


ORIGINAL RESEARCH

Clustering-based recommendation method with enhanced grasshopper optimisation algorithm

Zihao Zhao^{1,2} | Yingchun Xia^{1,2} | Wenjun Xu¹ | Hui Yu¹ | Shuai Yang^{1,2} |
Cheng Chen¹ | Xiaohui Yuan³ | Xiaobo Zhou¹ | Qingyong Wang^{1,2} | Lichuan Gu^{1,2} 

¹School of Information and Artificial Intelligence, Anhui Agricultural University, Hefei, China

²Anhui Provincial Engineering Research Center for Agricultural Information Perception and Intelligent Computing, Hefei, China

³University of North Texas, Denton, Texas, USA

Correspondence

Lichuan Gu and Qingyong Wang.
Email: glc@ahau.edu.cn and
wangqingyong@ahau.edu.cn

Funding information

Natural Science Research Project of Education Department of Anhui Province of China, Grant/Award Number: 2023AH051020; Key Project of Anhui Province's Science and Technology Innovation Tackle Plan, Grant/Award Number: 202423k09020040; National Key Research and Development Program of China, Grant/Award Number: 2023YFD1802200; Natural Science Foundation of Anhui Province, Grant/Award Number: 2308085MF21; National Natural Science Foundation of China, Grant/Award Numbers: 32472007, 62301006, 62306008; University Synergy Innovation Program of Anhui Province, Grant/Award Number: GXXT-2022-046

Abstract

In the era of big data, personalised recommendation systems are essential for enhancing user engagement and driving business growth. However, traditional recommendation algorithms, such as collaborative filtering, face significant challenges due to data sparsity, algorithm scalability, and the difficulty of adapting to dynamic user preferences. These limitations hinder the ability of systems to provide highly accurate and personalised recommendations. To address these challenges, this paper proposes a clustering-based recommendation method that integrates an enhanced Grasshopper Optimisation Algorithm (GOA), termed LCGOA, to improve the accuracy and efficiency of recommendation systems by optimising cluster centroids in a dynamic environment. By combining the K-means algorithm with the enhanced GOA, which incorporates a Lévy flight mechanism and multi-strategy co-evolution, our method overcomes the centroid sensitivity issue, a key limitation in traditional clustering techniques. Experimental results across multiple datasets show that the proposed LCGOA-based method significantly outperforms conventional recommendation algorithms in terms of recommendation accuracy, offering more relevant content to users and driving greater customer satisfaction and business growth.

KEYWORDS

artificial intelligence, clustering, recommender systems, swarm intelligence

1 | INTRODUCTION

Towards the close of the 20th century, the scholarly discourse first acknowledged the paradigm of recommendation systems, with collaborative filtering being expounded upon as a viable methodology for the provision of personalised suggestions [1]. Subsequently, collaborative filtering algorithms were applied in real-world situations, such as on movie platforms, to generate recommendations for users. Three commonly used types of

recommendation include collaborative filtering recommendation systems [2, 3], content-based recommendation systems [4] and model-based recommendation systems [5].

In the domain of personalised recommendation systems, traditional methods like collaborative filtering face significant challenges due to data sparsity and algorithm scalability issues. Recent years have seen various researchers attempting to enhance recommendation algorithm performance through diverse strategies, the algorithms such as singular value

Zihao Zhao and Yingchun Xia are contributed equally to this work.

This is an open access article under the terms of the [Creative Commons Attribution-NoDerivs](https://creativecommons.org/licenses/by/4.0/) License, which permits use and distribution in any medium, provided the original work is properly cited and no modifications or adaptations are made.

© 2025 The Author(s). *CAAI Transactions on Intelligence Technology* published by John Wiley & Sons Ltd on behalf of The Institution of Engineering and Technology and Chongqing University of Technology.

decomposition [6], Probabilistic Matrix Factorisation (PMF) [7] and Bayesian Probabilistic Matrix Factorization [8] have been employed. Although these techniques have improved recommendation effectiveness, they overlook the issue of centroid sensitivity that is often encountered in clustering, leading to reduced suggestion accuracy. To tackle this problem, we propose using the Grasshopper Optimisation Algorithm (GOA) [9], a population intelligence algorithm, to optimise clustering centroids. Additionally, we enhance GOA by incorporating a Lévy flight mechanism and multi-strategy co-evolution.

In recent years, the field of optimisation has witnessed a surge in the development of meta-heuristic algorithms, inspired by natural phenomena and behaviours observed in biological entities. These algorithms have been successfully applied to a wide range of problems, offering novel solutions to complex optimisation challenges. Among these, the GOA has demonstrated significant potential in optimising clustering centroids for recommendation systems. However, to further enhance our understanding and application of optimisation in recommendation systems, it is imperative to consider the rich landscape of recently proposed meta-heuristics.

Notably, the Monarch Butterfly Optimization [10] draws inspiration from the migration behaviour of monarch butterflies, offering a robust mechanism for global optimisation through the simulation of butterfly migration and adjustment phases. Similarly, the Slime Mould Algorithm [11] models the foraging behaviour of slime mould, efficiently navigating the search space by mimicking the mould's growth and movement patterns towards food sources. The Moth Search Algorithm [12] leverages the transverse orientation navigation method used by moths in nature, enabling an effective balance between exploration and exploitation in the search space. Meanwhile, the Hunger Games Search [13] algorithm, inspired by the concept of survival of the fittest, dynamically adjusts the search strategy based on the fitness landscape, promoting diversity and convergence towards optimal solutions. Further contributing to the array of meta-heuristics, the Colony Predation Algorithm [14] simulates the predation process in biological colonies, exploiting cooperative and competitive interactions among individuals to explore and exploit the search space effectively. Moreover, the Runge-Kutta method (RUN) [15] is a powerful and widely used numerical technique for solving ordinary differential equations by iteratively approximating solutions with high accuracy and stability. The Weighted Mean of Vectors (INFO) [16] proposes an innovative approach by integrating information from previous iterations to guide the search process, enhancing the algorithm's performance in navigating through multidimensional search spaces. Lastly, the rime Optimisation Algorithm (RIME) [17] offers a unique perspective by incorporating prime number theory into its search mechanics, exploring the search space through the distribution and properties of prime numbers.

Swarm intelligence optimisation algorithms have demonstrated extensive applications and notable successes across various fields, including computational biomedicine, medical image processing, and complex system optimisation. These algorithms enhance diagnostic precision and feature selection

efficiency in specific applications such as tumour classification and early-late stage diagnosis [18, 19], COVID-19 image segmentation [20], skin lesion diagnosis [21], advancing computational methods and technologies significantly [22, 23]. Swarm intelligence algorithms have also been explored by several scholars to improve the clustering model's accuracy and convergence speed, significantly enhancing the recommendation process's performance. As an example, Pan et al. [24] proposed a semi-supervised Particle Swarm Optimisation (PSO) clustering algorithm that utilises adaptive parameter optimisation to optimise the clustering model for collaborative filtering. The goal was to improve the performance of the clustering and nearest neighbour selection processes in collaborative filtering. Similarly, Yadav et al. [25] proposed a recommendation algorithm that incorporates trust information to overcome the limitations of collaborative filtering. They utilised PSO and Bat Algorithm (BA) to optimise the accuracy of recommendations. Logesh et al. [26] proposed a new heuristic clustering algorithm that combines swarm intelligence with fuzzy clustering models and applied it to a user-based collaborative filtering recommendation algorithm. Singh et al. [27] proposed a novel fuzzy clustering based recommendation method using improved grasshopper optimisation and Map-Reduce. Research has shown that collaborative filtering technology faces challenges related to data sparsity and scalability, whereas the swarm intelligence optimisation method has been successful in addressing the centroid-sensitive problem, but it is more susceptible to local optimal solutions.

In this paper, we propose LCGOA clustering-based recommendation. This innovative approach leverages swarm intelligence principles observed in nature, specifically the grasshopper's swarming behaviour, to optimise clustering centroids. By integrating the K-means algorithm with the LCGOA, our method adeptly addresses the challenges of data sparsity and scalability, paving the way for more accurate and efficient recommendations. The real-world benefits of such advancements are manifold, extending beyond mere technological enhancements to significantly improve user experiences and drive business success. Users enjoy more personalised and relevant content, leading to increased engagement and satisfaction, while businesses benefit from higher retention rates and potential revenue growth, showcasing the tangible impact of improved recommendation systems in today's digital landscape. The key contributions of this paper are:

- The integration of GOA enhanced with Lévy flight [28] and multi-strategy co-evolution in clustering for recommendation systems is a novel approach that optimises clustering centroids to overcome the limitations of traditional methods like K-means [29], addressing centroid sensitivity and scalability.
- By innovatively applying clustering techniques to tackle issues of data sparsity and scalability, the LCGOA method enhances both the accuracy and efficiency of recommendations, marking a significant improvement over traditional collaborative filtering methods.

- The LCGOA method is empirically validated across diverse datasets [30, 31], demonstrating its practical effectiveness and superiority over existing recommendation techniques, thus providing a robust proof of concept for its real-world applicability.
- Theoretical advancements are contributed through the detailed exploration of Lévy flight mechanisms in GOA for clustering within recommendation systems, promising substantial improvements in user engagement and content relevancy in commercial applications.
- Future research directions highlight the potential for problem-specific optimisations and real-world evaluations, suggesting ongoing development and refinement of the LCGOA model to expand its application in computational intelligence fields.

The remainder of this paper is organised as follows: Section 2 provides an overview of related work in this area. Section 3 outlines the application of the recommendation algorithm to the enhanced GOA. The presentation and analysis of experimental outcomes are presented in Section 4. Finally, Section 5 summarises the key findings of this paper and offers suggestions for future research.

