Applied Intelligence (2019) 49:188–205
https://doi.org/10.1007/s10489-018-1261-8

Chaotic dragonfly algorithm: an improved metaheuristic algorithm for feature selection

Gehad Ismail Sayed[1] · **Alaa Tharwat**[2,3] · **Aboul Ella Hassanien**[1,4]

Published online: 21 August 2018

Abstract
Selecting the most discriminative features is a challenging problem in many applications. Bio-inspired optimization algorithms have been widely applied to solve many optimization problems including the feature selection problem. In this paper, the most discriminating features were selected by a new Chaotic Dragonfly Algorithm (CDA) where chaotic maps embedded with searching iterations of the Dragonfly Algorithm (DA). Ten chaotic maps were employed to adjust the main parameters of dragonflies' movements through the optimization process to accelerate the convergence rate and improve the efficiency of DA. The proposed algorithm is employed for selecting features from the dataset that were extracted from the Drug bank database, which contained 6712 drugs. In this paper, 553 drugs that were bio-transformed into liver are used. This data have four toxic effects, namely, irritant, mutagenic, reproductive, and tumorigenic effect, where each drug is represented by 31 chemical descriptors. The proposed model is mainly comprised of three phases; data pre-processing, features selection, and the classification phase. In the data pre-processing phase, Synthetic Minority Over-sampling Technique (SMOTE) was used to solve the problem of the imbalanced dataset. At the features selection phase, the most discriminating features were selected using CDA. Finally, the selected features from CDA were used to feed Support Vector Machine (SVM) classifier at the classification phase. Experimental results proved the capability of CDA to find the optimal feature subset, which maximizing the classification performance and minimizing the number of selected features compared with DA and the other meta-heuristic optimization algorithms. Moreover, the experiments showed that Gauss chaotic map was the appropriate map to significantly boost the performance of DA. Additionally, the high obtained value of accuracy (81.82–96.08%), recall (80.84–96.11%), precision (81.45–96.08%) and F-Score (81.14–96.1%) for all toxic effects proved the robustness of the proposed model.

Keywords Toxic effects · Dragonfly algorithm · Feature selection · Optimization algorithm · Chaos theory

1 Introduction

Dimensionality is one of the main problems that may reduce the classification performance. There are many applications that depend on high dimensional datasets with hundreds or even thousands of features. Some of these features are redundant, noisy, or irrelevant, which may reduce the classification performance [1]. Hence, Feature Selection (FS) is an important preprocessing step in machine learning applications. The goal of the FS step is to eliminate redundant features to improve the effectiveness of learning algorithms and reduce the required memory and computational time [2].

Feature selection methods have two main categories. First, *filter methods* that depend on some data properties without using learning algorithms. On the contrary, *wrapper methods* which used learning algorithms for evaluating the selected subset of features. Wrapper methods are more accurate than filter methods; however, wrapper methods are computationally expensive [3].

The feature selection problem is formulated as a multi-objective optimization problem, and the goal of this problems

✉ Alaa Tharwat
aothman@fb2.fra-uas.de

Gehad Ismail Sayed
GehadIsmail_FCI@yahoo.com

1 Faculty of Computers and Information, Cairo University, Giza, Egypt

2 Faculty of Computer Science and Engineering, Frankfurt University of Applied Sciences, Frankfurt am Main, Germany

3 Faculty of Engineering, Suez Canal University, Ismailia, Egypt

4 Scientific Research Group in Egypt, (SRGE), http://www.egyptscience.net

is to (1) minimize the size of the selected features and hence decrease the required computational efforts, and (2) maximize the classification performance [3–5]. The size of the search space exponentially increases with the number of features; therefore, an exhaustive search to solve this problem is almost impossible in practice. Evolutionary computing algorithms and other swarm-based algorithms adaptively search the feature space by employing a set of search agents or candidates that communicate in a social manner to reach a global solution [6].

Many Bio-inspired optimization algorithms have been used for searching for the most discriminative features. For example, Ant Colony algorithm was employed for selecting features [7, 8]. Particle Swarm Optimization (PSO) has been widely used for extracting features. Moradi *et al.* enhanced the PSO algorithm by employing a local search for finding less correlated features [9]. A novel Artificial Bee Colony (ABC) has been used for extracting the most discriminative features [10, 11]. The Antlion optimizer has been utilized as a search strategy in wrapper feature selection method [12, 13]. Two optimization algorithms, namely, Differential Evolution (DE) and ABC algorithms, are combined for optimizing feature selection problem [14]. A modified version of the Salp Swarm Algorithm (SSA) was used for feature selection [3, 15].

In this paper, the Dragonfly Algorithm (DA) is used for feature selection. DA is one of the recent meta-heuristic optimization algorithms. It is used to solve single-objective and multi-objective problems [16]. In general, DA as many other optimization algorithms lends itself strongly to exploitation which may lead to the local optima problem. Hence, in some cases, DA fails to find the global optimal solution. Therefore, different strategies have been added to the optimization algorithms to improve its performance [17, 18]. In this research, chaos theory has been combined with DA and the proposed Chaotic Dragonfly Algorithm (CDA) was used to search for the most discriminating features.

The development of new drugs is a complex and an expensive process, and it has several steps, and toxicity assessment of drugs' components is one of these steps [19]. This step is vital as it is used to predict drug failures before any clinical trials. Therefore, measuring toxicity for thousands of compounds becomes a hot topic in the recent studies [20]. However, reliable high-throughput assays are expensive; thus, there is a high demand for computational models. Computational models offer a cheap and fast alternative to in-vivo and in-vitro bioassays. Moreover, the computational model saves experimental materials and protects animals [21]. The goal of computational models is to predict the toxicity of chemical compounds, the effect of different concentrations of the chemical compounds or the toxicological endpoints accurately. There are many examples of available computer models predicting toxicity such as Case [22], TOPKAT [23] and DEREK [24]. One of the main problems of computational toxicity models is the large amount of data, which makes the process of analyzing this data more difficult because not all the information is relevant. Selecting the relevant information is a complex problem. The feature selection technique is employed to identify a subset of the features for improving the classification performance and providing a faster classification, which leads to a comparable classification accuracy than using all features.

In this paper, a machine-learning model was proposed to evaluate the toxicity of hepatic drugs. The toxicity risks of the current drugs include mutagenic effect, tumorigenic effect, irritant effect and reproductive effect. The current dataset is imbalanced (see Section 2), i.e. the samples of one class significantly outnumber the samples of the other class. The proposed model consists of three phases. In the first phase, i.e. the data pre-processing phase, the Synthetic Minority Oversampling Technique (SMOTE) algorithm was used for creating minority class samples to obtain balanced samples in each class. In the second phase, i.e. the feature selection phase, the most discriminative features were selected using the proposed CDA algorithm. The selected features were then used to train the SVM classifier in the third phase, i.e. the classification phase. The SVM classifier was then used to classify an unknown drug into toxic, i.e. has one of the toxic effects, or non-toxic.

The rest of this paper is organized as follows: Section 2 presents a brief description of the dataset that are used in our proposed model. Theoretical background and steps of the proposed model are presented in Sections 3 and 4, respectively. Experimental scenarios and discussions are introduced in Section 5. Finally, conclusions and future work are presented in Section 6.

2 Description of the dataset

In this research, the data was extracted from the Drug bank database, which contained 6712 drugs. These drugs are classified as follows: 1448 FDA-approved small molecule drugs, 131 FDA-approved biotech (protein/peptide) drugs, 85 nutraceuticals and 5080 experimental drugs. We utilized 553 drugs that were biotransformed in liver [25]. Each drug is represented by 31 features, which were calculated using DataWarrior package [25]. These features are recorded in Table 1. The current dataset includes four toxic effects as presented in Table 2, where the mutagenic, irritant and tumorigenic effects have a high imbalance ratio (the ratio of the number of samples of the majority class to the number of samples of the minority class), while the reproductive effect has a low imbalance ratio.. In this paper,

Table 1 Dataset description

Feature No.	Name
1	Total Molecular Weight
2	Molecular Weight
3	Absolute Weight
4	cLogP
5	cLogS
6	H-Acceptors (Hydrogen bond Acceptor)
7	H-Donors (Hydrogen bond donor)
8	Total Surface Area
9	Polar Surface Area
10	Druglikeness
11	Shape Index
12	Molecular Flexibility
13	Molecular Complexity
14	Non H Atoms
15	Non C/H Atoms
16	Metal Atoms
17	Electron Negative Atoms
18	Stereo Centers
19	Rotatable Bonds
20	Rings
21	Aromatic Rings
22	Aromatic Atoms
23	sp3-Atoms
24	Symmetric atoms
25	Amides (acid amide)
26	Amines
27	AlkylAmines
28	Aromatic Amines
29	Aromatic Nitrogen
30	Basic Nitrogen
31	Acidic Oxygen

the positive class represents the minority class, which may have a negative impact on the sensitivity of the proposed model. Moreover, we considered each toxic effect as a separate dataset.

As shown in Table 2, the reproductive effect considered the top risk effect (33.82%); and both of mutagenic and tumorigenic effects are equal to (16.28%), finally irritant effect with (12.16%) for the current FDA drugs, which reflects the burden on the liver and such drugs should be replaced with safer medications.

3 Theory and methods

3.1 Imbalanced Datasets

The problem of imbalanced datasets appears when the number of samples of one class, i.e. majority class (S_{maj}), is significantly higher than the samples of the other one, i.e. minority class (S_{min}) [26–28]. This problem decreases the classification performance. This is because, (1) it is difficult for the classifier or learning algorithm to learn from a minority class; hence, the minority class samples are misclassified frequently; (2) using global assessment methods such as accuracy to evaluate the learning algorithm may provide an advantage to the majority class [28]. There are many methods used to solve the imbalanced dataset problem such as sampling methods [28], Kernel-Based methods [27], and Cost-Sensitive methods [27]. In this paper, due to the simplicity and efficiency of sampling methods, these methods are used for obtaining balanced data.

The aim of sampling methods is to modify the prior distribution of the majority and minority classes in the training phase to obtain more balanced samples in each class. There are two well-known sampling methods: *Random Under-sampling* (RUS) and *Random Over-sampling* (ROS). In the RUS method, the goal is to extract a small set of majority class samples to train the classifier while preserving all minority class samples. Therefore, the training data becomes more balanced and the training process becomes faster. However, the removed samples may have useful information; hence, the RUS method may decrease the classification performance [27]. In the ROS method, the aim is to increase the size of the minority class by replicating a set of minority samples. Thus, this method improves the minority class recognition. However, making exact copies of minority class samples increases the learning time and may lead to the overfitting problem [27].

