Neural Computing and Applications (2021) 33:7145–7164
https://doi.org/10.1007/s00521-020-05482-6

ORIGINAL ARTICLE

Hybrid crow search and uniform crossover algorithm-based clustering for top-*N* recommendation system

Walaa H. El-Ashmawi[1,2] · Ahmed F. Ali[1] · Adam Slowik[3]

Received: 3 July 2019 / Accepted: 27 October 2020 / Published online: 11 November 2020
© The Author(s) 2020

Abstract
Recommender systems (RSs) have gained immense popularity due to their capability of dealing with a huge amount of information available in various domains. They are considered to be information filtering systems that make predictions or recommendations to users based on their interests. One of the most common recommender system techniques is user-based collaborative filtering. In this paper, we follow this technique by proposing a new algorithm which is called hybrid crow search and uniform crossover algorithm (HCSUC) to find a set of feasible clusters of similar users to enhance the recommendation process. Invoking the genetic uniform crossover operator in the standard crow search algorithm can increase the diversity of the search and help the algorithm to escape from trapping in local minima. The top-*N* recommendations are presented for the corresponding user according to the most feasible cluster's members. The performance of the HCSUC algorithm is evaluated using the Jester dataset. A set of experiments have been conducted to validate the solution quality and accuracy of the HCSUC algorithm against the standard particle swarm optimization (PSO), African buffalo optimization (ABO), and the crow search algorithm (CSA). In addition, the proposed algorithm and the other meta-heuristic algorithms are compared against the collaborative filtering recommendation technique (CF). The results indicate that the HCSUC algorithm has obtained superior results in terms of mean absolute error, root means square errors and in minimization of the objective function.

Keywords Crow search algorithm · Uniform crossover · Recommendation system · User-based collaborative filtering

1 Introduction

Currently, there are more than four billion internet users all over the world who have access to more than one billion websites [1]. Due to the huge amount of information available, finding relevant information on the internet is an important issue. Among solutions that cope with this issue is a recommender system (RS). RS can be considered as an information filtering tool or support in the decision-making process that recommends items to users or filters and sorts information. Currently, recommendation algorithms have been widely used in Spotify, Facebook, TripAdvisor, and many others.

The RSs are categorized into three main filtering algorithms [2]. Collaborative filtering (CF) [3–6] is the simplest and most efficient algorithm of these. It has been adopted by many real-world systems such as Netflix and Amazon [7]. The CF can be classified into item and user-based CF based on the used prediction technique. The user-based CF finds users with similar preferences and predictions based on similar user interests, whereas the item-based recommends the most identical items to the user [8]. Another filtering technique is content-based filtering (CBF) [9, 10] which makes a recommendation based on the personal

✉ Adam Slowik
aslowik@ie.tu.koszalin.pl

Walaa H. El-Ashmawi
w.hashmawi@ci.suez.edu.eg;
walaa.hassan@miuegypt.edu.eg

Ahmed F. Ali
ahmed_fouad@ci.suez.edu.eg

[1] Department of Computer Science, Faculty of Computers and Informatics, Suez Canal University, Ismailia, Egypt

[2] Faculty of Computer Science, Misr International University, Cairo, Egypt

[3] Department of Electronics and Computer Science, Koszalin University of Technology, Koszalin, Poland

former choice from earlier. The CBF depends on the item description and user's profile [11]. The common characteristics between users like nationality, age, gender, etc., play a vital role in the demographic filtering technique [12]. The authors in [13] have discussed the other two filtering techniques (hybrid filtering and knowledge-based filtering). Hybrid filtering combines more than one existing filtering technique [14, 15] to overcome the limitation issues of RS, while knowledge-base filtering uses a relationship and inference about the user needs and preferences in the recommendation.

In this paper, we follow the user-based collaborative filtering recommendation model. In order to enhance the performance of CF, different clustering techniques are applied for better recommendations based on users' preferences [16, 17]. The aim of clustering is to collect identical users into a single cluster based on identical users' item ranking. A users' cluster is submitted to an active user who has a similar preference to the users in that cluster. Although many clustering techniques are obtainable in user-based CF such as *k*-means, fuzzy *c*-means, and others, the quality solution of the recommendation system is still an open issue.

Meta-heuristic algorithms have been proven effective for various real-world applications. The work in this paper has tried to utilize one of the recent meta-heuristic algorithms for clustering technique in user-based collaborative filtering.

The crow search algorithm has a good ability to balance between the exploration and the exploitation processes, and it has only two parameters, the flight length fL and the awareness probability AP, which makes it easy to implement. To the best of our knowledge, there are a few works for implementing the crow search algorithm to solve the top-*N* recommendation system.

The proposed algorithm is a hybrid of crow search algorithm (CSA) with a uniform crossover. The CSA is a population-based meta-heuristic algorithm that produces promising solutions for global optimum to optimization problems [18]. The proposed algorithm is called hybrid crow search and uniform crossover algorithm (HCSUC). Invoking the uniform crossover in the HCSUC algorithm can increase the diversity of search which can help the proposed algorithm to escape from trapping in local minima. Such a hybrid algorithm can achieve an improved clustering solution for a practical clustering-based recommender system.

The structure of this paper is organized as follows: Existing meta-heuristic algorithms for the recommendation system are reviewed in Sect. 2. Section 3 gives the mathematical formulation of the top-*N* recommendation system. The basic structure of the crow search algorithm is discussed in Sect. 4. The proposed HCSUC algorithm is illustrated in Sect. 5. The numerical experimental results and evaluation metrics are shown in Sect. 6. Section 7 summarizes the main points in this research work and gives an outlook for future work.

2 Literature review

The CF-based recommender systems are based on the collaboration of one user with other users. Some key decisions for evaluating the CF recommender systems are discussed in [19] such as user's task, dataset chosen, and evaluation of a recommender system's accuracy.

One of the popular approaches for making a recommendation is the use of clustering. The formed cluster must have a minimum inter and maximum intra similarities [13]. The better the clustering, the better the recommendation. The clustering techniques are very often used in the CF-based recommender systems to increase their performance. As an example, we mention the following papers.

Feng et al. [20] propose improving performances of top-*N* recommendation with the co-clustering method. The authors present the recommendation method based on collaborative filtering which is named UICDR (User-Item Community Detection based Recommendation). The main idea is related to the construction of a bipartite network with user-item interaction data. The users and the items are partitioned into several subgroups. When we have clusters with tightly linked users and items, the standard collaborative filtering models can be used for each cluster. The presented results show that the proposed method can significantly improve the performance of top-*N* recommendations of several standard collaborative filtering methods.

In an article [21] by Wasid et al., the clustering approach was used to incorporate multi-criteria ratings into traditional recommender systems effectively. Also, the intra-cluster user similarities were computed using a Mahalanobis distance method to make more accurate recommendations. The results obtained using the proposed method were compared with the results obtained using the traditional collaborative filtering method. The presented results show that the proposed approach can improve the accuracy of the recommendations.

The authors in [22] proposed an incremental CF system, based on a weighted clustering approach to provide a high quality of recommendations. They focused on the use of spherical *k*-means. The complexity of the presented approach does not depend on the number of users and items (in contrast to existing incremental methods). Therefore the proposed collaborative filtering system is more suitable for dynamic settings, involving huge databases, in which available information evolves rapidly. The experiments were made using several real-world datasets, and confirm

the efficiency and the effectiveness of the presented method in terms of scalability and recommendation quality.

In addition, in many research papers, it was noticed that the standard clustering methods (for example *k*-means clustering) can fall into local optima. Therefore, the results of recommendations can be optimized by applying clustering techniques together with nature-inspired global optimization algorithms. As an example, we mention the following articles.

Wang et al. [23] proposed a hybrid movie recommender system that utilized a genetic algorithm (GA) and *k*-means clustering to partition transformed user space. The presented approach employs principal component analysis (PCA) to dense the movie population space which could reduce the computation complexity. The presented results (which were performed on the Movielens dataset) indicate that the proposed method can provide high performance in terms of accuracy, and can generate more reliable and personalized movie recommendations than other existing methods.

The authors in [24] applied the PSO to optimize and enhance the result of a recommender system. In the proposed approach, they used a *k*-means algorithm which provides initial parameters to particle swarm optimization (PSO). In the next step, PSO provides initial seed and optimizes fuzzy *c*-means (FCM), for soft clustering of data items (users), instead of strict clustering behavior in *k*-means. The experiments were performed on the Movielens dataset. The results obtained using the proposed methods deliver more predictable and personalized recommendations.

Bedi et al. [25] proposed a recommender system based on the behavior of ants for generating a top-*N* recommendation. This method works in two phases. In the first phase, opinions from users collected in the form of the user-item rating matrix are clustered with the use of an ant-based clustering algorithm into a predetermined number of clusters and the obtained results are next stored in the database for future recommendations. In the second phase, the recommendation is generated for the active user. The efficiency of the proposed approach was evaluated using the Jester dataset and compared with the traditional collaborative filtering-based recommender system. The results obtained using the proposed method show a good performance in relation to the standard recommender system.

A grey wolf optimization algorithm and fuzzy *c*-means (FCM) clustering for a movie-based collaborative recommender system are presented in the paper [26] by Katarya et al. In the first step, the grey wolf optimizer algorithm was applied on the Movielens dataset to obtain the initial clusters. In the second step, the FCM is used to classify the users in the dataset by similarity of user ratings. The proposed collaborative recommender system was evaluated with the use of the Movielens dataset. The experiment results obtained by the presented recommender system demonstrated that efficiency is enhanced, and also offered better recommendations in comparison with other methods.

In the paper [27] by Senbagaraman et al., a recommender system which uses the collaborative filtering algorithm, *k*-means clustering, and cuttlefish optimization algorithm is presented. In the proposed approach, the *k*-means algorithm is used to group users of similar taste and then the cuttlefish algorithm is applied to propose optimal recommendations from the result of the *k*-means algorithm. The quality of the proposed system was evaluated using the Movielens dataset. The results obtained by the presented approach show that the developed system works more efficiently than the other recommender systems has high efficiency in the cross-validations, and is capable of recommending highly reliable outcomes.

A cuckoo search algorithm based on *k*-means for recommender systems is provided in [28] by Katarya et al. Hybridization of clustering technique and optimization approach is used to improve the movie prediction accuracy. The limitations of typical content-based and collaborative recommender systems have been overcome due to the proposed method. The approach of *k*-means clustering and cuckoo search algorithm were applied to the Movielens dataset. The results obtained using the proposed technique were observed for evaluation metrics such as mean absolute error, standard deviation, root mean square error, and *t*-value. In terms of accuracy and precision, the experiment results demonstrate that the proposed approach is capable of providing more reliable movie recommendations as compared to the existing cluster-based collaborative filtering methods.

In the article [29] by Katharya, a hybrid recommender system is proposed. In the presented technique, the *k*-means clustering algorithm is utilized together with an artificial bee colony optimization algorithm. In the first phase of the proposed method, the *k*-means clustering algorithm is applied to the Movielens dataset for the clustering of users into different clusters. In the second phase, the artificial bee colony optimization algorithm is employed to the results obtained by the *k*-means procedure for further optimization of these clustering results. The results obtained using the proposed approach show immense achievement regarding scalability and performance, and deliver accurate personalized movie recommendations by reducing the cold start problem.

