Evolutionary Intelligence (2022) 15:1777–1808
https://doi.org/10.1007/s12065-021-00590-1

RESEARCH PAPER

Chaotic vortex search algorithm: metaheuristic algorithm for feature selection

Farhad Soleimanian Gharehchopogh[1] · **Isa Maleki**[1] · **Zahra Asheghi Dizaji**[1]

Received: 2 July 2020 / Revised: 2 November 2020 / Accepted: 2 March 2021 / Published online: 20 March 2021

Abstract
The Vortex Search Algorithm (VSA) is a meta-heuristic algorithm that has been inspired by the vortex phenomenon proposed by Dogan and Olmez in 2015. Like other meta-heuristic algorithms, the VSA has a major problem: it can easily get stuck in local optimum solutions and provide solutions with a slow convergence rate and low accuracy. Thus, chaos theory has been added to the search process of VSA in order to speed up global convergence and gain better performance. In the proposed method, various chaotic maps have been considered for improving the VSA operators and helping to control both exploitation and exploration. The performance of this method was evaluated with 24 UCI standard datasets. In addition, it was evaluated as a Feature Selection (FS) method. The results of simulation showed that chaotic maps (particularly the Tent map) are able to enhance the performance of the VSA. Furthermore, it was clearly shown the fitness of the proposed method in attaining the optimal feature subset with utmost accuracy and the least number of features. If the number of features is equal to 36, the percentage of accuracy in VSA and the proposed model is 77.49 and 92.07. If the number of features is 80, the percentage of accuracy in VSA and the proposed model is 36.37 and 71.76. If the number of features is 3343, the percentage of accuracy in VSA and the proposed model is 95.48 and 99.70. Finally, the results on Real Application showed that the proposed method has higher percentage of accuracy in comparison to other algorithms.

Keywords Vortex Search Algorithm · Feature Selection · Chaotic Maps · Exploration · Exploitation · Accuracy

1 Introduction

The metaheuristic optimization algorithms were proposed over the past decades and implemented extensively to the problem of the complicated [1, 2]. The essential target in the optimization is the candidate the problem variables to minimize or maximize the objective function based on the global and local search [3, 4]. So as to triumph over the state-of-the-art goals in any problem, most of such algorithms were applied as an attempt to establish an approximate technique for attaining the optimum solution [5, 6]. A number of well-known new nature-inspired algorithms include the Invasive Weed Optimization (IWO) [7], the butterfly optimization algorithm (BOA) [8], the Artificial Bee Colony (ABC) [9], the Fruit Fly Optimization Algorithm (FOA) [10], the Firefly Algorithm (FA) [11], the Krill Herd (KH) algorithm [12], the Differential Evolution (DE) algorithm [13], the Flower Pollination algorithm (FPA) [14], etc. The distinction in nature is an essential factor why the algorithms have an alternate dimension of execution in delivering results [15, 16]. Besides, this factor might be the motivation behind why a few algorithms can best item an answer for specific issues, while others don't. Thus, it is according to this limitation that one algorithm is not good enough for solving every kind of problem.

During the past decade, an arithmetic framework and scientific branch, namely chaos, has been proposed, and is connected deeply with different scientific fields. Chaos involves three major dynamic properties: the quasi stochastic property, being sensitive against initial conditions, and ergodicity. The application of chaos theory in optimization research areas has attracted a lot of attention over the recent years. The Chaotic Optimization Algorithm (COA) [17] is among the applications of chaos, and uses the nature of chaos sequences. It has been indicated if random variables are replaced with chaotic variables, the performance of COA can be enhanced. Therefore, in the literature, there are a

✉ Farhad Soleimanian Gharehchopogh
bonab.farhad@gmail.com

[1] Department of Computer Engineering, Urmia Branch, Islamic Azad University, Urmia, Iran

number of studies on the hybridization of chaos with other algorithms for the purpose of improving the performance of COA. Some instances include the chaotic ACO [18], chaotic DE algorithm [19, 20], chaotic KH algorithm [21, 22], chaotic FPA [23], chaotic genetic algorithm [24, 25], chaotic PSO [26–28], chaotic gravitational search [29–31], chaotic bat algorithm [32], and etc.