3.1.1 Synthetic minority oversampling technique (SMOTE)

Instead of removing samples from the majority class as in the RUS method, or replicating a set of samples from

Table 2 Distribution of the two classes of each toxic effect

Toxic effect	# Samples in Positive Class	# Samples in Negative Class	Imbalance ratio
Mutagenic effect	90=16.28%	463=83.73%	5.14
Tumorigenic effect	90=16.28%	463=83.73%	5.14
Reproductive effect	187=33.82%	366=66.18%	1.96
Irritant effect	67=12.16%	486=87.88%	7.25

minority class as in the ROS method, the Synthetic Minority Over-sampling Technique (SMOTE) method creates data based on the similarities between existing minority samples. In the SMOTE method, minority class samples are over-sampled by creating synthetic samples. For each sample in the minority class $x_i \in S_{min}$, k nearest neighbors/samples are selected, and a synthetic sample can be generated as follow, $x_{new} = x_i + r_{ij} \times \delta = x_i + (\hat{x_{ij}} - x_i) \times \delta$, where $x_i \in S_{min}$ is one of the minority class samples, $\hat{x_{ij}}$ is one of the k-nearest neighbors for $x_i : \hat{x_{ij}} \in S_{min}$, $j = 1, 2, \ldots, k$, k is the number of selected neighbors, $\delta \in [0, 1]$ is a random number, and x_{new} is a point/sample along the line joining x_i and $\hat{x_{ij}}$. In other words, a synthetic sample is created by randomly selecting one of the k nearest neighbors ($\hat{x_{ij}}$) and multiplying a random number (δ) with the corresponding feature vector difference (r_{ij}) and then adding this vector to x_i [27, 29].

Figure 1 illustrates an example of the SMOTE method. Figure 1a shows a typical imbalanced data distribution, where the circles and squares represent samples of the majority and minority classes, respectively. In this example, assume $k = 4$. Figure 1a shows the created samples along the line segment between x_i and $\hat{x_{ij}}$. As shown, green squares highlight the created samples. These synthetic samples increase the number of minority samples and hence significantly improve the performance of a learning algorithm. Figure 1b shows the synthesized samples that are highlighted by solid squares. More details about the SMOTE algorithm can be found in [29].

3.1.2 Assessment methods for imbalanced datasets

Accuracy is a well-known assessment method, and it represents the ratio between the correctly classified samples to the total number of samples as in (1). Hence, the accuracy cannot distinguish between the number of correctly classified samples from different classes. Therefore, in imbalanced datasets, the accuracy may lead to erroneous conclusions [30, 31]. Therefore, instead of using accuracy in imbalanced datasets, different assessment methods can be used. *Precision* and *Recall* are two appropriate metrics to measure the performance of classification over imbalanced datasets. Precision is the proportion of positive samples that were correctly classified to the total number of positive predicted samples as denoted in (2). Recall, sensitivity, or True Positive Rate (TPR) measures how well the toxicity model detects the toxic effect or positive case as denoted in (3). However, the precision and recall are not sensitive to the changes in data distribution; hence, both measures can effectively evaluate the classification performance in imbalanced learning scenarios [27]. The F-Score metric combines both precision and recall and it is defined as denoted in (4). Therefore, F-Score is suitable in imbalanced scenarios than the accuracy metric [27].

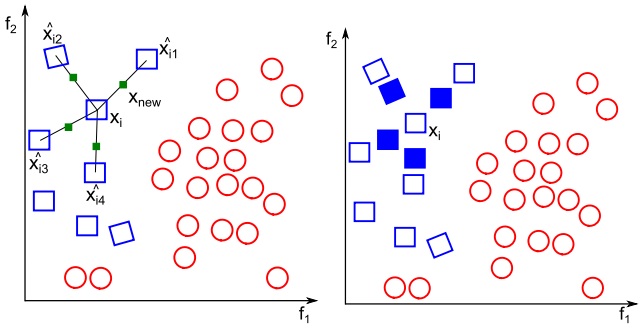

Fig. 1 An example of the SMOTE method; **a** before applying the SMOTE method and **b** after applying the SMOTE method

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}, \tag{1}$$

$$\text{Precision} = \frac{TP}{TP + FP}, \tag{2}$$

$$\text{Recall} = \frac{TP}{TP + FN}, \tag{3}$$

$$F\text{-Score} = \frac{2 * (\text{Precision} * \text{Recall})}{\text{Precision} + \text{Recall}} \tag{4}$$

where TP is the true positive (the number of correctly predicted toxic compounds), FN is the false negative (the number of toxic compounds but not predicted to be toxic), TN represents the true negative (the number of compounds that are correctly predicted to be not toxic) and FP is the false positive (the number of not toxic compounds, but predicted to be toxic).

3.2 Dragonfly algorithm

3.2.1 Inspiration

Dragonfly Algorithm (DA) is one of the recent bio-inspired optimization algorithms, and it was developed by S. Mirjalili in 2016 [16]. The main inspiration of this algorithm came from dynamic and static behaviors of swarming of dragonflies in nature. Dragonflies or Odonata are kinds of fancy insects. Nearly 3000 different kinds of this insect are found around the world. Nymph and adult are the two main stages of the life cycle of a dragonfly. However, they spend most of their lifespan in nymph and then undergo metamorphism to be an adult. Dragonflies are one of the small predators. As they feed in nature on almost all other small insects. Moreover, nymph dragonflies feed on small fishes and other kinds of marine insects [16].

Dragonfly algorithm has only two goals: hunting (static swarm) and migration (dynamic swarm). In the static

swarm, dragonflies create small groups and fly back and forth within a small area to hunt other flying preys. The main flying path characteristic of the static swarm is abrupt changes and local movements. However, this characteristic is different from the dynamic swarm, as large dragonflies, number of migrating propose over a long distance make the swarm in only one direction. Static and dynamic behaviors of dragonflies are very similar to exploitation and exploration of meta-heuristics phases. This similarity came from creating sub-swarms and fly over many regions in the dynamic swarm, which is the main purpose of the exploration phase while, in the static swarm, the dragonflies fly in bigger swarms along one direction which is the main purpose of the exploitation phase [16].

3.2.2 Mathematical model

There are three primitive principles of swarming behavior:-

1. **Separation**: this term refers to the avoidance of individuals' static collision from the others.
2. **Alignment**: this term refers to the individual velocity matching to other neighborhood individuals.
3. **Cohesion**: This term indicates to the individuals' tendency towards the neighborhood' mass center.

As survival is the main goal of any swarm; hence, all individuals are distracted outward enemies and attracted towards to food sources. Due to this kind of behavior, five main parameters affect the individuals' updating position: separation, alignment, cohesion, attraction towards a food source and distraction outwards an enemy. These parameters are mathematically defined as follows.

1. **Separation**: the separation is calculated as follows:

$$S_i = -\sum_{k=1}^{M} Y - Y_k, \tag{5}$$

where Y is defined as the individual's current position, Y_k is the position of the $k\text{-}th$ neighboring individual and M is the total number of neighboring individuals [16, 32].

2. **Alignment**: this parameter represents the mean of the velocities of all neighbors and it is calculated as follows:

$$A_i = \frac{\sum_{k=1}^{M} V_k}{M}, \tag{6}$$

where V_k is the velocity of $k\text{-}th$ neighboring individual.

3. **Cohesion**: this parameter is calculated as follows:

$$C_i = \frac{\sum_{k=1}^{M} Y_k}{M} - Y \tag{7}$$

4. **Attraction towards a food source**: this parameter represents the distance between the position of the current individual and the position of the food source (Y^+) and it is calculated as follow:

$$F_i = Y^+ - Y \tag{8}$$

5. **Distraction outwards an enemy**: this is the distance between the position of the current individual and the position of the enemy (Y^-) and it is calculated as follows:

$$E_i = Y^- - Y \tag{9}$$

The dragonfly behavior is the combination of these five parameters. Two vectors are used to update dragonflies' positions in the search space, namely, step vector (ΔY) and position vector (Y). Step vector is defined as follows:

$$\Delta Y_{t+1} = (aA_i + sS_i + cC_i + eE_i + fF_i) + w\Delta Y_t, \tag{10}$$

where a is defined as alignment weight, A_i is the alignment of the $i\text{-}th$ individual, s is defined as separation weight, S_i is the separation of the $i\text{-}th$ individual, c represents the cohesion weight, C_i is the cohesion of the $i\text{-}th$ individual, e is defined as enemy weight, E_i is the position enemy of the $i\text{-}th$ individual, f is defined as food weight, F_i is the food source of the $i\text{-}th$ individual, w is the inertia weight, and t is defined as the iteration number.

Overall, the individual position vector is defined as follows:

$$Y_{t+1} = Y_t + \Delta Y_{t+1} \tag{11}$$

During the optimization process, different exploitative and explorative behaviors are achieved using these parameters (a, s, c, e, and f). In other words, these parameters are used to balance between exploration and exploitation phases.

The dragonflies' convergence is guaranteed through algorithm iterations, as the parameters weight are changed adaptively. The flying path of dragonflies also adjusted as the optimization process proceeds.

The algorithm uses a random walk (Lèvy flight) to enhance the stochastic, randomness and DA exploration. The updating position of dragonflies is defined as follows:

$$Y_{t+1} = Y_t + Lèvy(d) \times Y_t, \tag{12}$$

$$Lèvy(y) = 0.01 \times \frac{n_1 \times \sigma}{|n_2|^{\frac{1}{\beta}}}, \tag{13}$$

$$\sigma = \left(\frac{\Gamma(1+\beta) \times sin(\frac{\pi\beta}{2})}{\Gamma(\frac{1+\beta}{2}) \times \beta \times 2^{(\frac{\beta-1}{2})}} \right)^{1/\beta}, \tag{14}$$

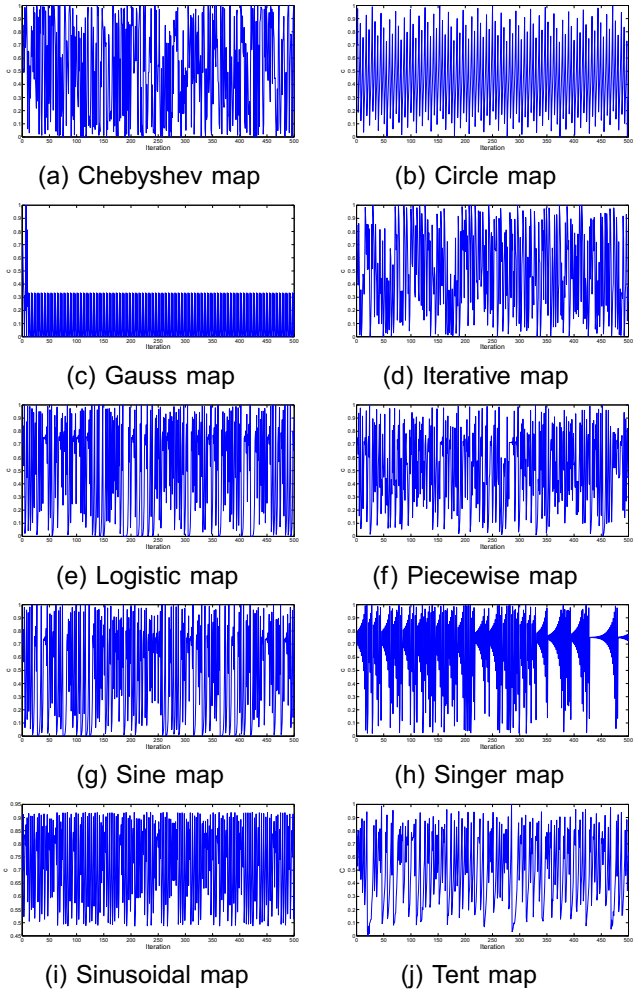

Fig. 2 Visualization of chaotic maps

where d represents the dimension of the position vector, n_1, n_2 are two random numbers in [0, 1], β is a constant, and $\Gamma(x) = (x-1)!$.