Logesh et al. [30] propose a hybrid quantum-induced swarm intelligence clustering for an urban trip recommendation. The authors present a novel user clustering approach based on quantum-behaved particle swarm optimization for the collaborative filtering-based recommender

system. Due to utilizing clustering mechanisms with collaborative filtering for grouping similar users as clusters, the efficiency of the generated recommendation was improved. The proposed approach has been evaluated on real-world large-scale datasets. The results obtained using the proposed method illustrate the advantageous performance of the presented method over other approaches.

In the paper [31], the authors present a hybrid collaborative filtering-based recommender system with adopted fuzzy *c*-mean clustering method and an artificial algae algorithm. The authors have used an advanced multilevel Pearson correlation coefficient to find the similarity between two users. The proposed system was evaluated using four real datasets such as Movielens 100k, Movielens 1M, Jester, and Epinion. The presented approach delivered superior results for almost all four datasets.

An evolutionary clustering algorithm based on temporal features for dynamic recommender systems is presented in the paper [32] by Rana et al. The clustering algorithm makes clusters of similar users and evolves them, depicting accurate and relevant user performances over time. The results obtained using the proposed approach were compared with the results obtained using standard recommendation algorithms. The presented method shows an improvement in terms of quality of recommendations and computational time.

In [33] by Chen et al., the evolutionary clustering for rating prediction based on user collaborative filtering is presented. The main objective of the presented method is to gather users with similar interests into the same cluster and to help users find items that fit their tastes best. After clustering operation, the collaborative filtering is adopted in each cluster. The target rating is calculated according to user-based collaborative filtering in its cluster. The experiments show the high efficiency of the proposed approach.

Alam et al. [34] present a hierarchical PSO clustering-based recommender system. In the paper, the method for patterns generation from user activities is presented. These patterns are generated efficiently for the recommender system using proposed hierarchical particle swarm optimization-based clustering (HPSO-clustering). The HPSO-clustering is a clustering approach based on the PSO algorithm which combines both the properties of hierarchical and partitional clustering. The users' sessions are grouped into different clusters. The recommendation for an active user is generated from these clusters. The authors said that in some clusters the achieved precision was equal to 100% when the top-5 ranked recommendations were selected.

In the paper [35] by Marung et al., methods including the visual-clustering recommendation method, the hybrid between the visual-clustering recommendation and user-based methods, and the hybrid between the visual-clustering recommendation and item-based methods are presented. The user-item clustering is based on a genetic algorithm. The results obtained using the proposed methods were compared with the results obtained using traditional methods. The results showed that the proposed recommendation methods were more efficient than traditional methods.

Most of the above research works have applied to the Movielens dataset while in this paper, the proposed HCSUC algorithm is applied to the Jester dataset to demonstrate the effectiveness of the results obtained.

3 Mathematical formulation of the top-*N* recommendation system

Formally, we have a set of n users; $U = \{u_1, u_2, \ldots, u_n\}$ and a set of m items; $I = \{i_1, i_2, \ldots, i_m\}$. A user u_j (i.e., $1 \leq j \leq n$) has a rating value on item i_k (i.e., $1 \leq k \leq m$). Therefore, the user-item rating matrix R can be represented as $n \times m$ matrix (i.e., $R = (r_{jk})$); r_{jk} represents the rating of user u_j on item i_k. The user-item rating values can be normalized in the scale 0 to 1, $r_{jk} \in [0, 1]$ where 0 indicates the item is not rated by a user, and 1 indicates the highest rating value of the item by the corresponding user. The users are partitioned to a pre-specified number of clusters $C = \{c_1, c_2, \ldots, c_l\}$. A cluster c_t (i.e., $1 \leq t \leq l$) contains a subset of users; $uc_t \subset U$. Therefore, the aim is to minimize the distance between each user and the center of belonging cluster as seen in Eq. (1).

$$\begin{aligned} &\text{Min} f(a,b) = \sum_{l=1}^{C}\sum_{j=1}^{U}\sum_{k=1}^{I} a_{jl}\|r_{jk} - b_{lk}\|^2 \\ &\text{Subject to}: \sum_{l=1}^{C} a_{jl} = 1 \quad \forall j = 1, \ldots, U \\ &\qquad\qquad \sum_{j=1}^{U} a_{jl} \geq 1 \quad \forall l = 1, \ldots, C \end{aligned} \tag{1}$$

where a is an assignment binary matrix of size $U \times C$; $a_{jl} = 1$ if user j is assigned to cluster l and 0 otherwise. Parameter b is the cluster center matrix of size $C \times I$; b_{lk} is the average of rating value of an attribute k in the cluster l. b_{lk} can be computed according to Eq. (2).

$$b_{lk} = \frac{\sum_{j=1}^{U} a_{jl} \times r_{jk}}{\sum_{j=1}^{U} a_{jl}} \tag{2}$$

The recommendation of items for an active user u_{act} from the set of unrated items I_a (i.e., $I_a \subset I$) is a primary goal of the recommendation system. The top-N recommendation [8] is based upon predicting rates for all unrated items in I_a

and then recommends the N items with the highest predicted ratings.

4 Crow search algorithm (CSA)

Crow search algorithm is a nature-inspired population-based algorithm, proposed by Askarzadeh [18]. In this section, we highlight the social behavior of the crows and we give a description of the crow search algorithm and how it works.

4.1 Social behavior and inspiration

Crows are intelligent birds that have a large brain relative to their body size. They live in a group (flock). They put their food in hiding places, and they can memorize these places and retrieve the hidden food even several months later. Crows carry out theft by following other crows to discover their food-hiding places and steal the hidden food. If a crow feels that another one is following it, it moves to another place far away from the food-hiding place to fool a thief. In the following subsection, we show how the crow search algorithm mimics the crow's social behavior.

4.2 Crow search algorithm implementation

The crow search algorithm (CSA) mimics the social behavior of crow birds as shown in the following items.

- *Initialization* The crows in the group represent the search agent (solution) in the population; the search space represents the environment. The population contains N solutions. The position of each crow i at iteration t is represented by a vector $\mathbf{x_i^t}$, where $\mathbf{x_i^t} = [x_{i1}^t, x_{i2}^t, \ldots, x_{id}^t]$ and d is a problem dimension. The whole population of size N with dimension d at iteration t can be represented as follows.

$$X = \begin{bmatrix} x_{11}^t & x_{12}^t & \ldots & x_{1d}^t \\ \vdots & \vdots & \vdots & \vdots \\ x_{N1}^t & x_{N2}^t & \ldots & x_{Nd}^t \end{bmatrix}$$

- *Memory initialization* The memory is used to store the best position for each solution at each iteration, and it is updated if the new solution's position is better than the position of the old one. The memory M of all crows (population) at iteration t for dimension d is initialized as follows.

$$M = \begin{bmatrix} m_{11}^t & m_{12}^t & \ldots & m_{1d}^t \\ m_{21}^t & m_{22}^t & \ldots & m_{2d}^t \\ \vdots & \vdots & \vdots & \vdots \\ m_{N1}^t & m_{N2}^t & \ldots & m_{Nd}^t \end{bmatrix}$$

- *Solution evaluation* Each solution in the population is evaluated by calculating its fitness by using the fitness function (objective function), $f(\mathbf{x_i^t})$.
- *Position update* The crow (solution) i can update its position based on the position of crow j (a randomly selected solution in the population) to discover its food's hiding place. During the movement of crow i toward crow j, two states can happen.

  State 1: If crow j does not watch crow i when it follows it, crow i will discover the food's hiding place of crow j and the crow i will update its position as follows.

$$\mathbf{x_i^{t+1}} = \mathbf{x_i^t} + r_i \times \mathrm{fl}_i^t \times (\mathbf{m_j^t} - \mathbf{x_i^t}) \tag{3}$$

  where r_i is a random number in the interval [0, 1], fl_i^t is the flight length, and $\mathbf{m_j^t}$ is the memory of crow j. The value of fl is responsible for the exploration and exploitation processes; if $\mathrm{fl} < 1$, it means the crow i will move toward crow j which leads to local search (exploitation) while if $\mathrm{fl} > 1$ it means the crow i will move far from crow j which leads to global search (exploration).

  State 2: The other state happens when the crow j knows that crow i is watching it and it has discovered its food's hiding place. In this case, the crow j moves randomly to fool crow i.

  The two states are based on the awareness probability AP_i^t of each crow in the population as follows.

$$\mathbf{x_i^{t+1}} = \begin{cases} \mathbf{x_i^t} + r_i \times \mathrm{fl}_i^t \times (\mathbf{m_j^t} - \mathbf{x_i^t}) & \text{if } r_j \geq \mathrm{AP}_i^t \\ \text{Move to a random position,} & \text{otherwise} \end{cases} \tag{4}$$

  where AP_i^t is the awareness probability. CSA balancing between exploration and exploitation processes according to the value of AP_i^t. Increasing the value of AP_i^t leads to global search while decreasing the value of AP_i^t leads to local search.
- *Memory update* Each crow (solution) updates its memory according to its fitness value. If the objective function value of the new crow's position is better than the current memory's value, then it updates its memory; otherwise, the memory will not be changed. The process of updating memory is shown as follows.

$$\mathbf{m_i^{t+1}} = \begin{cases} \mathbf{x_i^{t+1}} & \text{if } f(\mathbf{x_i^{t+1}}) \text{ is better than } f(\mathbf{m_i^t}) \\ \mathbf{m_i^t}, & \text{otherwise} \end{cases} \quad (5)$$

- *Solution feasibility* If the position of the new solution is feasible, then it is accepted; otherwise, the position of the new solution is not updated.
- *Stopping criteria* The overall processes are repeated until the termination criteria are satisfied.
- *Producing an optimal solution* Each solution in the population is evaluated by calculating its fitness function, and the overall (global) best solution is produced.

The main steps of the CSA are shown in the flowchart in Fig. 1.

5 The proposed HCSUC algorithm

In this section, we describe the main process of the proposed HCSUC algorithm. The HCSUC algorithm has two main phases. The first phase is the cluster forming phase, while the second phase is the top-N recommendation phase. We invoke the genetic uniform crossover in the proposed HCSUS algorithm to avoid trapping in local minima. In the following subsections, we present the genetic uniform crossover and how it works and we highlight in detail the main steps of the proposed HCSUC algorithm.

5.1 Hybrid crow search and uniform crossover algorithm (HCSUC)

The main structure of the HCSUC algorithms is shown in Algorithm 1 and its steps are represented as follows.

HCSUC contains two main phases. The first phase is starting to form clusters by grouping the number of users to a pre-specified number of feasible clusters based on the hybrid crow algorithm. Clustering helps in locating similar users' profiles. Once the clusters are obtained, the centroid (head) for each cluster is computed based on the similarity value between the cluster's members. The main steps of forming clusters are shown in the following stage.

