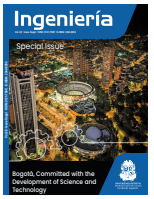
UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Ingeniería

https://revistas.udistrital.edu.co/index.php/reving/issue/view/1139

DOI: https://doi.org/10.14483/23448393.19925



Research

Optimization of Recommender Systems Using Particle Swarms

Optimización de sistemas recomendadores usando enjambre de partículas

Nancy Yaneth Gelvez-García[1]*, Jesús Gil-Ruíz[2], and Jhon Fredy Bayona-Navarro[3]

[1]Universidad Distrital Francisco José de Caldas (Bogotá-Colombia).
[2]Universidad Internacional de La Rioja (La Rioja-España).
[3]Universidad ECCI. (Bogotá-Colombia).

Abstract

Background: ecommender systems are one of the most widely used technologies by electronic businesses and internet applications as part of their strategies to improve customer experiences and boost sales. Recommender systems aim to suggest content based on its characteristics and on user preferences. The best recommender systems are able to deliver recommendations in the shortest possible time and with the least possible number of errors, which is challenging when working with large volumes of data.
Method: This article presents a novel technique to optimize recommender systems using particle swarm algorithms. The objective of the selected genetic algorithm is to find the best hyperparameters that minimize the difference between the expected values and those obtained by the recommender system.
Results: The algorithm demonstrates viability given the results obtained, highlighting its simple implementation and the minimal and easily attainable computational resources necessary for its execution.
Conclusions: It was possible to develop an algorithm using the most convenient properties of particle swarms in order to optimize recommender systems, thus achieving the ideal behavior for its implementation in the proposed scenario.

Keywords: recommender systems, optimization using particle swarm, collaborative filters, unsupervised systems

Article history

Received: 12th/Sep/2022
Modified: 5th/Dec/2022
Accepted: 30th/Jan/2023

Ing, vol. 28, no. suppl., 2023. e19925

*Correspondence: nygelvezg@udistrital.edu.co

Resumen

Contexto: Los sistemas recomendadores son una de las tecnologías más ampliamente utilizadas por comercios electrónicos y aplicaciones de internet como parte de sus estrategias para mejorar la experiencia de sus clientes y aumentar sus ventas. El sistema recomendador tiene por objetivo sugerir contenido basado en las características del mismo y en las preferencias de los usuarios. Los mejores sistemas recomendadores deben estar en la capacidad de entregar las recomendaciones en el menor tiempo y con el menor error posibles, lo cual constituye un desafío cuando se trabaja con grandes volúmenes de datos.
Método: En este artículo se presenta una técnica novedosa para optimizar sistemas recomendadores utilizando algoritmos de enjambre de partículas. El objetivo del algoritmo genético seleccionado es encontrar los mejores hiperparámetros que minimicen la diferencia entre los valores esperados y los obtenidos por el sistema recomendador.
Resultados: El algoritmo demuestra viabilidad dados los resultados obtenidos, destacando que su implementación es sencilla y los recursos computacionales necesarios para su ejecución son mínimos y de fácil acceso.
Conclusiones: Fue posible desarrollar un algoritmo utilizando las propiedades más convenientes del enjambre de partículas para optimizar los sistemas recomendadores, logrando el comportamiento ideal para su implementación en el escenario planteado.

Palabras clave: Sistemas Recomendadores, Optimización usando Enjambre de Partículas, Filtros Co- laborativos, Sistemas no Supervisados.

Table of contents

1. Introduction

Recommender systems constitute the basis for building technologies oriented towards improving user experience. Their use is part of our daily lives, given the popular use of applications such as Amazon, YouTube, and Google, whose recommender systems require a large volume of training information in order to provide effective suggestions (1).

To achieve the necessary volume of data, applications record user actions on platforms, with the aim to profile users. Nowadays, it is natural to record the time, the number of clicks, and the reactions of users while interacting with a specifc content (2). Additionally, it is important to gather content characteristics and quantify in order them to provide high-quality suggestions.

