. Theory, 13: 21-27.

Kumar, A. and M.P.S. Bhatia, 2012. Community expert based recommendation for solving first rater problem. Intl. J. Comput. Appl., 37: 7-13.

Lika, B., K. Kolomvatsos and S. Hadjiefthymiades, 2014. Facing the cold start problem in recommender systems. Expert Syst. Appl., 41: 2065-2073.

Melville, P., R.J. Mooney and R. Nagarajan, 2002. Content-boosted collaborative filtering for improved recommendations. Proceedings of the 8th AAAI National Conference on Artificial Intelligence Vol. 23, July 28-August 1, 2002, AAAI, Edmonton, Alberta, Canada, pp: 187-192.

Mulvenna. M.D., S.S. Anand and A.G. Buchner, 2000. Personalization on the net using web mining. Commun. ACM., 43: 123-125.

Resnick, P. and H.R. Varian, 1997. Recommender systems. Commun. ACM., 40: 56-58.

Zhang, T., R. Ramakrishnan and M. Livny, 1996. BIRCH: An efficient data clustering method for very large databases. Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data Vol. 25, June 04-06, 1996, ACM, Montreal, Quebec, ISBN:0-89791-794-4, pp: 103-114.