Egyptian Informatics Journal 18 (2017) 105–112

Contents lists available at ScienceDirect

Egyptian Informatics Journal

journal homepage: www.sciencedirect.com

Full length article

An effective collaborative movie recommender system with cuckoo search

Rahul Katarya *, Om Prakash Verma [1]

Department of Computer Science & Engineering, Delhi Technological University, Delhi, India

ARTICLE INFO

Article history:
Received 21 November 2015
Revised 8 October 2016
Accepted 18 October 2016
Available online 9 November 2016

Keywords:
Recommender system
Collaborative filtering
k-mean
Cuckoo search optimization
Movie

ABSTRACT

Recommender systems are information filtering tools that aspire to predict the rating for users and items, predominantly from big data to recommend their likes. Movie recommendation systems provide a mechanism to assist users in classifying users with similar interests. This makes recommender systems essentially a central part of websites and e-commerce applications. This article focuses on the movie recommendation systems whose primary objective is to suggest a recommender system through data clustering and computational intelligence. In this research article, a novel recommender system has been discussed which makes use of k-means clustering by adopting cuckoo search optimization algorithm applied on the Movielens dataset. Our approach has been explained systematically, and the subsequent results have been discussed. It is also compared with existing approaches, and the results have been analyzed and interpreted. Evaluation metrics such as mean absolute error (MAE), standard deviation (SD), root mean square error (RMSE) and t-value for the movie recommender system delivers better results as our approach offers lesser value of the mean absolute error, standard deviation, and root mean square error. The experiment results obtained on Movielens dataset stipulate that the proposed approach may provide high performance regarding reliability, efficiency and delivers accurate personalized movie recommendations when compared with existing methods. Our proposed system (K-mean Cuckoo) has 0.68 MAE, which is superior to existing work (0.78 MAE) [1] and also has improvement of our previous work (0.75 MAE) [2].

1. Introduction

A recommendation system has become an indispensable component in various e-commerce applications. Recommender systems collect information about the user's preferences of different items (e.g. movies, shopping, tourism, TV, taxi) by two ways, either implicitly or explicitly [3–7]. An implicit acquisition of user information typically involves observing the user's behavior such as watched movies, purchased products, downloaded applications. On the other hand, a direct procurement of information typically involves collecting the user's previous ratings or history. Collaborative filtering (CF) is the way of filtering or calculating items through the sentiments of other people [8–10]. It first gathers the movie ratings given by individuals and then recommends movies to the target user based on like-minded people with similar tastes and interests in the past. Additional impression on which some recommender systems are based is clustering. Clustering is a popular unsupervised data mining tool that is used for partitioning a given dataset into homogeneous groups based on some similarity or dissimilarity metric [11–14]. Collaborative filtering and clustering have been discussed in detail in the next section. Hybrid cluster and optimization approach is applied to improve movie prediction accuracy. Such a hybrid approach has been used to overcome the limitations of typical content-based and collaborative recommender systems. For clustering, k-means algorithm is applied and for optimization, cuckoo search optimization is implemented. K-means algorithm is an enormously greater clustering

* Corresponding author at: Department of Computer Science & Engineering, Delhi Technological University (Formerly Delhi College of Engineering), Shabad Daulatpur, Main, Main Bawana Road, Delhi 110042, India.
E-mail address: rahulkatarya@dtu.ac.in (R. Katarya).
URLs: http://www.dtu.ac.in/Web/Departments/CSE/faculty/ (R. Katarya), http://www.dtu.ac.in/Web/Departments/CSE/faculty/ (O.P. Verma).
[1] Department of Computer Science & Engineering, Delhi Technological University (Formerly Delhi College of Engineering), Shabad Daulatpur, Main, Main Bawana Road, Delhi 110042, India.

Peer review under responsibility of Faculty of Computers and Information, Cairo University.

Production and hosting by Elsevier

http://dx.doi.org/10.1016/j.eij.2016.10.002
1110-8665/

algorithm when compared to other clustering methods in relations of time, complexity or effectiveness for a particular number of clusters [12,15]. Clustering algorithm with a bio-inspired algorithm such as cuckoo search [16–21] delivers optimize results. The cuckoo search has shown best performance when compared with other algorithms such as genetic algorithms and particle swarm optimization. Simulations and comparison of the cuckoo search were greater to these existing algorithms for multimodal objective functions. To find the best results we have to find the most suitable weight among all possible ones. Cuckoo search was also performed well and showed good results that found the appropriate weights [22]. That is why cuckoo optimization algorithm is also used to obtain optimized weight in our work. Besides being one of the most efficient algorithms, it was found that it takes less time than other algorithms applied to the same dataset. The approach of k-means and cuckoo has been applied to the dataset, and the results have been observed regarding evaluation metrics such as mean absolute error (MAE), standard deviation (SD), root mean square error (RMSE) and t-value. These parameters examined and discussed to evaluate the performance of movie recommendation system. Regarding accuracy and precision, the experiment results reflect that the proposed approach is capable of providing more reliable movie recommendations as compared to the existing cluster-based CF methods. In numerous research, the clustering approaches are conducted with the entire dimensions of data which might lead to somewhat inaccuracy and results in more computation time. In general, designing expert movie recommendations is still a challenge, and discovering effective clustering method is a critical problem in this condition. To address aforementioned, a hybrid model- based movie recommendation approach is proposed to alleviate the issues of both extraordinary dimensionality and data sparsity. That is the reason we selected a cuckoo algorithm with k-means for optimization. On the comparison with some other optimization algorithms, the cuckoo was found to perform better than others. The major contributions of this research work are:

- We proposed a novel recommender system with K-means & cuckoo search optimization.
- Our system is innovative and efficient so far, as it employed Cuckoo search algorithm for excellent recommendations for Movielens Dataset.
- Our hybrid model has 0.68 MAE, which is superior to existing work (0.78 MAE).
- Our model also has excellent improvement of our previous work (0.75 MAE).
- The performance with respect to time is also better as compared to already existing systems.
- We used well known Movielens dataset (http://grouplens.org/datasets/movielens/100k/) to analyze the behavior of our proposed system.

The remainder of this article is planned as follows: Section 2 gives a brief explanation of the related work that was carried out on collaborative recommendation systems and clustering-based collaborative recommendation. The proposed approach called as a k-mean-cuckoo approach for movie recommender system is explained in Section 3. In Section 4, experiment results performed on Movielens dataset are described, and finally summarization of this article with future work are highlighted in Section 5.

2. Background and related work

Recommender systems are based on a variety of approaches such as content based [23,24], collaborative approach [9,25–27], hybrid [28,29]. Furthermost movie recommendation systems are centered on collaborative filtering and clustering. In movie recommender systems the user is asked to rate the movies which user has already seen then these ratings are applied to recommend other movies to the user that user has not perceived by utilizing collaborative filtering that is based on similar ratings. Collaborative filtering [9,10,30–32] is tremendously spreading in such a way that this approach influences most of the recommender systems. Collaborative filtering majorly classified into two principal classes such as memory-based collaborative filtering and model based collaborative filtering. Memory-based collaborative filtering [4,5,33] explores for nearest neighbors in the user space for an active user and dynamically recommend the movies. The shortcomings related to this method are computation complexity and data sparsity. Many authors [34] tried to reduce this computational complexity and memory bottleneck issues such as in item based collaborative filtering technique, in which relations between items were computed for neighborhood region around a target object. They showed in their empirical studies that item-based method could decrease the time of computation as well as deliver rationally correct prediction and accurateness. Model-based collaborative technique [3,4,8,33,35] produces a prebuilt model to collect rating patterns based on the database of users and ratings that can treat the issues of data sparsity and scalability. Model-based collaborative filtering is time-consuming and its offline in nature. Clustering based techniques are broadly used in movie recommendation systems to reduce the problem of scalability. Various researchers applied clustering-based methods on recommender systems that delivered expert recommendations [36–41]. The purpose of clustering is to partition objects into groups known as clusters in such a way that two objects within the same cluster have a minimum distance between them to identify similar objects then clustering process is performed offline to build the model. When a target user arrived, the online module allocates a cluster with a substantial similarity weight to the user, and the prediction rating of a specified item is computed based on the same cluster members instead of searching whole user space. The k-nearest neighbor (kNN) algorithm is the orientation algorithm in collaborative filtering recommendation process which is applied in recommendation process [42–45]. kNN based recommender systems for collaborative filtering recommendation process are reliable and with precise recommendations. A bio-inspired algorithm such as cuckoo search has exclusive background sensing abilities and employ a special method to facilitate the evolution of continuing resolutions into novel and quality recommendations by generating clusters with reduced time as discussed in next section.

3. K-means-cuckoo based collaborative filtering framework

To overcome the limitations of a collaborative recommender system, we propose a hybrid cluster and optimization based technique to improve movie prediction accuracy. Our motive is to design a unified model solution that incorporates user ratings from the Movielens dataset for predictions. We use K-means as clustering algorithm and cuckoo search as optimization algorithm and then apply to Movielens dataset for improved efficient recommender systems Initially k-means clustering algorithm is applied to Movielens dataset for clustering of users into different clusters. The clusters are selected randomly at first then users are inspected one by one by calculating the differences in their ratings and the centroid of the clusters, and if their difference is smallest, then the user gets allocated to the cluster to which they are closest. However, at this moment not assure that each user has been assigned to the real cluster with a minimum difference of centroid. So each user's distance is compared to its cluster mean and with other clusters mean and relocate the users according to the small-

est distance from any cluster's mean. Now this iterative relocation would now continue from this new partition until no more relocations occur. After a point, if no more relocations occurred then that point is the point of completion of the clustering process. The K-means algorithm is made of the following steps given in Fig. 1 [12,15,46,47].

Next cuckoo search optimization algorithm is applied to the resultant of the k-means algorithm for optimizing the results. The cluster is prepared with a fitness function that helps in improving the user's centroid distances, whereas fitness function changes previous centroids for a limited number of iteration (i.e. relocation of centroids to users). Then classify the users again by calculating the minimum centroid differences or applying k-means again. Cuckoo Search algorithm may be described using following three idealized rules [16–21]:

(a) Every cuckoo puts one egg at a time and dumps its egg in randomly chosen nest.
(b) The finest nests with the high quality of eggs will carry over to the subsequent generations.
(c) A number of existing host nests is fixed, and the egg laid by a cuckoo is exposed by the hosts birth a probability pa $\in [0,1]$.