After training with the aforementioned data, a vector of hyperparameters is generated for each user, which works as an input to calculate the suggestions resulting from the multiplication of the vector by the content characteristics. However, calculating each hyperparameter vector is rarely easy, since finding the function that reduces the difference between the values expected by the user and the values obtained by the model depends on iterative methods that are computationally expensive (3).

In light of the above, it is common to find novelties in the techniques for the optimization of recommender systems, such as those described below:

In (4), a model capable of generating effective recommendations using collaborative models is presented. This memory-based model introduces a hybrid recommendation strategy that uses particle swarm optimization (PSO) to efficiently learn the weights of the parameters for each user. In addition, it uses fuzzy sets to naturally represent user ratings.

In (5), the PSO algorithm is implemented in order to learn about the user's personal preferences. From this study, it is concluded that the use of this type of model specializes individual results for each user, thus generating greater precision in model predictions. On the other hand, in (6), low-rank matrices are implemented which introduce pairs of vectors and, via vector products, generalize user preferences towards one or more types of content. In conclusion, generalizing user preferences is possible, although the accuracy of the results is affected.

As opposed to the proposal made in this work, optimization using algorithm genetics is not performed in (7). The authors present a coevolutionary algorithm named *cooperative coevolutionary invasive weed optimization* (CCIWO), which optimizes functions with either local optima or Nash equilibria searches. The results show that implementing these algorithms optimizes the precision of the models and reduces their computational means.

The authors of (8) combine a semi-supervised optimization algorithm using a particle swarm with the clustering process from the recommender systems. They manage to overcome the expectations of the traditional recommender systems' algorithms, thus reducing the time spent, as well as its corresponding computational resources.

The work by (9) presents the Cognitive Information Filtering System (CIFS), which applies an evolutionary model while learning from user feedback. A CIFS filters emails based on user ratings and behavior monitoring.

Other techniques can be utilized to this effect, as is the case of the work presented in (10). This research demonstrates the use of networks based on the Bayesian Beliefs Network, which are also known as *Bayesian Networks*, allow reasoning under uncertainty, and combine the advantages of visual representation for decision-making. This work is crucial, as it allows for an approach towards creating a collaborative filter for recommender systems.

On the other hand, bio-inspired algorithms consist of a set of problem-solving strategies based on Darwin's natural selection theories, the functioning of neural networks, and bee swarms (11). These algorithms can represent an optimization problem as a function to be maximized or minimized according to a set of constraints. This type of problem is usually studied by operations research and techniques such as linear programming, which are generally employed to obtain a solution. However, it is not always possible to find a solution by analytical or mathematical means. Thus, iterative methods must sometimes be used.

Considering the importance of recommender systems, as well as their broad implementation, an optimization technique is herein suggested that utilizes all the advantages of genetic algorithms, which can solve optimization problems whenever iterative methods must be used. Thus, it is a model capable of recommending based on user preferences and particle swarm optimization. To test the algorithm, a movie database was used as an example. The algorithm succeded in consolidating a parameterizable and adaptive model according to data characteristics.

2. Materials and methods

2.1. Methodology

The first stage of the project consisted of appropriating the conceptual rationale of the theory regarding recommender systems and the optimization of a swarm of particles.

2.1.1. Recommender systems

Recommender systems establish a set of criteria and considerations regarding user data in order to make predictions about user preferences. According to (9) and (12), there are three types of recommender systems:

- **Popularity systems:** In this type of system, an object is referenced based on its interactions with users, and then it is recommended.
- **Content systems:** This system searches for objects similar to those viewed by the user and recommends them.
- **Collaborative systems:** This system relates a user's data with that of others in order to generate object recommendations. Within these, techniques are employed for assigning patterns to groups of users in such a way that each group is homogeneous and different from the others. In (13), an algorithm for generating the groups is presented.

