ISSN 2007-9737

Collaborative Recommender System Based on Improved Firefly Algorithm

Bharti Sharma[1], Adeel Hashmi[2], Charu Gupta[3], Amita Jain[4]

[1] Maharaja Surajmal Institute of Technology
Department of Information Technology
Delhi, India

[2] Maharaja Surajmal Institute of Technology,
Department of Computer Science and Engineering,
Delhi, India

[3] Bhagwan Parshuram Institute of Technology
Department of Computer Science and Engineering,
Delhi, India

[4] Netaji Subhas University of Technology,
Department of Computer Science and Engineering,
Delhi, India

charugupta0202@gmail.com, amita_jain_17@yahoo.com

Abstract. A recommendation system aims to capture the taste of the customer and predict relevant items which he/she may be interested in buying. There are many algorithms for generating recommendations in literature, however, most of them are non-optimal and do not have the capability to handle big data. In this paper, a collaborative recommendation system is proposed based on improved firefly algorithm. The firefly algorithm is used to generate optimal clusters which provide effective recommendations. The proposed algorithm works in two phases: Phase I which generates the clusters with firefly algorithm and Phase II gives real time recommendations. The firefly algorithm has been implemented in Apache Spark to give it the capability of handling big data. The combination of improved firefly-based clustering and Apache Spark makes it much faster and optimal than the state-of-the-art recommendation models. For experiments, movie-lens dataset has been utilized and different evaluation metrics have been used for performance analysis. The results show that the proposed method gives better results compared to existing methods.

Keywords. Clustering, collaborative filtering, firefly algorithm, recommender system, swarm intelligence

1 Introduction

Recommender systems aim to suggest the items a customer might like based on the information about his/her preferences and ratings. Recommendation system can be viewed as an extension to association/pattern mining. It has been observed if an item B is associated with item A then whenever any user buys item A, he is recommended item B and vice-versa [1, 2].

Recommendation systems are useful for both buyers and sellers since they reduce buyer's effort and increase sales. These systems are put to use in many fields like e-commerce websites, news filtering, web searches, online dating, social networking sites [3, 4, 5]. Movie recommendation or movie rating prediction is a popular use-case of recommender systems [6, 7]. It is analyzed that the state-of-the-art methods are slow, non-scalable and their achieved accuracy needs improvement. In this paper, a fast and scalable method to generate recommendations is proposed which is optimized by improved firefly algorithm.

The traditional methods of clustering like k-means algorithm are slow, so firefly optimization algorithm is used to create clusters. This firefly clustering algorithm is made scalable and parallelized by utilizing Apache Spark tool.

Firefly is a population based algorithm which has some additional advantages as compared to single point search algorithms. Some of the most important fields of its application are optimization of dynamic and noisy environment and constraints, combinatorial and multi-objective optimization. Apart from the field of optimization it is also capable of solving classification problems that we come across in the fields of neural network, data mining and machine learning. Clustering techniques are used to group similar items or objects together based on unsupervised learning. In this paper, the data set is divided based on random cluster heads and then the cluster-heads are re-calculated iteratively for optimal use in Firefly algorithm. The detailed working methodology and mathematical foundation is given in section 3.

The remainder of the paper is organized as follows. Section 2 surveys the various recommender algorithms. Section 3 introduces the vanilla version of firefly algorithm. Section 4 explains the working of proposed algorithm, improved firefly algorithm to generate optimal recommendations. Section 5 provides experimental results and analysis. Section 6 presents the conclusion along with future directions.

2 Recommender Algorithms

Recommender algorithms fall into three categories [8, 9]: content-based, collaborative and hybrid as shown in Figure-1. Collaborative filtering is based on the concept of user-ratings, where ratings given to the products by every user are stored, and for a user X the persons who have similar rating pattern are identified, and those products are recommended which were given high ratings by this identified group of people.

The recommender systems in addition to collaborative filtering also has approaches based on content-based methods on information retrieval, Bayesian inference, and case-based reasoning methods [10, 11]. These methods take the actual content or attributes of the items to make recommendation (instead of or in addition to patterns with user rating). Content-based algorithms recommend to a customer those items which are similar to items that the same customer has bought or searched in the past. Hybrid recommender systems [12] have also emerged as a recommendation technique combining content-based and collaborative algorithms into composite systems that build on the strengths of their algorithmic components.