- *Parameters initialization* In the beginning, we set the initial values for awareness probability AP parameters, the size of population N, and the maximum iterations number $\max_{itr}$.
- *Iteration counter initialization* We set the initial iteration's counter, where $t = 0$.
- *Population initialization* We generate the initial population randomly; each solution in the population is a vector where $\mathbf{x_i^t} \in [L, U]$ randomly, where L and U are the lower and upper domain of the search space, $i = 1, \ldots, N$, which indicates the cluster number (cluster_id) which is corresponding to a specific user. Figure 2 shows an example of solution representation. According to Fig. 2, the crow solution consists of 6 users (i.e., $U = 6$) and 3 clusters (i.e., $C = 3$). The first element indicates the first user (i.e., u_1) belongs to cluster number 2 (i.e., c_2), the second user belongs to class 1, and so on.
- *Solution evaluation* Each solution in the population is evaluated by calculating its fitness function as shown in Eq. (1).
- *Memory initialization.* The initial memory for each solution in the population is a vector $\mathbf{m_i^t}$, and it is assigned after calculating its fitness function.
- *Solution update* The main loop of the algorithm is repeated until the termination criteria are satisfied. In order to avoid trapping in local minima, we mate the best solution $\mathbf{m_j^t}$ with the current solution $\mathbf{x_i^t}$ by using the uniform crossover [36, 37]. An illustrative example of applying uniform crossover in HCSUS can be shown in Fig. 3. In Fig. 3, a mask allele "0" means, to the offspring, that the bit is copied from the second parent otherwise from the first parent. For the second offspring, the reversal of the mask is required [38]. The two offerings (i.e., $\tilde{X}$ and $\tilde{Y}$) are estimated, and the superior one is chosen as $\mathbf{m^t_{cross}}$. Therefore, each solution in the population is updated according to the value of the awareness probability as shown in Eq. (6).

$$\mathbf{x_i^{t+1}} = \begin{cases} \mathbf{x_i^t} \oplus r_i \times (\mathbf{m^t_{cross}} \ominus \mathbf{x_i^t}) & \text{if } r_j \geq \mathrm{AP}_i^t \\ \text{Move to a random position,} & \text{otherwise} \end{cases} \quad (6)$$

where r_i , r_j, and AP_i^t are random numbers in the range 0 and 1. $(\mathbf{m^t_{cross}} \ominus \mathbf{x_i^t})$ returns how far the best-crossed crow (solution) from the current source food in the form of a set of operators O_s. Figure 4 illustrates how to obtain the difference between two crows X and Y.

The result of $\ominus$ is a set of operators O_s which represents a vector $(\mathrm{user}_{id}, \mathrm{cluster}_{id})$. According to Fig. 4, as a result of $X \ominus Y$, $O_1(1, 2)$ means user 1 is assigned to cluster 2, $O_2(2, 2)$ means user 2 is assigned to cluster 2, and so on.

The operator $\otimes$ indicates the probability of r_i that all operators are selected from the resulting set of operators $(\mathbf{m^t_{cross}} \ominus \mathbf{x_i^t})$ while the operator $\oplus$ is used to update the position of the crow's information with the sequence of O_s.

- *Solution feasibility* The solution's new position is accepted if its value is feasible; otherwise, it is rejected and the current solution is kept.
- *Memory update* In Eq. (5), the memory of each solution is updated based on its fitness value.

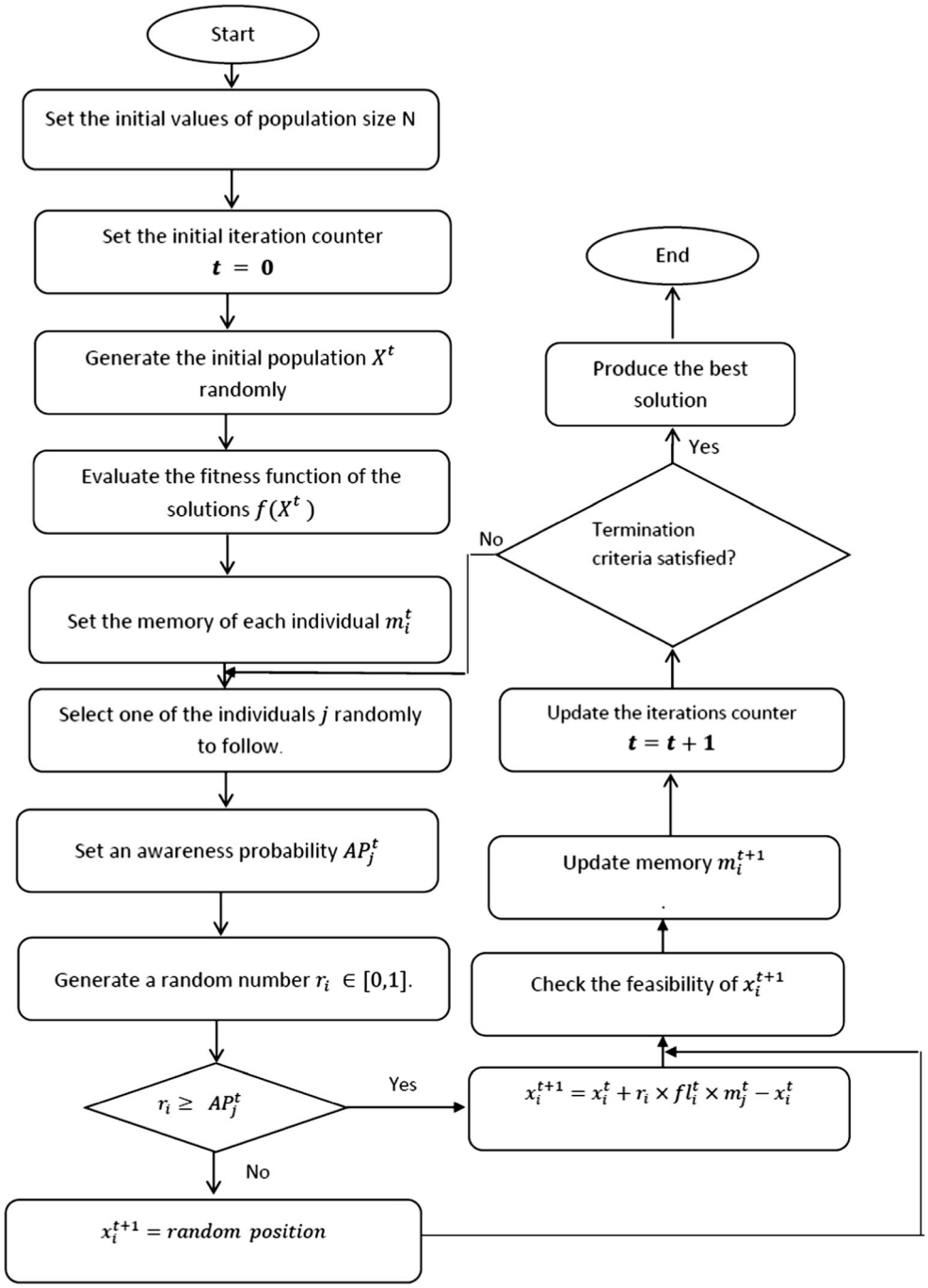


Fig. 1 The main process of the crow search algorithm

User_id	1	2	3	4	5	6
Cluster_id	2	1	1	2	3	2

Fig. 2 Solution representation

- *Best solution produced* Once the termination criteria are satisfied, the overall best solution is produced (i.e., users assigned to a set of clusters) and the centroid (head) of each cluster is based on the best solution. The centroid for each cluster is computed based on the Pearson correlation coefficient [39] as in Eq. (7).

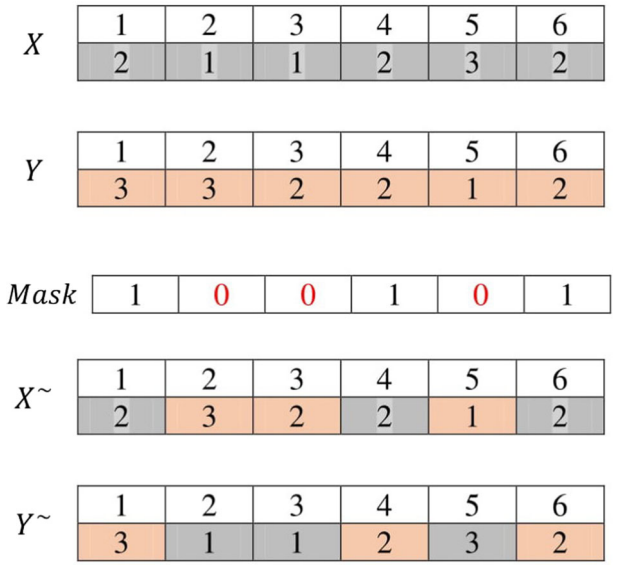


Fig. 3 Applying crossover in HCSUC algorithm

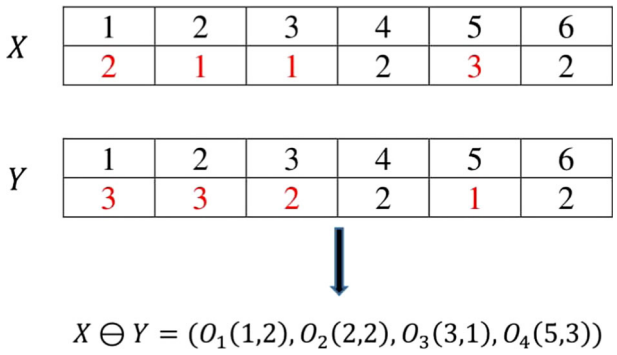


Fig. 4 An illustrative example on $X \ominus Y$

$$\forall c_t \in C, \mathrm{sim}(u,v) = \frac{\sum_{i=1}^{I_{uv}} (r_{ui} - \bar{r_u}) - (r_{vi} - \bar{r_v})}{\sqrt{\sum_{i=1}^{I_{uv}} (r_{ui} - \bar{r_u})^2 - (r_{vi} - \bar{r_v})^2}} \quad u, v \in uc_t \tag{7}$$

where $\mathrm{sim}(u,v)$ is the similarity between user u and v in cluster c, I_{uv} is the set of common items ranked by user u and v, r_{ui} and r_{vi} are the rated values of item i by user u and v, $\bar{r_u}$ and $(\bar{r_v})$ are the average rated values by user u and v, respectively.