In general, recommender systems translate into a better satisfaction of customer needs. These types of systems become personal assistants that encourage users to continue purchasing or viewing content. Additionally, they bring exceptional efficiency to the user experience and increase the likelihood that they will buy or consume the suggested content.

2.1.2. Bio-inspired algorithms

Natural selection is an evolutionary phenomenon described by Charles Darwin in his book *The Origin of Species*, which establishes that only the fittest or strongest survive in a natural environment, and that the least fit individuals and their characteristics are not transmitted to the following generations due to a lack of reproduction. Artificial neural networks consist of a set of interconnected nodes that transmit signals that excite or inhibit the nodes in order to obtain a result. Finally, particle swarms are often used to solve optimization problems. In these, a population of particles that communicate with each another is initialized and then directed towards the global minimum or maximum of a surface.

Among the bio-inspired algorithms, the following can be found:

- Artificial neural networks
- Evolutionary computing
- Genetic algorithms
- Particle swarms

Among the iterative methods are the well-known particle swarm approaches, which are based on the behavior of bees when searching for flower fields. Thus each particle moves over the surface that represents the function to be optimized until its global maximum or minimum. Fig. 1 shows the movement of the particles within the function. Each point of the function represents a solution for the optimization problem, but the goal is to find the best solution, *i.e.*, the one that reduces the difference between the values obtained by the model and the expected ones. For this reason, the points start to descend in the function until they reach the global minimum.

Taking advantage of the benefits of the techniques displayed in the literature (1–15), as well as considering the requirements for real-time recommendations, this proposal consists of a technique based on the optimization of particle swarms for optimizing the hyperparameters of recommender systems. For the development of this project, the following steps were executed:

2.1.3. Approach to the issue of optimization

Recommender systems base their operations on two aspects:

- **User attributes** correspond to user metadata and actions recorded within the platform, aiming to classify users into a group that allows profiling their preferences into a categorical group.
- **Element attributes:** These correspond to element metadata and allow grouping elements by categories, groups, currents, authors, *etc*.

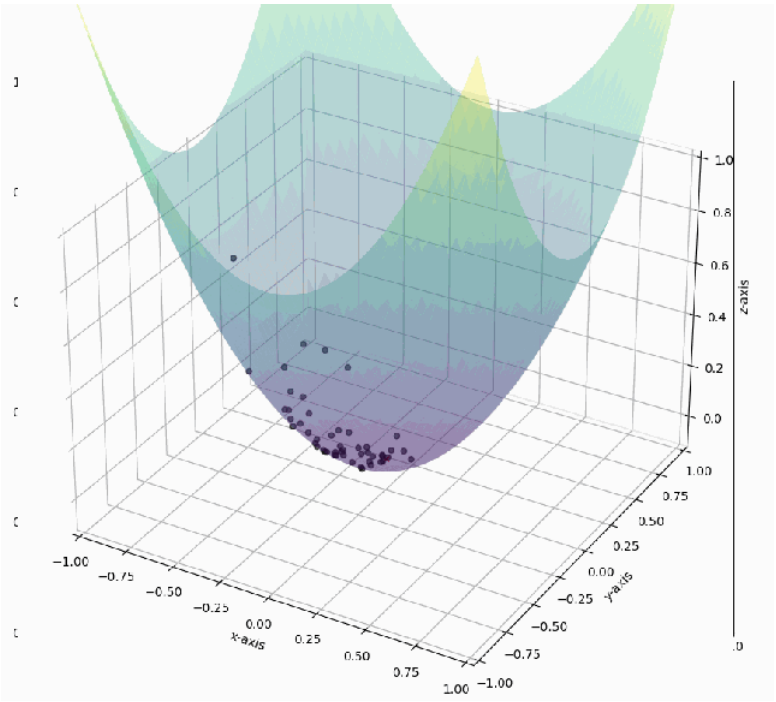


Figure 1. Movement of the particles in the function

The issue regarding the optimization of recommender systems consists of **predicting a user's rating of an element based on user and element attributes**. Mathematically, this can be represented as follows:

- n_u: number of users
- n_m: number of elements.
- $r(i,j) = 1$: whether the user j has rated the element i
- $y^{(i,j)}$: the rating given by the user j to the element i (defined only if $r\,(i,j) = 1$)
- $x^{(i)}$: characteristics of the element i
- $\theta^{(j)}$: a vector of parameters for the user j

Based on the above, the algorithm should try to predict $y^{(i,j)}$ for the values $r\,(i,j) = 0$ and recommend an element based on the result. Thus, for each user j, a parameter $\theta^{(j)}$ must be learned, and the rating of user j to the element i must be predicted with $\left(\theta^{(j)}\right)^T x^{(i)}$. Finally, the objective function for user j is constructed using the mean squared error:

$$\underset{\theta^{(j)}}{min}\frac{1}{2m^{(j)}}\sum_{i:r(i,j)=1}((\theta^{(j)})^T x^{(i)} - y^{(i,j)})^2 + \frac{\lambda}{2m^{(j)}}\sum_{k=1}^{n}{\theta_k^{(j)}}^2\,, \tag{1}$$

simplifying

$$\underset{\theta^{(j)}}{min}\,J(\theta^{(j)})\,, \tag{2}$$

the parameter λ is employed as a regularization parameter for the objective function.

2.2. Solution to the optimization problem using a particle swarm

Based on the study conducted by (14), it was concluded that particle swarms algorithms are useful when it comes to finding solutions faster than other bio-inspired algorithms such as evolutionary and genetic algorithms, which is why they were selected to solve this problem.

Within this algorithm, each one of the particles moves across the surface of possible solutions according to three coefficients:

- **Inertia coefficient (W):** it indicates the state of rest or motion of the particle.
- **Cognitive acceleration coefficient (C_1):** it indicates the speed of movement of the individual particle.
- **Social acceleration coefficient (C_2):**): it indicates the speed of movement of the particle in a social way.

Therefore, the motion x_i of the particle p_i at an instant of time t is given by:

$$x_i(t+1) = x_i(t) + V_i(t+1)\,, \tag{3}$$

and its velocity V_i at an instant of time t is

$$V_i(t+1) = WV_i(t) + C_1(P_i(t) - x_i(t)) + C_2(g(t) - x_i(t))\,, \tag{4}$$

where P_i is the best value obtained by the particle, and g is the best value obtained by a group of particles.

Thus, the particle moves in a vector parallel to the best individual value, the best global value, and its inertia at an instant of time t. Based on the above, each of the particles p corresponds to a solution $\theta^{(j)}$, and the result obtained after executing the algorithm is employed as the optimal solution.

2.3. Proposed algorithm

To develop the algorithm, the traditional model for training of recommender systems was considered, and its steps were adapted for the implementation of PSO. In general, these steps are as follows:

- **Data cleaning:** Deletion or correction of the data outside the average or standardized data.
- **Profiling and definition of object features:** The feature vector for each object $x^{(i)}$ is defined. Additionally, the user is asked to rate some random objects, capturing $r\,(i,j) = 1$ and $y^{(i,j)}$.
- **Training:** Initializing the particles p_i to a random value within the feature domain X. Subsequently, a number of maximum iterations and a minimum expected error are determined. The vectors $[W\,V_i(t), C_1(P_i(t) - x_i(t)), C_2(g(t) - x_i(t))]$ are calculated, and the particle is displaced based on the equation for V_i until the stopping condition is reached. The convergence position of the particles will be taken as $\theta^{(j)}$.
- **Suggestions** made to users based on the result of θX, taking the result as the probability that a user likes the content, with 1 being the most likely.

Fig. 2 depicts these steps.

