Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2019) 27: 2121 – 2136

doi:10.3906/elk-1812-175

Evolutionary approaches for weight optimization in collaborative filtering-based recommender systems

Sevgi YİĞİT SERT, Yılmaz AR, Erkan BOSTANCI
Department of Computer Engineering, Faculty of Engineering, Ankara University, Ankara, Turkey

Received: 24.12.2018 • **Accepted/Published Online:** 06.03.2019 • **Final Version:** 15.05.2019

Abstract: Collaborative filtering is one of the widely adopted approaches in recommender systems used for e-commerce applications, stating that users having similar tastes will have similar preferences in the future. The literature presents a number of similarity metrics such as the extended Jaccard coefficient to quantify these preference similarities. This paper aims to improve prediction accuracy by optimizing the similarity values computed using these metrics by adopting two biologically inspired approaches, namely artificial bee colony and genetic algorithms, with a bottom-up approach, suggesting that any improvement on a single-user basis will reflect on the overall prediction accuracy. Detailed statistical analysis was performed using the t-test, analysis of variance, and McNemar's test to see whether there were performance differences. The results show that statistically significant differences exist with high confidence levels.

Key words: Recommender systems, collaborative filtering, artificial bee colony, optimization, statistical evaluation

1. Introduction

Greater adoption of Internet technologies in daily life has been observed due to developments in both hardware and software systems, such as in mobile devices and the increasing number of applications published on various platforms. The way of shopping has evolved into a hybrid form in which people can actually visit shops and purchase goods easily on the Internet ubiquitously. The results of such shopping experiences both produce an economic buoyancy [1] and create large amounts of raw data that can also be used for designing marketing strategies to attract more customers.

The raw data related to the shopping experiences of customers may contain information regarding the date and amount of the purchase as well as the ratings given by customers to items they bought. These data can be used to identify the types of items bought and generate recommendations for similar items or the items that may also be needed or liked for future purchases. Data for a single customer may not constitute a large amount of information; however, online shopping websites are visited by millions everyday, given that 1.2 billion digital buyers were reported for 2015 by Statista,[1] generating enormous amounts of data to be processed. Therefore, automated systems are needed for generating recommendations for customers in order to enhance their shopping experience by providing them with buying options.

Recommender systems (RSs) [2] have been proposed in order to provide users with recommendations based on their past purchases or ratings. Such systems extract and analyze information from item features [3]

*Correspondence: ebostanci@ankara.edu.tr

[1] http://www.statista.com/statistics/251666/number-of-digital-buyers-worldwide/

or find similarities between different customers [4]. The reason behind the development of different approaches is that the utilization of various types of information may generate opportunities to improve the quality of the recommendation. Some RSs mentioned above give more weight to item features for deciding the preferences of customers and suggested relevant items that the customers chose in their past shopping experiences. Other systems attempt to infer an agreement between different customers on their item preferences in their shopping experiences with the assumption of users having similar tastes being likely to prefer similar items in the future.

Even though many of the proposed approaches tackle different aspects of RSs, there are still several problems to be solved for these systems. Examples of these problems can be stated as follows: the computation of similarity values between different users requires a certain amount of goods purchased or rated by both in the past. Furthermore, these similarity computations cannot make use of new users of the system with no purchase or rating history as well as new items introduced to the market recently. A trivial approach for assigning rating values for users that recently opted in or newly released items is calculating the average of the existing customer ratings. Previous studies show that there is still a place for improvement in the accuracy of commonly used similarity metrics.

Yet another problem that needs to be addressed by RS approaches is the problem of overspecialization, which can be described as the convergence to the mainstream or trending items, repeatedly recommending these items since these ratings will have a higher weight after a while [5]. This situation may disappoint some users who wish to receive recommendations for a variety of items rather than the popular ones. This problem can be overcome with approaches that can perform search operations over an extended space by looking for solutions that have not been considered yet. Evolutionary algorithms, namely the artificial bee colony (ABC) and genetic algorithm (GA), make use of exploitation and exploration operators [6], as shown in Table 1. Note that the literature suggests that there should be a balance between the exploration and exploitation phases in evolutionary algorithms [6]. The qABC algorithm inherently contains two exploitation phases.

Table 1. Search operators for ABC and GA.

Exploitation		Exploration	
ABC	**GA**	**ABC**	**GA**
Employee bee phase	Fitness evaluation	Scout bee phase	Recombination
Onlooker bee phase			Mutation

A number of different studies have employed evolutionary approaches with these considerations in mind [7–11]. The work in [7] presented a novel metric for computing recommendations. This metric uses various weights and a GA was used to optimize them. A genetic program was evolved in [9] in order to transform user-item relations to user-feature relations so that the dimensionality of sparse preference data was reduced. The work in [11] optimized clusters of users with ABC to avoid local optima. Similarly, topological features of social networks were exploited using the ABC algorithm for friend recommendations in [12]. More approaches are detailed in Section 2.

In this paper, we follow a different approach from the studies given above by aiming to optimize the weights computed using a variety of similarity metrics used in RSs such as Pearson's correlation coefficient (PCC), vector cosine similarity (VCS), and the extended Jaccard coefficient (EJC). The main idea behind this method is to improve the quality of the recommendation, i.e. reduce prediction error when compared to the one computed with original weights, in a bottom-up manner starting from an individual user to the complete set of

users. Two biologically inspired approaches, namely the ABC and GA, are compared according to a statistical framework. Differences in performance were identified using two statistical tests for the metrics mentioned above. The key contributions of our study are as follows:

- A novel approach for collaborative filtering using artificial bee colony is proposed.
- A modification is performed on the employee and onlooker phase to create variation.
- Our results are promising in yielding good results for different parameter settings.
- The results are also compared with a genetic algorithm-based approach.

The paper is structured as follows. In Section 2, we describe the CF method and the metrics used in this study. Details of the optimization approaches using ABC and GA are presented in Section 3. Section 4 describes the experimental framework to compare the evolutionary approaches with statistical tests. Results are presented in Section 5 and, finally, conclusions are drawn in Section 6.