2 | RELATED WORK

- Recommendation algorithm: The recommendation algorithm, which is responsible for determining the effectiveness and quality of the recommendation system, is a crucial component. Since the emergence of recommendation systems, these algorithms have been continuously developed and optimised and have been applied in various industries. Recently, both domestic and international academics have conducted extensive research on the challenges posed by recommendation technology. To address the issue of duplicated records in recommendation systems, Ghazanfar et al. [32] merged multiple recommendation system strategies and developed a cascaded hybrid recommendation system. Sajal Halder et al. [33] proposed the concept of movie group mining to manage and increase the viewing time of the most recent and popular films, thereby addressing the cold start problem. Panigrahi et al. [34] introduced a new hybrid algorithm that employs K-means, Alternating Least Squares and other dimensionality reduction techniques to address scalability and sparsity issues. Kumar et al. [35] proposed a two-class structure for binary matrix decomposition that generates a more accurate rating matrix than an ordinal matrix. To suggest favourite movies to active users, Bogdan and Vladimir [36] designed a monolithic hybrid recommendation system. Wu et al. [37] presented a comprehensive overview of personalised news recommendation, and proposed a novel perspective to understand personalised news recommendation based on its core problems and the associated techniques and challenges.
- Weighted Slope One (WSO) algorithm: In 2005, Professor Daniel Lemire [38] introduced the Slope One algorithm, a widely recognised item-based recommendation algorithm. This algorithm is known for its simplicity, efficiency and ease of implementation, and has demonstrated successful results in numerous applications. The algorithm begins by calculating the scoring deviation, as outlined in its calculation process. Given two items i and j ($i \neq j$), the deviation calculation equation dev_{ij} is shown below:

$$\text{dev}_{ij} = \frac{\sum_{u \in S_{ij}} (r_{ui} - r_{uj})}{|S_{ij}|} \quad (1)$$
 where r_{ui} and r_{uj} are the ratings of items i and j by user u , S_{ij} denotes the set of users who have ratings for both items i and j ($i \neq j$), and $|S_{ij}|$ denotes the number of users in the user set S_{ij} . After obtaining the deviation dev_{ij} between items i and j ($i \neq j$), user u can make a prediction for item i . The prediction equation pre_{ui} is as follows:

$$\text{pre}_{ui} = \frac{\sum_{j \in S(u) - \{i\}} (r_{uj} + \text{dev}_{uj})}{|S(u) - \{i\}|} \quad (2)$$
 where $S(u)$ denotes the set of all items rated by user u , $S(u) - \{i\}$ denotes the set of items in the set that have at least one item (i.e., items other than i) rated by user u at the same time as item i .
 However, the algorithm does not take into account the number of items being rated simultaneously, which can affect the accuracy of the predictions. To address this issue, Lemire et al. [38] proposed the WSO algorithm, which is outlined below:

$$\text{pre}_{ui} = \frac{\sum_{j \in S(u) - \{i\}} (r_{uj} + \text{dev}_{ij}) * |S_{ij}|}{\sum_{j \in S(u) - \{i\}} |S_{ij}|} \quad (3)$$
- Swarm intelligence optimisation: The swarm intelligence optimisation strategy is based on the core idea of generating intelligent group behaviours that can accomplish complex tasks by simulating the behaviour of individuals and constraining it with specific rules. This algorithm is inspired by the group behaviour observed in populations of organisms and the natural evolutionary processes that occur in nature. The concept of group intelligence was first introduced by Beni et al. [39]. Since then, many swarm intelligence algorithms have been developed, including Genetic Algorithm (GA) [40], Particle Swarm Optimization (PSO) [41], Bat Algorithm (BA) [42], Grey Wolf Optimiser (GWO) [43], Salp Swarm Algorithm (SSA) [44] and others. Swarm intelligence algorithms offer several advantages over conventional optimisation methods, including a straightforward framework, scalability, robustness and fast convergence. By simulating the collective intelligent behaviour of organisms, these algorithms provide a class of models with the ability to perform intelligent searches, which can be applied to various optimisation problems such as the Travelling Salesman Problem [45], Job-shop Scheduling Problem [46] and clustering problems [47].

- Lévy flight: The Lévy flight was first proposed by French mathematician Paul Lévy in 1937 [48]. There is sample evidence to suggest that the Lévy Flight can be used to describe the movement patterns of various animals [28]. During the flight, larger step sizes in the early stage contribute to increasing the complexity of biological populations and extending the activity search, which can make it more challenging to find the optimum solution locally. In the later stage, the step size decreases, allowing the biological population to quickly converge towards the global optimum solution in a smaller region. In this paper, the Lévy flight path is used to update the grasshoppers position using the stochastic equation given below [49]:

$$\vec{X}(t+1) = \vec{X}(t) + \mu \text{sign} \left[\text{rand} - \frac{1}{2} \right] \oplus L(s) \quad (4)$$

where $\vec{X}(t+1)$ denotes the position of the i grasshopper at the t iteration, μ is a uniformly distributed random number, \oplus is an entry wise multiplier, and rand is a random number in the interval $[0, 1]$.

The Lévy distribution is based on random steps to generate random numbers and random wanderings determined by the step size, which can usually be approximated as a simple power function distribution $L(s) \sim |s|^{-1-\beta}$ ($0 < \beta \leq 2$) is a parameter used to control the distribution, usually taken as 1.5, s is the step size and $L(s)$ is the probability of moving the step size s , so we can express the Lévy distribution as

$$L(s, \gamma, \mu) = \begin{cases} \sqrt{\frac{\gamma}{2\pi}} \exp\left[-\frac{\gamma}{2(s-\mu)}\right] \frac{1}{(s-\mu)^{\frac{1}{\beta}}}, & 0 < \mu < s < \infty \\ 0, & \text{otherwise} \end{cases} \quad (5)$$

where $\mu > 0$ is the minimum step size and γ is the scale parameter, when $S \rightarrow \infty$ the above equation can be written as

$$L(s, \gamma, \mu) = \sqrt{\frac{\gamma}{2\pi}} \frac{1}{s^{\frac{1}{\beta}}} \quad (6)$$

It is common to approximate $L(s)$ as

$$L(s) \rightarrow \frac{\alpha\beta\Gamma(\beta)\sin\left(\frac{\pi\beta}{2}\right)}{\pi|s|^{1+\beta}}, s \rightarrow \infty \quad (7)$$

where, Γ is the gamma function.

Usually we use the Mantegna algorithm [50, 51] to simulate its flight pattern. The Mantegna algorithm defines a random step size s as follows.

$$s = \frac{\mu}{|v|^{\frac{1}{\beta}}} \quad (8)$$

μ, v obeying the standard normal distribution where $\mu \sim N(0, \sigma_\mu^2), v \sim N(0, \sigma_v^2), \sigma_\mu$ and σ_v are calculated as follows.

$$\sigma_\mu = \left[\frac{\Gamma(1+\beta)\sin\left(\frac{\pi\beta}{2}\right)}{\Gamma\left(\frac{1+\beta}{2}\right)\beta 2^{\frac{\beta-1}{2}}}\right]^{\frac{1}{\beta}}, \sigma_v = 1 \quad (9)$$

Our work distinguishes itself from the referenced literature in several critical ways. We propose a clustering-based recommendation method that integrates an enhanced GOA, referred to as LCGOA. This novel approach marries K-means algorithm with GOA to optimise cluster centroids uniquely for conventional collaborative filtering recommendations. The incorporation of Lévy flight mechanism and multi-strategy co-evolution into the GOA significantly enhances the search accuracy and stability, thereby addressing the core issues of data sparsity and algorithm scalability more effectively than previous methods. Furthermore, our approach utilises user clustering to dramatically improve recommendation performance, a technique that has not been sufficiently explored in the existing literature. Our empirical tests on real-world datasets like MovieLens and Amazon showcase the superior accuracy of our method, achieving significant improvements over traditional collaborative filtering algorithms.

In summary, our contribution lies in developing a more accurate and scalable clustering-based recommendation system by innovatively applying swarm intelligence optimisation to address long-standing issues in the field.

3 | METHOD

Clustering can be formulated as a multi-objective optimisation problem, where the goal is to divide a set of data points into multiple groups such that the points within each group are more similar to each other and less similar to points in other groups. Swarm intelligence methods, such as GOA, have proven to be effective in solving optimisation problems in various fields, including data clustering. Grasshopper Optimisation Algorithm is a meta-heuristic swarm intelligence optimisation algorithm that simulates the behaviour of grasshopper populations in nature using mathematical models. In this chapter, we use the Lévy flight mechanism and multi-strategy co-evolution to improve GOA, which enhances search accuracy and provides a diverse set of solutions to the optimisation problem. The improved GOA is then utilised for clustering by optimising the cluster centroids, and finally applied in the recommendation system. The GOA was chosen as the enhanced algorithm for improving the K-means clustering method due to several specific advantages it holds over other swarm intelligence algorithms, such as exploration and exploitation balance, simulating swarm behaviour, low sensitivity to initial parameters, scalability and efficiency in handling nonlinearities, particularly in addressing the unique challenges of clustering in recommendation systems. These advantages

make GOA particularly suited to enhancing the K-means algorithm for clustering in recommendation systems, where the need to dynamically update clusters based on evolving user preferences and data environments is critical. The integration of GOA not only improves the accuracy and efficiency of clustering but also enhances the overall recommendation process by providing more relevant and personalised suggestions to users.