FS is the procedure of selecting a subset of features from an original feature set; it may be considered the most important pre-processing instrument to solve classification issues [33]. Figuring out a superior subset of features is a quite complicated challenge, and is decisive in the final results of the rates of classification error. The finalized feature subset will retain high rates of classification accuracy. The purpose is choosing an applicable subset including *d* features from a set of *D* features (d < D) in a given dataset [34]. *D* is made out of all features that are present in a given data set; it can encompass redundant, noisy, and misleading features. Consequently, an exhaustive search is performed within the whole solution environment, which usually takes a lot of time and cannot often be implemented in practice. To remedy this FS strategy, maintaining the best subset of *d* relevant features was taken into consideration. Inappropriate features are not only useless, but also can certainly worsen the classification performance. If irrelevant features are deleted, the computational efficiency can be advanced and classification accuracy expanded.

As indicated by search techniques of feature subsets, the current FS strategies can be classified into two classes: the filter-based approach and the wrapper-based approach. The filter method depends fundamentally on general qualities of datasets to assess and choose include subsets without taking into account an uncommon learning approach. Thus, the productivity of this methodology depends predominantly on the dataset itself instead of on the classifier [35, 36]. The wrapper method utilizes a classification calculation to assess feature subsets and embraces a search system to look for ideal subsets. It often leads to better since the wrapper approach takes into consideration a classifier with the evaluation or search process [37].

Each meta-heuristic algorithm has a unique search strategy. Meta-heuristic algorithms can find optimal solutions based on its own strategy such as balance between exploration and exploitation. Furthermore, VSA has advantages such as the smaller number of parameters and easy implementation. The VSA was embedded with chaotic maps to obtain a better compromise between exploitation and exploration. This paper uses hybrid methods based on CMs with the VSA for FS. The major contribution of the current paper is that a CMs model of VSA has been proposed to enhance the performance of VSA. In proposed methods, the chaotic seek method are followed to choose the ideal characteristic subset that maximizes the category accuracy and minimizes the feature subset duration. Ten one-dimensional CMs are adopted and changed with random movement parameters of the VSA. The performance of the proposed methods is tested on 24 benchmark datasets. Similarly, the performance of VSA is comparison with seven other metaheuristic algorithms. Based on mean criterion, the proposed method can obtain better solutions using the Tent Map in comparison with other metaheuristic algorithms.

The main contributions of this paper are as follows:

- VSA and Chaotic Maps are defined to FS.
- The proposed method has a faster convergence performance than the other algorithms. The proposed method has better convergence results on different datasets.
- The proposed method has been evaluated with 24 UCI standard datasets.
- The best VSA is State2 with VSAC101 that obtained by using the Tent map.
- The proposed method has been tested on author identification datasets
- The obtained results confirmed the validity and superiority of the proposed method in comparison to other algorithms.

The organization of this paper is as follows: Sect. 2 gives related works about chaotic and FS. Section 3 provides an introduction to VSA. The detailed description of the proposed method has been provided in Sect. 4, while the experimental results and discussion of the proposed VSA have been provided in Sect. 5. In Sect. 6, the proposed method has been applied on a real application (i.e., author identification). Finally, the conclusion and future work have been discussed in Sect. 7.

2 Related works

The Moth Swarm Algorithm (MSA) is among the most recently-developed nature-inspired heuristics for the purpose of the optimization problem. However, its shortcoming is that it has slow convergence rate, and the Chaos theory has been incorporated into it to eliminate this drawback. In [38], ten CMs have been embedded within the MSA for the purpose of finding the ideal number of prospectors to increase exploiting the most promising solutions. The proposed method was applied in solving the famous seven benchmark test functions. The results of simulation showed that CMs can enhance the performance of the original MSA with regard to the convergence speed. In addition, the sinusoidal map was found to be the best map for enhancing the performance of MSA.

The Cuckoo search algorithm (CSA) is a metaheuristic algorithm that has been inspired by nature and imitates

the obligate brood parasitic behavior of the cuckoo species. The method has been proven to have promising overall performance in solving optimization problems. Chaotic mechanisms were incorporated into CSA to make use of the dynamic features of the chaos theory, to further improve its search overall performance. However, in chaotic CSA (CCSA) [39], the best CM was applied in a single search of the new release, which restrained the exploitation capability of the search. The researchers considered utilizing multiple CMs at the same time to perform the nearby search inside the community of the global best solution that is found by CSA. To attain this goal, three kinds of multiple chaotic CSAs (MCCSA) were proposed via incorporating several CMs into the chaotic local search (CLS) parallel in a random or selective manner. The overall performance of MCCSA was validated using 48 broadly-used benchmark optimization features. The experimental results indicated that MCCSAs are generally better than CCSAs, and the MCCSA-P that makes use of the CMs has the best quality among all sixteen editions of the CSAs.