2. Related work and background

Recently, we have witnessed evolutionary changes in computing and Internet technologies. With the usage of these technologies in commerce, the number of products or items marketed on web-based commercial sites and the number of users searching and purchasing these products have increased dramatically. RS approaches aim to help users in narrowing down the possible range of items using different information filtering approaches. Among these approaches, collaborative filtering (CF) is the best known method, which is based on the assumption that the user prefers items similar to ones preferred in the past. Two widely used broad categories of CF algorithms are memory- and model-based methods [13]. The first one uses overall ratings during the training phase to generate a prediction for the rating of the current user on a target item. In a different manner, model-based approaches create models to predict unknown ratings using training data. A memory-based method exploits the similarities among users [13].

A rating prediction ($\hat{r}_{u,i}$) can be computed with the following formula for current user u on item i as a weighted sum computed from ratings of other users as given in [14]:

$$\hat{r}_{u,i} = \bar{r}_u + \kappa \sum_{v=1}^{n} w(u,v)(r_{v,i} - \bar{r}_v), \tag{1}$$

where n is the number of users who have a positive similarity value with the current user in the dataset. $w(u,v)$ is the measure of similarity for two users u and v. A normalization parameter κ is used for the magnitude of total weights as given in [14]. Average user ratings for users u and v are denoted by $\bar{r}_u$ and $\bar{r}_v$, respectively.

Given an active user u and a target item i, the approach aims to estimate the current user's rating for a specific item. The following steps are performed in this computation: first, the similarities between the active user and the rest of the users are found. Next, a list of users who have the highest similarity with the active user is created. This is called u's k-neighborhood, where k is the list size. Finally, similarities of ratings for u and his or her k-neighborhood are used as weights for computation of the predicted value as the weighted total ratings computed for neighbors of u.

Similarity metrics measure the closeness of rating vectors that belong to any two users. Used metrics PCC, EJC, and VCS are defined as shown in Table 2, where $w\langle PCC,\ EJC,$ or $VCS\rangle_{u,v}$ is the similarity

between two users, u and v. The list of items that were rated by both users is represented by I. $\bar{r}_u$ stands for the mean of the ratings given by user u. We also employed randomly assigned weights for similarities between users (RND) as a benchmark. The reason behind this usage is the need for investigating the amount of improvement in a setting that used random assignment of weights following the evolutionary approaches used here.

Table 2. Similarity metrics.

PCC	**EJC**	**VCS**
$w\langle PCC\rangle_{u,v} = \frac{a}{b.c}$	$w\langle EJC\rangle_{u,v} = \frac{d}{e+f-d}$	$w\langle VCS\rangle_{u,v} = \frac{g}{h.p}$
$a = \sum_{n\in I}(r_{u,n} - \bar{r}_u)(r_{v,n} - \bar{r}_v)$	$d = \sum_{n\in I}(r_{u,n})(r_{v,n})$	$g = \sum_{n\in I}(r_{u,n})(r_{v,n})$
$b = \sqrt{\sum_{n\in I}(r_{u,n} - \bar{r}_u)^2}$	$e = \sum_{n\in I}(r_{u,n}r_{u,n})$	$h = \sqrt{\sum_{n\in I}(r_{u,n}r_{u,n})}$
$c = \sqrt{\sum_{n\in I}(r_{v,n} - \bar{r}_v)^2}$	$f = \sum_{n\in I}(r_{v,n}r_{v,n})$	$p = \sqrt{\sum_{n\in I}(r_{v,n}r_{v,n})}$

In the k-neighborhood approach, the set of similar users is specific to each user u. The user ratings in the k-neighborhood for a specific item i are utilized to estimate the rating for user u for item i. The similarities calculated are used by this estimate as follows:

$$\hat{r}_{u,i} = \frac{\sum_{v\in V} w_{u,v} r_{v,i}}{\sum_{v\in V} w_{u,v}}, \tag{2}$$

where $\hat{r}_{u,i}$ is the estimated rating of current user u on specific item i. V represents the list of users similar to current user u. The rating of user v on item i is denoted with $r_{v,i}$. $w_{u,v}$ stands for the similarity of users u and v. Commonly rated items (CRI) were defined as the set indicating the list of items having a rating information from both users (u, v). This number can be used to determine the users in the k-neighborhood.

The literature presents a number of studies aiming to improve the accuracy of the base CF approach. Some of these studies utilized user-to-user relations [15, 16] while others incorporated interitem associations [17, 18] in their CF methods. The effect of the different similarity metrics (e.g., PCC or VCS) for users or items was also investigated in [19].

The method proposed in [14] employed the complete set of users in the training dataset for rating estimation. On the other hand, [15] employed a selected subset of the users for prediction, an approach known as k-neighborhood CF where the k most similar users are used for computing the prediction.

The authors of [17] used similarities calculated using items instead of users. This method assumes that the items that received similar ratings in the past are likely to get comparable ones in the future. The approach in [14] added the weighted deviation to the user's rating average as an offset for prediction; however, the method developed in [17] only used a simple weighted sum.

Several studies employing evolutionary computing in CF can be found in the literature [7, 9, 10, 20]. A novel approach to compute the weighted sum for estimation was defined in [7] and the GA was used here for refining the weights. The authors of [9] introduced a genetic programming (GP) method for converting the user-item domain to the user-feature domain. This study aimed to shrink the high dimensionality of sparse item preferences (selecting the discriminative set of features with GP), since the user-item space is a lot larger than the user-feature space. The work in [20] presented a hybrid RS to generate more accurate recommendations. The genes of a GA were encoded as the weights of implicit attributes in the learning materials. Using the rating

history, optimum weights were obtained using the individuals evolved by the GA. The nearest neighborhood algorithm (NNA) was then employed to produce recommendations using these optimized weights of implicit attributes [20]. A different direction was taken by Velez-Langs and Antonio [10]: adaptive learning capabilities of a GA were integrated with RS. This method improved the RS to find more suitable recommendations by introducing a new mechanism. A very recent study showed a significant amount of improvement in initial similarity weights using a GA [21].