The user with maximum similarity is chosen to be the centroid of a specific class and denoted as u_{cent} as shown in Eq. (8).

$$u_{\mathrm{cent}} = \max_{u,v \in uc_t} \mathrm{sim}(u,v) \tag{8}$$

- *Top-N recommendation for an active user* The second phase is the top-N recommendation for an active user. The recommendation process for an active user u_{act} is achieved based on the similarity metrics between an active user and the head (center) of the best cluster. The best-recommended cluster is assigned by evaluating the similarity distance between an active user and the head (center) of each cluster according to Eqs. (9) and (10) to find the best one with similar preferences.

$$\forall c_t \in C, \mathrm{sim}D(u_{\mathrm{act}}, u_{\mathrm{cent}}) = \frac{1}{\mathrm{dist}(u_{\mathrm{act}}, u_{\mathrm{cent}})} u_{\mathrm{cent}} \in c_t \tag{9}$$

$$\mathrm{dist}(u_{\mathrm{act}}, u_{\mathrm{cent}}) = \sqrt{\sum_{k=1}^{I} |r_{u_{\mathrm{act}},k} - r_{u_{\mathrm{cent}},k}|^2} \tag{10}$$

where $r_{u_{\mathrm{act}},k}$ is the rating of active user u_{act} on item k and $r_{u_{\mathrm{cent}},k}$ is the rating of centroid user u_{cent} on item k. The cluster centroid with maximum similarity is chosen to be the best cluster to the active user and denoted as c_t^* as shown in Eq. (11).

$$c_t^* = \max_{u_{\mathrm{cent}} \in uc_t} \mathrm{sim}D(u_{\mathrm{act}}, u_{\mathrm{cent}}) \forall c_t \in C \tag{11}$$

- *Predicted rating of items* The recommendation is based on the predicted rating score of unrated items to a recommender user according to Eq. (12).

$$r_{u_{\mathrm{act}},i_k} = \overline{r_{u_{\mathrm{act}}}} + \frac{\sum_{\tilde{u} \in c_t^*} (\mathrm{sim}(u_{\mathrm{act}}, \tilde{u}) * (r_{\tilde{u}i} - \overline{r_{\tilde{u}}}))}{\sum_{\tilde{u} \in c_t^*} \mathrm{sim}(u_{\mathrm{act}}, \tilde{u})} \forall i_k \in I_a \tag{12}$$

where r_{u_{act},i_k} is the predicted score of active user u_{act} on unrated item i_k and $\mathrm{sim}(u_{\mathrm{act}}, \tilde{u})$ is the similarity between the active user and best cluster's members c_t^* for an active user. The set of the predicted score for all unrated items is denoted as $p(I_a)$ for the recommended user (i.e., $r_{u_{\mathrm{act}},i_k} \in p(I_a)$).

- *Top-N recommended items* Once the predicted score rating is computed, the N items with the highest predicted score are recommended to the active user as defined in Eq. (13).

$$\mathrm{top}{-}N = \mathrm{argmax}_{I^* \subset I_a, |I^*| = N} P(I_a) \tag{13}$$

where top-N is a set containing the top-N recommended items; $|I^*|$ is the cardinality of recommended items which equals N, I_a is the set of unrated items and $p(I_a)$ is the predicted rating score for I_a according to Eq. (12).

Algorithm 1 HCSUC algorithm

Set the initial values for the awareness probability AP paraments, the size of population N and maximum iterations number max_{itr}.
Initialize the iteration numbers $t = 0$.
Generate the initial population X, where $\mathbf{x_i^t} \in [L, U]$ randomly, $i = 1, \ldots, N$.
Evaluate the fitness function of each solution $\mathbf{x_i^t}$.
Set the memory of each solution $\mathbf{m_i^t}$.
repeat
  for i=1 to N **do**
    Select one of the solutions j randomly to follow.
    Set an awareness probability AP_j^t.
    Generate a random number $r_i \in [0, 1]$.
    if $(r_i \geq AP_j^t)$ **then**
      $\mathbf{x_i^{t+1}} = \mathbf{x_i^t} + r_i \times fl_i^t \times (\mathbf{m_j^t} - \mathbf{x_i^t})$
    else
      $\mathbf{x_i^{t+1}}$= a random position.
    end if
  end for
  Check the feasibility of the new solutions $\mathbf{x_i^{t+1}}$
  Evaluate the fitness function of the new solutions $\mathbf{x_i^{t+1}}$.
  if $(\mathbf{f(x_i^{t+1})} \leq \mathbf{m_i^{t+1}})$ **then**
    $\mathbf{m_i^{t+1}} = \mathbf{x_i^{t+1}}$ {**Minimization problem**}
  else
    $\mathbf{m_i^{t+1}} = \mathbf{m_i^t}$
  end if{**Updating memory**}
  Set $t = t + 1$.
until Termination criteria satisfied.
Produce the best solution.
Chose the user with maximum similarity to be the centroid of a specific class using Equation 11
Predict rating of items as shown in Equation 12
Recommend top-N items using Equation 13

6 Experiment results

In order to examine the effectiveness of the HCSUC algorithm, a set of experiments were conducted on the Jester dataset [40], which are publicly available [41]. The Jester dataset has over four million ranked out of a hundred jokes from 73,421 users in the form of [userid], [itemid], [rating]. All the rating values are in the scale -10 to 10. All the experiments are implemented by using Eclipse Java Neon V-1.8 running on Intel(R) Core i7 CPU-2.80 GHz with 8 GB RAM and operating system (Windows 10). In addition, the performance analysis of the proposed algorithm is demonstrated against the most popular recommendation technique, collaborative filtering (CF), in addition to other popular meta-heuristic algorithms such as particle swarm optimization algorithm (PSO) [42, 44], African buffalo optimization (ABO) [43] and standard crow search algorithm (CSA) [18].

Each experiment is repeated five times and the average results are taken over a different number of iterations ranging from 10 to 25 iterations and the population size is set to 10 individuals, who are randomly generated for each algorithm. The parameters of the standard CSA, PSO, and ABO are the standard parameters for each algorithm.

The HCSUC algorithm against the others is tested on three experiments with the various number of users and joke items as shown in Table 1. Each experiment is conducted by taking a random number of users as active users (i.e., the ones for whom recommendations are generated). The number of users in each experiment is categorized into a different number of clusters and a different number of top-N jokes are recommended to the active users. For active users, some percentage θ of items are chosen randomly to be hidden ($\theta = 40\%$) for better evaluation.

The performance of the competitive algorithms is evaluated according to two important metrics as described below [45].

- *Mean Absolute Error (MAE)* is one of the most important evaluation metrics for RS. The minimization of MAE leads to a better recommendation.

$$\mathrm{MAE}_{u_{\mathrm{act}}} = \frac{\sum_{i=1}^{n} |p_i - a_i|}{n} \tag{14}$$

  where p_i and a_i are the predicted and actual rating of active user u_{act}, respectively, n is the total number of ratings.
- *Root mean square error (RMSE)* is a quadratic mean of the differences between predicted values and actual values.

Table 1 Experiment's parameters

Exp. No.	No. of users	No. of active users U_{act}	No. of jokes I	No. of clusters C	Top-N jokes N
Exp. 1	30	5	10	3, 5	4
Exp. 2	90	10	30	3, 5, 7, 9	4, 8, 12
Exp. 3	120	15	60	3, 5	4, 8, 12

$$\mathrm{RMSE}_{u_{\mathrm{act}}} = \sqrt{\frac{\sum_{i=1}^{n} |p_i - a_i|^2}{n}} \tag{15}$$

6.1 Comparison between HCSUC and other meta-heuristic algorithms

The proposed HCSUC is compared with three meta-heuristics algorithms, CSA, PSO, and ABO. The minimum (Min.), maximum (Max.), and the average (Avg.) of the fitness function as shown in Eq. (1) for each experiment, respectively, are reported in Tables 2, 3, 4, and 5. The best value for all algorithms is reported in bold text. The first test is applied at top-$N = 4$, $C = 3, 5$, $U = 30$, and $I = 10$ for the four algorithms, and the results are reported in Table 2

The values in Table 2 show that the proposed HCSUC algorithm obtains better average results than the other three algorithms.

The convergence curves of the three comparative algorithms for the average results are shown in Fig. 5 for 30 users $U = 30$ grouped to three clusters $C = 3$ by plotting the number of iterations versus fitness function values. The solid line represents the results of the proposed HCSUC algorithm, while the other dotted lines represent the results of the CSA and PSO algorithms.

In Fig. 5, the HCSUC algorithm has proven its efficiency in minimization up to 8% better than PSO, ABO, and CSA during iterations for 30 users which are grouped into three clusters.

Table 2 Experimental results of ($U = 30$ and $I = 10$)

Top-N	C	PSO			ABO		
		Min	Max	Avg	Min	Max	Avg
4	3	71.341	75.311	72.645	70.332	72.412	71.461
	5	63.919	68.414	66.257	63.919	69.478	66.708
Top-N	C	CSA			HCSUC		
		Min	Max	Avg	Min	Max	Avg
4	3	68.816	72.472	71.036	66.169	71.819	**69.379**
	5	63.891	66.521	65.074	62.420	64.474	**63.501**

With the increasing number of clusters as a result in Fig. 6 for 30 users $U = 30$ grouped to five clusters ($C = 5$), although CSA has obtained better results than PSO and ABO, the proposed HCSA still had results superior to them, with results better than PSO within the range 1–5%, better than ABO up to 6% and better than CSA up to 2%.

Moreover, the MAE and RMSE are computed for the number of active users over the number of runs as shown in Figs. 7 and 8, respectively.

For five active users with the top four recommended items, the MAE for the proposed algorithm is minimized and better than PSO by 10%, better than ABO by 8%, and better than CSA by 6% in the case of 3 clusters. With the increasing number of clusters, the MAE of HCSUC is better than PSO by 21%, better than ABO by 11%, and better than CSA by 14%.

According to RMSE, the proposed algorithm has obtained better results than the other three algorithms which range from 6 to 9% in the case of grouping the users into three clusters. In the case of five clusters, the active user has more opportunity for achieving better recommendations as the HCSUC has the best RMSE results with 12, 13, and 20% when compared with ABO, standard CSA, and PSO respectively.

The second test increases the users, jokes, and test users (i.e., active users) numbers. The HCSUC algorithm for top-N recommendations has achieved average results better than other compared algorithms for the different numbers of clusters and different top-N as reported in Table 3.

Figure 9 illustrates the convergence curve of the fitness function in the case of the number of users ($U = 90$) who are grouped into 3 clusters ($C = 3$) for top-12 recommended jokes. Although at the first four iterations as shown from the figure, the standard CSA has obtained results better than HCSUC by 4%, but with more iterations, the proposed algorithm has proven its superiority for achieving better results than CSA which reached up to 6% and better than PSO and ABO which reached up 9 and 7%, respectively.

For a better recommendation process with an increased the number of clusters as shown in Fig. 10, the HCSUC algorithm has obtained the best results for finding the set of feasible clusters of similarity users. It has better results

Table 3 Experimental results of ($U = 90$ and $I = 30$)

Top-*N*	C	PSO			ABO		
		Min	Max	Avg	Min	Max	Avg
4	3	175.829	183.956	179.173	172.947	179.624	175.545
	5	167.149	176.452	172.668	167.149	175.011	170.687
	7	165.385	170.739	169.265	164.728	167.684	166.513
	9	157.109	169.418	164.418	156.130	165.471	160.318
8	3	177.180	182.689	180.659	175.273	182.270	178.255
	5	171.615	175.230	173.553	170.083	174.658	172.055
	7	164.492	177.647	169.134	165.011	168.357	166.756
	9	160.039	172.619	165.074	157.089	167.155	161.608
12	3	178.555	182.631	180.519	170.475	178.047	175.652
	5	169.410	175.288	172.412	163.524	174.812	169.495
	7	167.377	174.537	170.842	165.611	171.156	168.591
	9	159.158	167.466	164.421	154.413	166.850	162.271

Top-*N*	C	CSA			HCSUC		
		Min	Max	Avg	Min	Max	Avg
4	3	173.448	178.871	175.724	172.044	176.167	**173.707**
	5	167.136	171.345	170.219	157.568	170.665	**165.927**
	7	163.006	167.758	165.463	154.769	164.393	**161.046**
	9	155.933	162.931	158.628	149.465	159.32	**155.157**
8	3	172.490	179.171	176.556	163.781	176.465	**171.053**
	5	164.461	174.961	170.084	164.419	170.006	**166.795**
	7	159.038	167.967	164.605	157.482	162.804	**160.114**
	9	156.071	165.923	159.719	153.668	158.893	**156.165**
12	3	173.289	180.069	177.204	165.991	174.141	**170.122**
	5	164.864	174.555	168.544	158.717	169.939	**165.324**
	7	159.618	170.570	166.351	152.873	165.238	**159.482**
	9	153.765	162.113	159.305	153.424	160.802	**157.944**

than PSO which reached up to 4%, better than ABO which reached up to 3%, and better than CSA which reached up to 5% during iterations. In addition, the performance of the proposed algorithm has been improved ranging from 2% from the first iteration to 5% at the last iteration.