For the framework, we considered a correlation where a user is considered as an egg. Each nest can be seen as a cluster. Fig. 2 shows a flowchart that displays the stepwise process that, how cuckoo search algorithm is applied. The procedure commences with the initialization where a random population of n host nests is introduced, and a levy flights behavior equation is obtained then fitness is obtained using the fitness function for obtaining an optimal solution. Levy flights is a random walk in which the step lengths are distributed according to a Levy distribution. The step length and Levy stable distribution can be calculated with the help of Laplace and Fourier transformations. It has been implemented in the cuckoo optimizing algorithm, and the flight length comes out to be $x(N) \sim N1/\alpha$ where $0 < \alpha < 2$, x is the random variable and N is step size. The distance that the cuckoo travels can be calculated using the above equation for each iteration. In order we select a random nest, say j then compared the fitness of the cuckoo egg (incoming new solution) with the fitness of the host eggs present in the nest. In case, the value of the fitness function of the cuckoo egg is less than or equal to the value of the fitness function of the randomly chosen nest, and then the randomly selected nest, j is replaced by the fresh resolution as given in Eq. (1).

$$\text{Fitness function} = \text{Current best solution} - \text{Previous best solution} \quad (1)$$

As the value of the fitness function tends to zero, the deviation between solutions decreases with increasing number of iterations and decision is that if the cuckoo egg is alike to a normal egg, it is difficult for the host bird to distinguish between the eggs. The fitness is the difference in solutions, and the new solution is replaced by the randomly chosen if the fitness of the cuckoo egg is greater than the randomly chosen nest, the host bird can distinguish between the host and the cuckoo egg.

Overall, our approach consists of an advanced k-means clustering technique optimized by a bio-inspired algorithm, cuckoo search. Fig. 3 contains the pseudo-code of the proposed framework. The clusters used to classify the users by interest similarity are analogous to the host bird nest and each egg is analogous to a user in the cuckoo optimization algorithm. The initially classified users are considered as host bird eggs. Some random users are selected to initialize the clusters (nests). The clusters are initialized and k-means is implemented to classify the initially selected users as shown in the algorithm. The users not selected are randomly chosen. For each randomly selected user a cluster is selected randomly and the fitness function. According to the fitness function calculated for that user an egg (user) may be detected by the host bird or it may remain in the nest. Once the cuckoo egg hatches then, it tries to throw other eggs randomly out the nest. This is done in the algorithm if the fitness of the cuckoo egg is better than that a predefined percentage of a number of users already present in the cluster.

Fig. 4 shows the above-described approach applied on the Movielens dataset. The Movielens dataset is recorded by reading the file and dataset is divided into clusters using k-means clustering into k clusters so that each cluster has a centroid. The distance between the user and the centroid is calculated, and the user is placed in the cluster whose centroid is the least distance away from him. When all such users have been relocated, the centroids are relocated and the new positions calculated. Consequently, the estimated rating that the user will give is calculated, and framework is optimized using cuckoo search algorithm. We have calculated various evaluation metrics for predicting the accuracy of recommender system such as: mean absolute error, standard deviation, root mean square error and t-value, which are satisfactory for comparing diverse recommender systems with the framework.

4. Experiment results and analysis

We study the public Movielens dataset to conduct the experiments, which is accessible online, having 100,000 ratings by 943 users or participants on 1682 movies, of scale 1–5. As discussed in the previous section, we presented a hybrid framework of k-means and cuckoo search algorithm to achieve an improved movie recommendation system. Framework mentioned in the previous section is applied to the Movielens dataset where data is considered from u1-u5 and U_a-U_b. The Movielens dataset is divided into different files. The dataset is divided into 80% training data and 20% test data for verification of the result. Movies are classified into 19 types viz. action, animation, horror, comedy, etc. which is also mentioned in 'u.item' file. Information about the user is present in 'u.user' file. To check the performance of recommender system framework, various metrics were calculated which include a MAE, SD, root mean squared error (RMSE) and t-value. Results have been shown in tables and graphs to deliver an enhanced understanding of the relationship between various parameters and the number of clusters which helps us to study the performance of our approach. Detailed analysis & behavior of recommender system framework is given below.

1. Place K points into the space specialized by the users that are being clustered. These points represent an initial set of centroids.
2. Assign each user to the group (cluster) that has the closest centroid.
3. When all users have been assigned, recalculate positions of K centroids for each cluster.
4. Repeat steps two and three till the centroids no longer move. This produces a separation of the users into a group (clusters) from which the metric to be minimized can be calculated.

Fig. 1. K-means algorithm approach.