The ABC algorithm is one of the swarm intelligence-based optimization algorithms inspired by honeybees [22]. It was developed based on observing the intelligent behavior of real honeybees in finding nectar-rich food sources and sharing the information of food sources with bees in the hive. Thus, the time spent on foraging and the used energy are minimized. ABC is a nature-inspired, robust, and easy-to-use optimization algorithm. Numerous applications have successfully implemented it for different problem domains [23–26].

Many variants of the standard ABC algorithm were also proposed in the literature. In some studies, an effort was made to improve the exploitation phase in the standard ABC. Zhu and Kwong [27] proposed the Gbest-guided ABC (GABC) algorithm, inspired by particle swarm optimization (PSO), which uses the global best solution in the exploitation phase. Gau and Lio [28] generated a new solution search equation inspired by differential evolution. The search in the exploitation phase is conducted only in the neighborhood of the best solution of the previous iteration. Banharnsakun [29] biased the direction of new candidate solutions toward the best-so-far position. The best solutions found so far are shared in the exploitation phase. These studies represent new solution search equations for onlooker bees. Quick ABC (qABC), proposed by Karaboga and Gorkemli [22], is one such approach. The qABC enhances the performance of the standard ABC in terms of convergence by defining a new formula for onlooker bees. It shows better local search ability, as will be detailed in Section 3.

Use of the ABC with a RS is a relatively new approach; hence, not so many such studies can be found. The work in [8] proposed a personalized auxiliary material recommendation system using the ABC. The system observes user learning activities on Facebook and conceptualizes the learner's individual learning style. The ABC selects the appropriate auxiliary materials in terms of their difficulty, number of 'likes', and course topics. That in [12] developed a friend recommendation system using an ABC exploiting structural properties and topological features of social networks. It first generates a subgraph of the network of the user and then suggests new links considering the parameters mentioned by optimizing the relative importance of the weights for these parameters using ABC. Ju and Xu [11] proposed a RS based on CF using k-means clustering to improve accuracy. The k-means algorithm was found to suffer from initialization dependence and get trapped in local optima, where evolutionary algorithms such as the ABC can be used to escape.

The authors believe, to the best of their knowledge, that an evolutionary algorithm employing the qABC to optimize the weights for CF does not exist in the literature.

3. Approach

The CF algorithm utilizes the similarities between users. CF uses these similarity weights to predict unknown ratings. Various well-known proximity metrics such as PCC, VCS, and EJC are used to compute user-to-user similarities. We will use evolutionary approaches (ABC and GA) to optimize the weights in order to improve the predictions. Note that qABC and ABC will be used in the following interchangeably. The overall approach is summarized in Figure 1.

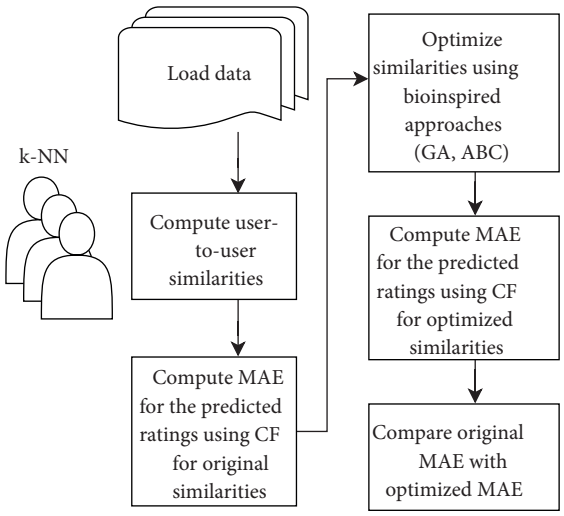

Figure 1. Algorithm used in optimizing the similarity metrics using bioinspired approaches.

3.1. qABC algorithm

The bee swarm in the qABC algorithm, as in the standard ABC, consists of 3 phases representing groups of bees to accomplish different tasks: employee, onlooker, and scout bees. An employee bee investigates nearby food sources, which are random solutions, around the hive. Back in the hive, they dance in the dancing area to inform the other bees about the locations of the food sources. Onlooker bees choose the best food source to exploit depending on a selection strategy that runs on probability basis. A probability value (p_i) is calculated as in Eq. (3) using fitness values ($fit(x_j)$) as an indicator of the quality of food sources (x_i) for a given number of food sources (SN). The higher the fitness value of a food source is, the more the probability of being chosen by onlooker bee increases:

$$p_i = \frac{fit(x_i)}{\sum_{j=1}^{SN} fit(x_j)}. \tag{3}$$

Briefly, onlooker bees are responsible for exploitation of food sources that are found in the employee bee phase. The last group of bees, scouts, fly around the hive and check random locations for food sources to avoid local optima.

The difference between the standard ABC and qABC is the modification of the equation in the onlooker bee phase, which allows the qABC to converge faster. In the standard algorithm, employee and onlooker bee phases use the same equation to find a neighbor of the food source (see Section 3.2 for a detailed explanation) (v_i) as

$$v_{ij} = x_{ij} + \phi(x_{ij} - x_{kj}), \tag{4}$$

where j is a randomly chosen dimension and x_k represents a food source selected randomly. As soon as a new candidate solution, v_i, is provided, the fitness values of v_i and x_i are calculated and compared to determine the better one. ϕ is a random number in [-1,1].

In qABC, the onlooker bee phase is illustrated in a way that is closer to real honeybees. In real life, onlooker bees watch the dances of employee bees and determine the region in the search space to be exploited [22].