By increasing the number of clusters, the HCSUC algorithm has gradually obtained better results when compared with other presented algorithms as in Fig. 11.

Although at the first number of iterations as seen in Fig. 11, PSO has achieved better results than ABO and HCSUC, it saturates after that. The CSA has achieved better results than PSO, ABO, and HCSUC. Overall, HCSUC has obtained better results than PSO which reached up to 10% with more iterations, better than ABO which reached up to 9%, and better than CSA which reached up to 8% with more iterations. The proposed algorithm has improved by 12% during the number of iterations.

Figure 12 illustrates the convergence curve of a fitness function for all algorithms in the case of 90 users grouped into 9 clusters to improve the top-8 recommendation process.

From Fig. 12, the PSO and standard CSA have obtained results close to each other's for the first number of iterations, but CSA has obtained better results than PSO with more iterations. Although the ABO has obtained better results than PSO and CSA, the proposed HCSUC algorithm has gradually decreased and reached better results than PSO within the range 4–7%, better than ABO within the range 1–3%, and better than CSA within the range 1–5% during iterations.

The MAE and RMSE for 10 active users are listed in Table 4 with a different number of clusters and number of top-*N* recommendations.

In Table 4, the results of MAE for the proposed HCSUC are better than PSO within the range from 2 to 10% for top-

Table 4 MAE and RMSE for 10 active users ($U_{act} = 10$)

Top-*N*	C	PSO		ABO	
		MAE	RMSE	MAE	RMSE
4	3	0.289	0.346	0.288	0.347
8		0.296	0.350	0.295	0.349
12		0.283	0.343	0.277	0.335
4	5	0.288	0.347	0.286	0.345
8		0.271	0.323	0.270	0.324
12		0.284	0.338	0.278	0.333
4	7	0.288	0.345	0.274	0.331
8		0.307	0.366	0.296	0.355
12		0.289	0.346	0.286	0.342
4	9	0.283	0.345	0.273	0.332
8		0.286	0.346	0.280	0.337
12		0.290	0.346	0.281	0.341

Top-*N*	C	CSA		HCSUC	
		MAE	RMSE	MAE	RMSE
4	3	0.291	0.349	**0.283**	**0.341**
8		0.292	0.345	**0.288**	**0.342**
12		0.274	0.333	**0.267**	**0.327**
4	5	0.278	0.339	**0.271**	**0.330**
8		0.269	0.322	**0.262**	**0.315**
12		0.271	0.327	**0.258**	**0.314**
4	7	0.271	0.329	**0.259**	**0.315**
8		0.288	0.348	**0.281**	**0.339**
12		0.280	0.337	**0.275**	**0.332**
4	9	0.268	0.327	**0.259**	**0.319**
8		0.271	0.327	**0.264**	**0.321**
12		0.276	0.334	**0.262**	**0.321**

4 recommendation with a different number of clusters. Also, HCSUC has obtained MAE results better than ABO and standard CSA up to 5 and 4%, respectively. The RMSE of the proposed algorithm reached up to 9% (i.e., 7 clusters with top-4 recommendations) when compared with PSO, better than ABO by 6% (i.e., 5 clusters with top-12 recommendations) and better than CSA by 4% (i.e., 9 clusters with top-12 recommendations).

The third test is increasing the number of users ($U = 120$) and the number of jokes ($I = 60$), and the results of the three comparative algorithms are shown in Table 5.

From Table 5, the HCSUC algorithm has obtained average results better than the other compared algorithms for enhancing the different top-*N* recommendations. Figures 13 and 14 illustrate the minimization of an average fitness function for forming a feasible set of 3 and 5 clusters, respectively.

It was observed from Fig. 13 that by iterations, both PSO and ABO results are minimized, while the standard crow search algorithm has gained better results from both PSO and ABO from the start to the end of iterations. The proposed hybrid crow has obtained the best minimization results over other compared algorithms and has an improvement which reached up to 4% along with a different number of iterations.

As shown in Fig. 14, ABO has obtained the worst results, while for the first iterations, CSA and PSO have obtained results near to each other and at the end of iterations. Of all over the iterations, the HCSUC has the superiority of obtaining the best results with an improvement up to 5% when compared with other algorithms.

For more illustrations, with 15 active users ($U_{act} = 15$), Figs. 15 and 16 show the MAE and RMSE for 3 and 5 clusters with top 4, 8, and 12 recommendations for each cluster.

In Fig. 15, the MAE of the proposed algorithm is minimized by 3% (i.e., in the case of 3 clusters with top 12 recommendations) when compared with PSO and CSA. In addition, it minimized by 7 and 4% (i.e., in the case of 3 clusters with top 8 recommendations) when compared with PSO and ABO, respectively. In the case of 3 clusters with top-4 recommendations, the MSE of HCSUC is minimized within the range 1–3% when compared with other algorithms. In the case of the number of users grouped to 5 clusters for better recommendations, the HCSUC has better MAE results up to 8% with top-8 recommendations when compared with PSO, up to 7% with top-4 and top-12 recommendations when compared with ABO and 3% when compared with standard CSA for different top-*N* recommendations.

According to Fig. 16, the HCSUC algorithm has proven its efficiency in terms of RMSE where it has better results than PSO ranging from 2 to 7% for different clusters and recommendations. In addition, it has superiority when compared with ABO and standard CSA with the best results reached being up to 6 and 3% respectively.

6.2 Comparison between CF and other meta-heuristic algorithms

In this subsection, the accuracy of the proposed algorithm and other meta-heuristics is compared against the conventional collaborative filtering. Table 6 shows the variation of MAE and RMSE for different numbers of randomly selected active users (e.g., 5 and 10) using all algorithms for recommending top-4 jokes.

From Table 6, the result of CF has the worst values when compared with meta-heuristic algorithms, whereas

Table 5 Experimental results of ($U = 120$ and $I = 60$)

Top-N	C	PSO			ABO		
		Min	Max	Avg	Min	Max	Avg
4	3	193.681	201.649	197.923	188.314	197.871	194.184
	5	191.001	195.714	193.227	190.255	195.496	193.155
8	3	193.118	202.609	197.555	193.971	198.945	195.768
	5	190.179	194.512	191.981	188.745	194.696	191.712
12	3	195.624	201.111	197.807	195.499	198.206	196.138
	5	187.414	198.415	192.837	186.521	191.786	189.435

Top-N	C	CSA			HCSUC		
		Min	Max	Avg	Min	Max	Avg
4	3	188.075	195.918	193.568	183.099	195.106	**191.046**
	5	185.685	193.234	190.490	182.455	187.425	**184.969**
8	3	191.533	197.169	194.227	185.801	193.491	**189.910**
	5	185.422	191.156	188.333	186.522	189.691	**188.273**
12	3	190.766	196.063	194.289	188.658	194.731	**192.081**
	5	180.37	191.652	186.997	177.884	189.003	**184.021**

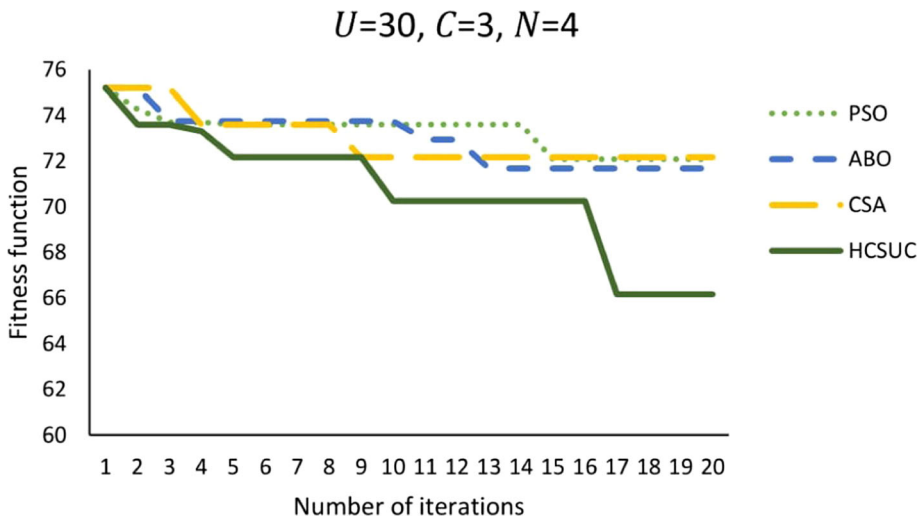


Fig. 5 Convergence curve for 30 users grouped into 3 clusters

the proposed HCSUC outperforms others in all cases due to the clustering-based method for a different number of clusters (i.e., better neighborhood choice of the active user). For example, for 30 users and 5 active users, the MAE of CF is 0.446 where the MAE of PSO, ABO, CSA, and HCSUC is 0.3424, 0.334, 0.3356, and 0.2948, respectively, in the case of 5 clusters. In the case of RMSE for U=90 and $U_{act} = 10$, the CF has achieved the worst value (0.3722), while the proposed algorithm has achieved the best value (0.3168). In addition, Figs. 17 and 18 illustrate the results of MAE and RMSE for the meta-heuristic algorithms (PSO, ABO, CSA, and HCSUC) against the collaborative filtering technique (CF) of various experiments to demonstrate the performance of the proposed algorithm.

From Fig. 17, the CF technique has achieved the maximum MAE, while the proposed algorithm has obtained the minimum value for recommendation with different experiments. The HCSUC has achieved better results than CF within the range from 18 to 34%, better than PSO ranging from 12 to 16%, better than ABO, and CSA up to 12%.