3.1 | Grasshopper optimisation algorithm based on Lévy flight

In this paper, we propose LGOA, by integrating the Lévy flight perturbation mechanism with GOA. The integration of Lévy flight enhances the algorithm's exploration and exploitation capabilities. The basic principle of LGOA is to first use the position update Equation (10) of GOA to update the positions of individual grasshoppers in one iteration of the optimisation search. Then, the individual grasshopper positions are disturbed using the Lévy flight unspecified selection to improve GOA's optimisation performance and avoid getting stuck in local optimum solutions.

$$x_{i+1}^d = c \left\{ \sum_{j=1, j \neq i}^N c \frac{ub_d - lb_d}{2} s(|x_j^d - x_i^d|) \frac{x_j - x_i}{d_{ij}} \right\} + \hat{T}_d \quad (10)$$

where d represents the dimension, ub_d and lb_d are the upper and lower limits of the d dimensional search space, and \hat{T}_d is the position of the current optimum individual in the d dimension.

The parameter c reduces the comfort zone proportional to the number of iterations and is calculated as follows [9]:

$$c = c_{\max} - l \frac{c_{\max} - c_{\min}}{L} \quad (11)$$

where c_{\max} is the maximum value of the adaptive coefficients, c_{\min} is the minimum value, usually taken as 1 and 0.00,001, l is the number of current iterations, and L is the maximum number of iterations.

The pseudo code for LGOA is shown below.

Algorithm 1 LGOA

1. Initialise a population of grasshoppers, X_i ($i = 1, 2, \dots, N$)
2. Initialise the number of populations N , the maximum number of iterations L , c_{\max} and c_{\min}
3. Calculate the fitness value for each search agent
4. Define T as the optimum search agent
5. **while** ($l < L$) **do**
6. Update the parameter c by Equation (11)

7. **for** (each search agent) **do**
 8. Use Equation (10) to update the current location of the search agent
 9. Calculate the current fitness value of the grasshopper based on the objective function and the current position
 10. Use Equation (4) to update the search agent optimum position
 11. The fitness value of each grasshopper is calculated again and the target value is updated according to the greedy law of merit
 12. **end for**
 13. Merit Update T
 14. **if** ($l < L$) **then**
 15. $l = l + 1$
 16. **end if**
 17. **end while**
 18. **return** T
-

3.2 | Multi-strategy and co-evolution Grasshopper Optimisation Algorithm

When using GOA to handle complex multi-modal test functions, it is common to obtain only one or two global optimum. This is because the population employed by GOA may not be able to locate all of the optimum solutions during the search. To address this issue, this paper proposes a co-evolution strategy [52] that involves dividing the initial population into multiple sub-populations that evolve independently. The grasshopper population is viewed as a subsystem of multiple populations that compete with each other for resources. The co-evolution strategy promotes evolution by promoting competition, constraints, coordination and utilisation among sub-populations. The optimal solutions are eventually shared between sub-populations to improve the balance between the populations' ability to explore both locally and globally.

The proposed method involves dividing the population of grasshoppers into several sub-populations, with each sub-population assumed to contain a single species. In this paper, three common adaptive coefficients of GOA species are selected c adaptation methods [53, 54], respectively, as follows.

$$c_1 = c_{\max} - l \frac{c_{\max} - c_{\min}}{L} \quad (12)$$

$$c_2 = \left(\frac{c_{\max} - l}{L} \right)^2 \quad (13)$$

$$c_3 = \left(\cos \left(\frac{\pi l}{L} \right) + c_{\max} \right) (c_{\max} + c_{\min}) \quad (14)$$

where, c_1 , c_2 and c_3 represent Linear, Arc and Cosine adaption respectively, c_{\max} is the maximum value of the adaptive

coefficients, c_{\min} is the minimum value, l is the number of current iterations and L is the maximum number of iterations.

The optimal parameter choice for each sub-population c may be affected by important parameters set in GOA and the magnitude of the problem being solved. To address this, we employ two strategies in this paper, namely the fixed division strategy and the random assignment strategy [55]. These strategies aim to choose the appropriate parameters for each sub-population.

The fixed division strategy involves giving each sub-population a fixed parameter c , as shown below, by allocating different parameters c to various sub-populations.

$$c_{S_j} = c_j, j \in 1, 2, \dots, n \quad (15)$$

where S_j represents the first i sub-population, c_j is the choice of one of the three adaptive coefficients c mentioned above, and n is the number of sub-population divisions. Before starting the search process, we fix a parameter c_j for each sub-population S_j , which remains constant throughout the iterative process of algorithm optimisation.

In each iteration, the random assignment strategy randomly chooses a parameter c for each sub-population. The random assignment method can be represented mathematically as:

$$c_{S_i} = \begin{cases} c_1, & \text{if rand}(1, n) = 1 \\ c_2, & \text{if rand}(1, n) = 2 \\ \dots & \\ c_n & \text{if rand}(1, n) = n \end{cases} \quad (16)$$

where $\text{rand}(1, n)$ is the random number generated by $1, 2, \dots, n$. In this way, each sub-population in each iteration has an equal chance of choosing any of the parameters c . This will successfully prevent the algorithm from entering a local optimum while also increasing the diversity of the populations.

To further enhance LGOA, we introduce multi-strategy co-evolution and propose a new algorithm, LCGOA. The LCGOA divides the grasshopper population into several sub-populations based on a population partitioning strategy. The parameters c are assigned to each sub-population using both the fixed division and the random assignment strategy. The algorithm updates the global optimum solution based on the optimum results obtained by each sub-population and the corresponding parameter c , and continuously updates the positions of each grasshopper until the maximum global optimum solution is reached. The pseudo code of LCGOA is presented below (see Algorithm 2).

Algorithm 2 LCGOA

1. Initialise a population of grasshoppers, X_i ($i = 1, 2, \dots, N$)
2. Initialise the number of populations N , the maximum number of iterations L , the number of sub-populations n , c_{\max} and c_{\min}
3. Dividing the population into n sub-populations, S_i ($i = 1, 2, \dots, n$)

4. Calculate the initial fitness value for each sub-population
 5. Define [*TargetFitness*, *TargetPosition*] as the initial optimum solution and its optimum position among all sub-populations
 6. **while** ($l < L$) **do**
 7. Update the evolution direction of the whole population by *TargetPosition*
 8. **for** (each sub-population S_i) **do**
 9. Use Equation (15) or (16) to select parameters c for each sub-population
 10. Update grasshopper position and target value using Algorithm 1
 11. **end for**
 12. Update [*TargetFitness*, *TargetPosition*]
 13. **if** ($l < L$) **then**
 14. $l = l + 1$
 15. **end if**
 16. **end while**
 17. **return** the final optimum [*TargetFitness*, *TargetPosition*]
-

3.3 | LCGOA clustering-based recommendation

This section proposes a user clustering recommendation algorithm with optimised clustering centroids that addresses the challenges of data sparsity and algorithm scalability in collaborative filtering recommendation. The proposed algorithm enhances the recommendation effect of the collaborative filtering algorithm by predicting unrated items in the data matrix using the WSO algorithm and pre-processing the initial data matrix to reduce its sparsity. The LCGOA clustering method is then used to cluster the populated rating data, which reduces the search space for the closest neighbours of the user and increases the scalability of the algorithm. Finally, the conventional recommendation algorithm is used to generate the final recommendation for the target user's class.

Clustering is a crucial process in recommendation algorithms that can improve the accuracy of recommendations for users in the same class cluster. It involves dividing a set of data into distinct groups of features. K-means is one of the 10 most popular clustering algorithms [29].

It is known that there are K centroids (u_1, u_2, \dots, u_K) and m data points (x_1, x_2, \dots, x_m) to be classified, each point is recorded with a vector r of the same dimensionality to which class it belongs. The objective function of the K-means algorithm is as follows.

$$\begin{aligned} & \min J(u_1, \dots, u_K, r_1, \dots, r_m) \\ & = \min \sum_{i=1}^m r_i^T \left[(x_i - u_1)^T \cdot (x_i - u_1), \dots, (x_i - u_K)^T \cdot (x_i - u_K) \right] \end{aligned} \quad (17)$$

In the K-means algorithm, optimising the objective function is typically achieved in two steps. First, assuming the coordinates u of the centroid are known, the algorithm solves for each category r . Second, assuming the category r of each point is known, the algorithm solves for the coordinates u of each centroid.

To improve the search efficiency of the K-means algorithm and maintain population diversity, we combine LCGOA with K-means. This algorithm utilises the balance between exploration and exploitation in LCGOA to optimise and search for the best cluster centroid. This process involves the following key steps [56]:

Step 1: Initialisation of cluster centroids: Initially, the population of possible solutions (i.e., cluster centroids) is generated randomly. Each solution represents a set of K centroids for the K-means algorithm.

Step 2: Evaluation of solutions: The quality of each solution is evaluated based on the objective function of the K-means algorithm, which typically involves minimising the within-cluster sum of squares.

Step 3: Optimisation with LCGOA: The LCGOA algorithm is applied to iteratively improve the solutions.

Step 4: Update of cluster centroids: Based on the outcomes of the LCGOA optimisation, the positions of the grasshoppers (solutions) are updated, which corresponds to the updating of the cluster centroids in the K-means algorithm.

Step 5: Convergence check: The algorithm checks for convergence based on a predefined criterion (i.e., whether the maximum number of iterations has been reached). If the convergence criterion is met, the algorithm terminates; otherwise, it goes back to **Step 2** for further iterations.

Step 6: Final cluster centroids: Once the algorithm converges, the final positions of the grasshoppers represent the optimised cluster centroids. These centroids are then used in the K-means algorithm to assign each data point to the nearest centroid, resulting in the final clustering of the data.