Then they examine the food sources in this region and choose the fittest one. This situation is materialized by

$$v_{N_ij}^{best} = x_{N_ij}^{best} + \phi(x_{N_ij}^{best} - x_{kj}), \tag{5}$$

where j is randomly chosen dimension and $x_{N_i}^{best}$ represents the best solution among the neighbors of x_i and itself. The neighbors of x_i are food sources in the region centered by x_i. The neighbors of x_i are computed as follows:

if $d(i,n) <= r * md_i$ **then**
  x_n is neighbor of x_i
else
  x_n is not neighbor
end if

Here, $d(i,n)$ is the Euclidean distance between food sources x_i and x_n with r being the radius of the neighborhood circle. md_i is the mean distance between x_i and each food source (Eq. (6)):

$$md_i = \frac{\sum_{k=1}^{SN} d(i,k)}{SN-1}. \tag{6}$$

If a food source is within the region calculated by multiplication of radius r and mean Euclidean distance md_i, this food source is a neighbor of x_i. It is obvious that when r is set to 0, the qABC converges to the standard ABC.

3.2. Food source representation

A food source stands as a possible solution for the problem to be optimized here and the hive represents the search space. A food source is represented as a list of (v_j, w_{u,v_j}) corresponding to user u's k-neighborhood, where v_j is the user id and w_{u,v_j} is the similarity value of user u and user j, which are the most similar k users $(1 \leq j \leq k)$ to user u. Similarities can take values from zero to unity.

A single food source has the following definition:

$$\mathbf{w}_u = [(v_1, w_{u,v_1}), \ldots, (v_k, w_{u,v_k})]. \tag{7}$$

Rows of such food sources constitute the food structure for the qABC algorithm. Note that it is trivial to represent this model as a chromosome in the GA, which is the second evolutionary algorithm presented in this paper. The evolutionary operators such as the phases in the qABC algorithm and cross-over or mutation in the GA are used to tackle the overspecialization problem by adapting the weights and improving solution diversity.

3.3. Fitness definition

The quality (fitness) of a food source is assessed depending on nectar amount, which is computed using an objective function. This function computes how accurate the prediction of the CF algorithm is. Mean absolute error (MAE) was selected as an objective function for this problem.

The fitness of a food source represents how likely it is that the food source will be chosen for the next iterations. This method is different from the work in [7] that employs a global fitness function, which encloses

all users, in that our method defines a fitness function separately for each user using MAE (8). Each user's similarity weight was enhanced independently of other users to obtain an overall decrease in MAE. Thus, the approach in this study is bottom-up rather than top-down.

$$MAE = \frac{1}{|I_u|} \sum_{i \in I_u} |\hat{r}_{u,i} - r_{u,i}|, \tag{8}$$

where I_u is the number of samples in the test data.

Greedy selection among food sources is applied based on this MAE value, taken as f_i, to model the fitness of the candidate solution that is evaluated using the following equation:

$$fit_i = \begin{cases} \frac{1}{1+f_i} & \text{if } f_i \geq 0, \\ 1 + |f_i| & \text{if } f_i < 0. \end{cases} \tag{9}$$

The same fitness definition was used in the GA as well.

3.4. Modification of employee and onlooker bee phases

As stated earlier, each food source represents the k-neighborhood of user u. To find k highly similar users of u, the bee colony is constructed using identical copies of the values for u. The principle of the ABC is that new food sources are produced by combining itself with a randomly selected food source among all other food sources at each iteration. Therefore, a modification has to be performed to generate new food sources; otherwise, exactly the same food sources would be reproduced.

The formulas used in employee and onlooker bee phases, Eqs. (10) and (11), were adjusted as follows:

$$v_{ij} = x_{ij} + \phi(\alpha * x_{ij} - (1 - \alpha) * x_{kj}), \tag{10}$$

$$v_{N_ij}^{best} = x_{N_ij}^{best} + \phi(\alpha * x_{N_ij}^{best} - (1 - \alpha) * x_{kj}), \tag{11}$$

where α was selected as 0.6, found by trial and error.

4. Experimental framework

4.1. Dataset

In this study, the dataset of [30] known as MovieLens 100K was employed to evaluate the evolutionary algorithms' performances. This dataset comprises the 100,000 ratings of 943 users on 1682 movies. The rating matrix density of the dataset is 0.0680. This sparse dataset includes ratings of each user for at least 20 movies.

4.2. Parameters

The following parameters inherent to the algorithms employed were empirically chosen and are listed as follows.

4.2.1. CF parameters

Important CF parameters are explained as follows:

- Neighborhood size (k): Evaluations were performed for $k \in (100, 200, \ldots, 800)$.

- CRI: The number of commonly rated items by two users. A default value of CRI was used to determine whether a similarity for two users can be calculated. The CRI value was chosen as 7, found empirically.
- The initial rating prediction was chosen as 3.0 for items for which a prediction cannot be computed since there is not a sufficient number of ratings for the same item.

4.2.2. qABC parameters

There are a few critical parameters to be specified in the qABC algorithm:

- SN: The number of food source positions.
- l: The number of trials to abandon a food source.
- MCN: Maximum number of cycles.
- r: Radius of the current solution's circular neighborhood.

SN is set to 60 and MCN is 250. For the scout bee phase, the limit value l is calculated as:

$$l = SN * D, \tag{12}$$

where D is the problem dimension, which is selected as 100 in our problem. The empirical results in [22] show that an r value around unity is appropriate for the qABC, selected as 1 in our implementation.

4.2.3. GA parameters

The following parameters are used to adjust the operation of the GA:

- Mutation rate: The likelihood of a mutation to happen for a gene to create diverse solutions, chosen as 0.015.
- Termination criterion: Number of generations required to stop the GA's evolving process. It is chosen as 70.
- Elite individual count: Individuals with best fitness values are reserved not to be affected by mutations for the next generation. This number was set to 5 considering 10 individuals for the initial case and 50 individuals in the maximum case.
- Alpha (α'): The weight of each parent for whole arithmetic recombination [6], 0.4, was found empirically for the first one and $1 - \alpha'$ was used for the second.

4.3. Statistical tests

4.3.1. The t-test

Randomness or statistical significance of the difference between two sets of data can be determined using the t-test.

4.3.2. Analysis of Variance (ANOVA)

As a generalization of the t-test for more than two samples, ANOVA performs hypothesis testing by comparing the means of several groups to see whether or not they are equal. There are two reasons for using ANOVA instead of running t-tests between pairs of groups: one reason is that pairwise comparisons between samples is time-consuming when there are more than two samples. Secondly and more importantly, one can expect to have one of twenty t-tests performed to be incorrect since this is a consequence of the 5% acceptance level.