According to the results of RMSE, as shown in Fig. 18, the proposed algorithm has obtained results better than other CF and other meta-heuristic algorithms. When compared with CF, the HCSUC has achieved improvements in the RMSE ranging from 15 to 29% along with

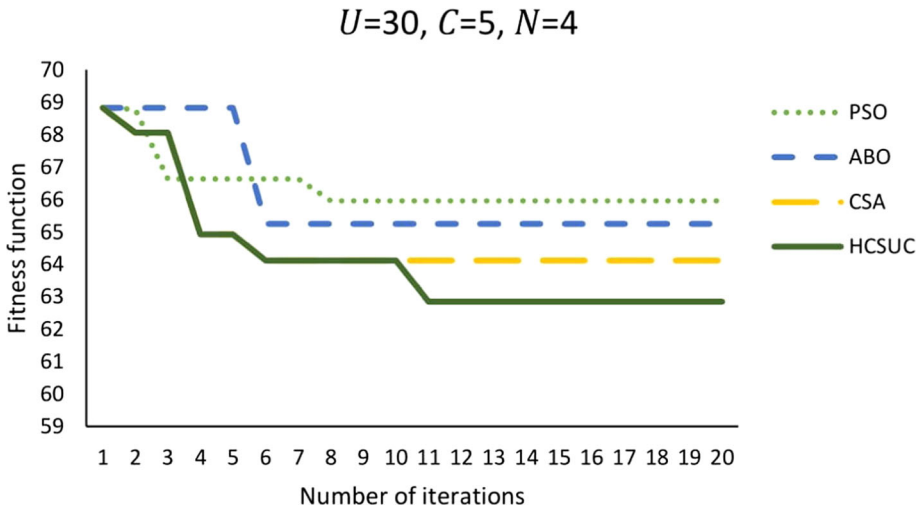
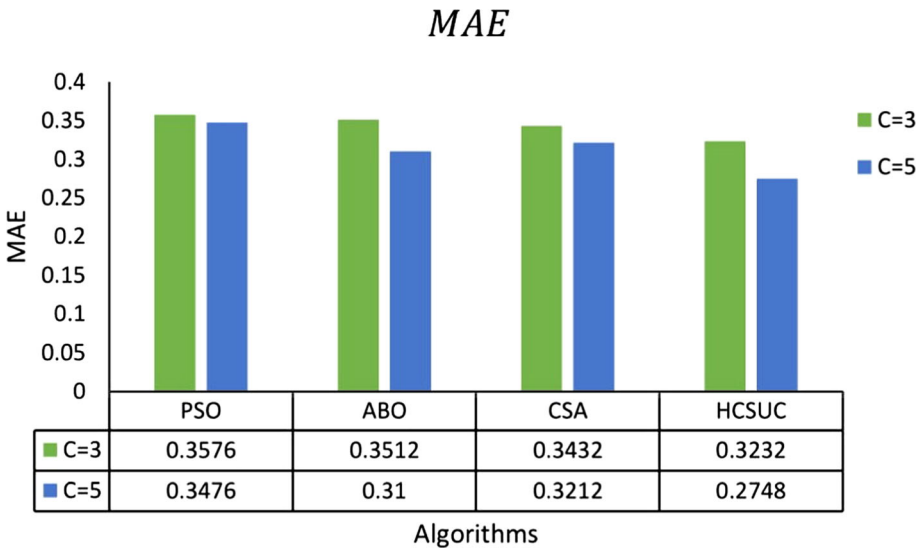
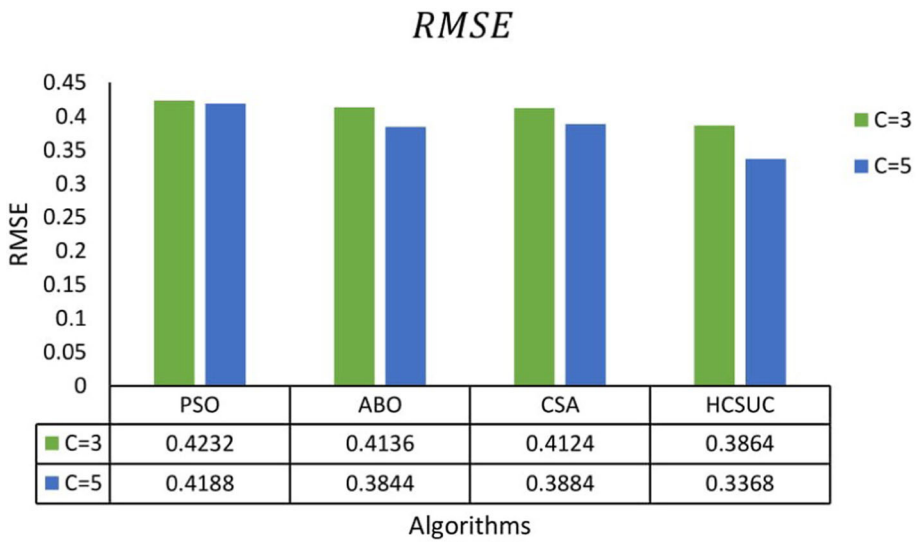


Fig. 6 Convergence curve for 30 users grouped into 5 clusters



	PSO	ABO	CSA	HCSUC
C=3	0.3576	0.3512	0.3432	0.3232
C=5	0.3476	0.31	0.3212	0.2748

Fig. 7 Mean absolute error for 5 active users



	PSO	ABO	CSA	HCSUC
C=3	0.4232	0.4136	0.4124	0.3864
C=5	0.4188	0.3844	0.3884	0.3368

Fig. 8 Root mean square error for 5 active users

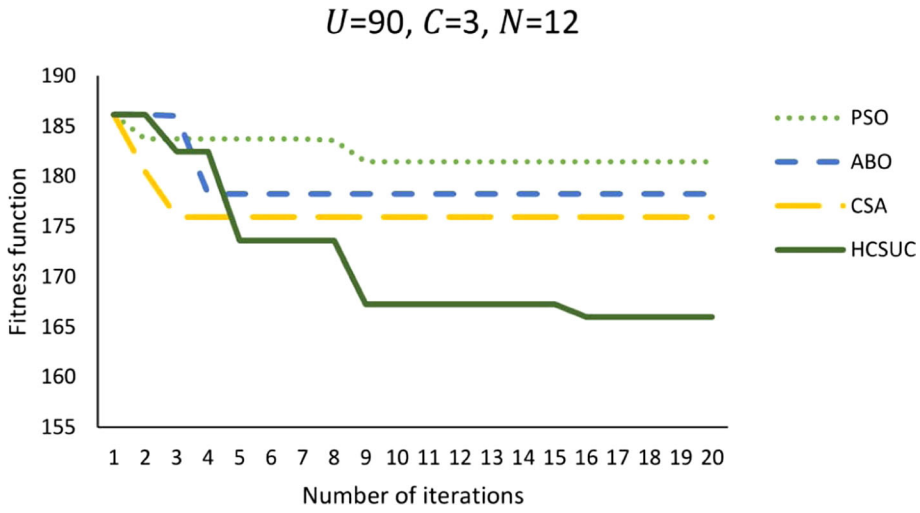


Fig. 9 Convergence curve for 90 users grouped into 3 clusters (top-$N = 12$)

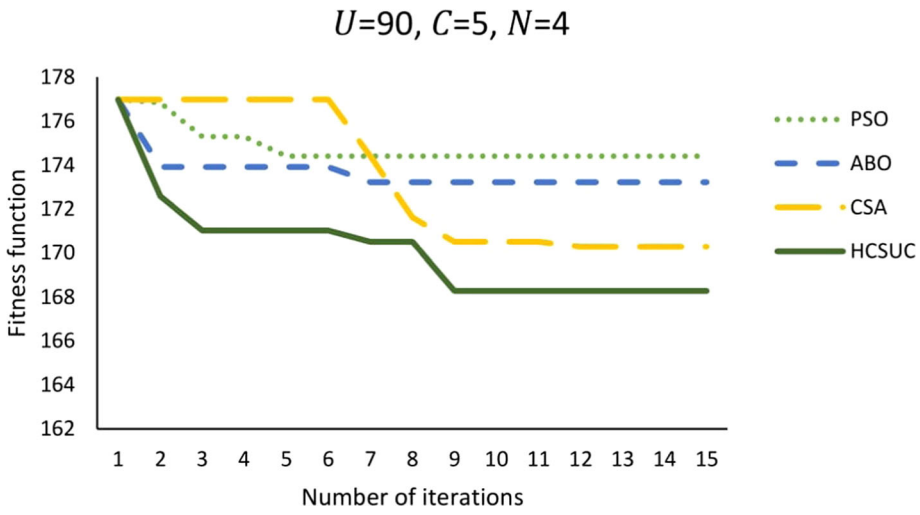


Fig. 10 Convergence curve for 90 users grouped into 5 clusters (top-$N = 4$)

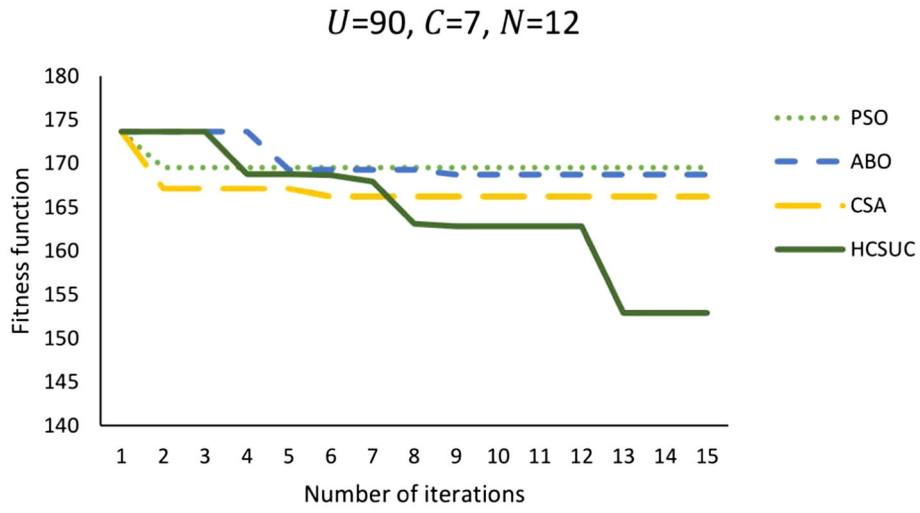


Fig. 11 Convergence curve for 90 users grouped into 7 clusters (top-$N = 12$)

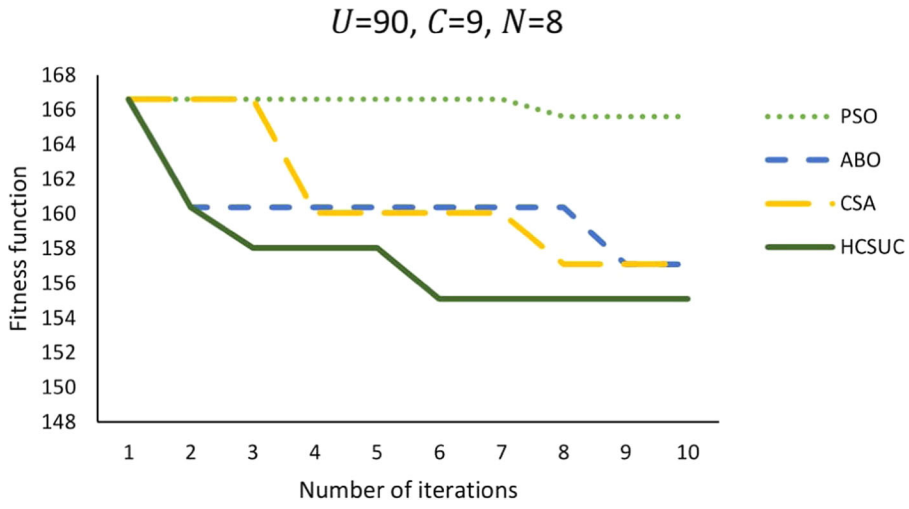


Fig. 12 Convergence curve for 90 users grouped into 9 clusters (top-$N = 8$)