This integrated approach, combining K-means with LCGOA, aims to optimise cluster centroids more effectively than the standard K-means algorithm alone, thereby enhancing the accuracy of clustering-based recommendation methods by addressing the issue of centroid sensitivity. The pseudo code of the algorithm is presented below (see Algorithm 3).

Algorithm 3 LCGOA Clustering

1. Initialise the population X_i ($i = 1, 2, \dots, N$)
2. Initialise c_{\max} , c_{\min} , L , number of clustering k and clustering centroid
3. Coding the k clustering centres to get

- the initial position of grasshoppers
 4. Calculate the cluster centroids for each category and encode them as grasshopper locations
 5. Use Equation (17) to evaluate the location of grasshoppers and retain their optimal location
 6. Update LCGOA parameter c and grasshopper location
 7. **while** ($l < L$) **do**
 8. LCGOA updates parameter and optimises the best solution
 9. **end while**
 10. The new grasshopper positions are reverse coded to obtain k clustering centroids
 11. Using the best clustering centroid to clustering
-

Let the set of users be $U = \{u_1, u_2, \dots, u_m\}$, and the set of users generated using LCGOA clustering is denoted as $U' = \{C_1, C_2, \dots, C_i, \dots, C_K\}$. Where, K is the number of clusters and C_i denotes the first i class. The steps of LCGOA clustering-based recommendation are as follows.

Step 1: Input user u , user-rating matrix $R_{m \times n}$ and initialise the number of clusters K .

Step 2: Eliminate the 0 elements in the matrix $R_{m \times n}$ using Equation (3) to obtain the matrix $R'_{m \times n}$.

Step 3: Use $X_i (i \in 1, 2, \dots, K)$ as the initial clustering centroid and divide the data in $R_{m \times n}$ into K classes by LCGOA clustering.

Step 4: Calculate the similarity between the clustering centroids of u and K using the Equation (18), and add u to the class with the highest similarity to it.

$$d(u_i, X_i) = \sqrt{(u_i - X_i)^T (u_i - X_i)} \quad (18)$$

Step 5: Calculate the similarity between u and other users in the class, and derive its nearest neighbour set $N_{uj} (j = 1, 2, \dots, m)$. The similarity calculation for user u and v is defined as follows:

$$\text{sim}(u, v) = \frac{\sum_{i \in I_{uv}} (r_{ui} - \bar{r}_u)(r_{vi} - \bar{r}_v)}{\sqrt{\sum_{i \in I_{uv}} (r_{ui} - \bar{r}_u)^2} \sqrt{\sum_{i \in I_{uv}} (r_{vi} - \bar{r}_v)^2}} \quad (19)$$

where I_{uv} denotes the set of items with common ratings of users u and v , r_{ui} and r_{vi} denote the ratings of item i by users u and v , respectively. \bar{r}_u and \bar{r}_v are the average ratings of users u and v on I_{uv} , respectively.

Step 6: Based on the ratings of items by similar nearest neighbours, use the Equation (20) to find the predicted ratings of recommended items by users u and rank them, and recommend the top N items to u .

$$P_{ui} = \frac{\sum_{n \in N_u} (r_{ni} - \bar{r}_n) \text{sim}(u, n)}{\sum_{n \in N_u} |\text{sim}(u, n)|} + \bar{r}_u \quad (20)$$

where, \bar{r}_u and \bar{r}_n denote the average ratings of user u and n , N_u is the set of nearest neighbour for user u , r_{ni} is the ratings of items i by user n , and $\text{sim}(u, n)$ is the similarity between user u and n .

In this paper, we enhance the standard GOA by integrating a Lévy flight mechanism and introducing a multi-strategy co-evolution approach. The Lévy flight mechanism diversifies the search patterns of the algorithm, enabling it to escape local optima and explore the search space more thoroughly with variable step sizes. This not only broadens the exploration capabilities but also significantly reduces the search complexity by avoiding exhaustive searches in non-promising regions. Furthermore, the multi-strategy co-evolution divisions the overall population into sub-populations, each adopting a different strategy for search space exploration. This division allows for a more focused search within each segment of the solution space, effectively decreasing the search complexity by reducing redundant search efforts and enhancing the convergence speed.

4 | EXPERIMENTS

4.1 | Datasets

To compare the performance metrics of the recommendation algorithm based on LCGOA clustering with traditional collaborative filtering recommendation and to verify its efficacy, we conducted comparison experiments using the Movielens and Amazon datasets. The datasets are described below.

- (1) Movielens movie dataset ml-100k [30]: This dataset consists of 100,000 ratings (1–5) from 943 users on 1682 movies, each user has rated at least 20 movies, and simple demographic info for the users (age, gender, occupation, zip). The sparsity level of the user rating set is 0.9370. The data was collected through the MovieLens web site (movielens.umn.edu) during the 7-month period from September 19th, 1997 through April 22nd, 1998. Users were selected at random for inclusion.
- (2) Amazon product reviews and metadata [31]: This dataset includes reviews (ratings, text), product metadata (description, category information, price, brand and image features) and links. In this case, the identity of the user has been anonymised and replaced by userID, and the products correspond by ID number. The rows in the file are first sorted by userID, and then by productID in user. Rating represents the user's rating of them item, and the rating takes values from 1 to 5, with 1 representing the lowest and 5 being the highest.

4.2 | Recommendation performance comparison

To conduct experiments, we used a five-fold cross-validation method to sample the dataset. The dataset was evenly divided into five parts, and each of the five parts was used for validation in different experiments. Each experiment was repeated five times under the same conditions, and the results of the five experiments were averaged for analysis. The Mean Absolute Error (MAE), Root Mean Square Error (RMSE), Precision and Recall were used as evaluation metrics in this paper. Mean Absolute Error was calculated as the magnitude of error between the true and predicted ratings of items for users. Root Mean Square Error was evaluated by dividing the dataset into training and testing sets and calculating the error between them to assess the quality of the recommendation results. Precision is the ratio of the number of items predicted by the recommendation method to the total number of recommended items of interest to the user, while Recall is the ratio of the number of items predicted by the recommendation method to the total number of items of interest to the user. The specific equation for these evaluation metrics are as follows.

$$\text{MAE} = \frac{\sum_{u,i \in T} |r_{ui} - \hat{r}_{ui}|}{|T|} \quad (21)$$

$$\text{RMSE} = \frac{\sqrt{\sum_{u,i \in T} (r_{ui} - \hat{r}_{ui})^2}}{|T|} \quad (22)$$

$$\text{Precision} = \frac{\sum_{u \in U} |R(u) \cap T(u)|}{\sum_{u \in U} |R(u)|} \quad (23)$$

$$\text{Recall} = \frac{\sum_{u \in U} |R(u) \cup T(u)|}{\sum_{u \in U} |T(u)|} \quad (24)$$

where \hat{r}_{ui} represents the predicted ratings of the user u on item i , and r_{ui} is the true rating. The lowerer the MAE and RMSE value, the better the recommendation. $R(u)$ is the list of recommendations made to the user based on the user's behaviour on the training set, while $T(u)$ is the list of the user's behaviour on the test set. The higher Precision and Recall value, the better the recommendation.

To address the issue of sparsity in the user-item rating matrix of traditional collaborative filtering recommendation, this paper proposes using LCGOA to first cluster users, thereby reducing the search space and difficulty of finding the nearest neighbours, and improving the accuracy of recommendations. By clustering users, it becomes easier to identify the real neighbours and form the final recommendation list.

To compare the performance of the three algorithms, we conducted experiments on two datasets and evaluated the MAE and RMSE. The algorithms tested were the traditional user-based collaborative filtering algorithm (UB-CF), the K-means-based collaborative filtering algorithm (K-means-CF), and the LCGOA clustering-based collaborative filtering