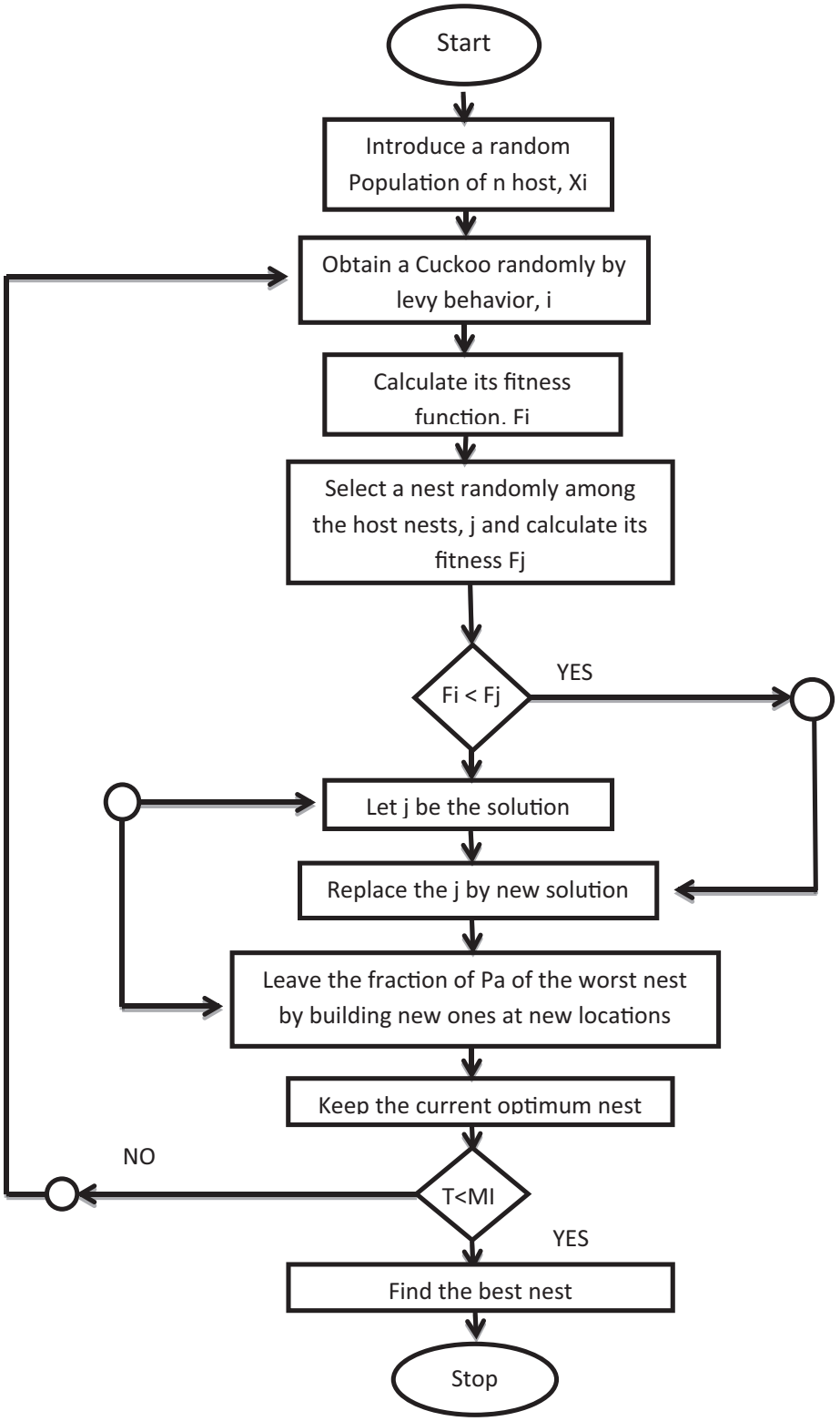

Fig. 2. A flowchart depicting the process involved in the application of cuckoo search algorithm.

4.1. Mean absolute error (MAE)

The mean absolute error is calculated for the Movielens dataset.

$$\mathrm{MAE} = \frac{\sum \left|\widetilde{P}_{i,j} - r_{i,j}\right|}{M} \tag{2}$$

where M is the entire number of expected movies, P_{ij}, represents the predicted value for user i on item j, and rij is the true rating.

$$\mathrm{MAE} = \frac{\mathrm{AE}}{\mathrm{n} * \mathrm{m}} \tag{3}$$

$|e_i| = |f_i - y_i|$, where f_i is the prediction value, and y_i is the true value.

```
Initialization Parameters
k=number of clusters,
n=number of users,
m=number of movies,
itr=number of iterations,
remaining_pool[n]= bool value, remaining_pool[i] representing ith user assigned to any
cluster or not;
no_of_elements_in_clus[i]-number of users present in cluster i
no_of_users_in_remaining_pool=no of users without assigning to any cluster
1. start
2. randomly select some users
3. fill the remaining_poll with the users left
4.  initialize the nests(clusters)
5. for each user i among previously selected
       min_diff=INT_MAX
       for each cluster j
       diff = Euclidian distance between user and cluster
            If diff<min_diff:
               min_diff=diff
               min_index=j
        assign cluster min_index to i
6. while itr>0
      for each cluster i
         For each movie l rated by j
            For each user j belonging to i
                Calculate mean rating for each movie
7.  For each user i
        For each cluster j
           Calculate diff(i,j)
             Assign user to cluster with min diff
               Update mean for each movie in assigned cluster
   Itr--
8. from each user in remaining_pool
      Randomly select a user i
         Randomly select a cluster j
            Calculate fitness of the user in that cluster
            If fitness(i,j)>no_of_elements_in_clus(j)*including factor
               Replace worst fit user with probability p
               With probability (1-p) add the user to that group without replacing
                  no_of_users_in_remaining_pool--;
9. calculate the predicted rating by each user for each user
10. compare the ratings with actual rating
11. compute the essential factors for efficiency comparison
```

Fig. 3. Pseudo-code of k means-cuckoo search framework.