Content-Based Filtering systems recommend an item based on the contents of that item. If a user has previously searched for, or looked at some items with attribute ‘A’ then more items with attribute A will be recommended. Thus recommendations are made by comparing the contents of an item with the profile of the target user. The profile of a user is built from his history of interaction with the system by modeling the user’s preferences. The attributes can be assigned automatically or manually. The attributes have to be represented such that the user profile and the items can be compared to extract meaningful relations. A learning algorithm which can create the user profile based on items bought/viewed is also needed [13].

Collaborative Filtering (CF) is a process in which ratings are obtained from the users and recommendations to a new user are given based on opinions of other users with similar taste. The items that are recommended to a user are based upon his/her similarity to other users. For example, if two users X and Y have shown similar preferences in the past then the items which are liked by X in the future will be recommended to Y and vice versa. So basically, this algorithm assumes that if some user A has the same view as user B on an issue then A is more likely to have the same view as B on any other issue as well [14]. Collaborative Filtering based approach can be further divided into two categories: Memory-based and Model-based.

Memory-Based approach is a simple approach which makes use of a similarity measure to find users/items which are related to the active user or

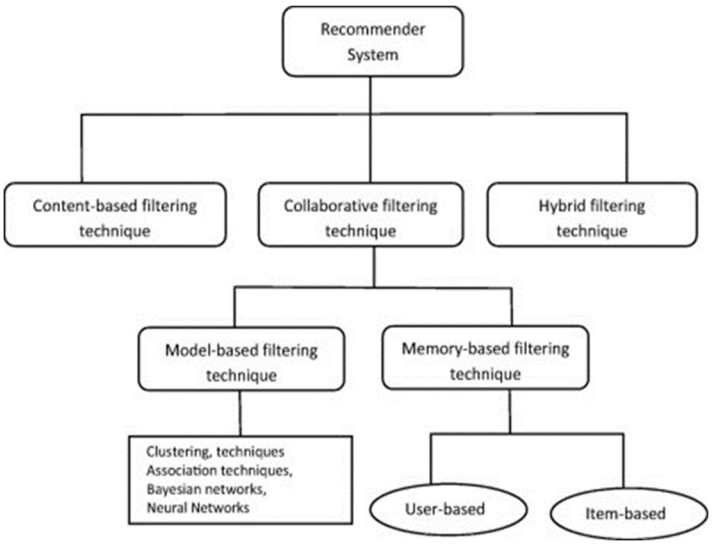

Fig. 1. Types of Recommender Systems

to the items bought/viewed by the active user. The methods that can be used to find out the similarity between two users are Euclidean distance, cosine similarity, correlation, etc.

i. User-based approach: User–user collaborative filtering was the first of the automated CF approach. It was first introduced in the GroupLens Usenet article recommender [15]. Then those items which have been highly rated by most of these users are identified and recommended to the active user [16, 17, 18].

ii. Item-based approach: Item-based collaborative filtering models the ratings item-wise and not user-wise. An item-item matrix is built to determine relationship between every pair of items. A similarity measure like correlation is used to build this matrix. When a customer rates an item A then this matrix is looked up to find the items which have highest similarity with A in the matrix. Slope [19, 20]

Model-Based approach is basically dependent upon machine learning, data mining algorithms to make predictions. In this approach the aim is not to find most similar users/items, but to develop a model to classify the user and recommend highly rated items of other users belonging to the same class. The machine learning algorithms used in model-based approach are clustering, Bayesian networks, singular value decomposition (SVD), etc. This model has advantage over memory-based approach in the sense that it provides faster recommendations, handles sparsity better than memory-based ones, scales with dataset and has better prediction performance.

Many recommender systems combine the memory-based and model-based collaborative filtering algorithms which can be called **hybrid collaborative filtering**. This type overcomes the limitations of both the other types but increases complexity and is expensive. The hybrid approach can be used to overcome some of the common problems that occur when either of the other two approaches is used independently. It has been observed that the hybrid approach provides more accurate recommendations than either of the two approaches [21]. A popular example of hybrid approach is content-boosted collaborative filtering [22]. Apart from these popular techniques, there are some other recommendation techniques as well.

Knowledge-Based Recommenders. These recommender systems area specific kind of

recommender systems that are based on prior knowledge about all the items that are available and also knowledge about user preferences [23].

Demographic recommenders. As the name suggests, these recommender systems provide recommendations based on a demographic profile of the user. The ratings given by users in a particular demographic section are used to provide recommendations to a user of that particular section. Here are also few problems which are encountered by recommender systems like cold start, sparsity, trust and privacy [24].