3.3 Chaotic map

Chaos can be defined as a phenomenon where any small change in its initial condition can lead to non-linear change for the future behavior. Moreover, it is mathematically defined as a semi-random behavior produced by nonlinear deterministic systems [33]. Chaos Optimization Algorithm (COA) is one of the recent search algorithms. The main idea of it is to transform parameters/variables from the chaos to the solution space. It depends on searching of global optimum on chaotic motion properties such as ergodicity, regularity, and stochastic properties. COA has attracted more attention in the last decade due to simplicity, convergence speed, and its capability for avoiding local minima. All of these reasons are used to improve the performance of evolutionary algorithms significantly [34].

In this research, ten distinguished non-invertible one-dimensional maps are used to obtain chaotic sets. These maps are widely used in much related works as in [3, 33–35]. As shown in Fig. 2, the maps have different behaviors and hence the behavior of DA can be tested using different maps. The used chaotic maps are defined in Table 3. In this table, q denotes the index of the chaotic sequence, and p_q is the $q\text{-}th$ number in the chaotic sequence. The initial point p_0 is set to 0.7 for all chaos maps. As the initial values of the chaotic map have a great influence on some of the chaotic maps' fluctuation pattern, we used the initial values as in

Table 3 The ten adapted chaotic maps

Map No.	Name	Definition	Range
M1	Chebyshev	$p_{q+1} = cos(q\cos^{-1}(p_q))$	(−1,1)
M2	Circle	$p_{q+1} = mod(p_q + r - (\frac{l}{2\pi})sin(2\pi p_q), 1)$, $l = 0.5$ and $r = 0.2$	(0,1)
M3	Gauss/mouse	$p_{q+1} = \begin{cases} 1, & p_q = 0 \\ \frac{1}{mod(p_q,1)}, & \text{otherwise} \end{cases}$	(0,1)
M4	Iterative	$p_{q+1} = \sin(\frac{l\pi}{p_q})$, $l = 0.7$	(-1,1)
M5	Logistic	$p_{q+1} = lp_q(1-p_q)$, l=4	(0,1)
M6	Piecewise	$p_{q+1} = \begin{cases} \frac{p_q}{l}, & 0 \le p_q < l \\ \frac{p_q - l}{0.5-l}, & l \le p_q < 0.5 \\ \frac{1-l-p_q}{0.5-l}, & 0.5 \le p_q < 1-l \\ \frac{1-p_q}{l}, & 1-l \le p_q < 1 \end{cases}$, $l = 0.4$	(0,1)
M7	Sine	$p_{q+1} = \frac{l}{4}\sin(\pi p_q)$, $l = 4$	(0,1)
M8	Singer	$p_{q+1} = \mu(7.86p_q - 23.31p_q^2 + 28.75p_q^3 - 13.302875p_q^4)$, $\mu = 1.07$	(0,1)
M9	Sinusoidal	$p_{q+1} = lp_q^2 sin(\pi p_q)$, $l = 2.3$	(0,1)
M10	Tent	$p_{q+1} = \begin{cases} \frac{p_q}{0.7}, & p_q < 0.7 \\ \frac{10}{3}(1-p_q), & p_q \ge 0.7 \end{cases}$	(0,1)

[35]. Figure 2 shows the visualization of these maps for c cohesion weight through 500 iterations.

4 Proposed model

This section describes the proposed model in detail. The model consists of three main phases: data pre-processing, features selection, and the classification phase. Figure 3 shows the overall architecture of the proposed model. The details of each phase are explained below.

4.1 Data pre-processing phase

In this phase, three different sampling methods were used: (1) RUS method; (2) ROS method; and (3) SMOTE method, the details of the three methods are presented in Section 3.1. The aim of these methods is to obtain more balanced samples in each class.

4.2 Feature selection phase using chaotic dragonfly algorithm (CDA)

In the features selection phase, the Chaotic Dragonfly Algorithm (CDA) was used for feature selection. As mention before, there are five main parameters that affect the updating position of the individuals, namely, separation, alignment, cohesion, attraction towards a food source, and distraction outwards an enemy. However, the corresponding weights of these five parameters are initialized randomly. This randomness can affect badly the algorithm stability and performance as it will be discussed later in the next section. In order to solve these problems, chaotic maps are used where the random parameters are replaced by chaotic values. Such a combination between the dragonfly algorithm and chaotic maps can be defined as Chaotic Dragonfly Algorithm (CDA). In this paper, ten different chaotic maps were used. The mathematical formulas of these maps are defined in Section 3.3. The pseudo code of CDA is defined at Algorithm 1.

In CDA, the updating position, which is defined in (10), is reformulated as follows:

$$\Delta Y_{t+1} = (B(i)A_i + B(i)S_i + B(i)C_i + B(i)E_i + B(i)F_i) + B(i)\Delta Y_t, \quad (15)$$

where $B(i)$ is the obtained value of a chaotic map of the $i\text{-}th$ iteration.

It is worth mentioning that the complexity of DA as many optimization algorithms is $O(dM + M\mathcal{C})$, where d is the dimension of the problem, M is the number of solutions/particles/agents, and $\mathcal{C}$ represents the complexity of the objective function. The complexity of CDA is approximately identical to the DA algorithm. This is because the complexity of the additional step that generates chaotic numbers in the CDA algorithm is $O(Max_{iter})$, where Max_{iter} is the maximum number of iterations. The CDA algorithm uses this step only at the beginning for generating all chaotic numbers which are required during the iterations. However, the complexity of the step that generates random numbers in the DA algorithm is also $O(Max_{iter})$ which is identical to the CDA algorithm. As a consequence, both DA and CDA algorithms approximately have the same time complexity.

In this paper, CDA is implemented as a feature selection algorithm based on the wrapper method where the chaotic search is embedded in the searching iterations of CDA in order to find the optimal feature subset describing the dataset. The aim of applying feature selection is to improve the classification performance, reduce the number of selected features, and reduce the classification computational cost. The detailed descriptions of CDA are defined as follows:-

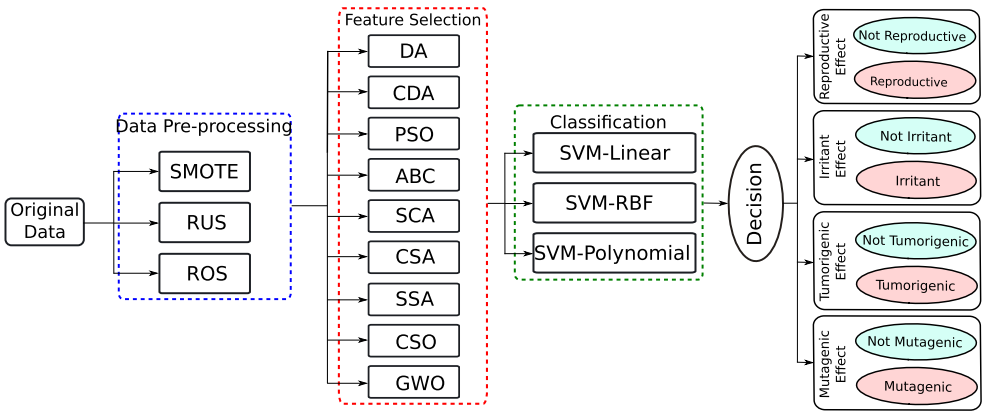

Fig. 3 The overall architecture of the proposed model

4.2.1 Parameters initialization

In the beginning, CDA starts with setting CDA parameters and randomly initializes dragonfly positions (solutions) within the search space. Each position represents a feature subset with different numbers of features and different lengths. The initial parameters settings are presented in Table 4.

4.2.2 Fitness function

At each iteration, each dragonfly position is evaluated using a predefined fitness function F (see (16)). In this paper, two objective criteria are used for evaluating each position, i.e. a subset of features, namely, classification accuracy and the number of selected features where the objective function is to maximize the classification accuracy and minimize the number of selected features. The adopted fitness function equation combines the two criteria into one by setting a weight factor as in (16), where Acc represents the classification accuracy, w_f is the weight factor which has value in $[0, 1]$, L_f is the length of selected feature subset, and L_t is the total number of features. The factor w_f is used to control the importance of classification accuracy and number of selected features. As for any classifier, improving the accuracy is the first objective, the weight factor is usually set to value near one [36]. Thus, we set w_f for all the following experiments to 0.9.

$$F = \max\left(Acc + w_f\left(1 - \frac{L_f}{L_t}\right)\right) \tag{16}$$

In this study, the data is divided randomly into two different parts: training and testing set using 10-fold cross validation method to ensure the stability of the obtained results. The training set is used to train a classifier through optimization and the testing set is used to evaluate the selected features.

Moreover, k-Nearest Neighbor (k-NN) was the used classifier to measure the goodness of the selected features, where k represents the number of nearest neighbors. We used the k-NN classifier due to two main reasons. First, k-NN is simple and easy to implement classifier. Second, in comparison with many learning algorithms, the computational time of k-NN is small because there is no a training phase in the k-NN classifier [37]. In k-NN, an unknown pattern is distinguished or classified based on the similarity to the known samples, i.e. training samples, by computing the distances from the unknown sample to all labeled samples and select the k-nearest samples as the basis for classification [38]. The unknown sample is assigned to the class which has the most samples among the k-nearest samples [37]. The best solution is the one, which maximizes the classification accuracy and minimizes the number of selected features.

Table 4 Parameters settings for CDA

Parameter	Value
β	1.5
d	31
M	50
Lower Bound	1
Upper Bound	31
Maximum Iteration	50

4.2.3 Termination criteria

The positions of CDA are updated according to (11 and 15). The optimization process terminates when it reaches the maximum number of iterations or when the best solution is found.

Algorithm 1 Chaotic dragonfly algorithm (CDA)

1: Set the initial values of the dragonflies (population) size (M), dimensions (d), lower and upper boundaries and the maximum number of iterations (Max_{iter}).
2: Initialize the dragonfly positions ($Y_k(k = 1, 2, \ldots, Max_{iter})$) randomly.
3: Initialize the step vector $\Delta Y_k(k = 1, 2, \ldots, Max_{iter})$
4: Calculate the fitness value of each position $F(Y_k)$.
5: Set $t := 1$. {Counter initialization}.
6: **repeat**
7: **for** $(j = 1 : j \leq M)$
8: Calculate the value of chaotic map $B(t)$.
9: Update the values of w, s, a, f and e.
10: Calculate A, S, C, F, and E using (5, 6, 7, 8 and 9).
11: **if** Number of neighboring dragonfly≤ 1
12: Update the velocity vector using (15).
13: Update the position vector using (11).
14: **else**
15: Update the position vector using (12).
16: **end if**
17: Check if the new positions go out of the search space boundaries and bring it back.
18: **end for**
19: Set $t = t + 1$. {Iteration counter increasing}.
20: **until**$(t < Max_{iter})$. {Termination criteria satisfied}.
21: Produce the best solution N.