The MAE has been calculated for different values of k (number of clusters), and the result has been shown in Table 1. It is detected that there is a gradual decrease in the value of MAE as we increase the number of clusters from 4 to 68. When the number of clusters is 4, the MAE is 0.825169 and it drops to 15% as the number of clusters is increased by 64. Thus, we can say that MAE decreases with increase in the number of clusters. The possible description for such behavior is that as we increase the clusters, the closeness between the objects going into the cluster increases. Close elements remain in the same cluster and prediction becomes more and more accurate. As the number of clusters increase, each user gets more choice to be assigned to a cluster, and each cluster may get less number of users assigned to it as the number of clusters increase since the number of users is fixed. The closeness of the user with the cluster assigned to it increases as the number of clusters increase. Hence, the difference between the calculated value and the actual value decrease that decrease the mean absolute error.

4.2. Standard deviation (SD)

The standard deviation is obtained for the Movielens dataset.

$$SD = \frac{\sum_{i=1}^{k}\left\{\sum_{j=1}^{\text{no. of elements in i}}\left(\sum_{l=1}^{m}\sqrt{\frac{(\text{expect l}-\text{mean Kn l})^2}{m}}\right)\right\}}{\text{no. of elements in i}} \quad (4)$$

As the number of clusters increase then standard deviation also decreases in Table 2. At k = 4, the value of standard deviation is 0.235333 which gets approximately halved when the number of clusters is increased by 64. Exceptional performance is observed when standard deviation increases with an increase in the cluster from 28 to 32. Such behavior is attributed to non-uniform nature of the Movielens dataset. As the numbers of cluster increase, each user gets more choice to be assigned to a cluster. Each cluster may get less number of users assigned to it as the number of clusters increase since the number of users is fixed.

4.3. Root mean squared error (RMSE)

The Root Mean Squared Error (RMSE) is a frequently used measure of the deviation between values predicted by a model and the values observed from the environment that is being modeled. The difference between the calculated and observed values are squared

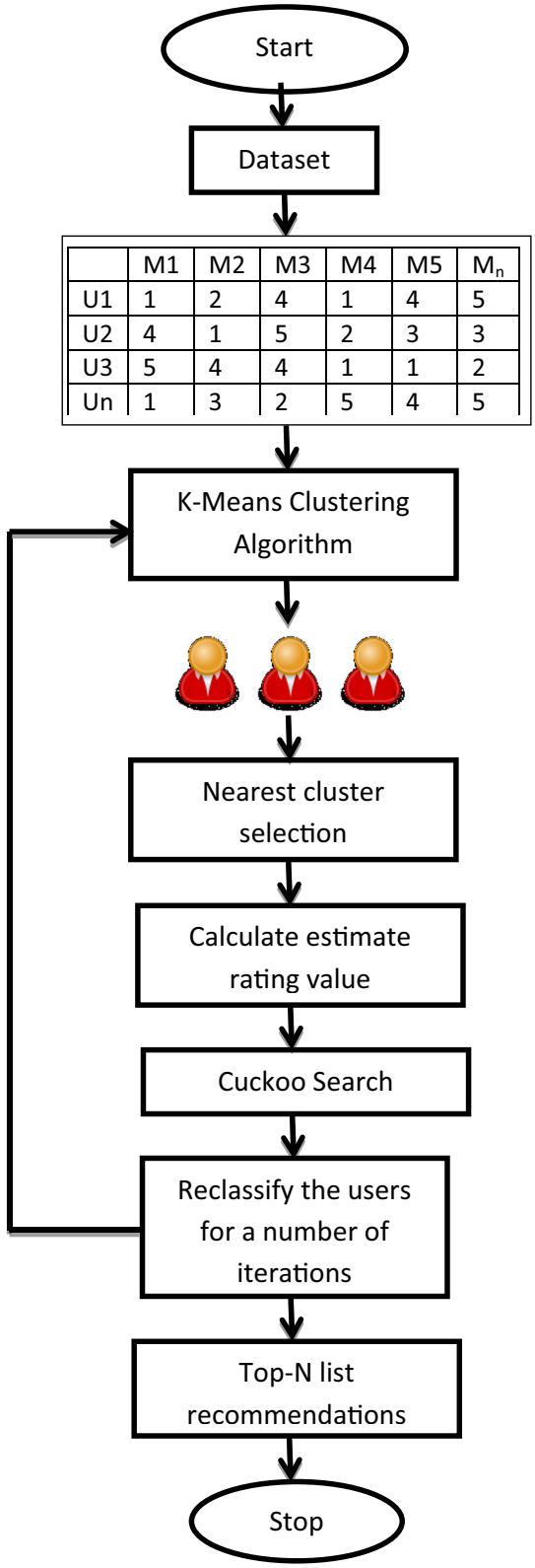

Fig. 4. Movie recommender system framework.

and then summed n times. n divides the result and then raised to power half. The RMSE of a model prediction on the estimated variable Xmodel is defined as the square root of the mean squared error. Where Xobs has detected values and Xmodel is modeled values at time i.