3 Firefly Algorithm

Firefly algorithm developed by Yang in 2008 [25] is a meta-heuristic algorithm used to solve optimization problems. Firefly algorithm is among those stochastic algorithms which follow randomization approach to search the solution in the data set. In this section, the biological, mathematical foundation and behavior of firefly is presented. It also explains the intuition and foundation of clustering with firefly algorithm.

3.1 Biological Foundation and Behavior

Fireflies are distinguished by their flashing light which is produced by a biochemical process also known as bioluminescence [26, 27, 28] . The rhythmic flashes are used as signals for mating [29, 30]. Apart from attracting the mating partners, these bright lights are used as warning signals from potential predators.

Firefly algorithm produced use the following assumptions [25]:

- The brightness of the firefly corresponds to the objective function.
- Each firefly is attracted to all other fireflies as they are unisex.
- A brighter firefly is more attractive, and a less bright firefly will move towards a firefly which is brighter. The attractiveness/brightness decreases as the distance between the fireflies' increases.

3.2 Mathematical Formulation

The light intensity I(r) varies according to the inverse square law:

$$I(r) = I_s / r^2 , \quad (1)$$

where, I_s is the light intensity at source.

The light intensity I varies with distance r for a stated medium with absorption coefficient λ, acc. to the equation:

$$I = I_0 e^{-\lambda r} , \quad (2)$$

where, I_0 is the actual light intensity.

The Eq-1 and Eq-2 can be combined to give the following equation:

$$I = I_0 e^{-\lambda r^2} , \quad (3)$$

Taking the above equations into consideration, the attractiveness β of a firefly can be defined as:

$$\beta = \beta_0 e^{-\lambda r^2} , \quad (4)$$

where β_0 is the attractiveness of the firefly at r=0.

In the real time environment, the attractiveness function of the firefly i.e. $\beta(r)$ can be any monotonically deceasing function described in the generalize form as:

$$\beta(r) = \beta_0 e^{-\lambda r^m} \ (m \geq 1) , \quad (5)$$

The Cartesian distance (*r*) is the distance between any two random fireflies *i* and *j* at location x_i and x_j, respectively:

$$r_{ij} = \left\| x_i - x_j \right\| = \sqrt{\sum_{k=1}^{d} (x_{i,k} - x_{j,k})^2} , \quad (6)$$

where, $x_{i,k}$ is the k^{th} dimension of the spatial coordinate x of the i^{th} firefly.

In 2-dimensional case, we have:

$$r_{ij} = \sqrt{(x_i - x_j)^2 - (y_i - y_j)^2}\,, \tag{7}$$

Firefly *i* is attracted to another more attractive firefly *j* according to equation (8):

$$x_i = x_i + \beta e^{-\lambda r^2}(x_j - x_i)\,, \tag{8}$$

The value of the parameters λ plays a significant role determining the speed of convergence and it follows the range of 0 to 10.

The pseudo-code for firefly algorithm is given below in Figure-2.

The above algorithm is only for exploitation part (finding the local best solution). For exploration part (to find global solution), we make use of the Levy flight instead of the traditional method. To make the process of exploitation faster, the less bright fireflies are moved towards brightest firefly only instead of all the brighter fireflies [31, 32].

3.3 Clustering Using Firefly Algorithm

In this section, the main aim is to calculate cluster heads by minimizing the sum of calculated distances of the patterns with their cluster heads [34, 35]. The function to be minimized during clustering process can be described as given in Eq-9:

$$J(k) = \sum_{k=1}^{M} \sum_{i \in c_k} (x_i - c_k)\,, \tag{9}$$

where M is the no. of clusters, c_k is the cluster head of k^{th} cluster, and x_i is a data point belonging to the cluster.

The cluster head of a cluster is the centroid of the cluster. The centroid of a cluster with n points can be calculated by Eq-10:

$$c_k = \sum_{i \in c_k} \frac{x_i}{n_k}\,, \tag{10}$$

where n_k is number of points in the k^{th} cluster.

By performing clustering, we can divide a dataset into different groups based on some similarity measures. Most widely used similarity measures are based on distance calculation between the dataset and the cluster heads [36].

The cluster heads are calculated by minimizing the Euclidean distance between each data instance X_i and the cluster center c_k. The cost function for the pattern *i* is given by Eq-11:

$$f_i = \frac{1}{D} \sum_{j=1}^{D} d(x_j, p_i^{C_q})\,, \tag{11}$$

where D is the count of data instances, and $p_i^{C_q}$ defines the class *q* to which the instance *i* belongs. The proposed pseudo-code for clustering through firefly algorithm is given in Figure-3.