4.3.3. McNemar's test

An alternative method that can be used to analyze pairwise data is McNemar's test. While being a nonparametric test, McNemar's test is similar to the χ^2 test in that it uses 2×2 contingency tables to compute z scores. It is widely adopted by different researchers working in medical science. This test was recently employed for performance evaluation in machine learning [31]. This is because the test is quite robust against type-I error, which occurs when the evaluation concludes that there is a performance difference between algorithms when there is none.

5. Results

The experiments were conducted on the dataset given in Section 4.1 for 10 folds. The number of neighbours (k) was selected as 100 to 800. Tables 3a–3d represent the mean values of 10 folds for the original metrics as well as the ones optimized using the evolutionary approaches presented in this paper, i.e. the GA and ABC, employing different similarity metrics. For the PCC metric shown in Table 3a, evolutionary approaches provide better results than the original weights (*Org*). While the GA yields the best performance from $k = 200$ to 700, the ABC has the best results for $k = 100$ and 800. Table 3b indicates that average results for the optimized weights using the GA and ABC reduce the original MAE values. The ABC has shown the best average for the EJC metric. The performance of the ABC for $k = 400$ to 800, i.e. 0.7597, is the lowest prediction error among all results. For $k = 200$, the GA used for optimization has superior accuracy, yielding the minimum value of 0.7552. Table 3c shows that VCS results in the lowest errors for all of the neighborhood sizes when the GA is used for optimization. When the ABC is used, it yields slightly inferior results than the GA but better results than the original weights. It can be seen that when the weights are optimized using the ABC, the EJC results in the lowest error values. On the other hand, the GA approach produced the lowest error values for the PCC and VCS metrics. Note that there are also results for randomly generated weights (RND) in Table 3d. The result of the optimization performed by ABC and GA can be clearly seen whereby both yield smaller error values than (*Org*). In conclusion, the improvement of the optimization is clearly visible for all metrics.

Figure 2 shows the best and worst case results for different metrics over varying number of neighbors comparing the evolutionary approaches with the original weights. In order to draw statistically meaningful conclusions, detailed analysis was performed using ANOVA, the t-test, and McNemar's test. The first test that will be discussed is ANOVA, which was used to find out whether there are performance differences between the metrics and the optimization approaches given here. Table 4 shows the ANOVA results. These tests were performed on two groups of data, which are grouped by the approaches and the metrics. The metrics were compared in the former case (RND was excluded), while the approaches are compared in the latter. Note that for both tables, `f-crit` is found as 3.0533 for ANOVA since the same number of samples was used. One can observe that for both tables $f \gg$ `f-crit`, with very small P-values indicating high confidence levels. These results show that there are indeed statistically significant performance differences between the different metrics

Table 3. Mean MAE of 10 folds ($\mu \pm \sigma$) for a- PCC, b- EJC, c- VCS, and d- RND. Bold values represent the smallest MAE for different k-neighborhoods.

(a)

K	PCC Org	PCC GA	PCC ABC
100	0.8317±0.0028	0.7921±0.0187	**0.7847±0.023**
200	0.8145±0.0024	**0.7734±0.0169**	0.7846±0.0229
300	0.8115±0.0026	**0.7709±0.0165**	0.7846±0.0229
400	0.8110±0.0025	**0.7736±0.0164**	0.7846±0.0229
500	0.8111±0.0027	**0.7751±0.0162**	0.7848±0.0231
600	0.8111±0.0027	**0.7795±0.0142**	0.7847±0.0229
700	0.8112±0.0027	**0.7805±0.0137**	0.7845±0.0228
800	0.8371±0.0690	0.8068±0.0716	**0.7846±0.0226**

(b)

K	PCC Org	PCC GA	PCC ABC
100	0.7968±0.0029	0.7799±0.0164	**0.7597±0.0240**
200	0.7877±0.0024	**0.7552±0.0235**	0.7595±0.0239
300	0.7913±0.0018	0.7683±0.0129	**0.7597±0.0240**
400	0.7985±0.0022	0.7740±0.0115	**0.7597±0.0241**
500	0.8064±0.0021	0.7807±0.0109	**0.7597±0.0238**
600	0.8126±0.0023	0.7858±0.0108	**0.7597±0.0239**
700	0.8152±0.0024	0.7881±0.0106	**0.7596±0.0239**
800	0.8161±0.0022	0.7915±0.0101	**0.7597±0.0239**

(c)

K	PCC Org	PCC GA	PCC ABC
100	0.8297±0.0029	**0.7716±0.0358**	0.7886±0.0218
200	0.8139±0.0024	**0.7574±0.0301**	0.7885±0.0219
300	0.8106±0.0018	**0.7573±0.0276**	0.7888±0.0219
400	0.8123±0.0020	**0.7604±0.0251**	0.7887±0.0218
500	0.8153±0.0021	**0.7647±0.0237**	0.7886±0.0217
600	0.8181±0.0024	**0.7676±0.0233**	0.7886±0.0219
700	0.8194±0.0024	**0.7693±0.0232**	0.7886±0.0219
800	0.8198±0.0023	**0.7703±0.0231**	0.7886±0.0216

(d)

K	PCC Org	PCC GA	PCC ABC
100	0.8526±0.0031	0.8098±0.0168	**0.8039±0.0207**
200	0.8360±0.0022	**0.7957±0.0139**	0.8040±0.0206
300	0.8296±0.0022	**0.7916±0.0132**	0.8039±0.0205
400	0.8269±0.0024	**0.7896±0.0123**	0.8039±0.0206
500	0.8254±0.0024	**0.7890±0.0126**	0.8039±0.0207
600	0.8248±0.0024	**0.7878±0.0124**	0.8039±0.0206
700	0.8245±0.0023	**0.7931±0.0079**	0.8038±0.0206
800	0.8244±0.0023	**0.7937±0.008**	0.8040±0.0207