Table 1
MAE for different values of k (number of clusters) between 4 and 68.

S. no.	Numbers of clusters (k)	Mean absolute error
1	4	0.825169
2	8	0.806630
3	10	0.795371
4	12	0.790461
5	16	0.781465
6	20	0.776206
7	28	0.768038
8	32	0.761306
9	36	0.754507
10	40	0.744677
11	44	0.738921
12	48	0.732927
13	52	0.726212
14	56	0.716778
15	60	0.712400
16	64	0.684224
17	68	0.697293

Table 2
Movielens dataset with standard deviation and numbers of clusters.

S. no.	Numbers of clusters (k)	Standard deviation
1	4	0.235333
2	8	0.211460
3	10	0.184073
4	12	0.184289
5	16	0.150996
6	20	0.126688
7	28	0.121020
8	32	0.127900
9	36	0.126319
10	40	0.133401
11	44	0.127355
12	48	0.130083
13	52	0.119024
14	56	0.121663
15	60	0.112847
16	64	0.109447
17	68	0.119283

Table 3
RMSE with numbers of clusters for Movielens dataset.

S. no.	Number of clusters (k)	RMSE
1	4	1.30652
2	8	1.29750
3	10	1.29394
4	12	1.28899
5	16	1.28522
6	20	1.28070
7	28	1.27693
8	32	1.27110
9	36	1.26650
10	40	1.25802
11	44	1.25345
12	48	1.25150
13	52	1.24576
14	56	1.24162
15	60	1.23921
16	64	1.23639
17	68	1.23104

RMSE for each element in test case

$$= \sqrt{\left(\sum(\text{predicted rating} - \text{actual rating})(\textit{predicted rating} - \textit{actual rating})/\text{n}\right)} \quad (5)$$

In Table 3, it is observed that the value of RMSE decreases gradually as we increase the number of clusters. At k = 4, the value of RMSE is 1.3062 which decreases by 6% approx. Such behavior is observed because increasing the number of clusters increase the similarity between the users and the cluster assigned to them.

Table 4
t-value with a number of clusters for Movielens dataset.

S. no.	Numbers of clusters (k)	t-value
1	4	4.12583
2	8	3.99549
3	10	3.36057
4	12	3.39319
5	16	3.39319
6	20	2.81489
7	28	2.79138
8	32	2.79138
9	36	2.81693
10	40	2.81693
11	44	2.81693
12	48	2.81693
13	52	2.81693
14	56	2.81693
15	60	2.81693
16	64	2.81693
17	68	2.81693

Table 5
Comparisons of evaluation metrics with different methods.

Method	Mean	Standard deviation
PCA-SOM	0.98	0.07
SOM-CLUSTER	0.75	0.06
UPCC	0.81	0.11
KMEANS-CLUSTER	0.69	0.10
PCA-KMEANS	0.93	0.12
GAKM-CLUSTER	0.76	0.05
PCA-GAKM	0.98	0.17
K-means-cuckoo (Proposed)	0.68	0.10

4.4. t-value

The t-value for a movie recommendation system is obtained as follows

$$\text{T-value} = \sum_{i=1}^{k}\sum_{j=1}^{k}\left(\frac{\overline{Xi}-\overline{Xj}}{\sqrt{\frac{(SD)^2 i}{(no.\ of\ elements\ in\ i)}+\frac{(SD)^2 j}{(no.\ of\ elements\ in\ j)}}}\right) \quad (6)$$

The t-value depend on upon the values of the mean of different clusters and their standard deviation as shown in Table 4. The change in values of these factors has been explained. The t-value decrease due to the same reasons. However, it is observed to be more or less constant after a fixed number of clusters. The experiment results compared with current existing methods are given in Table 5 [1]. Where all the methods were evaluated on 64 clusters: In PCA-SOM mean value is 0.98. Principal component analysis (PCA) is a scientific procedure that converts correlated variables into a smaller number of uncorrelated variables called principal components. The self-organizing map (SOM) defines a mapping from a higher-dimensional input space to a lower dimensional map space. SOM-Cluster has the mean value as 0.75. UPCC has the average value as 0.81. Kmeans-cluster method has the average as 0.69. PCA with k-means combination as PCA-K-Means has the mean value as 0.93. Genetic Algorithms (GAs) are adaptive heuristic search procedure based on the evolutionary concepts of natural selection and genetics. GAKM-Cluster combination has the mean value as 0.76 and similarly PCA-GA-KM has mean value as 0.98. However, K-means-cuckoo provides best mean as 0.68 when compared to other methods.