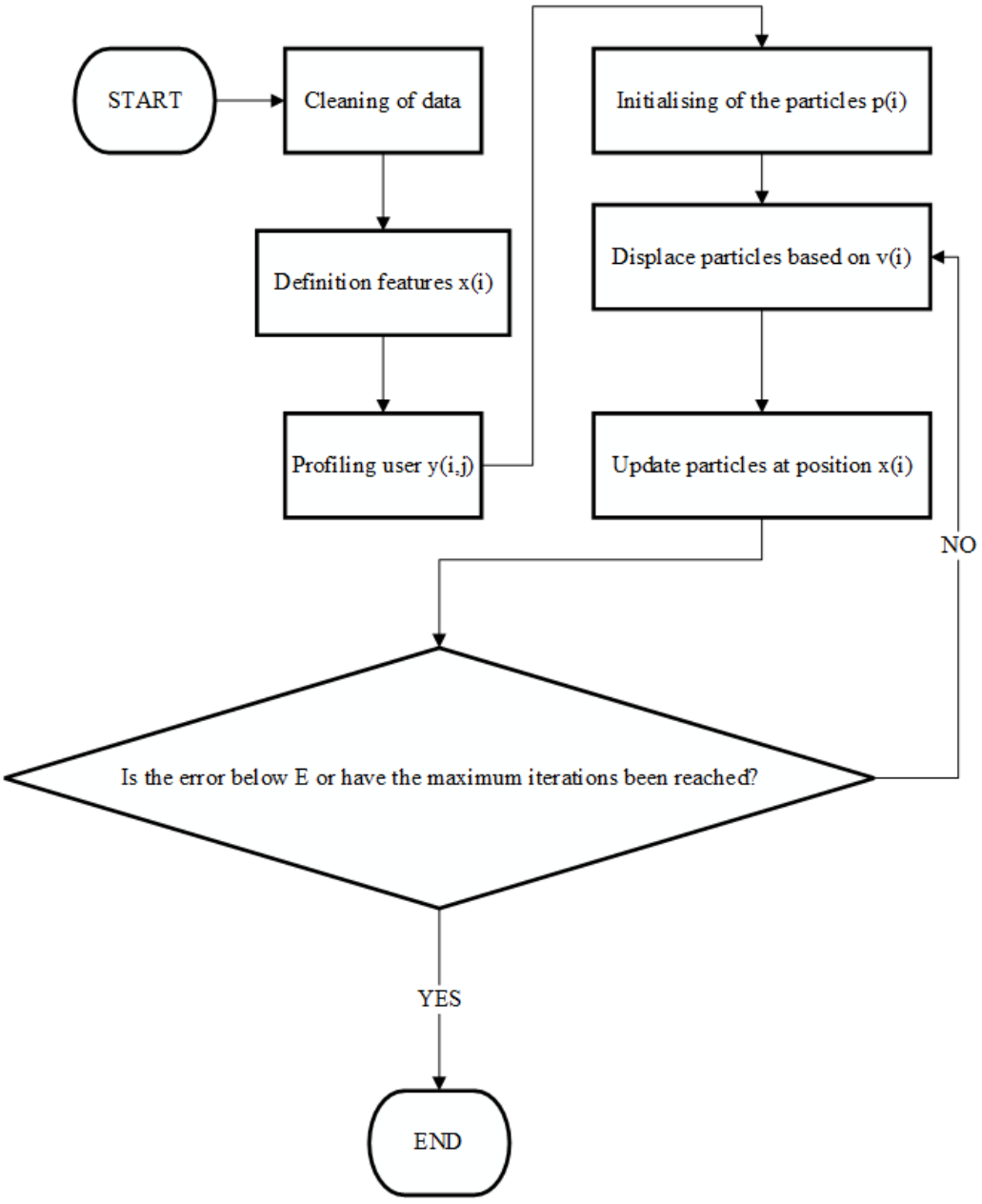


Figure 2. Description of the algorithm used to perform the training process

3. Results

To test the algorithm, a set of metadata was used which comprised 45.000 films that were released on July 2017 or before MovieLens. The metadata includes the cast, crew, plot keywords, budget, revenue, posters, release dates, languages, production companies, countries, and The Movie Database (TMDB) vote counts and averages. These are available in the Movies Dataset from Kaggle. We used two groups of regularization parameters for the particle swarm in order to generate two variations of the algorithm:

- In the first variation, the Cognitive Acceleration Coefficient (C_1) was prioritized with the values of W =1, C_1 =3, and C_2 =1,5, hereafter referred to as the *school of fish* (CP).
- In the second variation, the Social Acceleration Coefficient (C_2) was prioritized with the values of W =1, C_1 =1,5, and C_2 =3, hereafter referred to as the *swarm of particles* (SP).

Each of the algorithm's variations was tested on a dataset that increased in volume by 50 records after each run. The Python programming language was used to develop the algorithm. Fig. 3 shows the execution of the algorithm on the terminal device. For the execution, a computer with an Intel Core i5 4 GHz 4-core processor and 16 Gb RAM memory were used.

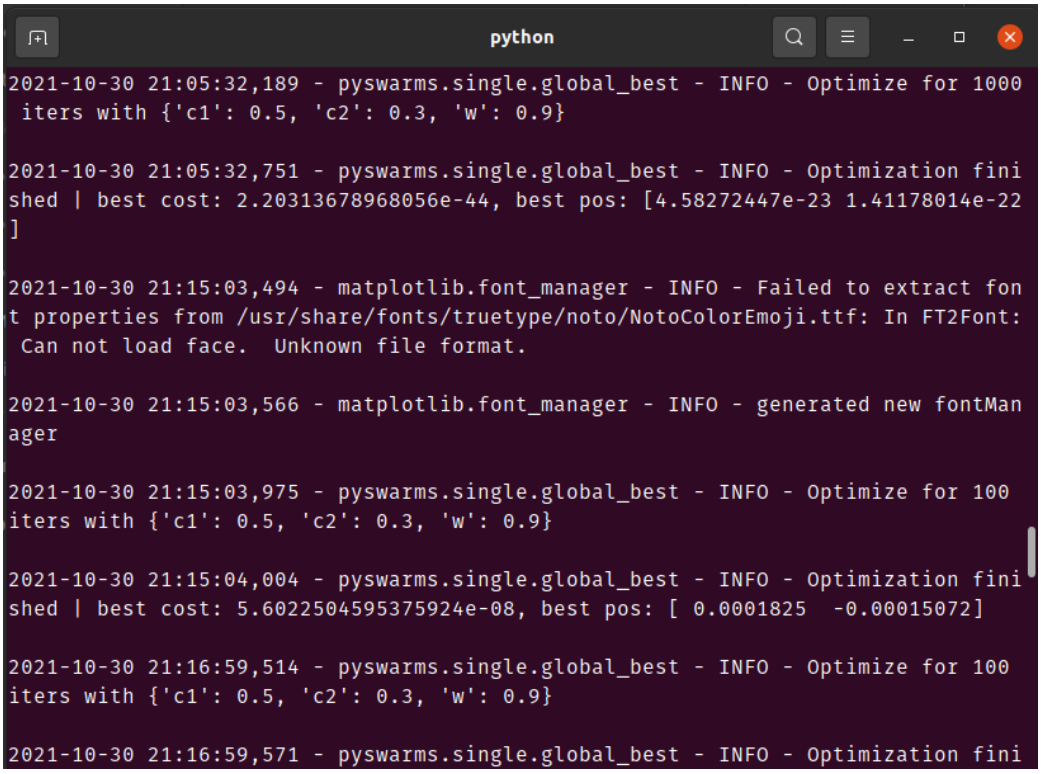


Figure 3. Example of the algorithm's execution

The results were recorded in Table I, which provides evidence that the CP model requires less time than the EP one. However, more iterations are needed.