4.3 Classification phase

In this phase, the SVM classifier with different kernel functions was used. SVM is one of the most widely-used

learning algorithms in many research fields such as chemoinformatics [26], Power systems [39], and bioinformatics [40]. This is because SVM obtains high classification results with both linear and nonlinear data. With nonlinear data, kernel functions are used for transforming the data into a higher dimensional space where the data can be linearly separable [33]. The selected features obtained from the previous phase were used as the input to the SVM classifier in order to test the robustness of these selected features.

The goal of SVM is to orientate a hyperplane, $w^T x + b = 0$, to separate different classes, where w is a weight vector, x is the input vector or the training pattern, and b is the bias or threshold. The SVM classifier achieved high classification accuracy when the data was linearly separable. However, the SVM classifier cannot classify a non-linearly separable data; hence, kernel functions are used to map the data onto a new higher dimensional space, where the data can be separated linearly [41]. There are different kernel functions such as Linear Kernel ($\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{x}_i, \mathbf{x}_j \rangle$), Radial Basis Function (RBF) ($\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2 / 2\sigma^2)$), and polynomial kernel of degree d ($\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = (\langle \mathbf{x}_i, \mathbf{x}_j \rangle + c)^d$) are used, where x_i and x_j represent two samples or patterns [42]. In this research, Radial Basis Function (RBF), polynomial, and linear kernel functions were used [33].

5 Results and analysis

In this section, three main experiments were conducted to evaluate the proposed model. The first experiment in Section 5.2 aims to evaluate the performance of DA and CDA algorithms using different chaotic maps. The second experiment in Section 5.3 was conducted to compare the CDA algorithm with seven different meta-heuristic optimization algorithms: PSO [43], Artificial Bee Colony (ABC) [10], Grey Wolf Optimizer (GWO) [44], Chicken Swarm Optimization (CSO) [45], Salp Swarm Algorithm (SSA) [15], Crow Search Algorithm [46] and Sine Cosine Algorithm (SCA) [47]. The third experiment in Section 5.4 aims to compare the selected features

Table 5 PC specifications

Name	Detailed settings
Hardware	
CPU	Core (TM) i3
Frequency	2.13 GHZ
RAM	2 GB
Software	
Operating System	Windows 7
Language	MATLAB R2012R

Table 6 A comparison between DA and CDA using different chaotic maps in terms of worst fitness value, the best fitness value, the mean fitness value, standard deviation (SD), average features length and p-value

Algorithm	Min	Max	Mean	SD	Average feature length (%)	p-value
Irritant effect						
DA	1.28	1.40	1.33	0.06	50.29	
CDA-1	1.26	1.54	1.42	0.08	40.68	0.0043
CDA-2	1.29	1.46	1.39	0.06	45.23	0.0266
CDA-3	1.29	**1.57**	**1.47**	0.08	**32.19**	6.79E-06
CDA-4	1.29	1.46	1.39	0.06	40.39	0.0016
CDA-5	1.26	1.48	1.40	0.06	41.16	1.24E-04
CDA-6	1.28	1.48	1.39	0.06	41.87	0.002
CDA-7	**1.34**	1.55	1.43	0.07	37.23	1.09E-05
CDA-8	1.31	1.47	1.38	**0.05**	42.97	2.45E-04
CDA-9	1.32	1.49	1.40	0.07	40.19	0.0058
CDA-10	1.28	1.45	1.40	**0.05**	32.84	2.92E-04
Mutagenic effect						
DA	1.26	1.40	1.34	0.05	47.23	
CDA-1	1.31	1.42	1.37	**0.04**	42.65	0.1723
CDA-2	1.30	1.48	1.43	0.06	38.58	3.52E-04
CDA-3	1.27	**1.62**	**1.50**	0.09	31.29	1.46E-06
CDA-4	1.28	1.52	1.43	0.08	39.81	0.00138
CDA-5	1.28	1.44	1.38	**0.04**	42.32	0.0532
CDA-6	1.32	1.49	1.43	0.05	39.03	1.61E-03
CDA-7	1.29	1.54	1.47	0.05	**30.39**	2.64E-05
CDA-8	1.25	1.51	1.42	0.07	39.55	4.67E-05
CDA-9	1.30	1.43	1.36	0.05	46.26	0.5636
CDA-10	**1.34**	1.50	1.43	0.07	38.58	1.07E-05
Reproductive effect						
DA	1.15	1.38	1.28	0.11	40.94	
CDA-1	1.14	1.35	1.29	0.08	35.74	0.1115
CDA-2	1.11	1.41	1.29	0.10	34.13	0.0421
CDA-3	1.11	1.47	1.35	0.10	31.42	7.90E-04
CDA-4	1.11	1.49	1.35	0.11	30.19	4.50E-04
CDA-5	1.16	1.38	1.31	0.06	36.90	0.0724
CDA-6	1.12	1.47	1.39	0.08	26.77	9.94E-05
CDA-7	1.16	**1.54**	**1.48**	0.07	13.87	1.18E-13
CDA-8	1.13	1.44	1.33	0.10	32.65	1.21E-03
CDA-9	**1.22**	1.33	1.27	**0.04**	37.03	0.2224
CDA-10	1.18	1.50	1.38	0.10	**12.78**	2.70E-03
Tumorigenic effect						
DA	1.26	1.38	1.33	0.04	46.39	
CDA-1	1.28	1.50	1.43	0.06	35.74	3.71E-05
CDA-2	1.26	1.46	1.39	0.06	38.32	2.94E-05
CDA-3	1.24	1.56	**1.48**	0.08	30.90	5.42E-07
CDA-4	1.30	1.45	1.37	0.05	38.90	6.60E-04
CDA-5	1.28	1.53	1.41	0.09	38.45	1.30E-04
CDA-6	1.21	1.40	1.33	0.06	45.03	0.8492
CDA-7	**1.34**	1.45	1.43	**0.02**	39.10	2.71E-11

Table 6 (continued)

Algorithm	Min	Max	Mean	SD	Average feature length (%)	*p*-value
CDA-8	1.24	1.50	1.40	0.08	35.68	1.20E-04
CDA-9	1.23	1.45	1.35	0.09	45.54	0.014
CDA-10	1.22	**1.57**	1.45	0.10	**29.81**	3.40E-04

from CDA with the features that were selected from the other optimization algorithms in terms of classification performance. In this experiment, the SVM classifier was used to classify unknown samples into toxic or non-toxic. This experiment divided into three sub-experiments. The first sub-experiment aims to determine the best kernel function of SVM. In the second sub-experiment, the proposed model was evaluated before/after data pre-processing phase. In this sub-experiment, three well-known data sampling methods were used, namely, SMOTE, RUS, and ROS. The third sub-experiment aims to evaluate the performance of the features that were selected from DA, CDA, PSO, ABC, CSO, and GWO algorithms in terms of classification performance, i.e. accuracy, precision, recall, and F-Score, and classification computational cost. All

Table 7 Selected features of CDA using Gauss chaotic map

Toxic effect	Features indices
Irritant	5, 6, 8, 9, **10**, **11**, **12**, 13, **19**, 20, **22**, 24
Mutagenic	2, 5, 7, **10**, **11**, **12**, 14, 16, **19**, **22**, 23
Reproductive	2, 6, 9, **10**, **11**, **12**, 13, 14, 17, **19**, 20, 21, **22**
Tumorigenic	4, 5, **10**, **11**, **12**, 13, 16, 18, **19**, 20, **22**, 24

these experiments were performed on the same PC with the same specification settings as presented in Table 5.

5.1 Performance metrics

In this paper, two types of evaluation criteria were used. The first evaluation was used for evaluating the performance of the proposed features selection algorithms. Six different statistical measurements were adopted for this evaluation. These measurements were the worst fitness value, the best fitness value, the mean fitness value, standard deviation (SD), average features length and *p*-value from Wilcoxon's rank sum test (nonparametric statistical test) with 5% significance level [48]. The statistical test is necessary in order to prove that the proposed algorithm provides a significant improvement compared with the other

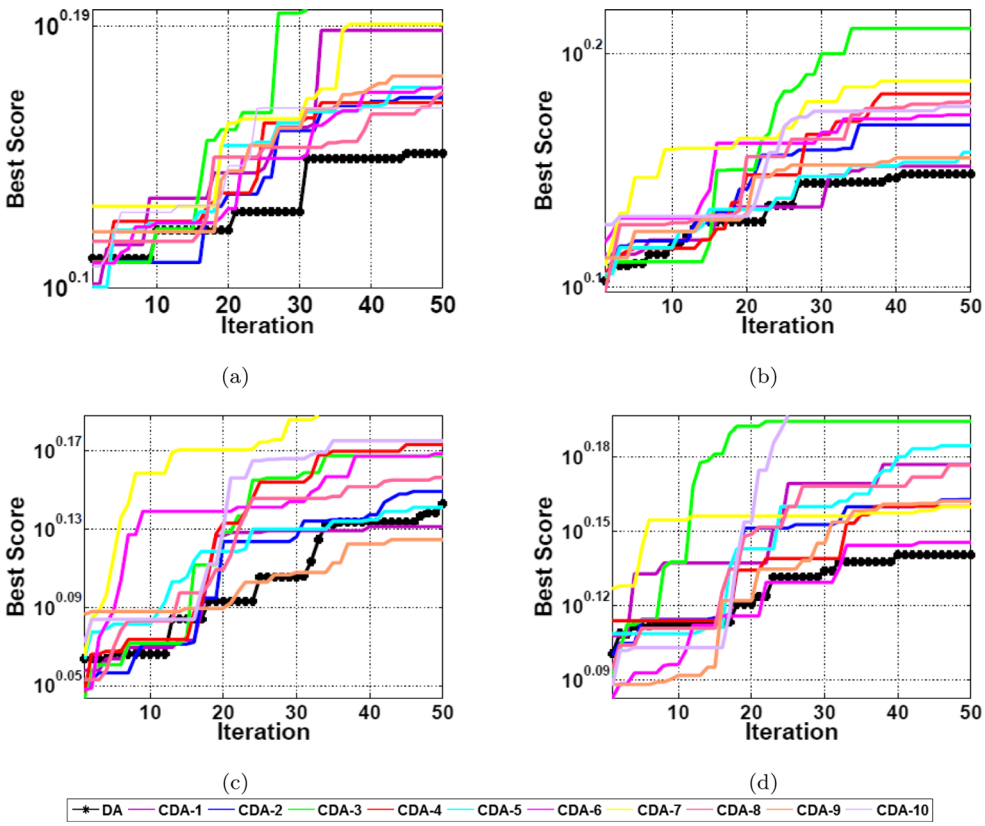

Fig. 4 Convergence curves of the DA and CDA algorithms using different chaotic maps; **a** Irritant Effect, **b** Mutagenic Effect, **c** Reproductive Effect and **d** Tumorigenic Effect