As represented in Table 5, it is observed that the mean absolute error is observed to be the least for K-means-cuckoo framework as compared to other methods. Here, a number of clusters used in Table 5 are 64 for every method that was evaluated on Intel i5 processor machine with 4 GB RAM. Thus, the current framework may be used to provide better performance in recommending movies to users. The t-value and RMSE values are significantly low as compared with other approaches thereby proving that the hybrid of k means and cuckoo search algorithm has better performance and accuracy as compared to other methods on Movielens dataset [1]. Table 6 shows the values of mean absolute error obtained by different methods with the various values of k (number of clusters) and Table 7 presents the finish time for various methods.

5. Conclusion and future work

In this article hybrid of k-means and cuckoo search is applied to the Movielens dataset to achieve an improved movie recommendation system. We measured the performance of our approach regarding MAE, RMSE, SD, and t-value. The experiment outcomes on the Movielens dataset discussed indicated that the approach that we discussed provide high performance regarding accuracy and were capable of providing reliable and personalized movie recommendation systems with the specific number of clusters. Evaluation metrics (for a given number of clusters) originated to be lesser than those of other methods. Some limitations of our proposed approach are that if the initial partition does not turn out to work well then, efficiency may decrease. For future work use of various other nature inspired algorithms in place of cuckoo search algorithm can be used. The greatest promising method is the one in which the algorithms used in clustering and optimization provide the best presentation regarding accuracy and speed.

Table 6
MAE values for different approaches for different values of k (number of clusters).

No. of clusters (k)	Mean absolute error (MAE)							
	PCA-GAKM	PCA-SOM	SOM-CLUSTER	UPCC	KMEANS-CLUSTER	PCA-K-MEANS	GAKM-CLUSTER	K-MEANS CUCKOO
8	0.94	1.3	0.78	0.82	0.80	0.92	0.85	0.80
16	0.94	1.1	0.80	0.82	0.77	0.92	0.85	0.78
32	0.94	0.98	0.78	0.80	0.75	0.94	0.86	0.76
64	0.98	0.98	0.75	0.81	0.69	0.93	0.76	0.68

Table 7
Comparison of speed of different approaches for Movielens dataset.

Methods	PCA-GAKM	PCA-SOM	SOM-CLUSTER	UPCC	k-means-Cluster	PCA-K-MEANS	GAKM CLUSTER	K-Means Cuckoo
Speed (in seconds)	29.32	147.73	86.92	179.34	21.25	65.56	325.71	63.22