Table I. Comparison of training results for the proposed models

Number of data points	CP time	EP time	CP generation	EP generation
50	81,5	120,5	54	48
100	180,6	250,45	68	65
150	270,1	394,5	87	78
200	375,9	520,4	90	84
250	458,7	740,7	121	98

Let us recall that a good recommender system must be able to answer as quickly as possible, and, although the CP model demands more iterations, it meets the requirements for recommender systems training.

As part of the training process, the data were divided into three groups:

- 60 % used for data training
- 20 % used for model testing
- 20 % used for model validation

Table II shows the errors obtained when testing the model. It can be seen that the error rate decreases and converges as the data increase. Thereupon, the conclusion is that the model correctly generalizes each of the data groups presented in the previous model.

Table II. Comparison of validation mistakes for the different data sizes using a particle swarm

Data	Training error	CV error	Trial error	EP generation
50	3,541	3,213	3,124	48
100	2,174	1,981	1,741	65
150	1,871	1,441	1,651	78
200	1,974	1,445	1,745	84
250	1,744	1,745	1,742	98

Finally, the algorithm was compared to the traditional recommender systems training method via the descending gradient. Table III shows the training results and validation errors for the different datasets. According to the results, it can be seen that the error obtained is greater and requires a higher number of iterations in comparison with the particle swarm algorithm.

Table III. Comparison of validation errors for different data sizes using the descending gradient

Data	Training error	CV error	Trial error	Iterations
50	4,892	4,913	4,897	73
100	3,974	3,911	3,741	89
150	2,511	2,712	2,592	98
200	2,142	2,456	2,514	103
250	2,102	2,105	2,096	124

4. Conclusions

Based on these results, the following conclusions were reached:

- Through bio-inspired algorithms, specifically particle swarms, it is possible to solve the problem regarding the optimization of recommender systems. Moreover, its implementation was demonstrated using a case study.

- It was found that it is possible to generate a wide range of solutions by varying the parameters with which particle swarm algorithms are executed. However, their assessment will depend on the defined metrics.
- The training of recommender systems using particle swarms allows for the implementation of parallelization strategies different from the traditional training employed.

Future works should experiment with other types of bio-inspired algorithms and variations of particle swarms.

5. Author contributions

All authors contributed equally to the research.

References

[1] Z. Zhang and S. Qian, "The research of e-commerce recommendation system based on collaborative filtering technology," in *Advances in Computer Science and Information Engineering*, D. Jin and S. Lin, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 507–512. https://doi.org/10.1007/978-3-642-30126-1_80 ↑2, 5

[2] L. Cegan and P. Filip, "Advanced web analytics tool for mouse tracking and real-time data processing," in *2017 IEEE 14th International Scientific Conference on Informatics*, 2017, pp. 431–435. https://doi.org/10.1109/INFORMATICS.2017.8327288 ↑3, 5

[3] A. Janusz, G. Hao, D. Kaluza, T. Li, R. Wojciechowski, and D. Slezak, "Predicting escalations in customer support: Analysis of data mining challenge results," in *2020 IEEE International Conference on Big Data (Big Data)*, 2020, pp. 5519–5526. https://doi.org/10.1109/BigData50022.2020.9378024 ↑3, 5

[4] M. Wasid and V. Kant, "A particle swarm approach to collaborative filtering-based recommender systems through fuzzy features," *Procedia Comput. Sci.*, vol. 54, pp. 440–448, 2015. https://doi.org/10.1016/j.procs.2015.06.051 ↑3, 5

[5] S. Ujjin and P. J. Bentley, "Particle swarm optimization recommender system," *Proceedings of the 2003 IEEE Swarm Intelligence Symposium. SIS'03* (Cat. No.03EX706), Indianapolis, IN, USA, 2003, pp. 124-131, https://doi.org/10.1109/SIS.2003.1202257 ↑3, 5

[6] J. Abernethy, F. Bach, T. Evgeniou, and J.-P. Vert, "Low-rank matrix factorization with attributes," ArXiv, https://doi.org/10.48550/arXiv.cs/0611124 ↑3, 5