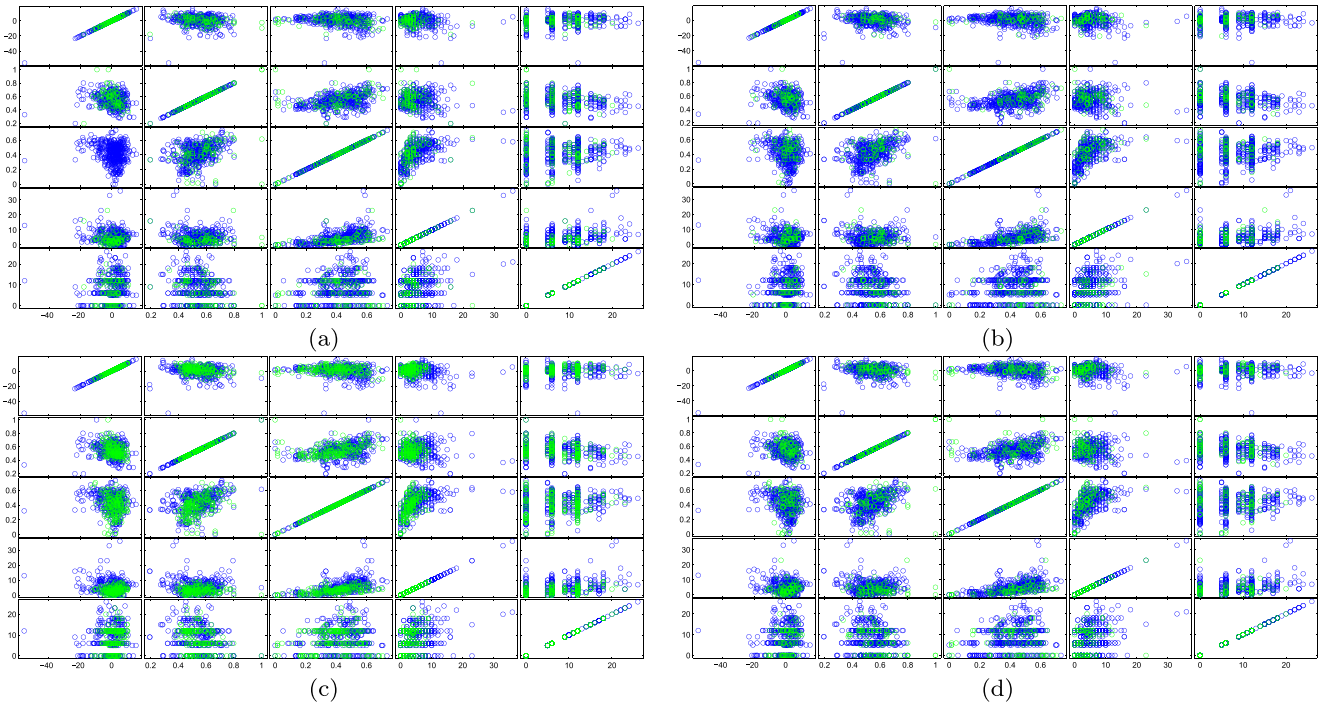

Fig. 5 Visualization of the plot matrix for all datasets (**a**) Tumorigenic, (**b**) Irritant, (**c**) Reproductive, (**d**) Mutagenic, using the most five important selected features which are highlighted in Table 7

algorithms [49]. Wilcoxon's rank sum test is more sensitive than the t-test as it assumes proportionality of differences between two paired samples. Moreover, it is safer than the t-test as it does not assume the normal distributions. Additionally, the outliers affecting less on the Wilcoxon's test than the t-test [49]. The best values of p when p-value < 0.05 which can be considered sufficient evidence against the null hypothesis. In this paper, this test was used to evaluate different versions of the CDA algorithm with different chaotic maps and determine the best one. The worst fitness value, the best fitness value, the mean fitness value, standard deviation, and the average features length are mathematically defined as follows:

$$\text{Standard Deviation}(SD) = \sqrt{\frac{\sum_{i=1}^{M}(BS_i - \mu)^2}{Max_{iter}}}, \tag{17}$$

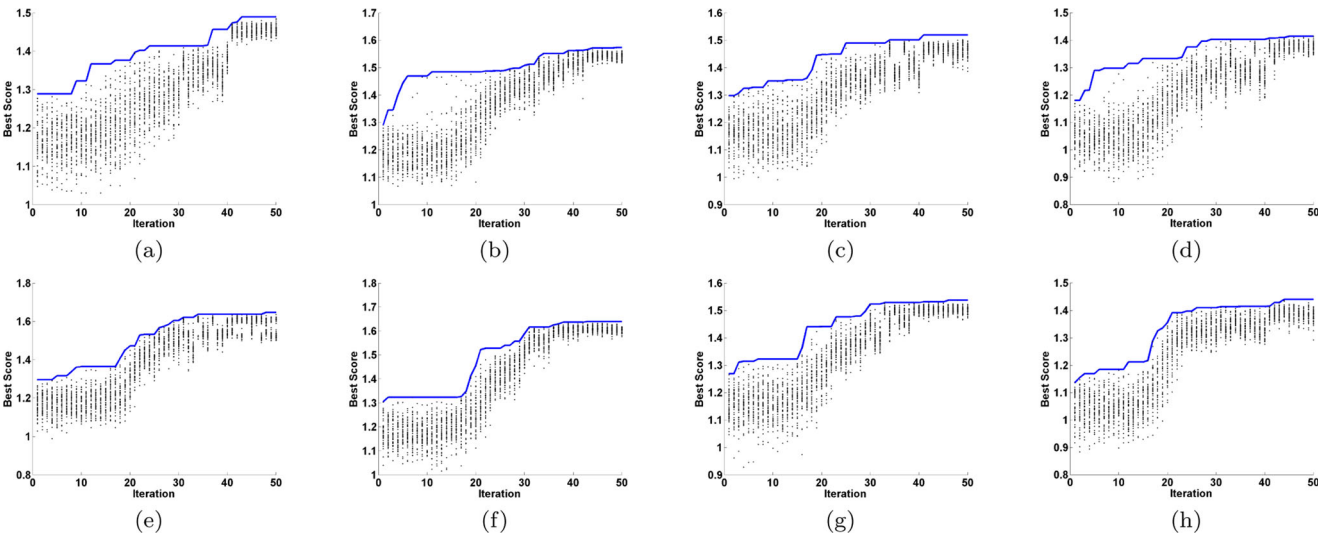

Fig. 6 Visualization of the search history of the fitness values for the whole population with respect to the best score obtained so far for both DA and CDA algorithms. (**a** and **e**) Irritant effect, (**b** and **f**) Mutagenic Effect, (**c** and **g**) Reproductive Effect and (**d** and **h**) Tumorigenic Effect. Top figures (**a**, **b**, **c**, and **d**) for the DA algorithm, and bottom figures (**e**, **f**, **g**, and **h**) for the CDA algorithm

$$\text{Best Fitness} = max_{i=1}^{Max_{iter}} BS_i, \tag{18}$$

$$\text{Worst Fitness} = min_{i=1}^{Max_{iter}} BS_i, \tag{19}$$

$$\text{Mean Fitness} = \frac{1}{Max_{iter}} \sum_{i=1}^{Max_{iter}} BS_i, \tag{20}$$

$$\text{Average Features Length} = \frac{\sum_{i=1}^{Max_{iter}} \frac{length(BS_i)}{L}}{Max_{iter}}, \tag{21}$$

where BS is the best score obtained so far for each iteration and L is the number of features in the original dataset.

The second evaluation was used to evaluate the selected features using classification measures. These measures are accuracy, precision, recall, and F-Score. These measures are defined in Section 3.1.2.

5.2 The performance of CDA with different chaotic maps

The main objective of this experiment is to evaluate the performance of DA and CDA algorithms using different chaotic maps and determining the optimal chaotic map for each dataset. This experiment aims to evaluate the performance of DA with different versions of CDA in terms of statistical measurements. Table 6 shows the obtained results of this experiment. The rank sum test for this experiment is applied on DA and all versions of CDA. Moreover, Fig. 4 shows the convergence curves of DA and CDA using different chaotic maps.

From Table 6 many notices can be seen. As shown, the best results of statistical measurements are highlighted in bold text, while the worst results of p-Value where $p < 0.05$ are underlined. It is worth mentioning that the marks CDA-1, CDA-2, ..., CDA-10 in Table 6 are short for the DA algorithm using the ten chaotic maps that are listed in Table 3. As observed in Table 6, different versions of CDA outperformed the standard version of DA. Moreover, it can be observed that in most cases, CDA-3 obtained the highest statistical results compared with the other algorithms. Moreover the obtained $p\text{-}value$ also indicates

Table 8 Parameter settings for PSO [50], ABC [51], CSO [56], GWO [52], CSA [53] and SCA [54] optimization algorithms

Algorithm	Parameter	Value
PSO	An inertial weight	1
	An inertia weight damping ratio	0.9
	Personal learning coefficient	1.5
	Global learning coefficient	2.0
ABC	A number of colony size	10
	A number of food source	5
	A number of limit trials	5
CSO	A number of chicken updated	10
	The percent of roosters population size	0.15
	The percent of hens population size	0.7
	The percent of mother hens population size	0.05
GWO	a	2
SCA	b	2
CSA	Awareness probability	0.1
	Flight length	0.01

Table 9 A comparison between CDA, PSO, ABC, GWO, CSO, CSA, SCA and SSA algorithms in terms of statistical measurements