In [40], a chaos-based Crow Search Algorithm (CCSA) has been proposed to solve the fractional optimization problems (FOPs). The proposed CCSA integrated the chaos theory (CT) into the CSA for the purpose of refining the global convergence velocity and enhance the exploration/exploitation inclinations. CT was utilized to track the standard CSA parameters, which yielded four versions and the high-quality chaotic variant was investigated. The incorporation of CT was able to improve the overall performance of the proposed CCSA and allow the search process to perform better speeds. The overall performance of the CCSA method was proven on twenty fractional benchmark problems. Furthermore, it was further tested on a fractional monetary environmental power dispatch problem via attempting to limit the ratio of the overall emissions to general gasoline cost. Ultimately, the proposed CCSA was compared with the PSO, standard CSA, FA, Dragonfly Algorithm (DA), and GWO. In addition, the efficiency of the proposed CCSA was justified by the non-parametric Wilcoxon signed-rank test. The experimental results proved that the proposed CCSA performs better than similar algorithms with regard to efficiency and reliability.

In [41], a new hybrid algorithm for solving optimization problems based on chaotic ABC and chaotic simulated annealing has been proposed. The chaotic ABC reveals new locations chaotically. Chaos may additionally improve the exploration of the search space. Really, the proposed hybrid method affords a hybrid of nearby search accuracy of simulated annealing and the capacities of global seek of ABC. Moreover, they used an exclusive method for producing the initial population. Sincerely preliminary populace is of brilliant significance for populace-based techniques, because it immediately influences the rate of convergence and nice of the outcomes. It is established the usage of 12 benchmark functions. The effects are as compared with those of the artificial bees' algorithm, the hybrid algorithm of ABC and simulated annealing and PSO. Simulation effects display the performance of the proposed method.

In [42], an adaptive chaotic Bacterial Foraging Optimization (BFO) is presented. The improved BFO consisted of two new features, the adaptive chemotaxis step setting, and the chaotic perturbation operation in all chemotactic events. The former feature results in fast convergence rate and the acceptable convergence accuracy in the algorithm, while the latter further allows the search to avoid the local optima and attain better convergence accuracy. Firstly, an idea of adaptive exponential decease chemo taxis step is presented, in which the natural exponential function variable is a function about the iterations and nutritive ratio between the current bacterium position and the best bacterium position in each iteration. Secondly, when each bacterium reaches a new position through swim behavior, chaotic perturbation is applied to avoid entrapping into local optima. With five benchmark functions, Chaotic BFO is proved to have a better performance than the original BFO and BFO with linear deceasing chemo taxis step (BFO-LDC).

Jia et al. [43] proposed an effective memetic DE algorithm (DECLS), which makes use of a CLS with a 'shrinking' strategy. The shrinking strategy for the CLS search space was introduced in that paper. In addition, the local search length was determined according to the feedback of the fitness of the objective functions in a dynamic manner in order to save the function evaluations. Furthermore, the parameter settings of the DECLS were adapted in the process of evolution so as to further enhance the optimization efficiency. The hybrid form of the DE and a CLS as well as a parameter adaptation mechanism seemed very reasonable. The CLS is helpful in enhancing the local search capability of DE, whereas the parameter adaptation can improve the global optimization quality. The CLS is helpful in improving the optimization performance of the canonical DE through exploring a very large search space in the early phases so as to avoid the occurrence of premature convergence, and exploiting a tiny region in later phases to refine the finalized solutions. In addition, the settings of parameters in the DECLS were controlled adaptively to further improve the search capability. To assess the efficiency and effectiveness of the proposed DECLS algorithm, it was compared with four state-of-the-art DE variants and the IPOP-CMA-ES algorithm on a set of 20 selected benchmark functions. The findings showed that the DECLS is significantly superior, or at least comparable, to other optimizers with regard to the convergence performance and solution accuracy. Furthermore, the DECLS was shown to have certain advantages in terms of solving problems with high dimensions.

In [44], a modified DE algorithm based on the Opposition-based Learning (OBL) and a chaotic sequence named the OBL Chaotic DE (OBL-CDE) was proposed. The proposed OBL-CDE algorithm is different from the basic DE in two ways. The first one is related to the generation of the initial population that follows the OBL rules, while the second one is the dynamic adaption of the scaling factor F through using the chaotic sequence. The numerical results obtained by the OBL-CDE compared to the results of DE and the opposition-based DE algorithms on 18 benchmark functions indicated that the OBL-CDE is capable of finding more superior solutions and maintaining reasonable convergence rates at the same time.