4 Proposed Firefly Recommendation System (FRS)

The proposed Firefly Recommendation System i.e. (FRS) works in *two phases* which includes training phase and recommendation phase. **Phase I** is an offline process in which rating matrix is produced from the collected data and clusters are obtained using firefly clustering algorithm. **Phase II** is a real-time process in which the recommendations for current user are generated.

In this phase, the active user is assigned a recommendation cluster and recommendations are generated.

Phase I: Training Phase

The movie-lens dataset has 100,000 ratings of 943 users on 1682 movies. The movies are classified into 19 genres viz. action, comedy, horror, etc. The dataset is divided into two parts, 80% as training data and 20% as test data.

The data is converted into a 943X1682 matrix. The dataset need not be normalized as the ratings are in the scale of 1-5. However, the dataset is sparse (only 100,000 ratings out of possible 1,586,126 available), so we need to replace the missing values by 0.

The rating matrix is divided into K clusters using firefly clustering technique. N fireflies are initially generated, each having K cluster-heads.

Firefly Meta-heuristic Algorithm(FFA)

Begin
Initialize the algorithm parameters:
MaxGen: the maximal number of generations
λ : the light absorption coefficient
r : distance from the light source
d : the domain space
Identify the objective principle of f(x), where $x=(x_1,.....,x_d)^T$
Generate the initial population of fireflies x_i (i=1, 2 ,..., n)
Verify the light intensity of I_i at x_i via $f(x_i)$
While (t<MaxGen)
    For i = 1 to n (all n fireflies);
        For j=1 to n (n fireflies)
            If ($I_j > I_i$), move firefly i towards j by using equation (8);
            end if
            Attractiveness varies with distance r via $\exp[-\gamma r^2]$;
            Calculate new result and update light intensity;
            End for j;
        End for i;
        Rank the fireflies and find the current best;
    End while;
    Display results;
End procedure

Fig. 2. Firefly Meta-heuristic

FIREFLY_CLUSTERING (normalized_dataset, k_clusters, N_fireflies)

1. Generate fireflies
   - Create N fireflies, each having k cluster-heads (D-dimensional) with random normalized values.
   - Create clusters for each firefly (based on shortest Euclidean distance).
2. Calculate the fitness of each firefly. The fitness of a firefly is the average of Euclidean distance of all the points from their assigned cluster-head.
   Intensity of firefly (I) = Fitness of firefly
3. Rank the fireflies according to their intensity (higher intensity, higher rank).
4. Compare each firefly (starting with lowest ranked) with other fireflies. Move the firefly towards brighter firefly otherwise move it randomly. (Check that the new position lies in range 0-1).

$$\beta = \beta_0 e^{-\lambda r^2},$$

$$x_i = x_i + \beta e^{-\lambda r^2}(x_j - x_i).$$

5. Obtain the updated cluster-heads of each firefly after step-2. Re-form clusters of each firefly.
6. Re-calculate fitness/intensity of each firefly. Replace old firefly with new firefly if it has higher intensity. Repeat step-2.
7. Repeat Step 3-6 for MaxGenerations (e.g. 1000 times).
8. Display the cluster-heads of the brightest firefly.

Fig. 3. Proposed Firefly clustering algorithm

Each cluster-head has 1682 dimensions having values in the range 1-5, which are generated at random.

For each firefly, K clusters are generated by assigning each point in the dataset to the nearest cluster-head in the firefly (similarity measure used is Euclidean distance), and the WCSS (within-cluster sum of squares) is calculated.

The firefly with the lowest WCSS is considered to be the brightest firefly, and the less bright fireflies are moved towards the brightest fireflies.

The brightest firefly is also moved at random to a position which further increases the intensity of the brightest firefly.

This process is repeated to certain number of iterations, and the fittest firefly after all these

iterations is considered to be the final solution (clusters).

Phase-II: Process of recommendation for active users.

To generate recommendations for an active user, a cluster (among k-clusters) is to be selected. A simple approach is to select the cluster whose centroid has highest similarity with the active user e.g. the centroid with lowest Euclidean distance with the active user. If there are large numbers of clusters, then multiple clusters can also be used for better results. In such a case, the probability that a cluster is chosen for generating recommendations is given by Eq-12:

$$P_i = \frac{\rho_i * d_i}{\sum_{i=1}^{k} \rho_i * d_i}, \tag{12}$$

where, ρ_i is the density of the cluster, and d_i is the Euclidean distance between active user profile and centroid of cluster:

$$\rho_i = \frac{N_i}{\sum_{i=1}^{k} N_i}, \tag{13}$$

where N_i is the number of users in cluster i.