Algorithm	Min	Max	Mean	SD	Average Feature Length (%)	$p\text{-}value$
Irritant effect						
CDA	**1.29**	**1.57**	**1.51**	0.08	32.19	
PSO	1.25	1.5	1.43	0.06	**24.58**	3.20E-04
ABC	1.27	1.53	1.46	0.09	35.48	0.046
CSO	1.25	1.47	1.43	**0.03**	35.1	4.60E-04
GWO	1.25	1.39	1.37	0.05	46.65	8.70E-03
CSA	**1.29**	1.47	1.40	0.06	53.23	8.04E-06
SCA	1.28	1.50	1.48	0.04	45.74	1.36E-04
SSA	1.27	1.54	**1.51**	0.04	26.8	9.30E-03
Mutagenic Effect						
CDA	1.30	**1.48**	**1.43**	0.06	38.58	
PSO	1.3	1.4	1.37	**0.04**	43.94	2.40E-03
ABC	1.24	1.47	1.42	0.05	45.39	1.03E-06
CSO	**1.32**	1.5	**1.43**	0.05	**34.65**	1.31E-06
GWO	1.27	1.45	1.4	0.06	43.23	3.70E-04
CSA	1.24	1.40	1.37	**0.04**	55.74	1.03E-05
SCA	1.25	1.47	1.40	0.05	43.10	8.17E-06
SSA	1.28	**1.48**	**1.43**	0.05	41.80	0.952
Reproductive Effect						
CDA	1.16	**1.54**	**1.48**	0.07	**13.87**	
PSO	**1.18**	1.21	1.2	0.04	45.16	1.82E-16
ABC	1.12	1.32	1.28	**0.03**	41.61	1.76E-13
CSO	1.17	1.37	1.34	0.06	30.39	1.79E-13
GWO	1.11	1.28	1.25	0.04	45.48	2.62E-15
CSA	1.12	1.29	1.25	**0.02**	44.68	1.25E-05
SCA	1.15	1.38	1.26	0.03	40.26	2.62E-15
SSA	1.14	1.51	1.45	0.06	18.80	0.512
Tumorigenic Effect						
CDA	1.24	**1.56**	**1.48**	0.08	**30.90**	
PSO	1.28	1.41	1.38	0.05	37.23	6.70E-03
ABC	1.16	1.44	1.41	**0.03**	39.29	4.20E-04
CSO	**1.31**	1.48	1.46	0.04	**30.90**	0.04248
GWO	1.3	1.44	1.38	0.05	45.11	1.40E-03
CSA	1.29	1.54	1.45	0.05	45.48	9.50E-03
SCA	1.27	1.49	1.38	**0.03**	40.05	7.80E-03
SSA	1.23	1.53	1.47	0.04	33.25	0.076

that CDA-3 outperformed the other algorithms, which means that the Gauss chaotic map can significantly improve the performance of DA. Moreover, CDA-7 achieved the second-best results. On the other hand, CDA-9, in most cases, could not enhance the performance of DA, and the p-value proved that there is no big difference between DA and CDA-9, as the p-value exceeds 0.05.

In terms of convergence speed and best score obtained, Fig. 4 shows the graphical representation of the convergence curves of DA with different versions of CDA for 50 iterations. As shown, after embedding chaotic maps, many modifications are noticed. It can be observed from this figure that CDA with different chaotic maps obtained better results as their curves were higher than the standard DA 'colored with black'. Moreover, it can be noticed that the convergence speed of these CDAs' versions against the standard DA. Additionally, the CDA-3 curve obtained the highest results while CDA-9 obtained the lowest results. These results were consistent with the results obtained from Table 6.

The best feature subset for each data is shown in Table 7. From the table, it can be remarked that some features are highlighted in bold text. This is because these features were selected in all toxic effects, i.e. mutagenic, tumorigenic, reproductive and irritant effects, which reflects the importance of these features. Figure 5 shows the plot matrix for all datasets (tumorigenic, irritant, reproductive, and mutagenic) with only the five selected features which are highlighted in Table 7. As shown, the features in all datasets are not well discriminated and hence linear classifiers will not be suitable for the classification.

Figure 6 shows a comparison between the DA and CDA algorithms. The figure shows the search history of the fitness values for the whole population with respect to the best score obtained so far. Therefore, it is easy to observe the movement of the candidate solutions through the iterations. As it can be seen, the fluctuation of fitness values of the dragonflies for the original DA algorithm is higher than in CDA. Moreover, it can be observed that there are abrupt changes in the fitness values of dragonflies in case of the exploration phase and gradually changes in the exploitation phase. In addition, it can be observed that the proposed CDA algorithm obtained results better than DA. Furthermore, CDA converges faster than DA. Hence, in Fig. 6, it can be concluded the good balancing between exploration and exploitation of CDA.

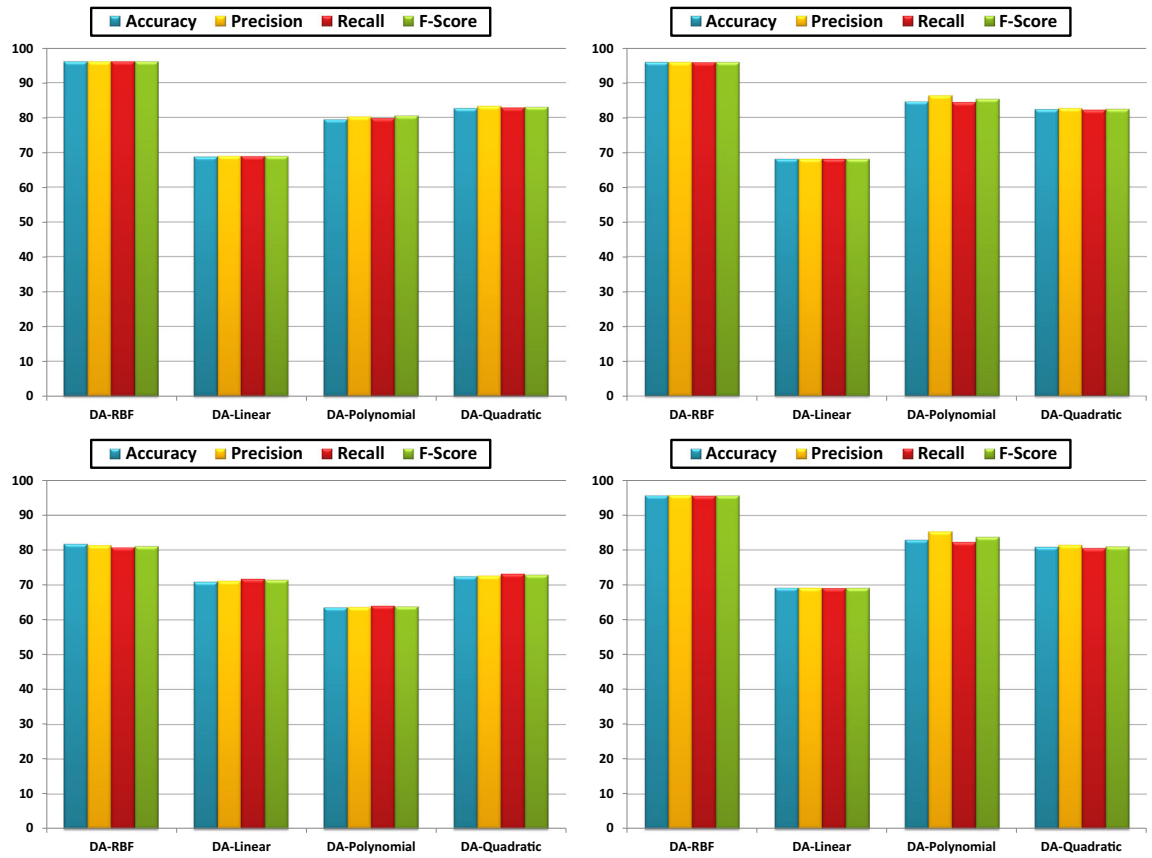

Fig. 7 Comparison between the results of the proposed model using SVM with different kernel functions in terms of accuracy, precision, recall, and F-Score; **a** Irritant Effect, **b** Mutagenic Effect, **c** Reproductive Effect and **d** Tumorigenic Effect

From these findings, it can be concluded that embedding the value of the chaotic sequence on the main parameters of movement equations enhanced the DA algorithm by avoiding the local optima and reaching the global optima much faster. Based on these results, the Gauss chaotic map is selected as the appropriate map. This map will be evaluated in more details in the following section.

5.3 CDA vs. Other meta-heuristic optimization algorithms

The objective of this experiment is to compare the performance of CDA with Gauss chaotic map with seven other optimization algorithms, namely, PSO [50], ABC [51], GWO [52], CSA [53], SCA [54], SSA [55], and CSO [56]. The parameters settings for all meta-heuristic optimization algorithms are shown in Table 8. As it can be observed from this table, SSA has no parameters to set. This is because SSA has only three parameters ($c1$, $c2$, and $c3$), $c1$ is assigned according to the iteration number, $c2$ and $c3$ are assigned with random values at each iteration. Moreover, the default parameters for all algorithms were the maximum generation = 50, the number of search agents = 50, lower bound = 1, and upper bound = 31. The results of this experiment are summarized in Table 9.

Table 9 shows the statistical measurements of the tumorigenic effect, mutagenic effect, irritant effect, and reproductive effect datasets. The highest obtained results are highlighted in bold text. From the table, it can be seen that in most cases, the CDA algorithm obtained the highest best and mean score. However, in most cases, CDA obtained the lowest stability. This is because there are many parameters that were used in the updating position of dragonflies including A, S, C, E, and F and their weight factors a, s, c, e, f, w and step vector. Moreover, the p-value shows the superiority of CDA compared with the other algorithms as in most cases the p-value was less than 0.05. Also, it can be observed that SSA is a very competitive algorithm with CDA. SSA was the second place after CDA. Additionally, it can be seen that SSA, CSA, and SCA were obtained results nearly similar to CDA. This is because these algorithms have few parameters to be tuned.

From these findings, it can be concluded that the CDA is better than PSO, GWO, ABC, CSO, SSA, CSA, and

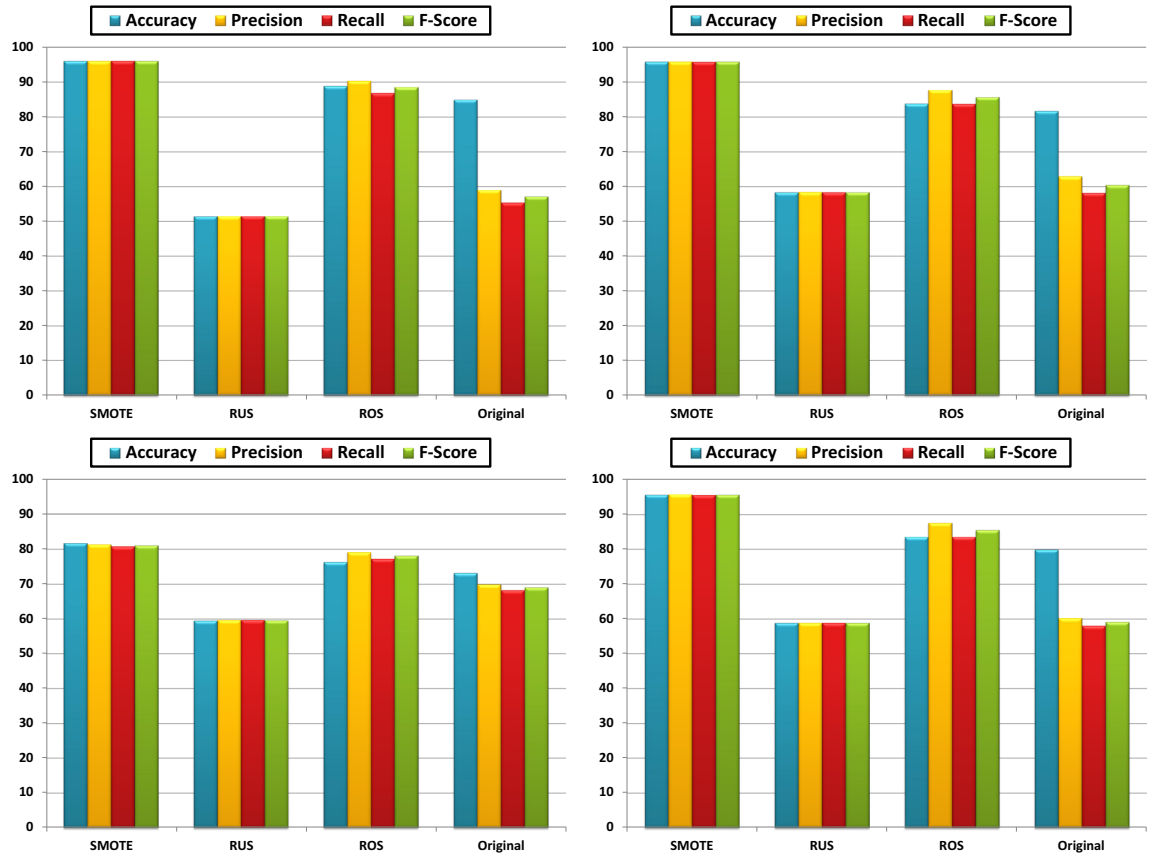

Fig. 8 Comparison between original, SMOTE, RUS and ROS in terms of accuracy, precision, recall, and F-Score; **a** Irritant Effect, **b** Mutagenic Effect, **c** Reproductive Effect and **d** Tumorigenic Effect