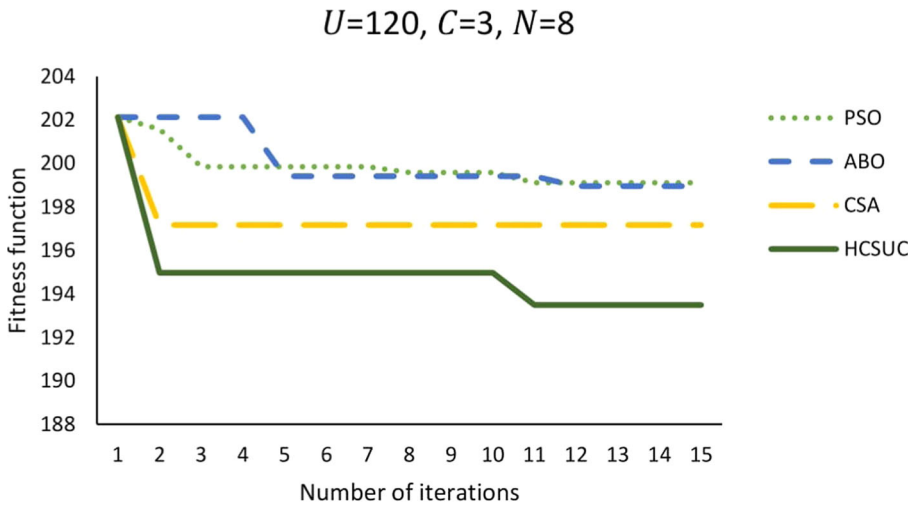


Fig. 13 Convergence curve for 120 users grouped into 3 clusters (top-$N = 8$)

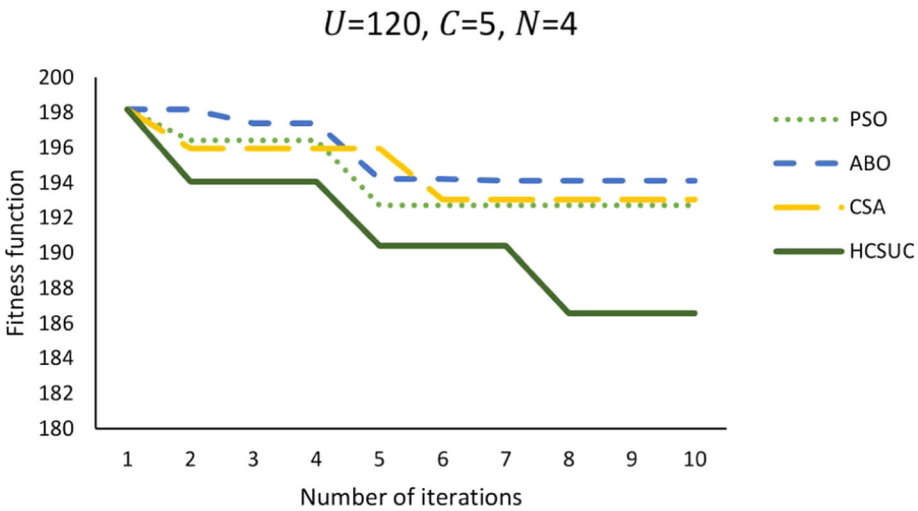
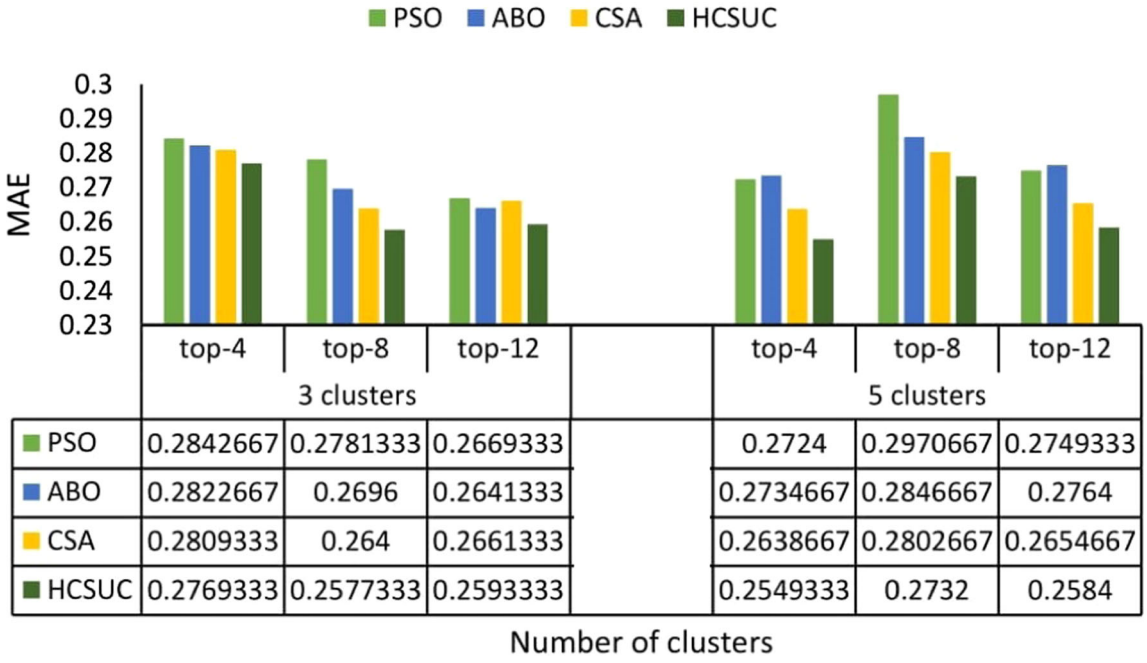
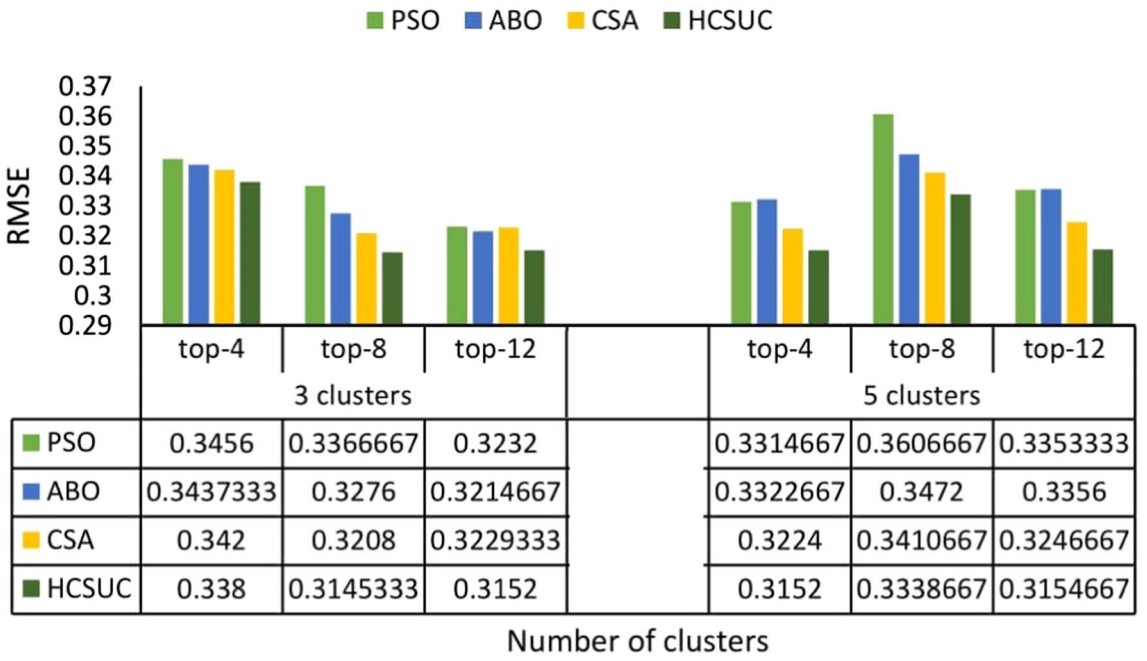


Fig. 14 Convergence curve for 120 users grouped into 5 clusters (top-$N = 4$)



	top-4 (3 clusters)	top-8 (3 clusters)	top-12 (3 clusters)	top-4 (5 clusters)	top-8 (5 clusters)	top-12 (5 clusters)
PSO	0.2842667	0.2781333	0.2669333	0.2724	0.2970667	0.2749333
ABO	0.2822667	0.2696	0.2641333	0.2734667	0.2846667	0.2764
CSA	0.2809333	0.264	0.2661333	0.2638667	0.2802667	0.2654667
HCSUC	0.2769333	0.2577333	0.2593333	0.2549333	0.2732	0.2584

Fig. 15 Mean absolute error (MAE) for active users ($U_{act} = 15$)



	top-4 (3 clusters)	top-8 (3 clusters)	top-12 (3 clusters)	top-4 (5 clusters)	top-8 (5 clusters)	top-12 (5 clusters)
PSO	0.3456	0.3366667	0.3232	0.3314667	0.3606667	0.3353333
ABO	0.3437333	0.3276	0.3214667	0.3322667	0.3472	0.3356
CSA	0.342	0.3208	0.3229333	0.3224	0.3410667	0.3246667
HCSUC	0.338	0.3145333	0.3152	0.3152	0.3338667	0.3154667

Fig. 16 Root mean square error (RMSE) for active users ($U_{act} = 15$)

various experiments. When compared with other meta-heuristics, the proposed algorithm has a minimized RMSE that reaches up to 9% (in the case of ABO and CSA) and up to 12% (in the case of PSO).

From all the above results, we can deduce that the proposed hybrid crow search and uniform crossover algorithm-based clustering outperformed other algorithms in all test cases of the Jester dataset.