The recommendations are provided from the cluster with highest probability or from multiple clusters which lie in particular probability range. The latter approach may provide the active user recommendations which are different and make him interested in trying something new.

After selecting the clusters for recommendations, next step is to predict the ratings for un-rated items of the active user and recommending the items whose predicted value is high. If there is only one chosen cluster, then the values of unrated items is simply the average of the ratings given for corresponding item by all the users in the cluster.

But if multiple clusters have been selected then we also consider the quality of ratings in each chosen cluster. A criterion of the rating quality of a cluster is the number of ratings available to each item in the cluster, higher the density of ratings better the quality of the cluster:

$$Q_i = \frac{\sum_{p=1}^{t} r_{ip}}{n_i * t}, \tag{14}$$

where, Q_i is the quality of cluster i, n_i is the number of users in the cluster i, t is the number of items, and r_{ip} is the count of ratings available for item p in the cluster i.

5 Experimental Results and Analysis

For performance analysis of our recommendation system framework we calculate various metrics like MAE, SD, RMSE and t-value. Various graphs and tables of the calculated results are shown for better understanding of the framework.

MAE: Mean Absolute Error

We calculated mean absolute error on the dataset of movielens dataset by using Eq-15:

$$MAE = \frac{\sum |p_{ij} - t_{ij}|}{M}, \tag{15}$$

where, M is no. of movies in the dataset, p_{ij} is predicted value for *i* user on *j* items, and t_{ij} is true rating.

The results are shown in Table 4 for the calculated MAE for different values of K. The outcome as observed from this table is that as we increase the number of clusters, MAE values gradually decrease.

SD: Standard Deviation

By using Eq-16, we calculate SD on movie lens dataset:

$$SD = \frac{\sum_{i=1}^{k} \left\{ \sum_{j=1}^{n_i} \left(\sqrt{\frac{\sum_{l=1}^{D} l - \overline{l}}{D}} \right) \right\}}{n_i}, \tag{16}$$

The results of SD with different cluster count are shown in Table-4. The outcome of this calculated metrics is that as the number of cluster increases their SD value decreases.

Table 1. Snapshot of Movielens dataset

	Movie	Movie	…	Movie
	#1	**#2**		**#1682**
User#1	5	3	…	0
User#2	4	0	···	0
.	.	.	…	.
.	.	.		.
.	.	.		.
User#943	0	5	…	0

Table 2. Sample Snapshot of Fireflies (20 fireflies with 3 cluster-heads each)

		Movie	Movie	...	Movie
		#1	**#2**		**#1682**
Firefly#1	**Cluster-Head #1**	2	3	...	3
	Cluster-Head #2	3	1	...	5
	Cluster-Head #3	4	1	...	2
.	.. .	.. .	.. .	...	. .
Firefly#20	**Cluster-Head #1**	5	1	...	1
	Cluster-Head #2	1	2	...	2
	Cluster-Head #3	2	4	...	4

Table 3. Sample of cluster assigned to each user in fittest firefly (assuming that there are 3 clusters in each firefly)

	User	User	User	…	User
	#1	**#2**	**#3**		**#943**
Cluster#	3	1	3	…	2

RMSE: Root Mean Square Error

We calculated RMSE on movie lens dataset by using Eq-17:

$$RMSE = \sqrt{\frac{\sum (p - t)^2}{n}}, \tag{17}$$

where, p is the predicted value, t is the actual

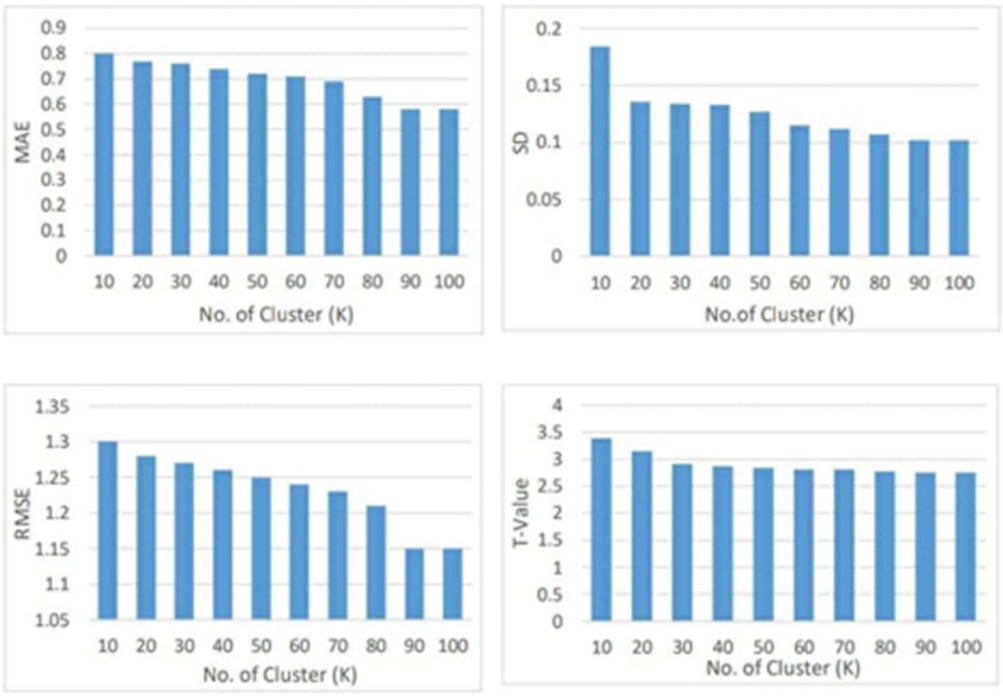

Fig. 4. Performance of proposed algorithm on changing cluster size

value, n is the number of predicted ratings.

The results after calculation of RMSE for different cluster count are represented in Table-4. It is observed that the RMSE value gradually decreases as we increase the number of clusters like other metrics like MAE and SD.

t-value

This t-value basically depends on the values of mean obtained for different clusters and their calculated SD values. We calculate t-value (for significance level of 5%) of the dataset by using Eq 18:

$$t-value=\sum_{i=1}^{k}\sum_{j=1}^{k}\left(\frac{\overline{X}_i-\overline{X}_j}{\sqrt{\frac{(SD_i)^2}{n_i}+\frac{(SD_j)^2}{n_j}}}\right) \quad (18)$$

.

Similar to the other matrices, t-value also decreases for the same reason as mentioned above. Results are shown in Table 4.

The performance of proposed firefly-based recommendation system was also compared with the other popular clustering-based techniques like k-means, PSO (Particle Swarm Optimization), and ACO (Ant Colony Optimization), Bat algorithm, Cuckoo search. All the algorithms were run for 100 iterations. The performance of firefly-based recommendation was slightly better than all other techniques as can be seen in the Table 5 and Figure-4.

6 Conclusion and Future Work

This paper proposed an improved firefly meta-heuristic based clustering approach for recommendation systems. A clustering based recommender system should be able to generate optimal clusters, hence firefly algorithm was

Table 4. Performance based on cluster size

No. of clusters (k)	MAE	SD	RMSE	t-value
10	0.8	0.184	1.3	3.39
20	0.77	0.136	1.28	3.15
30	0.76	0.134	1.27	2.91
40	0.74	0.133	1.26	2.87
50	0.72	0.127	1.25	2.84
60	0.71	0.115	1.24	2.81
70	0.69	0.112	1.23	2.81
80	0.63	0.107	1.21	2.77
90	0.58	0.102	1.15	2.75
100	0.58	0.102	1.15	2.75

Table 5. Performance comparison with other algorithms (k=90)

	k-means	PSO	ACO	Bat	Cuckoo	Firefly
MAE	0.69	0.7	0.7	0.67	0.71	0.58
SD	0.113	0.113	0.112	0.107	0.114	0.102
RMSE	1.23	1.23	1.22	1.19	1.23	1.15
t-value	2.81	2.81	2.81	2.76	2.81	2.75
Precision	0.53	0.52	0.51	0.54	0.52	0.58
Recall	0.43	0.41	0.41	0.44	0.41	0.47

utilized. The original firefly algorithm has been improved by making it faster by moving the less bright firefly towards only the brightest firefly instead of all the brighter fireflies.

For exploration, Levy flight has been used instead of random function. For fast results, the algorithm is parallelized using map-reduce to enable it to be executed in a scalable environment.

The performance of the proposed approach is evaluated using various metrics and the results indicate that the approach generates highly relevant recommendations. In the future work, other swarm optimization methods like whale

optimization, shark smell optimization, etc. can be utilized.