SCA algorithms, in selecting optimal feature subset which maximizes the classification performance and minimizes the number of selected features.

5.4 Toxicity effects classification experiment

This experiment was conducted to evaluate the selected features that were selected from the proposed feature selection algorithm, i.e. CDA. In this experiment, the SVM classifier with different kernel functions was used to evaluate the selected features. This experiment is divided into three sub-experiments. The aim of the first sub-experiment is to compare different kernel functions of SVM and determine the optimal kernel function, which will be used in the next sub-experiments. Figure 7 shows the obtained results in terms of recall, accuracy, precision, and F-Score from CDA with Gauss map using different kernel functions, namely, linear, RBF, and polynomial. The order of the polynomial kernel was two, i.e. quadratic kernel, and three. Moreover, the value of σ parameter for the RBF kernel was set to 1 and the value of C was 20. From this figure, it can be observed that the RBF kernel function achieved the best results when the irritant, mutagenic, reproductive, and tumorigenic effect datasets were used. This is because as shown in Fig. 5, the features are not well-discriminated and hence the linear kernel function obtained the worst results. On the other hand, transforming the data into a higher dimensional space such as in RBF or polynomial kernels improves the classification performance. Due to the good results of the RBF kernel function, it will be used for the next sub-experiments.

In the second sub-experiment, RUS, ROS and SMOTE algorithms were used to obtain a balanced distribution of classes, and the proposed model was evaluated before/after data pre-processing phase. In this experiment, the features that were selected using CDA and SVM with RBF kernel were used. Figure 8 shows the results of this sub-experiment. From this figure, many notices can be seen. The RUS method did not improve the performance of the proposed model, as it obtained low results compared with the original data. On the contrary, the ROS algorithm achieved results better than the RUS algorithm and the original dataset. Moreover, as it can be seen from the figure,

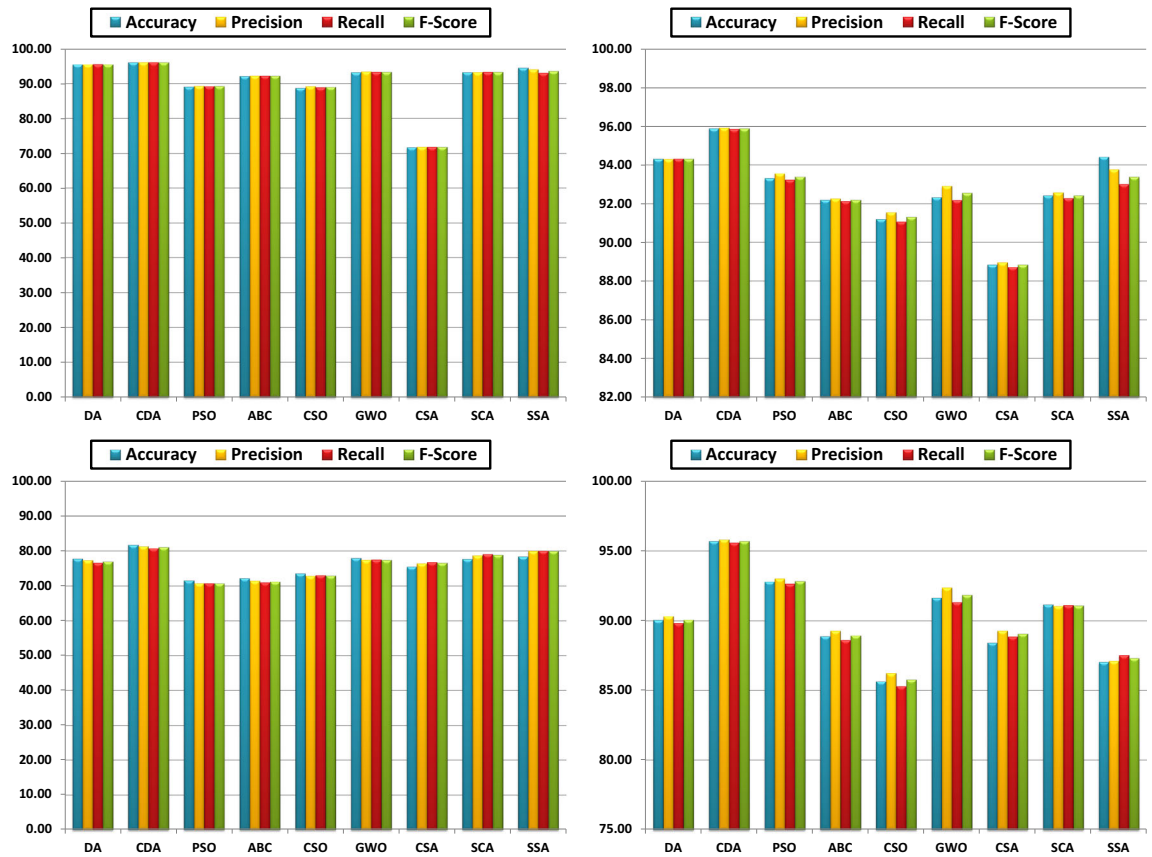

Fig. 9 Comparison between DA, CDA, PSO, ABC, CSO, GWO, CSA, SCA and SSA in terms of accuracy, precision, recall, and F-Score; **a** Irritant Effect, **b** Mutagenic Effect, **c** Reproductive Effect and **d** Tumorigenic Effect

SMOTE outperformed the other two sampling methods, i.e. RUS and ROS, and it obtained higher accuracy, recall, precision, and F-Score. This is because, (1) in the RUS algorithm, samples are randomly removed from the majority class; (2) in the ROS algorithm, the replication of minority class samples do not cause its decision boundary to spread or extend into the majority class region. On the other hand, the SMOTE algorithm preserves all samples and generates new minority class samples to obtain more balanced dataset. As shown in Fig. 8, the precision of the proposed model when the SMOTE algorithm was used ranged from 82% to 96%, while in RUS, ROS, and original dataset the results were (51% to 59%), (83% to 87%), and (56% to 68%), respectively. Moreover, the SMOTE algorithm achieved results in terms of recall and F-score better than RUS, ROS, and the original dataset.

The third sub-experiment was conducted to compare the features that were selected by CDA with the standard DA, PSO, ABC, GWO, and CSO in terms of classification measurements. In this sub-experiment, the features that were selected are evaluated using the SVM classifier with RBF kernel function. Figure 9 shows the results of this sub-experiment. From the figure, it can be seen that the selected features using CDA outperformed the other algorithms. Additionally, the performance of SSA, SCA, and DA are comparable, as the SSA algorithm outperformed DA for the reproductive and mutagenic toxic datasets while DA outperformed SSA for the tumorigenic and irritant toxic datasets. However, SCA outperformed DA for tumorigenic. Moreover, it can be observed that the obtained results of the irritant toxic effect are higher than other toxic effects where the best results obtained were accuracy 96.08%, precision 96.08 %, recall 96.11%, and F-Score 96.1%, while the worst

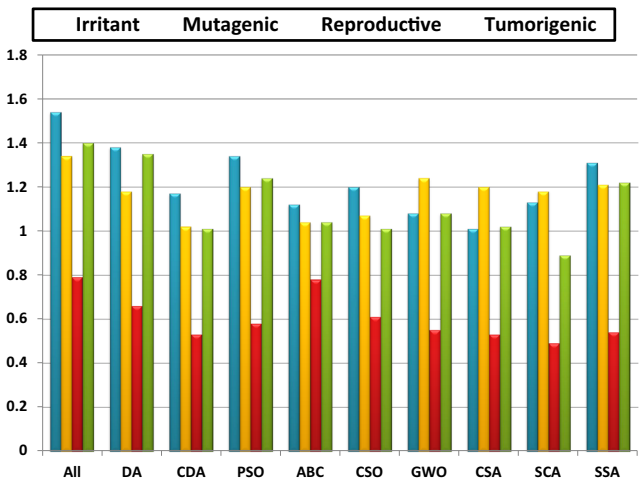

Fig. 10 Classification computational time of the original features and the selected features using DA, CDA, PSO, ABC, CSO, GWO, SCA, CSA, and SSA

results obtained were accuracy 81.82%, precision 81.45 %, recall 80.84%, F-Score 81.14% for the reproductive toxic effect. Moreover, the mutagenic toxic effect was in the second place after the irritant effect, and the tumorigenic toxic effect is in the third place.

In terms of classification computational time, Fig. 10 shows the results using the original features and selected features. As shown, the computational time reduced after applying feature selection algorithms. Moreover, it can be observed that the CDA algorithm reduced the time significantly than DA.

From these findings, we can conclude that the SVM classifier with RBF kernel function improved the classification performance than the other kernel functions. Moreover, the selected features using the CDA algorithm achieved results better than the other optimization algorithms and the selected features reduced the computational time required for the classification.

6 Conclusions

In this paper, a novel hybridization of chaos with dragonfly algorithm, namely, Chaotic Dragonfly Algorithm (CDA) was proposed. Ten chaotic maps were used in this study to enhance the performance of DA. CDA is applied to one of the challenging problems, namely, feature selection. The proposed CDA feature selection algorithm was used to select the most discriminative features (chemical compounds) for four toxic effects of liver drugs, namely, irritant, mutagenic, reproductive, and tumorigenic effects. The performance of CDA was compared with four other meta-heuristic optimization algorithms: PSO, CSO, GWO, and ABC. The experimental results showed that the CDA algorithm outperformed the other optimization algorithms. Moreover, the experimental results proved that the adjusted variable using the Gauss map significantly enhanced the DA algorithm in terms of classification performance, stability quality, number of selected features, and convergence speed. Moreover, it showed that the proposed model was fast, robust, efficient, and coherent. Moreover, it could be used for further drug toxicity prediction in the early drug development stage.