Table 6 MAE and RMSE results of $U_{act} = 5$ and $U_{act} = 10$

U	U_{act}	C	PSO		ABO	
			MAE	RMSE	MAE	RMSE
30	5	3	0.4072	0.4756	0.3864	0.4472
		5	0.3424	0.4128	0.334	0.4012
90	10	7	0.2806	0.3378	0.2804	0.3398
		9	0.2758	0.3338	0.275	0.3338
U	U_{act}	C	CSA		HCSUC	
			MAE	RMSE	MAE	RMSE
30	5	3	0.3796	0.4444	**0.36**	**0.424**
		5	0.3356	0.3996	**0.2948**	**0.364**
90	10	7	0.2806	0.3394	**0.2638**	**0.3224**
		9	0.2744	0.3326	**0.2588**	**0.3168**

U	U_{act}	CF	
		MAE	RMSE
30	5	0.4912	0.5576
		0.446	0.5128
90	10	0.3234	0.3844
		0.3144	0.3722

Bold fonts the best results are marked

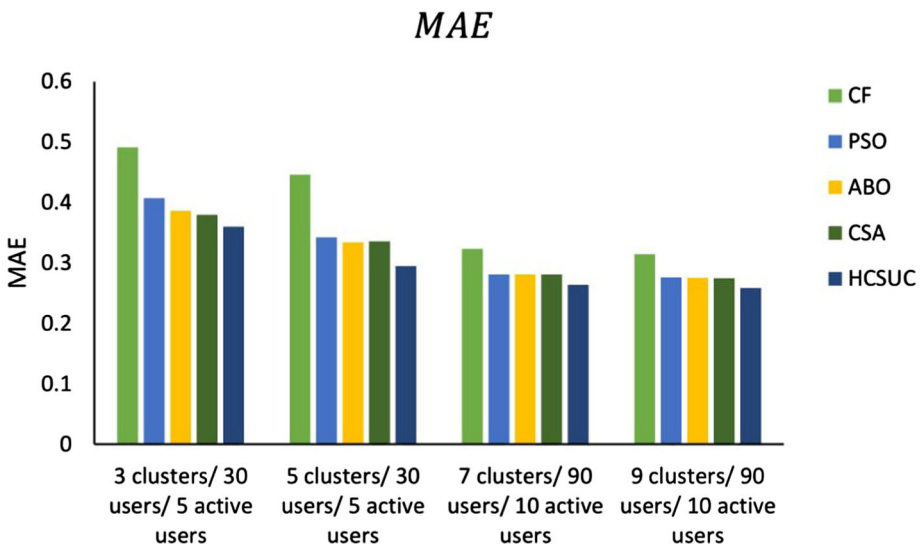


Fig. 17 Mean absolute error (MAE) for comparative algorithms

7 Conclusion and future work

In this paper, we proposed a new recommendation system based on user-based collaborative filtering by combining the crow search algorithm and a genetic uniform crossover operator. The proposed algorithm is called hybrid crow search and uniform crossover algorithm (HCSUC). Such a combination between the crow search algorithm and genetic uniform crossover operator can increase the diversity of the search and avoid trapping in local minima. The HCSUC algorithm has two phases; the first phase is cluster forming, and the second phase is a top-N recommendation for an active user. HCSUC is compared with the standard crow search, African buffalo optimization and

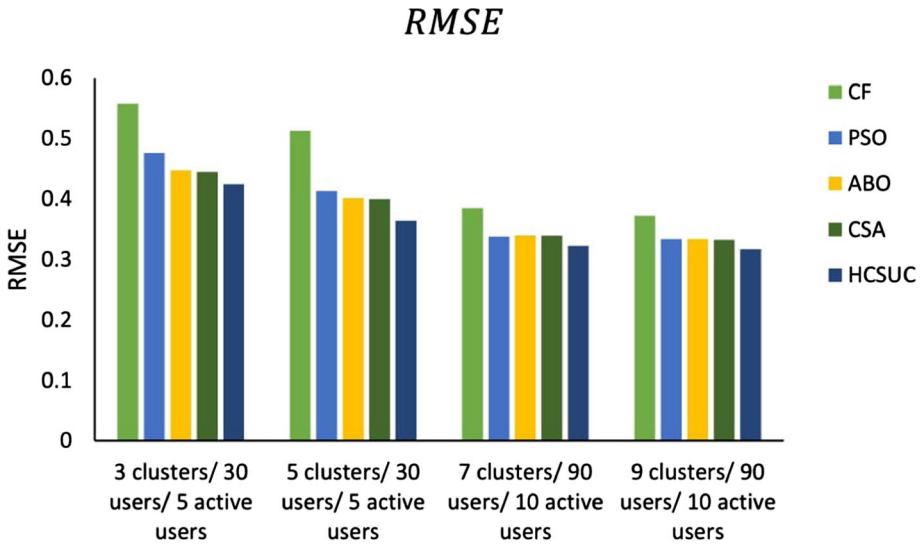


Fig. 18 Root mean square error (RMSE) for comparative algorithms

particle swarm optimization algorithms. HCSUC is tested on the Jester dataset with different numbers of users and a different number of jokes. The results show that the HCSUC algorithm can work as a recommender system with high-quality results. In our future work, we will add some modifications to the proposed HCSUC algorithm to work in a cluster system or grid computing system so that it is suitable for a large data size.

Compliance with ethical standards

Conflict of interest In the present work, we have not used any material from previously published. So we have no conflict of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

1. http://www.internetlivestats.com/. Last retrieved May 2019
2. Bobadilla J, Ortega F, Hernando A, Gutiérrez A (2013) Recommender systems survey. Knowl Based Syst 46:109–132
3. Adomavicius G, Tuzhilin A (2005) Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. IEEE Trans Knowl Data Eng 17(6):734–749
4. Fahad A et al (2014) A survey of clustering algorithms for big data: taxonomy and empirical analysis. IEEE Trans Emerg Top Comput 2(3):267–279
5. Herlocker J, Konstan JA, Riedl J (2002) An empirical analysis of design choices in neighborhood-based collaborative filtering. Inf Retr 5(4):287–310
6. Konstan JA, Riedl J (2012) Recommender systems: from algorithms to user experience. User Model User-Adap Inter 22:101–123
7. Wang W, Zhang G, Lu J (2015) Collaborative filtering with entropy-driven user similarity in recommender systems. Int J Intell Syst 30(8):854–870
8. Sarwar BM, Karypis G, Konstan JA, Riedl J (2001) Item-based collaborative filtering recommendation algorithms. WWW10, pp 285–295
9. Salter J, Antonopoulos N (2006) CinemaScreen recommender agent: collaborative and content based filtering. Intell Syst 21(1):35–41
10. Van Meteren R, Van Someren M (2000) Using content-based filtering for recommendation. In: Proceedings of ECML/MLNET workshop: machine learning in new information age, pp 47–56
11. Poonam B, Goudar RM, Barve S (2015) Survey on collaborative filtering, content-based filtering and hybrid recommendation system. Int J Comput Appl 110(4):31–36
12. Krulwich B (1997) Lifestyle finder: intelligent user profiling using large-scale demographic data. AI Mag 18(2):37–46
13. Singh SP, Solanki S (2019) Recommender system survey: clustering to nature inspired algorithm. In: Proceedings of 2nd international conference on communication, computing and networking, pp 757–768
14. Burke R (2007) Hybrid web recommender systems. Adapt Web 4321:377–408
15. Porcel C, Tejeda-Lorente A, Martinez MA, Herrera-Viedma E (2012) A hybrid recommender system for the selective dissemination of research resources in a technology transfer office. Inf Sci 184(1):1–19
16. Park DH, Kim HK, Choi I, Kim JK (2012) A literature review and classification of recommender systems research. Expert Syst Appl 39(11):10059–10072
17. Tsai CF, Hung C (2012) Cluster ensembles in collaborative filtering recommendation. Appl Soft Comput 12(4):1417–1425
18. Askarzadeh A (2016) A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm. Comput Struct 169:1–12

19. Herlocker JL, Konstan JA, Terveen LG, Riedl JT (2004) Evaluating collaborative filtering recommender systems. ACM Trans Inf Syst 22(1):5–53
20. Feng L, Zhao Q, Zhou C (2020) Improving performances of Top-N recommendations with co-clustering method. Expert Syst Appl 143:113078
21. Wasid M, Ali R (2018) An improved recommender system based on multi-criteria clustering approach. Proc Comput Sci 131:93–101
22. Salah A, Rogovschi N, Nadif M (2016) A dynamic collaborative filtering system via a weighted clustering approach. Neurocomputing 175:206–215
23. Wang Z, Yu X, Feng N, Wang Z (2014) An improved collaborative movie recommendation system using computational intelligence. J Vis Lang Comput 25(6):667–675
24. Katarya R, Verma OP (2016) A collaborative recommender system enhanced with particle swarm optimization technique. Multimedia Tools Appl 75:9225–9239
25. Bedi P, Sharma R, Kaur H (2009) Recommender system based on collaborative behaviour of ants. J Artif Intell 2:40–55
26. Katarya R, Verma OP (2018) Recommender system with grey wolf optimizer and FCM. Neural Comput Appl 30:1679–1687
27. Senbagaraman M, Senthilkumar R, Subasankar S, Indira R (2017) A movie recommendation system using collaborative approach and cuttlefish optimization. In: Proceedings of international conference on emerging trends in engineering, science and sustainable technology, pp 95–99
28. Katarya R, Verma OP (2017) An effective collaborative movie recommender system with cuckoo search. Egypt Inform J 18(2):105–112
29. Katarya R (2018) Movie recommender system with metaheuristic artificial bee. Neural Comput Appl 30(6):1983–1990
30. Logesh R, Subramaniyaswamy V, Vijayakumar V, Gao X-Z, Indragandhi V (2018) A hybrid quantum-induced swarm intelligence clustering for the urban trip recommendation in smart city. Future Gener Comput Syst 83:653–673
31. Katarya R, Verma OM (2017) Effectual recommendations using artificial algae algorithm and fuzzy c-mean. Swarm Evolut Comput 36:52–61
32. Rana C, Jain SK (2014) An evolutionary clustering algorithm based on temporal features for dynamic recommender systems. Swarm Evolut Comput 14:21–30
33. Chen J, Uliji S, Wang H, Yan Z (2018) Evolutionary heterogeneous clustering for rating prediction based on user collaborative filtering. Swarm Evolut Comput 38:35–41
34. Alam S, Dobbie G, Riddle P, Koh YS (2012) Hierarchical PSO clustering based recommender system. In: Proceedings of IEEE congress on evolutionary computation, pp 1–8
35. Marung U, Theera-Umpon N, Auephanwiriyakul S (2016) Top-N recommender systems using genetic algorithm-based visual-clustering methods. Symmetry 54(8):1–19
36. Hussain A, Muhammad YS, Sajid MN (2018) An efficient genetic algorithm for numerical function optimization with two new crossover operators. Int J Math Sci Comput 4(4):1–17
37. Umbarkar AJ, Sheth PD (2015) Crossover operators in genetic algorithms: a review. ICTACT J Soft Comput 6(1):1083–1092
38. Syswerda G (1989) Uniform crossover in genetic algorithms. In: Proceedings of the third international conference on Genetic algorithms, pp 2–9
39. Lü L, Medo M, Yeung CH, Zhang Y-C, Zhang Z-K, Zhou T (2012) Recommender systems. Phys Rep 519(1):1–49
40. Goldberg K, Roeder T, Gupta D, Perkins C (2001) Eigentaste: a constant time collaborative filtering algorithm. Inf Retr 4(2):133–151
41. http://eigentaste.berkeley.edu/dataset/. Last retrieved April 2019
42. Kennedy J, Eberhart RC (1995) Particle swarm optimization. In: Proceedings of the IEEE international conference on neural networks, vol 4, pp 1942–1948
43. Odili JB, Kahar MNM (2015) African buffalo optimization (ABO): a new meta-heuristic algorithm. J Adv Appl Sci 3(3):101–106
44. Yan J, He W, Jiang X, Zhang Z (2017) A novel phase performance evaluation method for particle swarm optimization algorithms using velocity-based state estimation. Appl Soft Comput 57:517–525
45. Schroder G, Thiele M, Lehner W (2011) Setting goals and choosing metrics for recommender system evaluations. In: Proceedings of UCERSTI2 workshop at the 5th ACM conference on recommender systems, vol 23, pp 78–85

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.