The optimization methods other than swarm optimization like neural networks can also be utilized. There are various ways in which the recommendations from optimal clusters can be generated, these also can be explored.

References

1. **Kidzinski, L. (2011).** Statistical foundations of recommender systems. Master Thesis, Faculty of Mathematics, Informatics and Mechanics, University of Warsaw.
2. **Deshpande, M., Karypis, G. (2004).** Item-based top-N recommendation algorithms. ACM Transactions on Information System, Vol. 22, No. 1, pp. 143–177. DOI: 10.1145/963770.963776.
3. **Li, P., Yamada, S. (2004).** A movie recommender system based on inductive learning. IEEE Conference on Cybernetics and Intelligent Systems, Vol. 1, pp. 318–323. DOI: 10.1109/ICCIS.2004.1460433.
4. **Wei, K., Huang, J., Fu, S. (2007).** A survey of e-commerce recommender systems. International Conference on Service Systems and Service Management, pp. 1–5. DOI: 10.1109/ICSSSM.2007.4280214.
5. **Porcel, C., Herrera-Viedma, E. (2010).** Dealing with incomplete information in a fuzzy linguistic recommender system to disseminate information in university digital libraries. Knowledge-Based Systems, Vol. 23, No. 1, pp. 32–39. DOI: 10.1016/j.knosys.2009.07.007.
6. **Porcel, C., Moreno, J.M., Herrera-Viedma, E. (2009).** A multi-disciplinary recommender system to advice research resources in University Digital Libraries. Expert Systems with Applications, Vol. 36, No. 10, pp. 12520–12528. DOI: 10.1016/j.eswa.2009.04.038.
7. **Goldberg, D., Nichols, D., Oki, B.M., Terry, D. (1992).** Using collaborative filtering to weave an information tapestry. Communications of the ACM, Vol. 35, No. 12, pp. 61–70. DOI: 10.1145/138859.138867.
8. **Deshpande, P.K., Banchhor, C. (2014).** Survey on recommender systems. International Journal of Engineering Research and Development, Vol. 10, No. 6, pp. 49–54.
9. **Felfernig, A., Jeran, M., Ninaus, G., Reinfrank, F., Reiterer, S., Stettinger, M. (2014).** Basic approaches in recommendation systems. Recommendation Systems in Software Engineering, Springer, pp. 15–37. DOI: 10.1007/978-3-642-45135-5_2.
10. **Schafer, J.B., Konstan, J.A., Riedl, J. (2001).** E-commerce recommendation applications. Data Mining and Knowledge Discovery, Vol. 5, pp. 115–153. DOI: 10.1023/A: 1009804230409.
11. **Pazzani, M., Billsus, D. (2007).** Content-based recommendation systems. The Adaptive Web, 325–341.
12. **Burke, R. (2002).** Hybrid recommender systems: Survey and experiments. User Modelig and User-Adapted Interaction, Vol. 12, No. 4, pp. 331–370. DOI: 10.1023/A: 1021240730564.
13. **Smyth, B. (2007).** Case-based recommendation. The Adaptive Web, Lecture Notes in Computer Science, Vol. 4321, pp. 342–376. DOI: 10.1007/978-3-540-72079-9_11.
14. **Hill, W., Stead, L., Rosenstein, M., Furnas, G. (1995).** Recommending and evaluating choices in a virtual community of use. Proceedings of the SIGCHI conference on Human factors in computing systems, pp. 194–201.
15. **Resnick, P., Lacovou, N., Suchak, M., Bergstrom, P., Riedl, J. (1994).** GroupLens: an open architecture for collaborative filtering of netnews. ACM conference on Computer supported cooperative work, pp. 175–186. DOI: 10.1145/192844.192905.
16. **Vucetic, S., Obradovic, Z. (2000).** A regression-based approach for scaling-up personalized recommender systems in e-commerce. Workshop on Web Mining for E-Commerce (WEBKDD'00), pp. 1–9.
17. **Zhao, Z.D, Shang, M.S. (2010).** User-based collaborative-filtering recommendation algorithms on Hadoop. Third International Conference on Knowledge Discovery and

Data Mining, pp. 478–481. DOI: 10.1109/WKDD.2010.54.