The standard Glowworm Swarm Optimization (GSO) shows poor ability in global search and easily gets trapped into local optima. A Quantum GSO algorithm based on CMs was proposed [45] in order to solve such problems. First of all, a chaotic sequence was generated to initialize the population. This process results in higher probability to cover more local optimal areas, and provides the ground for further optimization and tuning. Next, the quantum behavior was applied to the elite population, which made it possible for individuals to locate any position of the solution space randomly with a certain probability. This greatly enhanced the capability of the algorithm in global search and avoiding local optima. Finally, it adopted the single dimension loop swimming instead of the original fixed-step movement mode. This not only improved the solution precision and convergence speed, but also solved GSO problems that were too sensitive to the step-size, and indirectly enhanced the robustness of the algorithm. The simulation results indicated that the proposed method was feasible and effective.

The Fruit Fly Algorithm (FOA) has recently been proposed as a metaheuristic technique, and is inspired by the behavior of fruit flies. Mitic et al. [46] improved the standard FOA through introducing the novel parameter in combination with chaos. The performance of this chaotic FOA (CFOA) was studied on ten famous benchmark problems using 10 different CMs. In addition, comparison studies with the basic FOA, FOA with Levy flight distribution, and other recently-published chaotic algorithms were made. Statistical findings on each optimization task showed that the CFOA results in a very high convergence rate. In addition, CFOA is compared with recently developed chaos enhanced algorithms such as chaotic bat algorithm, chaotic-accelerated PSO, chaotic FA, chaotic ABC, and chaotic CSA. Research findings generally indicate that FOA with Chebyshev map show superiority to the similar methods in terms of the reliability of global optimality and the algorithm success rate.

In addition, Gandomi et al.[47] proposed a chaos-enhanced version of the accelerated PSO. Some other instances of chaos-enhanced metaheuristic algorithms include the chaotic Genetic Algorithm [48], Chaotic PSO [49, 50], Chaotic Salp Swarm Algorithm [51], Chaotic Elephant Herding Optimization (EHO) algorithm [52], Chaotic Bat Algorithm[53], Chaotic FOA[46], Chaotic GSO Algorithm [45, 54], Chaotic Black Hole algorithm [55], Chaotic Simulated Annealing PSO Algorithm (CSAPSO) [56], Chaotic Social Spider Optimization Algorithm[57], Chaotic Bean Optimization Algorithm[58], Chaotic Quantum CSA [59], Chaotic Antlion Algorithm[60], Chaotic Hybrid Cognitive Optimization Algorithm[61], Chaotic Simulated Annealing [62], Chaotic Based Quantum Genetic Algorithm [63], Chaotic Teaching Learning Algorithm[64], Chaotic DE algorithm [65], Chaotic Grey Wolf Optimization Algorithm[66], Chaotic Fractal Search[67], Chaotic Brain Storm Optimization Algorithm [68], Multi-Objective CCSA [69], Chaotic Grasshopper Optimization Algorithm [70], Chaotic Krill Herd [21, 71, 72], Chaotic DE[73], Chaotic Firefly Algorithm [74, 75], Chaotic Starling PSO Algorithm[76], Chaotic CCSA [77], Chaotic Grey Wolf Optimization Algorithm [78] and etc. Table 1 shows a comparison of different models of meta-heuristic algorithms based on chaotic map.

3 Vortex search algorithm

The VSA is a recent metaheuristic optimization algorithm that changes into the stimulated mode by the vertical flow of the stirred fluids. Its processes consist of the simplified generation phases similar to other single-solution algorithms. The generation of VSA populations is modified to any generations with the aid of the usage of values completely shape the modern single solution. Furthermore, the performance of every update and seek of iteration pass at the seek space is an essential section in rendering single-solution. Inside the proposed VSA, this stability is performed with the aid of using a vortex-like search pattern. The strategies of vortex sample are simulated through some of the nested circles. The info of VSA techniques may be in brief defined in 4 steps as follow [79].

3.1 Generating the initial solution

The preliminary procedure initials 'center' μ_0 and 'radius' r_0. In this phase, the initial *center* (μ_0) can be calculated using Eq. (1).

$$\mu_0 = \frac{upperlimit + lowerlimit}{2} \tag{1}$$

where *upperlimit* and *lowerlimit* are the bound constraints of the problem, which can be defined in vector of $d \times 1$ dimensional-space. In addition, σ_0 is the initial *radius* r_0 generated with Eq. (2).