Acknowledgment We would like to thank Dr. Yasmine S. Momen of the clinical pathology department; national liver institute for providing the database that has been used in this work and for her great effort for getting understanding the used dataset.

Compliance with Ethical Standards

Conflict of interests The authors declare that there is no conflict of interests regarding the publication of this paper.

References

1. Liu H, Motoda H (2012) Feature selection for knowledge discovery and data mining, vol 454. Springer Science & Business Media, Berlin
2. Yu L, Liu H (2003) Feature selection for high-dimensional data: A fast correlation-based filter solution. In: Proceedings of the 20th international conference on machine learning (ICML-03), pp 856–863
3. Faris H, Mafarja MM, Heidari AA, Aljarah I, Ala'm AZ, Mirjalili S, Fujita H (2018) An efficient binary salp swarm algorithm with crossover scheme for feature selection problems. Knowledge-Based Systems
4. Tharwat A, Gaber T, Ibrahim A, Hassanien AE (2017) Linear discriminant analysis: a detailed tutorial. AI Commun 30(2):169–190
5. Tharwat A (2016) Principal component analysis-a tutorial. Int J Appl Pattern Recogn 3(3):197–240
6. Xue B, Zhang M, Browne WN (2013) Particle swarm optimization for feature selection in classification: a multi-objective approach. IEEE Trans Cybern 43(6):1656–1671
7. Dorigo M, Gambardella LM (1997) Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Trans Evol Comput 1(1):53–66
8. Kashef S, Nezamabadi-pour H (2013) A new feature selection algorithm based on binary ant colony optimization. in: 5th conference on information and knowledge technology (IKT). IEEE, pp 50–54
9. Moradi P, Gholampour M (2016) A hybrid particle swarm optimization for feature subset selection by integrating a novel local search strategy. Appl Soft Comput 43:117–130
10. Karaboga D, Basturk B (2007) A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm. J Glob Optim 39(3):459–471
11. Moayedikia A, Jensen R, Wiil UK, Forsati R (2015) Weighted bee colony algorithm for discrete optimization problems with application to feature selection. Eng Appl Artif Intell 44:153–167
12. Mirjalili S (2015) The ant lion optimizer. Adv Eng Softw 83:80–98
13. Zawbaa HM, Emary E, Parv B (2015) Feature selection based on antlion optimization algorithm. in: Third world conference on complex systems (WCCS). IEEE, pp 1–7
14. Zorarpacı E, Özel SA (2016) A hybrid approach of differential evolution and artificial bee colony for feature selection. Expert Syst Appl 62:91–103
15. Mirjalili S, Gandomi AH, Mirjalili S, Saremi S, Faris H, Mirjalili S (2017) Salp swarm algorithm: a bio-inspired optimizer for engineering design problems. Adv Eng Softw 114:163–191
16. Mirjalili S (2016) Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. Neural Comput Appl 27(4):1053–1073
17. Wang G, Guo L, Wang H, Duan H, Liu L, Li J (2014) Incorporating mutation scheme into krill herd algorithm for global numerical optimization. Neural Comput Appl 24(3-4):853–871
18. Gandomi AH, Yang XS (2014) Chaotic bat algorithm. J Comput Sci 5(2):224–232
19. Pereira M, Costa VS, Camacho R, Fonseca NA, Simões C, Brito RM (2009) Comparative study of classification algorithms using molecular descriptors in toxicological databases. In: Advances in Bioinformatics and Computational Biology. Springer, Berlin, pp 121–132
20. Huang R, Southall N, Xia M, Cho MH, Jadhav A, Nguyen DT, Inglese J, Tice RR, Austin CP (2009) Weighted feature significance (wfs): a simple, interpretable model of compound toxicity based on the statistical enrichment of structural features. Toxicol Sci 112(2):385–393
21. Tharwat A, Gaber T, Fouad MM, Snasel V, Hassanien AE (2015) Towards an automated zebrafish-based toxicity test model using machine learningProceedings of International Conference on Communications, management, and Information technology (ICCMIT'2015). Proced Comput Sci 65:643–651
22. Klopman G (1984) Artificial intelligence approach to structure-activity studies. computer automated structure evaluation of biological activity of organic molecules. J Am Chem Soc 106(24):7315–7321
23. Prival MJ (2001) Evaluation of the topkat system for predicting the carcinogenicity of chemicals. Environ Mol Mutagen 37(1):55–69
24. Woo YT, Lai DY, Argus MF, Arcos JC (1995) Development of structure-activity relationship rules for predicting carcinogenic potential of chemicals. Toxicol Lett 79(1):219–228
25. Sander T, Freyss J, von Korff M, Rufener C (2015) Datawarrior: an open-source program for chemistry aware data visualization and analysis. J Chem Inf Model 55(2):460–473
26. Tharwat A, Moemen YS, Hassanien AE (2017) Classification of toxicity effects of biotransformed hepatic drugs using whale optimized support vector machines. J Biomed Inform 68:132–149
27. He H, Garcia EA (2009) Learning from imbalanced data. IEEE Trans Knowl Data Eng 21(9):1263–1284
28. López V, Fernández A, García S, Palade V, Herrera F (2013) An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. Inf Sci 250:113–141
29. Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP (2002) Smote: synthetic minority over-sampling technique. J Artif Intell Res 16:321–357
30. López V, Fernández A, Moreno-Torres JG, Herrera F (2012) Analysis of preprocessing vs. cost-sensitive learning for imbalanced classification. open problems on intrinsic data characteristics. Expert Syst Appl 39(7):6585–6608
31. López V, Fernández A, Del Jesus MJ, Herrera F (2013) A hierarchical genetic fuzzy system based on genetic programming for addressing classification with highly imbalanced and borderline data-sets. Knowl-Based Syst 38:85–104
32. Reynolds CW (1987) Flocks, herds and schools: a distributed behavioral model. ACM SIGGRAPH Comput Graph 21(4):25–34
33. Tharwat A, Hassanien AE (2018) Chaotic antlion algorithm for parameter optimization of support vector machine. Appl Intell 48(3):670–686
34. Zhang Q, Li Z, Zhou CJ, Wei XP (2013) Bayesian network structure learning based on the chaotic particle swarm optimization algorithm. Genet Mol Res 12(4):4468–4479
35. Saremi S, Mirjalili S, Lewis A (2014) Biogeography-based optimization with chaos. Neural Comput Appl 25(5):1077–1097
36. Sarafrazi S (2013) Facing the classification of binary problems with a gsa-svm hybrid system. Math Comput Model 57:270–278
37. Tharwat A, Mahdi H, Elhoseny M, Hassanien AE (2018) Recognizing human activity in mobile crowdsensing environment using optimized k-nn algorithm. Expert Syst Appl 107:32–44
38. Tharwat A (2016) Linear vs. quadratic discriminant analysis classifier: a tutorial. Int J Appl Pattern Recogn 3(2):145–180
39. Liu Z, Cui Y, Li W (2015) A classification method for complex power quality disturbances using eemd and rank wavelet svm. IEEE Trans Smart Grid 6(4):1678–1685
40. Sun L, Liu H, Zhang L, Meng J (2015) lncrscan-svm: a tool for predicting long non-coding rnas using support vector machine. PloS one 10(10):e0139654
41. Keerthi SS, Lin CJ (2003) Asymptotic behaviors of support vector machines with gaussian kernel. Neural Comput 15(7):1667–1689
42. Tharwat A, Hassanien AE, Elnaghi BE (2017) A ba-based algorithm for parameter optimization of support vector machine. Pattern Recogn Lett 93:13–22

43. Poli R, Kennedy J, Blackwell T (2007) Particle swarm optimization. Swarm Intell 1(1):33–57
44. Mirjalili S, Mirjalili S, Lewis A (2014) Grey wolf optimizer. Adv Eng Softw 69:46–61
45. Meng X, Liu Y, Gao X, Zhang H (2014) A new bio-inspired algorithm: chicken swarm optimization. In: International conference in swarm intelligence. Springer, Berlin, pp 86–94
46. Askarzadeh A (2016) A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm. Comput Struct 169:1–12
47. Mirjalili S (2016) Sca: a sine cosine algorithm for solving optimization problems. Knowl-Based Syst 96:120–133
48. Wilcoxon F (1945) Individual comparisons by ranking methods. Biom Bull 1:80–83
49. Derrac J, García S., Molina D, Herrera F (2011) A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm Evoloutionary Comput 1(1):3–18
50. Lin SW, Ying KC, Chen SC, Lee ZJ (2008) Particle swarm optimization for parameter determination and feature selection of support vector machines. Expert Syst Appl 35(4):1817–1824
51. Schiezaro M, Pedrini H (2013) Data feature selection based on artificial bee colony algorithm. EURASIP J Image Video Process 2013(1):1–8
52. Sayed GI, Soliman M, Hassanien AE (2016) Bio-inspired swarm techniques for thermogram breast cancer detection. In: Medical Imaging in Clinical Applications. Springer, pp 487–506
53. Hafez AI, Zawbaa HM, Emary E, Mahmoud HA, Hassanien AE (2015) An innovative approach for feature selection based on chicken swarm optimization. In: 2015 7th international conference of soft computing and pattern recognition (SoCPaR). IEEE, pp 19–24
54. Hafez AI, Zawbaa HM, Emary E, Hassanien AE (2016) Sine cosine optimization algorithm for feature selection. In: International symposium on INnovations in intelligent systems and applications (INISTA). IEEE, pp 1–5
55. Sayed GI, Khoriba G, Haggag MH (2018) A novel chaotic salp swarm algorithm for global optimization and feature selection. Appl Intell, pp 1–20
56. Sayed GI, Hassanien AE, Azar AT (2017) Feature selection via a novel chaotic crow search algorithm. Neural Comput Applic, pp 1–18

Gehad Ismail Sayed is a Ph.D. student since 2016. She received her B.Sc. with honours in 2013 and M.Sc degree in 2016, both from Cairo University, Faculty of Computers and Information, Information Technology Department, Cairo, Egypt. Her research areas include computational intelligence, medical image analysis, swarm intelligence and data mining.

Alaa Tharwat received his MSc in 2008 Mansoura University, and PhD in 2017 from Suez Canal University. He worked as a researcher at Gent University (2015). Currently, he is a Postdoc in Faculty of Computer Science and Engineering, Frankfurt University of Applied Sciences, Frankfurt am Main, Germany.

Aboul Ella Hassanien is the Founder and Head of the Egyptian Scientific Research Group (SRGE) and Professor of Information Technology at the Faculty of Computer and Information, Cairo University. He has more than 500 scientific research papers published in prestigious international journals.