18. **Zhu, X., Ye, H., Gong, S. (2009).** A personalized recommendation system combining case-based reasoning and user-based collaborative filtering. Chinese Control and Decision Conference, pp. 4026–4028. DOI: 10.1109/CCDC.2009.5192712.
19. **Sarwar, B., Karypis, G., Konstan, J., Riedl, J. (2001).** Item-based collaborative filtering recommendation algorithms. Proceedings of the 10th international conference on World Wide Web, pp. 285–295.
20. **Sun, Z., Luo, N. (2010).** A new user-based collaborative filtering algorithm combining data-distribution. Information Science and Management Engineering (ISME), Vol. 2, pp. 19–23. DOI: 10.1109/ISME.2010.48.
21. **Murugasamy, K., Murugasamy, K. (2016).** Hybrid clustering using firefly optimization and fuzzy c-means algorithm. Circuits and Systems, Vol. 7, No. 9, p. 2339–2348. DOI: 10.4236/cs.2016.79204.
22. **Shardanand, U., Maes, P. (1995).** Social information filtering: algorithms for automating “word of mouth”. Proceedings of the SIGCHI conference on Human factors in computing systems, pp. 210–217.
23. **Vozalis, E.G., Margaritis, K.G. (2003).** Recommender systems: An experimental comparison of two filtering algorithms. Proceedings of the 9th Panhellenic Conference in Informatics-PCI, pp. 152–166.
24. **Sarwar, B.M. (2001).** Sparsity, scalability, and distribution in recommender systems. Doctoral Dissertation, University of Minnesota.
25. **Yang, X.S. (2010).** Nature-inspired metaheuristic algorithms, 2nd ed. Luniver Press.
26. **Zang, H., Zhang, S., Hapeshi, K. (2010).** A review of nature-inspired algorithms. Journal of Bionic Engineering, Vol. 7, No. 4, pp. S232--S237. DOI: 10.1016/S1672-6529(09)60240-7.
27. **Yang, X.S. (2010).** Firefly algorithm, stochastic test functions and design optimisation. International Journal of Bio-Inspired Computation, Vol. 2, No. 2, pp. 78–84.
28. **Sajwan, M., Acharya, K., Bhargava, S. (2014).** Swarm intelligence based optimization for web usage mining in recommender system. International Journal of Computer Applications Technology and Research, Vol. 3, No. 2, pp. 119–124. DOI: 10.7753/IJCATR0302.1007.
29. **Yang, X.S., He, X. (2013).** Firefly algorithm: recent advances and applications. International Journal of Swarm Intelligence, Vol. 1, No. 1, pp. 36–50.
30. **Fister, I., Fister Jr., I., Yang, X.S., Brest, J. (2013).** A comprehensive review of firefly algorithms. Swarm and Evolutionary Computation, Vol. 13, pp. 34–46. DOI: 10.1016/j.swevo.2013.06.001.
31. **Khan, W.A., Hamadneh, N.N., Tilahun, S.L., Ngnotchouye, J.M. (2016).** A review and comparative study of firefly algorithm and its modified versions. Optimization Algorithms-Methods and Applications, IntechOpen. DOI: 10.5772/62472.
32. **Kumar, M.S., Prabhu, J. (2019).** Hybrid model for movie recommendation system using fireflies and fuzzy c-means. International Journal of Web Portals (IJWP), Vol. 11, No. 2, pp. 1–13. DOI: 10.4018/IJWP.2019070101.
33. **Yang, X.S. (2010).** Firefly algorithm, Lévy flights and global optimization. Research and development in intelligent systems XXVI, Springer, pp. 209–218. DOI: 10.1007/978-1-84882-983-1_15.
34. **Senthilnath, J., Omkar, S.N., Mani, V. (2011).** Clustering using firefly algorithm: performance study. Swarm and Evolutionary Computation, Vol. 1, No. 3, pp. 164–171. DOI: 10.1016/j.swevo.2011.06.003.
35. **Hassanzadeh, T., Meybodi, M.R. (2012).** A new hybrid approach for data clustering using firefly algorithm and K-means. 16th CSI International Symposium on Artificial Intelligence and Signal Processing (AISP), pp. 7–11. DOI: 10.1109/AISP.2012.6313708.
36. **Mohammed, A.J., Yusof, Y., Husni, H. (2015).** Determining Number of Clusters Using Firefly Algorithm with Cluster Merging for Text Clustering. Advances in Visual Informatics (IVIC), Lecture Notes in Computer Science,

Vol. 9429, pp. 14–24. DOI: 10.1007/978-3-319-25939-0_2.

Article received on 15/06/2021; accepted on 17/11/2021.
Corresponding author is Bharti Sharma.