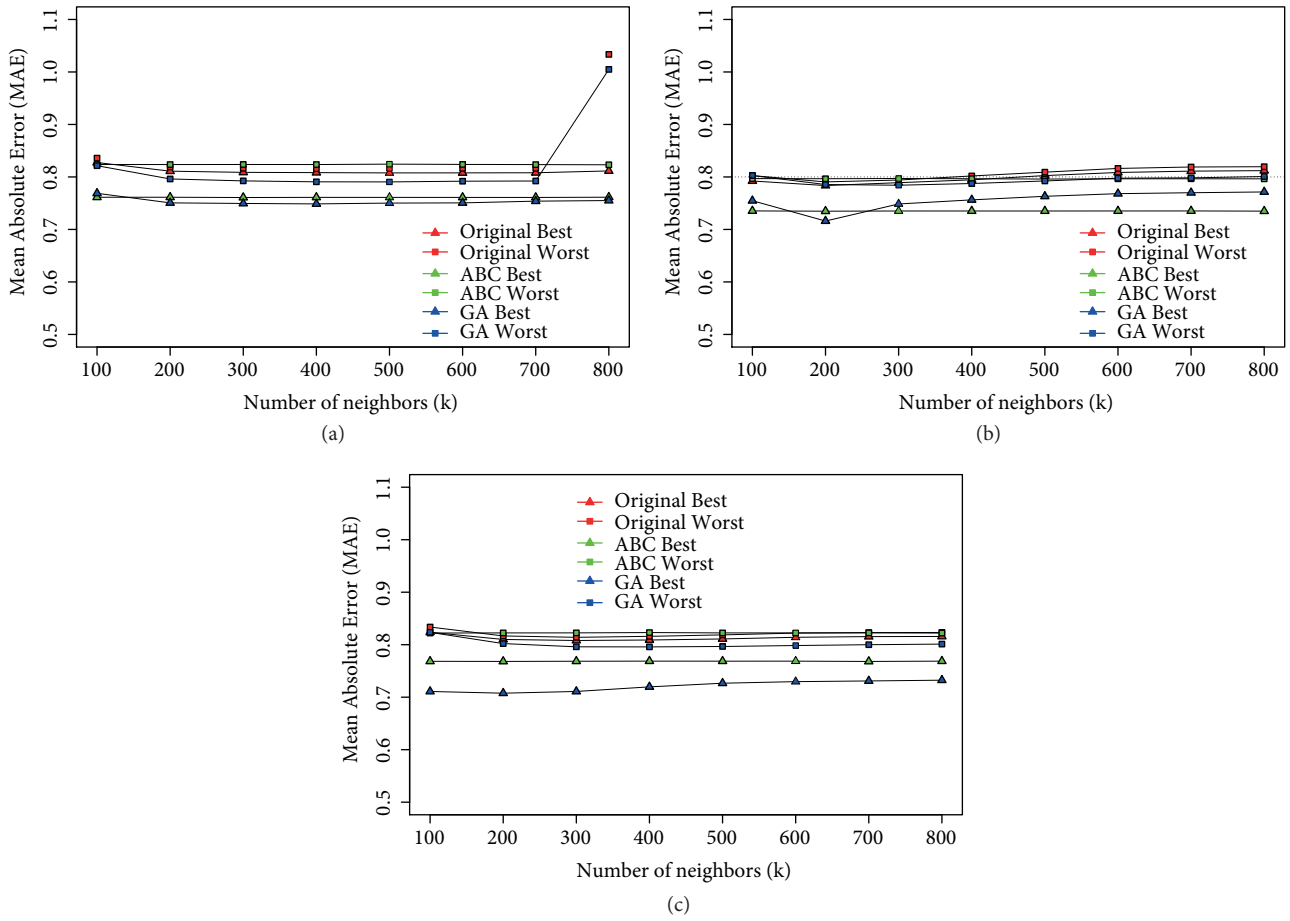

Figure 2. Best and worst results for optimized weights with ABC and GA for PCC, EJC, and VCS: a- PCC results, b- EJC results, c- VCS results.

and approaches, rejecting the null hypothesis H_0 that suggests that different approaches or metrics would yield similar results.

Table 4. a- ANOVA results for metrics, b- ANOVA results for approaches.

(a)

Approaches	f	**P**
Original	23.7318	9.72E-10
ABC	4830.9610	2.1E-142
GA	14.0334	2.46E-06

(b)

Metrics	f	**P**
PCC	64.5340	3.26E-21
EJC	148.4380	5.25E-37
VCS	162.8738	4.06E-39

ANOVA suggested that there are variations between groups; however, it is important to identify pairwise differences as well, and the t-test was used for this purpose. t-test results (see Table 5) show that MAE values obtained using both ABC and GA approaches are significantly better than the MAE values for the original weights computed using PCC. The confidence level is very high, $\mathrm{P}(T \leq t) = 0$ for both one- and two-tailed predictions, in the *Original* versus ABC and GA comparison. However, there is no significant difference between

the performances of ABC and GA approaches with respect to the PCC metric. For EJC and VCS metrics, both ABC and GA outperformed the *Original* approach with high confidence levels. ABC generates significantly better predictions than GA on the EJC metric. On the other hand, MAE values for the GA are significantly lower than that of the ABC. The RND column has similar statistical values. In this case, it is clear that the GA and ABC approaches are significantly superior to the *Original* approach and the GA is significantly better than the ABC.

Table 5. The t-test results for all approaches grouped by metrics.