[7] H. Hajimirsadeghi, A. Ghazanfari, A. Rahimi-Kian, and C. Lucas, "Cooperative coevolutionary invasive weed optimization and its application to Nash equilibrium search in electricity markets,"*2009 World Congress on Nature and Biologically Inspired Computing (NaBIC)*, Coimbatore, India, 2009, pp. 1532-1535, https://doi.org/10.1109/NABIC.2009.5393669 ↑3, 5

[8] W. Zhou, X. Pan, R. Li, and Y. Lu, "The recommendation system based on semi-supervised pso clustering algorithm," in *Proceedings of the 2016 International Forum on Mechanical, Control and Automation (IFMCA 2016)*. Atlantis Press, 2017/03, pp. 63-71. [Online]. Available: https://doi.org/10.2991/ifmca-16.2017.11 ↑3, 5

[9] A. Tjoa, M. Hofferer, G. Ehrentraut, and P. Untersmeyer, "Applying evolutionary algorithms to the problem ofinformation filtering," in *Database and Expert Systems Applications. 8th International Conference, DEXA '97. Proceedings*, 1997, pp. 450–458. ↑3, 4, 5

[10] D. Heckerman, D. Chickering, C. Meek, R. Rounthwaite, and C. Kadie, "Dependency networks for collaborative filtering and data visualization." *J. Mach. Learn. Res.*, 01 2000, pp. 264–273. ↑4, 5

[11] D. Gutiérrez, A. Tapia Córdoba, and A. Rodríguez del Nozal. "Algoritmos Genéticos con Python: Un enfoque practico para resolver problemas de ingeniería", Marcombo. Spain. 2020. ↑4, 5

[12] D. Nichols, "Implicit rating and filtering," in *Proceedings of the Fifth DELOS Workshop on Filtering and Collaborative Filtering. ERCIM*, 1998, pp. 31–36, fifth DELOS Workshop on Filtering; Collaborative Filtering; Conference date: 01-01-1900. ↑4, 5

[13] L. Ungar and D. Foster, "Clustering methods for collaborative filtering," *Proceedings of the Workshop on Recommendation Systems, Conference: of the 10th International Conference on Knowledge Management and Knowledge Technologies (I-KNOW 2010)*, 1-3 September, 2010, Graz, AustriaAt: Graz, Austria. ↑4, 5

[14] E. M.-M. Blanca Cecilia López-Ramírez, "Estudio del comportamiento en-línea de algoritmos bio-inspirados usando medidas de desempeño en optimización con restricciones" *COMCEV'07*, 3-5 october, 2007, Aguascalientes, México. ↑5, 7

[15] D. A. Demissie, "A Hybrid Movie Recommendation System using Particle Swarm Optimization and K-means Clustering Algorithm," Ph.D. dissertation, Adama Science and Technology University (ASTU), June 2020. ↑5

Nancy Yaneth Gelvez-García

PhD Student in Engineering at Universidad Distrital Francisco José de Caldas; Master in Information Sciences and Communications, Universidad Distrital Francisco José de Caldas; Systems Engineer, Universidad ECCI; professor at Universidad Distrital Francisco José de Caldas; research professor of the GIIRA Research group.
Email: nygelvezg@udistrital.edu.co

Jesús Gil-Ruiz

International PhD Student in Computer Science at the International University of La Rioja; Superior Industrial Engineer and Civil Engineer from the University of Cádiz; Industrial Organization Engineer from the International University of La Rioja; academic coordinator of the Expert and Digital Project Management program at Marconi University of Miami (MIU);

professor at the International University of La Rioja; part of the Drive Data Systems (DDS) research group.
Email: Jesus.gil@unir.net

Jhon Fredy Bayona-Navarro

Master in Electronic Engineering, Universidad Javeriana; electronic engineer, Universidad Antonio Nariño; professor at Universidad ECCI; research professor of the INDETECA research group.
Email: jbayonan@ecci.edu.co