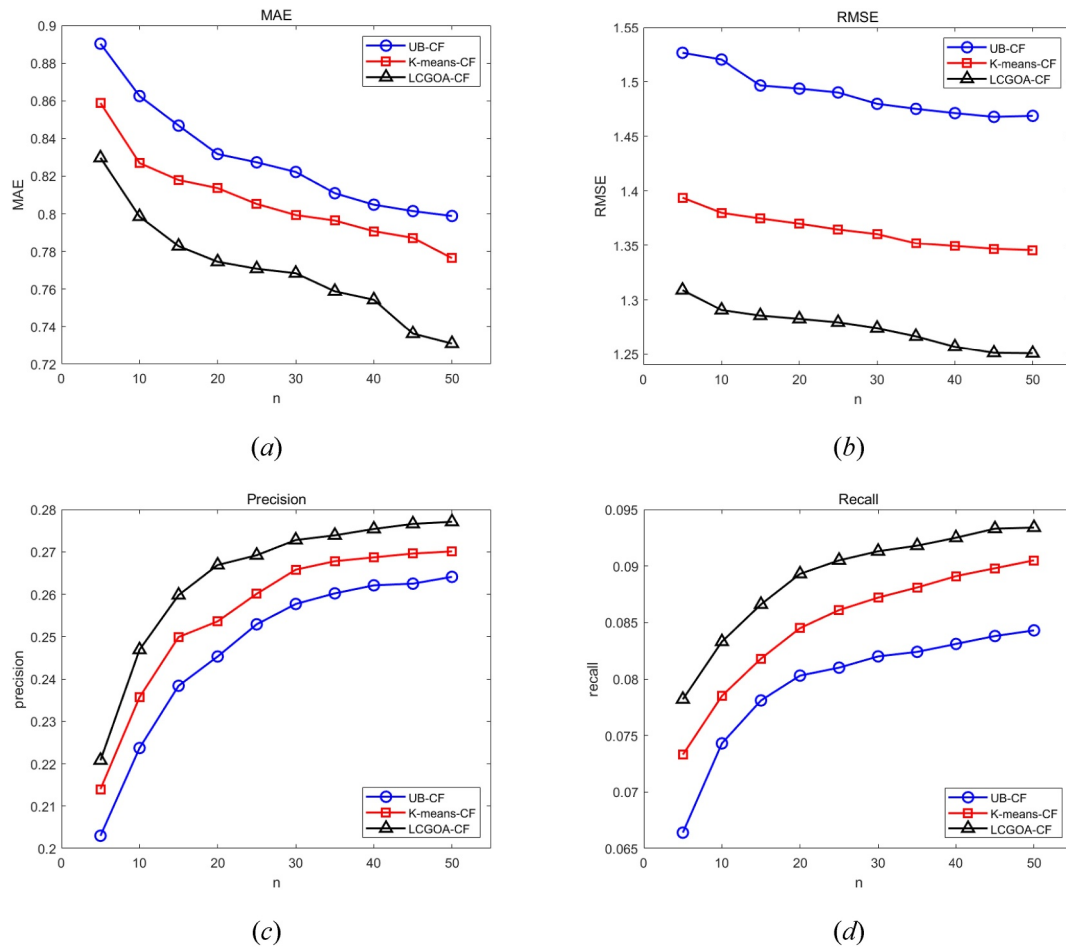


FIGURE 1 Comparison of results with different numbers of nearest neighbours on Movielens dataset.

algorithm (LCGOA-clustering-CF). The experimental results are presented in Figures 1 and 2, where the vertical axis represents the corresponding score prediction index values for the three algorithms, and the horizontal axis represents the number of chosen nearest neighbours n , ranging from 5 to 50 in increments of 5.

On the Movielens dataset, the overall trend of the Precision and Recall of the three collaborative filtering recommendation algorithms increases with the increase of the number of nearest neighbours n value, and then gradually plateaus. The LCGOA clustering-based recommendation algorithm we proposed has the highest Precision and Recall values, which indicates that clustering users at an early stage can effectively improve the recommendation accuracy. The MAE and RMSE values of all three collaborative filtering recommendation algorithms decrease as the number of nearest neighbours of the target user increases, indicating that the accuracy of the recommendation can be effectively improved by increasing the number of nearest neighbours. Our proposed method achieved the lowest MAE and RMSE values for different numbers of neighbours, demonstrating that the algorithm reduces data sparsity by filling the initial data and clustering user data, which makes the nearest neighbour search

range more objective and yields more accurate user neighbours before performing traditional recommendation. The experimental results show that the LCGOA clustering-based recommendation algorithm proposed in this paper outperforms the traditional collaborative filtering recommendation algorithms and has better performance. Similarly, on the Amazon dataset, the LCGOA clustering-based recommendation algorithm has the lowest MAE and RMSE values and the highest Precision and Recall values, indicating the best recommendation performance.

4.3 | More validation experiments

In this section, through a series of test functions with different characteristics performance comparison of GOA improvement strategies (see Table 1 for test functions information [57]).

Where functions F1–F3 are unimodal test functions, which have only one global optimum; F4–F5 are multi-modal test functions with several local optimum solutions (we set the dimension of F1–F5 to 30); F6–F8 are fixed dimensional multi-modal test function [57]. The settings of the key parameters are shown in Table 2.

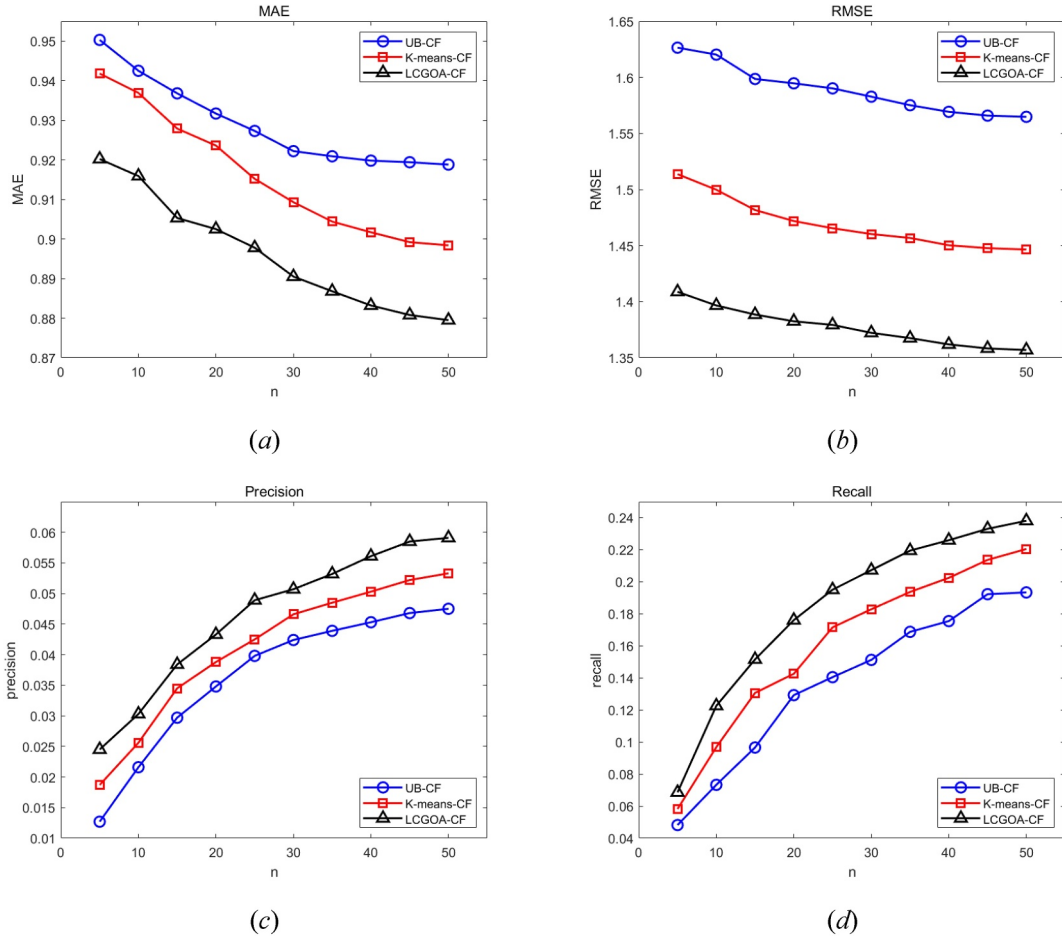


FIGURE 2 Comparison of results with different numbers of nearest neighbours on Amazon dataset.

TABLE 1 Different test functions information.

Function	Name	Dim	Range	F_{\min}
$F_1(x) = \sum_{i=1}^n x_i^2$	Sphere	30	$[-100,100]$	0
$F_2(x) = \sum_{i=1}^n \left(\sum_{j=1}^i x_j \right)^2$	Schwefel	30	$[-100,100]$	0
$F_3(x) = \sum_{i=1}^n i x_i^4 + \text{random}$	Quartic	30	$[-1.28,1.28]$	0
$F_4(x) = -20 \exp\left(-0.2 \sqrt{\frac{1}{n} \sum_{i=1}^n x_i^2}\right) - \exp\left(\frac{1}{n} \sqrt{\sum_{i=1}^n \cos(2\pi x_i)}\right) + 20 + e$	Ackley	30	$[-32,32]$	0
$F_5(x) = \sin^2(\pi \omega_1) + \sum_{i=1}^{n-1} (\omega_i - 1)^2 (1 + 10 \sin^2(\pi \omega_i + 1)) + (\omega_n - 1)^2 (1 + \sin^2(2\pi \omega_n))$	Lévy	30	$[-10,10]$	0
$F_6(x) = \left(\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{\sum_{i=1}^2 (x_i - a_{ij})^6} \right)^{-1}$	Shekel Foxholes	2	$[-65.536, 65.536]$	≈ 1
$F_7(x) = \sum_{i=1}^{11} \left(a_i - \frac{x_1 (b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4} \right)^2$	Kowalik	4	$[-5,5]$	≈ 0.0003075
$F_8(x) = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2$	Booth	2	$[-10,10]$	0

4.3.1 | Lévy flight validity experiments

In this section, we compare the efficiency of LGOA and GOA to demonstrate the validity of the Lévy flight mechanism perturbation. Table 3 presents a comparison of the optimum, mean and standard deviation values obtained from running GOA and LGOA on test functions.

Table 3 shows that LGOA performs better than GOA in terms of optimum values, mean values and standard deviation. While the multi-modal test function has local optimum solutions and is prone to settling towards the local optimum during

TABLE 2 Key parameter settings.

Parameter name	Parameter setting
N (population size)	150
L (maximum number of iterations)	1000
c_{\max}	1
c_{\min}	0.00001
n (number of sub-populations)	3

TABLE 3 Grasshopper Optimisation Algorithm (GOA) and LGOA comparison results.