PCC				**EJC**			
	Original	***ABC***	***GA***		***Original***	***ABC***	***GA***
Mean	0.8174	0.7846	0.7815	Mean	0.8031	0.7597	0.7779
Variance	0.0006	0.0005	0.0009	Variance	0.0001	0.0005	0.0003
	Orig.-ABC	**Orig.-GA**	**ABC-GA**		**Orig.-ABC**	**Orig.-GA**	**ABC-GA**
t Stat	**8.7314**	**8.1027**	**0.757011**	**t Stat**	**15.4143**	**11.0172**	**-5.69238**
P(T≤t) one-tail	0.0000	7.95E-14	0.225144	P(T≤t) one-tail	0.0000	1.11E-20	3.26E-08
t Crit. one-tail	**1.6548**	**1.6549**	**1.655579**	**t Crit. one-tail**	**1.6586**	**1.6566**	**1.655215**
P(T≤t) two-tail	0.0000	1.59E-13	0.450289	P(T≤t) two-tail	0.0000	2.21E-20	6.51E-08
t Crit. two-tail	**1.9755**	**1.9756**	**1.976692**	**t Crit. two-tail**	**1.9814**	**1.9782**	**1.976122**
VCS				**RND**			
	Original	***ABC***	***GA***		***Original***	***ABC***	***GA***
Mean	0.8174	0.7886	0.7648	Mean	0.8305	0.8039	0.7938
Variance	0.0000	0.0004	0.0007	Variance	0.0001	0.0004	0.0002
	Orig.-ABC	**Orig.-GA**	**ABC-GA**		**Orig.-ABC**	**Orig.-GA**	**ABC-GA**
t Stat	**11.8644**	**17.5037**	**6.367019**	**t Stat**	**10.9022**	**19.8573**	**3.784936**
P(T≤t) one-tail	0.0000	1.51E-30	1.11E-09	P(T≤t) one-tail	0.0000	5.55E-43	0.000114
t Crit. one-tail	**1.6616**	**1.6626**	**1.655076**	**t Crit. one-tail**	**1.6583**	**1.6557**	**1.655811**
P(T≤t) two-tail	0.0000	3.01E-30	2.22E-09	P(T≤t) two-tail	0.0000	1.11E-42	0.000227
t Crit. two-tail	**1.9861**	**1.9876**	**1.975905**	**t Crit. two-tail**	**1.9810**	**1.9769**	**1.977054**

Final statistical results were obtained from McNemar's test. This test both identifies which method is performing better than others and how this result is statistically meaningful. Tables 6a and 6b show that there are significant performance differences. Looking at Table 6a, one can confidently say that EJC performed better than both PCC and VCS for the original weights with z scores of 3.4659 and 8.8325, respectively. It is worth mentioning that z scores higher than 2.576 suggest 99.5% and 99% confidence levels for one- and two-tailed predictions, correspondingly [31]. Similar results were obtained for the optimized weights when ABC was employed. On the other hand, PCC showed better performance than EJC when the GA was used; however, this result is not conclusive since the z score is less than 1.645. When comparing PCC and VCS, a z score of 1.9007 suggests more than 95% and 90% confidence for one- and two-tailed predictions. When $z = 0$, one can conclude that there are no statistically significant performance differences between EJC and VCS. The results of Table 6 actually confirm the t-test results. One example of this is that the test could not find a statistically meaningful difference between ABC and GA for the PCC metric. The rest of the z scores for this table indicate significant performance differences with high confidence levels. As another interesting aspect, the GA works better with VCS, while the ABC works better with EJC.

It is also interesting to observe that the results obtained by the bioinspired approaches presented in our study are quite comparable with a benchmark library given in [32]. For instance, MyMediaLite's UserKNN-

Table 6. a- McNemar's test results for metrics grouped by approaches. Arrowheads indicate the approach/metric giving superior performance. b- McNemar's test results for approaches grouped by metrics. Arrowheads indicate the approach/metric giving superior performance.

(a)

Original		
	EJC	**VCS**
PCC	3.4659 ↑	3.4659 ←
EJC		8.8325 ←
ABC		
	EJC	**VCS**
PCC	8.8325 ↑	5.2548 ←
EJC		8.8325 ←
GA		
	EJC	**VCS**
PCC	0.559 ←	1.9007 ↑
EJC		0

(b)

PCC		
	ABC	**GA**
Original	5.7020 ↑	8.8325 ↑
ABC		0.1118 ↑
EJC		
	ABC	**GA**
Original	7.4908 ↑	8.6089 ↑
ABC		3.9131 ←
VCS		
	ABC	**GA**
Original	5.7020 ↑	8.8325 ↑
ABC		3.9131 ↑

Pearson method resulted in a MAE of 0.7281, while our ABC and GA approaches yielded 0.7845 and 0.7709, respectively. It is worth mentioning here that the original weight for this approach was 0.8115. The methods presented here improve the user-to-user similarities on given original weights computed with different similarity metrics. The results given therein are better here since the original weights are computed with a different formula in CF. The authors believe that the bioinspired approaches are capable of improving their results as well.

6. Conclusion

This study employed two biologically inspired approaches, namely the ABC and GA, to optimize the computed weights using various similarity metrics such as PCC, EJC, and VCS in CF. In the proposed approaches, a bottom-up method was followed in such a way that each user was considered individually with the expectation that if each user's similarity weights with the other users were optimized independently, overall improvements could be achieved.

The experiments were conducted using 10-fold cross-validation of the MovieLens-100K dataset. First, user-to-user similarity values were computed and used to produce predictions with those weights for varying neighborhood sizes in the CF framework. Then these *Original* weights were optimized using the qABC (a derivative of ABC) and GA approaches. CF was applied with these weights to generate predictions. Finally, detailed statistical tests indicated that the GA and ABC approaches produce significantly better improvements on similarity weights. It can be seen that the results are quite comparable to the work in [32]. When using the qABC, a modification was also employed to prevent the reproduction of the same food sources since the original weights were cloned to construct the initial bee colony. This modification was required in employee and onlooker bee phases. A simple heuristic was used here to overcome this problem by combining the two food sources arithmetically.

In both approaches, the sets of users initially found using various metrics were not changed during the optimization process; only the similarity values were adjusted. It would be an interesting research direction to

improve these approaches by allowing new users, other than the initially computed ones, to get into the similar users' neighborhoods. A second possible direction would be integrating user-specific information (age, gender, etc.) into the selection of initial similar users.

References

[1] Simon D. Social media equals social customer: managing customer experience in the age of social media. iUniverse, 2013.