References

[1] Wang Z, Yu X, Feng N, Wang Z. An improved collaborative movie recommendation system using computational intelligence. J Vis Lang Comput 2014;25:667–75.
[2] Katarya R, Verma OP. A collaborative recommender system enhanced with particle swarm optimization technique. Multimed Tools Appl 2016;75:1–15.
[3] Lu J, Wu D, Mao M, Wang W, Zhang G. Recommender system application developments: a survey. Decis Support Syst 2015;74:12–32.
[4] Bobadilla J, Ortega F, Hernando a, Gutiérrez a. Recommender systems survey. Knowl-Based Syst 2013;46:109–32.
[5] Chen L, Chen G, Wang F. Recommender systems based on user reviews: the state of the art. User Model. User-Adapt Interact 2015;25:99–154.
[6] Konstan Ja, Riedl J. Recommender systems: from algorithms to user experience. User Model. User-Adapt Interact 2012;22:101–23.
[7] Katarya R, Verma OP. Recent developments in affective recommender systems. Phys A Stat Mech Appl 2016;461:182–90.
[8] Isinkaye FO, Folajimi YO, Ojokoh Ba. Recommendation systems: principles, methods and evaluation. Egypt Inform J 2015.
[9] Su X, Khoshgoftaar TM. A survey of collaborative filtering techniques. Adv Artif Intell 2009;2009:1–19. , <http://www.hindawi.com/journals/aai/2009/421425/> [accessed May 21, 2013].
[10] Shi YUE, Larson M, Hanjalic A. Collaborative filtering beyond the user-item matrix: a survey of the state of the art and future challenges. ACM Comput Surv 2014;47:1–45.
[11] Fahad a, Alshatri N, Tari Z, Alamri a, Khalil I, Zomaya a, et al. A survey of clustering algorithms for big data: taxonomy & empirical analysis. IEEE Trans Emerg Top Comput 2014;2. 1–1.
[12] Hartigan Ja, Wong Ma. A K-means clustering algorithm. J R Stat Soc 1979;28:100–8.
[13] Jain aK, Murty MN, Flynn PJ. Data clustering: a review. ACM Comput Surv 1999;31:264–323.
[14] Mukhopadhyay A, Maulik U, Bandyopadhyay S. A survey of multiobjective evolutionary clustering. ACM Comput Surv 2015;47.
[15] Kanungo T, Mount DM, Netanyahu NS, Piatko CD, Silverman R, Wu aY. An efficient k-means clustering algorithm: analysis and implementation. IEEE Trans Pattern Anal Mach Intell 2002;24:881–92.
[16] Yang XS, Deb S. Multiobjective cuckoo search for design optimization. Comput Oper Res 2013;40:1616–24.
[17] Walton S, Hassan O, Morgan K, Brown MR. Modified cuckoo search: a new gradient free optimisation algorithm. Chaos, Solitons Fractals 2011;44:710–8.
[18] Yang X-S, Deb S. Cuckoo search via levy flights, <http://arxiv.org/abs/1003.1594>; 2010.
[19] Yang X-S, Deb S. Cuckoo search: recent advances and applications. Neural Comput Appl 2014;24:169–74.
[20] Yang X-S, Deb S. Cuckoo search via levy flights. Chaos, Solitons Fractals 2013;44:1561–6.
[21] Gandomi AH, Yang X-S, Alavi AH. Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems. Eng Comput 2013;29:17–35.
[22] Hatami M, Pashazadeh S. Improving results and performance of collaborative filtering-based recommender systems using cuckoo optimization algorithm. Int J Comput Appl 2014;88:46–51.
[23] Lops P, De Gemmis M, Semeraro G. Content-based recommender systems: state of the art and trends. Recomm Syst Handb 2011:1–33. <http://link.springer.com/chapter/10.1007/978-0-387-85820-3_3>.
[24] Pazzani MJ, Billsus D. Content-based recommendation systems 2007:325–41.
[25] Huang Z, Zeng D, Chen H. A comparison of collaborative-filtering algorithms for e-commerce. IEEE Intell Syst 2007;22:68–78.
[26] Ekstrand MD. Collaborative filtering recommender systems. Found Trends® Human-Comput Interact 2010;4:81–173.
[27] Adomavicius G, Tuzhilin A. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. Adapt Web 2005;69:253–60.
[28] Burke R. Hybrid web recommender systems. Adapt Web 2007:377–408. doi: http://dx.doi.org/10.1007/978-3-540-72079-9_12.
[29] Kim BM. Clustering approach for hybrid recommender system. In: Proc IEEE/WIC Int Conf Web Intell (WI 2003). p. 33–8.
[30] Krzywicki a, Wobcke W, Kim YS, Cai X, Bain M, Mahidadia a, et al. Collaborative filtering for people-to-people recommendation in online dating: data analysis and user trial. Int J Hum Comput Stud 2015;76:50–66.
[31] Xu Y, Yin J. Collaborative recommendation with user generated content. Eng Appl Artif Intell 2015;45:281–94. doi: http://dx.doi.org/10.1016/j.engappai.2015.07.012.
[32] Adomavicius G, Tuzhilin a. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. IEEE Trans Knowl Data Eng 2005;17:734–49.
[33] Lü L, Medo M, Yeung CH, Zhang Y-C, Zhang Z-K, Zhou T. Recommender Syst 2012:97. , <http://arxiv.org/abs/1202.1112> [accessed August 22, 2014].
[34] Sarwar B, Karypis G, Konstan J, Riedl J. Item-based collaborative filtering recommendation algorithms. Proceedings of the 10th international conference on World Wide Web. ACM; 2001. p. 285–95. doi: http://dx.doi.org/10.1145/371920.372071.
[35] Zhang R, Bao H, Sun H, Wang Y, Liu X. Recommender systems based on ranking performance optimization. Front Comput Sci China 2015;1–11.
[36] Nilashi M, Jannach D, Bin Ibrahim O, Ithnin N. Clustering- and regression-based multi-criteria collaborative filtering with incremental updates. Inf Sci (Ny) 2014;293:235–50.
[37] Guo G, Zhang J, Yorke-Smith N. Leveraging multiviews of trust and similarity to enhance clustering-based recommender systems. Knowl-Based Syst 2014.
[38] Thong NT, Son LH. HIFCF: an effective hybrid model between picture fuzzy clustering and intuitionistic fuzzy recommender systems for medical diagnosis. Expert Syst Appl 2015;42:3682–701.
[39] Kohrs A, Merialdo B. Clustering for collaborative filtering applications. Comput Intell Model Control Autom. IOS Press; 1999. , <http://citeseerx.ist.psu.edu/viewdoc/summary?>.
[40] Xue G-R, Lin C, Yang Q, Xi W, Zeng H-J, Yu Y, et al. Scalable collaborative filtering using cluster-based smoothing. Scalable Collab Filter Using Clust Smoothing 2005:114.
[41] Georgiou O, Tsapatsoulis N. Improving the scalability of recommender systems by clustering using genetic algorithms. Lect Notes Comput Sci 2010;6352:442–9.
[42] Z. Lu, H. Shen, A Security-assured Accuracy-maximised Privacy Preserving Collaborative Filtering Recommendation Algorithm, (2015) 1–9. http://arxiv.org/abs/1506.0001.
[43] Borràs J, Moreno A, Valls A. Intelligent tourism recommender systems: a survey. Expert Syst Appl 2014.
[44] Soares M, Viana P. Tuning metadata for better movie content-based recommendation systems. Multimed Tools Appl 2014:1–22.
[45] Borràs J, Moreno A, Valls A. Intelligent tourism recommender systems: a survey. Expert Syst Appl 2014;41:7370–89.
[46] Ahmad A, Dey L. A k-mean clustering algorithm for mixed numeric and categorical data. Data Knowl Eng 2007;63:503–27.
[47] Pham DT, Dimov SS, Nguyen CD. Selection of K in K-means clustering. Proc Inst Mech Eng Part C J Mech Eng Sci 2005;219:103–19.