Test functions	Optimum		Mean		Standard deviation	
	GOA	LGOA	GOA	LGOA	GOA	LGOA
F1	1.6911×10^{-2}	1.9465×10^{-20}	0.21317	1.8968×10^{-16}	0.18535	2.7914×10^{-16}
F2	3.5099×10^2	1.1163×10^{-19}	1.1265×10^3	8.471×10^{-16}	1.543×10^3	9.502×10^{-16}
F3	6.1847×10^{-3}	5.6738×10^{-6}	1.0411×10^{-2}	4.3813×10^{-5}	3.9879×10^{-3}	3.5779×10^{-5}
F4	1.7793	3.2885×10^{-11}	2.6528	2.5086×10^{-9}	0.86197	2.2116×10^{-9}
F5	2.1755	0.2689	6.8258	0.58325	3.5575	0.2217
F6	4.369×10^{-14}	1.4577×10^{-14}	1.8547×10^{-13}	1.3952×10^{-13}	1.1959×10^{-13}	7.8204×10^{-14}
F7	0.998	0.998	0.998	0.998	2.3984×10^{-16}	2.1579×10^{-16}
F8	5.5947×10^{-4}	3.0749×10^{-4}	6.7683×10^{-3}	3.0767×10^{-4}	9.3864×10^{-3}	1.5488×10^{-7}

Note: Bold values indicate that the algorithm obtained optimal results in comparison with the other algorithms.

TABLE 4 Comparison of optimum values of different improvement strategies.

Test functions	LGOA-1	LGOA-2	LGOA-3	LGOA-F	LGOA-R
F1	1.9842×10^{-20}	1.7447×10^{-28}	8.8261×10^{-28}	3.0129×10^{-29}	1.7679×10^{-30}
F2	1.5685×10^{-19}	1.9225×10^{-28}	4.2286×10^{-27}	1.6026×10^{-28}	1.4211×10^{-29}
F3	4.3859×10^{-6}	7.0432×10^{-7}	1.0223×10^{-5}	5.7634×10^{-6}	3.4301×10^{-7}
F4	2.9982×10^{-11}	4.4409×10^{-15}	7.9936×10^{-15}	4.4409×10^{-15}	8.8818×10^{-16}
F5	0.2689	0.71966	0.45206	0.17913	0.090294
F6	0.998	0.998	0.998	0.998	0.998
F7	3.075×10^{-4}	3.0826×10^{-4}	3.0778×10^{-4}	3.075×10^{-4}	3.0749×10^{-4}
F8	4.6417×10^{-15}	3.9794×10^{-9}	5.4892×10^{-18}	3.3267×10^{-13}	3.6424×10^{-18}

Note: Bold values indicate that the algorithm obtained optimal results in comparison with the other algorithms.

optimisation, the unimodal test function only has one global optimum, making LGOA substantially better than the original GOA. The experimental results indicate that LGOA has a better ability to jump out of local optimum and a stronger global search capability. Moreover, the standard deviation comparison demonstrates that LGOA has greater stability. The advantages of the Lévy flight mechanism are clearly demonstrated by the experimental results.

4.3.2 | Comparison experiments of different improvement strategies

In this section, we compare the performance of LGOA with three different parameter c selection forms and two different multiple-group co-evolution strategies, namely fixed division (LGOA-F) and random assignment (LGOA-R), using Linear, Arc and Cosine adaptive methods. Table 4 presents a comparison of the optimum and mean values obtained by applying various strategies to a test function. Similarly, Table 5 shows the comparison of the optimum and mean values obtained by applying different c selection forms and co-evolution strategies to the same test function.

Analysing Tables 4 and 5, it can be observed that LGOA-F and LGOA-R generally perform better than the original LGOA in terms of optimum search accuracy. LGOA-R achieves the best optimum values among all 8 test functions, while LGOA-F outperforms LGOA-1, 2 and 3 in some test functions. Similarly, in terms of mean values, LGOA-R outperforms all other strategies on all 8 test functions, while LGOA-F also performs better than LGOA-1, 2 and 3 in some test functions. Figure 3 demonstrates the average convergence curve for the test functions, where the fixed division strategy (LGOA-F) is represented by the red curve, and the random assignment strategy (LGOA-R) is represented by the black curve.

The experimental results demonstrate that LGOA-F and LGOA-R improve the algorithm's optimisation-seeking ability by enabling information sharing among sub-populations, while also ensuring efficient mining and the ability to jump out of local optimum. These benefits are achieved through the multi-strategy co-evolution strategy and are evident in both the unimodal test function with a single global optimum and the multi-modal test function with locally optimum solutions. Comparing LGOA-F and LGOA-R, it is obvious that LGOA-R has better performance.

4.3.3 | Comparison with other swarm intelligence algorithms

In this section, we combine the previously improved LCGOA (LGOA-R) with the multiple swarm co-evolution based GOA (GOA-MC) [55] and several other newer swarm intelligence optimisation algorithms, such as GWO [43], PSO [41] and SSA [44]. Tables 6 and 7 show the comparison of the optimum values and mean values obtained by various swarm intelligence algorithms run on the test functions.

From Tables 6 and 7, it is easy to see that in the comparison with other swarm intelligence algorithms, LCGOA does not perform optimally only on F1, and the rest of the test functions reach the optimal optimum and mean values, which indicates that the algorithm has a strong search capability and high overall search accuracy. The distinctive swarming behaviour simulation, balance between exploration and exploitation, simplicity, scalability and ease of enhancement make GOA a compelling choice for solving complex optimisation problems, such as those encountered in clustering-based recommendation systems. In addition, the LCGOA was overall superior in the comparison with the GOA-MC, which further demonstrates the advantages of introducing the Lévy flight mechanism.

TABLE 5 Comparison of mean values of different improvement strategies.

Test functions	LGOA-1	LGOA-2	LGOA-3	LGOA-F	LGOA-R
F1	1.2229×10^{-16}	1.8531×10^{-24}	2.0823×10^{-24}	4.842×10^{-25}	3.478×10^{-26}
F2	5.7938×10^{-16}	1.032×10^{-23}	9.7245×10^{-24}	6.889×10^{-24}	6.6172×10^{-25}
F3	8.4892×10^{-6}	1.3799×10^{-5}	3.5214×10^{-5}	6.3158×10^{-6}	4.3859×10^{-6}
F4	1.9501×10^{-9}	1.7994×10^{-13}	1.6396×10^{-13}	1.0498×10^{-13}	2.8955×10^{-14}
F5	0.58367	1.2397	0.68113	0.49807	0.42271
F6	0.99801	0.99801	0.99801	0.998	0.998
F7	3.0874×10^{-4}	4.591×10^{-4}	3.1781×10^{-4}	3.0826×10^{-4}	3.0787×10^{-4}
F8	1.4342×10^{-13}	1.2064×10^{-6}	1.2225×10^{-14}	6.3243×10^{-12}	2.2401×10^{-15}

Note: Bold values indicate that the algorithm obtained optimal results in comparison with the other algorithms.

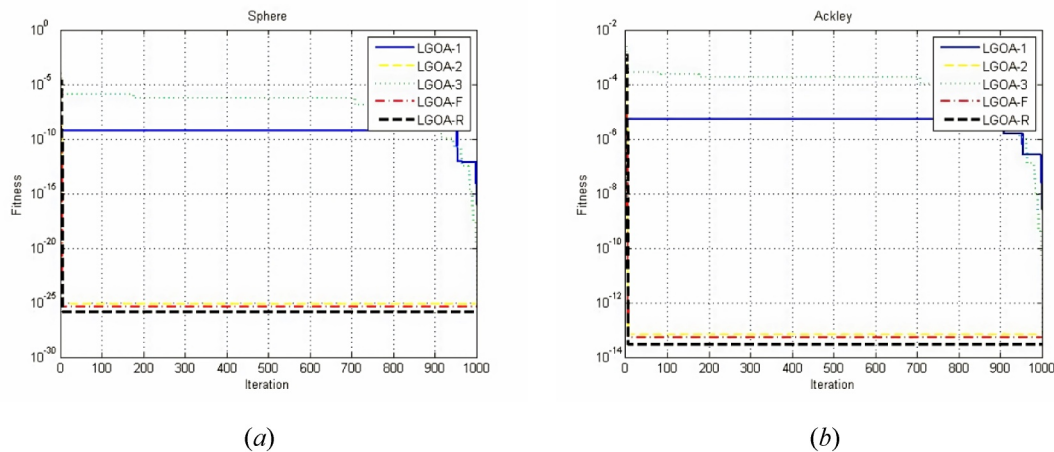


FIGURE 3 The average convergence curve for the test functions.

TABLE 6 Comparison of optimum values.

Test functions	LCGOA	GOA-MC	GWO	PSO	SSA
F1	1.6529×10^{-30}	9.6585×10^{-3}	1.8137×10^{-82}	1.2549×10^{-5}	5.4129×10^{-9}
F2	1.3617×10^{-29}	56.6844	1.8212×10^{-18}	0.1753	2.058
F3	3.4843×10^{-7}	1.0233×10^{-2}	2.2688×10^{-4}	4.0396×10^{-2}	2.6341×10^{-2}
F4	8.8818×10^{-16}	2.0433	8.8818×10^{-16}	9.7236×10^{-2}	2.2819×10^{-5}
F5	0.090294	0.2954	0.36126	0.29241	1.1773
F6	0.998	0.998	0.998	4.2225	0.998
F7	3.0749×10^{-4}	3.0787×10^{-4}	3.075×10^{-4}	3.215×10^{-4}	3.078×10^{-4}
F8	4.0287×10^{-20}	1.3856×10^{-14}	1.1203×10^{-10}	8.5212×10^{-9}	2.9286×10^{-16}

Note: Bold values indicate that the algorithm obtained optimal results in comparison with the other algorithms.

TABLE 7 Comparison of mean values.