[2] Resnick P, Varian HR. Recommender systems. Communications of the ACM 1997; 40: 56-58.

[3] Lops P, De Gemmis M, Semeraro G. Content-based recommender systems: state of the art and trends. In: Ricci F, Rokach L, Shapira B, Kantor PB (editors). Recommender Systems Handbook. Berlin, Germany: Springer, 2011, pp. 73-105.

[4] Schafer JB, Frankowski D, Herlocker J, Sen S. Collaborative filtering recommender systems. In: Brusilovsky P, Kobsa A, Nejdl W (editors). The Adaptive Web. Berlin, Germany: Springer-Verlag, 2007, pp. 291-324.

[5] Balabanovic M, Shoham Y. Combining content-based and collaborative recommendation. Communications of the ACM 1997; 40: 66-72.

[6] Eiben AE, Smith JE. Introduction to Evolutionary Computing. Berlin, Germany: Springer-Verlag, 2003.

[7] Bobadilla J, Ortega F, Hernando A, Alcalá J. Improving collaborative filtering recommender system results and performance using genetic algorithms. Knowledge-Based Systems 2011; 24: 1310-1316.

[8] Hsu CC, Chen HC, Huang KK, Huang YM. A personalized auxiliary material recommendation system based on learning style on facebook applying an artificial bee colony algorithm. Computers and Mathematics with Applications 2012; 64: 1506-1513.

[9] Anand D. Feature extraction for collaborative filtering: a genetic programming approach. International Journal Computer Science Issues 2012; 9 (5): 348.

[10] Velez-Langs O, De Antonio A. Learning user's characteristics in collaborative filtering through genetic algorithms: some new results. Advance Trends in Soft Computing 2014; 312: 309-326.

[11] Ju C, Xu C. A new collaborative recommendation approach based on users clustering using artificial bee colony algorithm. Scientific World Journal 2013; 2013: 869658.

[12] Akbari F, Tajfar A, Nejad A. Graph-based friend recommendation in social networks using artificial bee colony. In: IEEE Conference on Dependable, Autonomic and Secure Computing; 2013. pp. 464-468.

[13] Breese JS, Heckerman D, Kadie C. Empirical analysis of predictive algorithms for collaborative filtering. In: Fourteenth Conference on Uncertainty in Artificial Intelligence; San Francisco, CA, USA; 1998. pp. 43-52.

[14] Resnick P, Iacovou N, Suchak M, Bergstrom P, Riedl J. Grouplens: An open architecture for collaborative filtering of netnews. In: 1994 ACM Conference on Computer Supported Cooperative Work; Chapel Hill, NC, USA; 1994. pp. 175-186.

[15] Herlocker JL, Konstan JA, Borchers A, Riedl J. An algorithmic framework for performing collaborative filtering. In: 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval; New York, NY, USA; 2017. pp. 227-234.

[16] Golbeck J, Hendler J. Filmtrust: Movie recommendations using trust in web-based social networks. In: IEEE Consumer Communications and Networking Conference; Las Vegas, NV, USA; 2006. pp. 282-286.

[17] Sarwar B, Karypis G, Konstan J, Riedl J. Item-based collaborative filtering recommendation algorithms. In: 10th International Conference on WWW; Nanjing, China; 2012. pp. 285-295.

[18] Linden G, Smith B, York J. Amazon.com recommendations: item-to-item collaborative filtering. Internet Computing IEEE 2003; 7 (1): 76-80.

[19] Konstan JA, Miller BN, Maltz D, Herlocker JL, Gordon LR et al. Grouplens: Applying collaborative filtering to usenet news. Communications of the ACM 1997; 40: 77-87.

[20] Salehi M, Pourzaferani M, Razavi SA. Hybrid attribute-based recommender system for learning material using genetic algorithm and a multidimensional information model. Egyptian Informatics Journal 2013; 14: 67-78.

[21] Ar Y, Bostanci E. A genetic algorithm solution to the collaborative filtering problem. Expert Systems with Applications 2016; 61: 122-128.

[22] Karaboga D, Gorkemli B. A quick artificial bee colony (qABC) algorithm and its performance on optimization problems. Applied Soft Computing 2014; 23: 227-238.

[23] Draa A, Bouaziz A. An artificial bee colony algorithm for image contrast enhancement. Swarm and Evolutionary Computation 2014; 16: 69-84.

[24] Agrawal S, Sahu O. Artificial bee colony algorithm to design two-channel quadrature mirror filter banks. Swarm and Evolutionary Computation 2015; 21: 24-31.

[25] Yigit S, Tugrul B, Celebi FV. A complete cad model for type-I quantum cascade lasers with the use of artificial bee colony algorithm. Journal of Artificial Intelligence 2012; 5: 76-84.

[26] Subotic M, Mialn T, Stanarevic N. Different approaches in parallelization of the artificial bee colony algorithm. International Journal of Mathematical Models and Methods in Applied Sciences 2011; 5: 755-762.

[27] Zhu G, Kwong S. Gbest-guided ABC algorithm for numerical function optimization. Applied Mathematics and Computation 2010; 217: 3166-3173.

[28] Gao WF, Liu SY. A modified artificial bee colony algorithm. Computers and Operations Research 2012; 39: 687-697.

[29] Banharnsakun A, Achalakul T, Sirinaovakul B. The best-so-far selection in artificial bee colony algorithm. Applied Soft Computing 2011; 11: 2888-2901.

[30] Harper FM, Konstan JA. The MovieLens datasets: history and context. ACM Transactions on Interactive Intelligent Systems 2015; 5 (4): 19:1-19:19.

[31] Bostanci B, Bostanci E. An evaluation of classification algorithms using McNemar's test. In: Seventh International Conference on Bio-Inspired Computing: Theories and Applications, Advances in Intelligent Systems and Computing; India; 2013. pp. 15-26.

[32] Gantner Z, Rendle S, Freudenthaler C, Schmidt-Thieme L. MyMediaLite: A free recommender system library. In: 5th ACM Conference on Recommender Systems; Chicago, IL, USA; 2011. pp. 305-308.