Test functions	LCGOA	GOA-MC	GWO	PSO	SSA
F1	4.1634×10^{-26}	3.2793×10^{-2}	8.0914×10^{-81}	8.65×10^{-3}	8.2659×10^{-9}
F2	7.8189×10^{-25}	1.0904×10^2	3.2622×10^{-16}	0.50163	18.0533
F3	5.6802×10^{-6}	1.569×10^{-2}	7.6697×10^{-4}	7.7025×10^{-2}	4.5282×10^{-2}
F4	8.4572×10^{-14}	4.1584	3.7303×10^{-13}	0.61736	1.5106
F5	0.42271	5.0778	0.66244	1.2187	5.001
F6	0.998	1.0974	1.1964	10.8219	0.998
F7	3.079×10^{-4}	7.5382×10^{-4}	4.4102×10^{-4}	5.8638×10^{-4}	7.3808×10^{-4}
F8	2.9657×10^{-15}	3.1243×10^{-11}	1.306×10^{-8}	5.389×10^{-8}	7.3294×10^{-15}

Note: Bold values indicate that the algorithm obtained optimal results in comparison with the other algorithms.

5 | CONCLUSION

In this paper, we propose a collaborative filtering recommendation algorithm based on a swarm intelligence algorithm for improved clustering to address the issues of data sparsity and algorithm scalability. Our experiments demonstrate the effectiveness of LCGOA clustering in maximising recommendation accuracy. The specific steps taken in this work are as follows:

- This paper aimed to address the limitations of traditional recommendation systems by improving recommendation accuracy and efficiency. Specifically, the research sought to enhance the clustering process in collaborative filtering by integrating the K-means algorithm with an optimised GOA, namely LCGOA, to dynamically determine optimal cluster centroids.
- The research methodology involved a novel application of LCGOA to optimise the selection of cluster centroids in the K-means algorithm, a popular method used in collaborative filtering recommendation systems. The research detailed the process of integrating LCGOA with K-means, emphasising the iterative optimisation of centroids to minimise intra-cluster variance and enhance recommendation relevance. Comparative experiments were conducted against baseline

recommendation algorithms across multiple datasets, utilising a comprehensive set of evaluation metrics.

- The experimental results demonstrated significant improvements in recommendation accuracy and efficiency with the LCGOA-enhanced K-means method. Notably, the method showed a notable reduction in MAE and RMSE, alongside gains in Precision and Recall when compared to traditional approaches. These findings indicate that the LCGOA integration effectively addresses the challenges of initial centroid selection and local optima, common in standard K-means applications.
- From a theoretical perspective, this paper contributes to the literature on swarm intelligence applications in recommendation systems, providing a novel integration strategy that enhances clustering algorithms. Practically, the improved recommendation accuracy and efficiency translate into tangible benefits for both users and businesses, including enhanced personalisation, user satisfaction and operational scalability. This enhanced accuracy ensures users receive more relevant content recommendations, thereby increasing engagement and loyalty, while businesses benefit from optimised content delivery and resource utilisation, leading to better customer retention and potential revenue growth.

- While the experimental results demonstrate the effectiveness of our proposed method, there are still some shortcomings to consider. For instance, the No Free Lunch Theorem [58] highlights the impossibility of developing a universal optimisation technique, which means that we must be problem-specific when addressing optimisation challenges. Furthermore, our current evaluation metrics may not fully reflect the performance of the algorithms. In the future, we should consider incorporating qualitative and quantitative multidimensional evaluation criteria, such as coverage, diversity, surprise and user satisfaction, to make the recommendation results more realistic and trustworthy. Additionally, one promising direction is the real-world application of our method in various industries. For instance, in e-commerce platforms like Amazon and Alibaba, personalised product recommendations are crucial for driving sales and improving user satisfaction. The dynamic nature of user preferences in these platforms makes clustering-based methods like LCGOA particularly relevant, as they can efficiently adapt to changes in customer behaviour and enhance the recommendation process. This will be reflected in our future work.

In conclusion, by addressing the limitations of traditional recommendation algorithms, our proposed LCGOA-based method offers enhanced clustering capabilities, making it particularly useful in dynamic environments where user preferences evolve rapidly. This improvement is particularly relevant for large-scale e-commerce platforms, streaming services, and social media networks, where the accuracy and relevance of recommendations directly impact user engagement and business performance. The findings underscore the potential of leveraging swarm intelligence for data clustering challenges, marking a step forward in the pursuit of more intelligent, adaptive and user-centric recommendation systems.

ACKNOWLEDGEMENTS

We would like to express our gratitude to Yingchun Xia, who contributed equally to this work as a co-first author. Zhao and Xia developed the algorithms and drafted the manuscript. The authors would like to thank the Anhui Agricultural University Application Technology Research Institute of Agricultural Big Data for their helps. This work is supported by the National Natural Science Foundation of China under Grants 32472007, 62306008 and 62301006, the National Key Research and Development Program under Grant 2023YFD1802200, the Key Project of Anhui Province's Science and Technology Innovation Tackle Plan under Grant 202423k09020040, the Natural Science Research Project of Education Department of Anhui Province of China under Grant 2023AH051020, the Natural Science Foundation of Anhui Province under Grant 2308085MF21, the University Synergy Innovation Program of Anhui Province under Grant GXXT-2022-046.

CONFLICT OF INTEREST STATEMENT

The authors declare no conflicts of interest.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are openly available in MovieLens Datasets at <https://grouplens.org/datasets/movielens/>.

ETHICS APPROVAL

No ethics approval were required for the study.

PATIENT CONSENT

Not applicable.

PERMISSION TO REPRODUCE MATERIAL FROM OTHER SOURCES

Not applicable.

CLINICAL TRIAL REGISTRATION

Not applicable.

ORCID

Lichuan Gu  <https://orcid.org/0000-0002-3768-8203>

REFERENCES

1. Xi, L., Hu, Q., Liu, H.: Graph-embedding-inspired article recommendation model. *Expert Syst. Appl.* 214, 119100 (2023). <https://doi.org/10.1016/j.eswa.2022.119100>
2. Phalle, T.S., Bhushan, S.: Content based filtering and collaborative filtering: a comparative study. *J. Adv. Zool.* 45, 96–100 (2024). <https://doi.org/10.53555/jaz.v45is4.4158>
3. Alharbe, N., Rakrouki, M.A., Aljohani, A.: A collaborative filtering recommendation algorithm based on embedding representation. *Expert Syst. Appl.* 215, 119380 (2023). <https://doi.org/10.1016/j.eswa.2022.119380>
4. Aggarwal, C.C., Aggarwal, C.C.: Content-based recommender systems. In: *Recommender Systems: The Textbook*, pp. 139–166 (2016). https://doi.org/10.1007/978-3-319-29659-3_4
5. Hrnjica, B., Music, D., Softic, S.: Model-based recommender systems. In: *Trends in Cloud-Based IoT*, pp. 125–146 (2020)
6. Wang, J.F., et al.: An algorithm of collaborative filtering based on SVD and trust factors. *J. Chin. Comput. Syst.* 38(06), 1290–1293 (2017)
7. Zhang, Y., Xu, H., Yu, X.: The recommendation algorithm based on improved conditional variational autoencoder and constrained probabilistic matrix factorization. *Appl. Sci.* 13(21), 12027 (2023). <https://doi.org/10.3390/app132112027>
8. Wang, G., et al.: An improved constrained Bayesian probabilistic matrix factorization algorithm. *Soft Comput.* 27(9), 5751–5767 (2023). <https://doi.org/10.1007/s00500-022-07799-x>
9. Saremi, S., Mirjalili, S., Lewis, A.: Grasshopper optimisation algorithm: theory and application. *Adv. Eng. Software* 105, 30–47 (2017). <https://doi.org/10.1016/j.advengsoft.2017.01.004>
10. Wang, G.G., Deb, S., Cui, Z.: Monarch butterfly optimization. *Neural Comput. Appl.* 31(7), 1995–2014 (2019). <https://doi.org/10.1007/s00521-015-1923-y>
11. Li, S., et al.: Slime mould algorithm: a new method for stochastic optimization. *Future Generat. Comput. Syst.* 111, 300–323 (2020). <https://doi.org/10.1016/j.future.2020.03.055>
12. Wang, G.G.: Moth search algorithm: a bio-inspired metaheuristic algorithm for global optimization problems. *Memet. Comput.* 10(2), 151–164 (2018). <https://doi.org/10.1007/s12293-016-0212-3>
13. Yang, Y., et al.: Hunger games search: visions, conception, implementation, deep analysis, perspectives, and towards performance shifts. *Expert Syst. Appl.* 177, 114864 (2021). <https://doi.org/10.1016/j.eswa.2021.114864>

14. Tu, J., et al.: The colony predation algorithm. *JBE* 18(3), 674–710 (2021). <https://doi.org/10.1007/s42235-021-0050-y>
15. Ahmadianfar, I., et al.: RUN beyond the metaphor: an efficient optimization algorithm based on Runge Kutta method. *Expert Syst. Appl.* 181, 115079 (2021). <https://doi.org/10.1016/j.eswa.2021.115079>
16. Ahmadianfar, I., et al.: INFO: an efficient optimization algorithm based on weighted mean of vectors. *Expert Syst. Appl.* 195, 116516 (2022). <https://doi.org/10.1016/j.eswa.2022.116516>
17. Su, H., et al.: RIME: a physics-based optimization. *Neurocomputing* 532, 183–214 (2023). <https://doi.org/10.1016/j.neucom.2023.02.010>
18. Thawkar, S., et al.: Breast cancer prediction using a hybrid method based on butterfly optimization algorithm and ant lion optimizer. *Comput. Biol. Med.* 139, 104968 (2021). <https://doi.org/10.1016/j.compbimed.2021.104968>
19. Zhou, L., et al.: Multi-omics graph convolutional networks for digestive system tumour classification and early-late stage diagnosis. *CAAI Trans. Intell. Technol.* 9(6), 1572–1586 (2024). <https://doi.org/10.1049/cit2.12395>
20. Xing, J., et al.: Boosting whale optimizer with quasi-oppositional learning and Gaussian barebone for feature selection and COVID-19 image segmentation. *J. Bionic Eng.* 20(2), 797–818 (2023). <https://doi.org/10.1007/s42235-022-00297-8>
21. Dillshad, V., et al.: D2LFS2Net: multi-class skin lesion diagnosis using deep learning and variance-controlled Marine Predator optimisation: an application for precision medicine. *CAAI Trans. Intell. Technol.* (2023). <https://doi.org/10.1049/cit2.12267>
22. Sayed, G.I., Soliman, M.M., Hassanien, A.E.: A novel melanoma prediction model for imbalanced data using optimized SqueezeNet by bald eagle search optimization. *Comput. Biol. Med.* 136, 104712 (2021). <https://doi.org/10.1016/j.compbimed.2021.104712>
23. Piri, J., Mohapatra, P.: An analytical study of modified multi-objective Harris Hawk Optimizer towards medical data feature selection. *Comput. Biol. Med.* 135, 104558 (2021). <https://doi.org/10.1016/j.compbimed.2021.104558>
24. Pan, X., et al.: User collaborative filtering recommendation algorithm based on adaptive parametric optimisation SSPSO. *Int. J. Comput. Sci. Math.* 8(6), 580–592 (2017). <https://doi.org/10.1504/ijcsm.2017.088977>
25. Yadav, S., et al.: Trust aware recommender system using swarm intelligence. *J. Comput. Sci.* 28, 180–192 (2018). <https://doi.org/10.1016/j.jocs.2018.09.007>
26. Logesh, R., et al.: Enhancing recommendation stability of collaborative filtering recommender system through bio-inspired clustering ensemble method. *Neural Comput. Appl.* 32(7), 2141–2164 (2020). <https://doi.org/10.1007/s00521-018-3891-5>
27. Singh, V.K., Sabharwal, S., Gabrani, G.: A new fuzzy clustering-based recommendation method using grasshopper optimization algorithm and Map-Reduce. *Int. J. Syst. Assur. Eng. Manag.* 13(5), 2698–2709 (2022). <https://doi.org/10.1007/s13198-022-01740-z>
28. Dhawale, P.G., Kamboj, V.K., Bath, S.K.: A lévy flight based strategy to improve the exploitation capability of arithmetic optimization algorithm for engineering global optimization problems. *Trans. Emerg. Telecommun. Technol.* 34(4), e4739 (2023). <https://doi.org/10.1002/ett.4739>
29. Zubair, M., et al.: An improved K-means clustering algorithm towards an efficient data-driven modeling. *Ann. Data Sci.* 11(5), 1525–1544 (2024). <https://doi.org/10.1007/s40745-022-00428-2>
30. Harper, F.M., Konstan, J.A.: The movielens datasets: history and context. *ACM Trans. Interact. Intell. Syst. (THIS)* 5(4), 1–19 (2015). <https://doi.org/10.1145/2827872>
31. McAuley, J., et al.: Image-based recommendations on styles and substitutes. In: *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 43–52 (2015)
32. Ghazanfar, M.A., Prugel-Bennett, A.: A scalable, accurate hybrid recommender system. In: *2010 Third International Conference on Knowledge Discovery and Data Mining*, pp. 94–98. IEEE (2010)
33. Halder, S., Sarkar, A.M.J., Lee, Y.K.: Movie recommendation system based on movie swarm. In: *2012 Second International Conference on Cloud and Green Computing*, pp. 804–809. IEEE (2012)
34. Panigrahi, S., Lenka, R.K., Stitpragyan, A.: A hybrid distributed collaborative filtering recommender engine using Apache spark. *Procedia Comput. Sci.* 83, 1000–1006 (2016). <https://doi.org/10.1016/j.procs.2016.04.214>
35. Kumar, V., et al.: Collaborative filtering using multiple binary maximum margin matrix factorizations. *Inf. Sci.* 380, 1–11 (2017). <https://doi.org/10.1016/j.ins.2016.11.003>
36. Walek, B., Fojtik, V.: A hybrid recommender system for recommending relevant movies using an expert system. *Expert Syst. Appl.* 158, 113452 (2020). <https://doi.org/10.1016/j.eswa.2020.113452>
37. Wu, C., et al.: Personalized news recommendation: methods and challenges. *ACM Trans. Inf. Syst.* 41(1), 1–50 (2023). <https://doi.org/10.1145/3530257>
38. Lemire, D., Maclachlan, A.: Slope one predictors for online rating-based collaborative filtering. In: *Proceedings of the 2005 SIAM International Conference on Data Mining*, pp. 471–475. Society for Industrial and Applied Mathematics (2005)
39. Beni, G., Wang, J.: *Swarm intelligence in cellular robotic systems*. In: *Robots and Biological Systems: Towards a New Bionics?* pp. 703–712. Springer Berlin Heidelberg, Berlin (1993)
40. Holland, J.H.: Genetic algorithms. *Sci. Am.* 267(1), 66–73 (1992). <https://doi.org/10.1038/scientificamerican0792-66>
41. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: *Proceedings of ICNN'95-international Conference on Neural Networks*, vol. 4, pp. 1942–1948. IEEE (1995). <https://doi.org/10.1109/icnn.1995.488968>
42. Yang, X.S.: A new metaheuristic bat-inspired algorithm. In: *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, pp. 65–74. Springer Berlin Heidelberg, Berlin (2010)
43. Mirjalili, S., Mirjalili, S.M., Lewis, A.: Grey wolf optimizer. *Adv. Eng. Software* 69, 46–61 (2014). <https://doi.org/10.1016/j.advengsoft.2013.12.007>
44. Mirjalili, S., et al.: Salp Swarm Algorithm: a bio-inspired optimizer for engineering design problems. *Adv. Eng. Software* 114, 163–191 (2017). <https://doi.org/10.1016/j.advengsoft.2017.07.002>
45. Pop, P.C., et al.: A comprehensive survey on the generalized traveling salesman problem. *Eur. J. Oper. Res.* 314(3), 819–835 (2024). <https://doi.org/10.1016/j.ejor.2023.07.022>
46. Dauzère-Pères, S., et al.: The flexible job shop scheduling problem: a review. *Eur. J. Oper. Res.* 314(2), 409–432 (2024). <https://doi.org/10.1016/j.ejor.2023.05.017>
47. Zheng, T., et al.: Hybridizing multi-objective, clustering and particle swarm optimization for multimodal optimization. *Neural Comput. Appl.* 34(3), 1–28 (2022). <https://doi.org/10.1007/s00521-021-06355-2>
48. Kamaruzaman, A.F., et al.: Lévy flight algorithm for optimization problems—a literature review. *Appl. Mech. Mater.* 421, 496–501 (2013). <https://doi.org/10.4028/www.scientific.net/amm.421.496>
49. Baweja, D., et al.: Lévy flight based fire Hawk optimizer. In: *2024 IEEE Region 10 Symposium (TENSymp)*, pp. 1–6. IEEE (2024)
50. Mantegna, R.N.: Fast, accurate algorithm for numerical simulation of Lévy stable stochastic processes. *Phys. Rev.* 49(5), 4677–4683 (1994). <https://doi.org/10.1103/physreve.49.4677>
51. Barua, S., Merabet, A.: Lévy Arithmetic Algorithm: an enhanced metaheuristic algorithm and its application to engineering optimization. *Expert Syst. Appl.* 241, 122335 (2024). <https://doi.org/10.1016/j.eswa.2023.122335>
52. Che, W., et al.: Dynamic constrained multi-objective optimization algorithm based on co-evolution and diversity enhancement. *Swarm Evol. Comput.* 89, 101639 (2024). <https://doi.org/10.1016/j.swevo.2024.101639>
53. Heidari, A.A., et al.: Harris hawks optimization: algorithm and applications. *Future Generat. Comput. Syst.* 97, 849–872 (2019). <https://doi.org/10.1016/j.future.2019.02.028>

54. Jia, H., et al.: Hybrid grasshopper optimization algorithm and differential evolution for global optimization. *J. Intell. Fuzzy Syst.* 37(5), 6899–6910 (2019). <https://doi.org/10.3233/jifs-190782>
55. Wang, C., et al.: Multi-objective grasshopper optimization algorithm based on multi-group and co-evolution. *Math. Biosci. Eng.* 18(3), 2527–2561 (2021). <https://doi.org/10.3934/mbe.2021129>
56. Annas, M., Wahab, S.N.: Data mining methods: K-means clustering algorithms. *Int. J. Cyber IT Serv. Manag.* 3(1), 40–47 (2023). <https://doi.org/10.34306/ijcitsm.v3i1.122>
57. Surjanovic, S., Bingham, D.: Virtual Library of Simulation Experiments: Test Functions and Datasets (2013). URL <https://www.sfu.ca/~ssurjano/optimization.html>. 2020
58. Adam, S.P., et al.: No Free Lunch Theorem: A Review. *Approximation and Optimization: Algorithms, Complexity and Applications*, pp. 57–82 (2019)

How to cite this article: Zhao, Z., et al.: Clustering-based recommendation method with enhanced grasshopper optimisation algorithm. *CAAI Trans. Intell. Technol.* 1–16 (2025). <https://doi.org/10.1049/cit2.12408>