Table 1 A Comparison of Different Models of Meta-heuristic Algorithms based on Chaotic Maps

Refs	Models	Application	Chaotic map	Improved
[38]	Chaotic MSA	Optimization problem	Sinusoidal	*Convergence speed
[39]	CCSA	Solving optimization problems	Gaussian map	*Exploration and exploitation
[40]	CCSA	Optimization problems	Circle map	*Obtaining the global optimum *accelerate the convergence performance
[41]	Chaotic artificial bee colony and chaotic simulated annealing	Solving optimization problems	Sinusoidal map	*Faster convergence *better exploration
[42]	Adaptive chaotic BFO (ACBFO)	Solving optimization problems	Logistic map	*Convergence speed
[43]	DE algorithm based on chaotic local search	20 benchmark functions	Logistic chaotic function	*Convergence performance *solution accuracy
[44]	Opposition based Chaotic DE (OCDE)	Benchmark function	Logistic chaotic function	*Obtaining the global optimum *accelerate the convergence performance
[45]	Quantum GSO algorithm based on Chaotic Sequence (QCSGSO)	Solving optimization problems	Logistic chaotic function	*Avoid from prematurity
[46]	Chaotic FOA	Multi-mode functions	Logistic chaos mapping	*Convergence speed
[47]	Chaos-enhanced accelerated PSO	*Global optimization *three engineering problems	*Sinusoidal map *Singer map	*Global optimality *convergence speed
[48]	Chaos-Genetic Algorithm	Grid scheduling	Logistic map	*Convergence performance *solution accuracy
[49]	An improved chaotic PSO algorithm based on adaptive inertia weight (AIWCPSO)	Benchmark functions	Logistic map	*Convergence accuracy
[50]	Chaotic PSO algorithm (CS-PSO)	Recommendation system	Chaos methods	*Global search capability; *avoiding the premature convergence;
[51]	Chaotic Salp Swarm Algorithm (CSSA)	*Global optimization; *feature selection;	Logistic map	*Minimizing the number of selected features; *maximizes the classification accuracy;
[52]	Chaotic EHO algorithm	15 benchmark functions from CEC 2013	Tent map	*Convergence speed
[53]	Chaotic bat algorithm	Robust global optimization	Thirteen different chaotic maps	*Global search capability; *avoiding the premature convergence;
[54]	Chaotic GSO algorithm	Eight standard test functions	Logistic map	*Convergence performance *solution accuracy
[55]	Chaotic Inertia Weight Black Hole Algorithm (CIWBH)	Twenty-three benchmark functions	Logistic map	*Enhance the global search *trade-off between exploitation and exploration *better convergence
[56]	Chaotic Simulated Annealing PSO Algorithm	Complex optimization problems	Logistic map	*Global search ability; *convergence precision;
[57]	Chaotic Social Spider Optimization Algorithm	Robust Clustering	Logistic map	*Convergence speed
[58]	Chaotic bean optimization algorithm (CBOA)	CEC2014 benchmark functions	Logistic map	*Global optimization

$$\sigma_0 = \frac{\max(upperlimit) - \min(lowerlimit)}{2} \tag{2}$$

3.2 Generating the candidate solutions

The procedure of producing candidate solutions is applied for the purpose of rendering the generation of populations $C_t(s)$ in any iterations, where t is the t-th iteration. The VSA is randomly generated around the initial center μ_0 by using a Gaussian distribution, where $C_0(s) = \{s_1, s_2, \ldots, s_m\} m = 1, 2, 3, \ldots, n$ represents the solution and n is the overall number of candidate solutions. The equation of multivariate Gaussian distribution has been shown in Eq. (3).

$$p(x|\mu,\Sigma) = \frac{1}{\sqrt{(2\pi)^d|\Sigma|}} exp\left\{ -\frac{1}{2}(x-\mu)^T \sum^{-1} (x-\mu) \right\} \quad (3)$$

In Eq. (3) d indicates the dimension, while x is the d × 1 vector of a random variable, μ indicates the d × 1 vector of the sample mean (i.e., center), and Σ indicates the covariance matrix. Equation (4) indicates that when the diagonal elements (i.e., variances) of the Σ values are equal and the off-diagonal elements (i.e., covariance) equal zero (uncorrelated), the resulting shape of the distribution will be spherical. Thus, the value of Σ is computed through utilizing equal variances with zero covariance.

$$\sum = \sigma^2.[I]_{d\times d} \quad (4)$$

where the representation in Eq. (4), σ^2 is the variance of the distribution, I represent the $d \times d$ identity matrix and σ_0 is the initial radius (r_0) as can see in Eq. (2).

3.3 Replacement of the current solution

The replacement of the current solution is conducted for the selection process. A solution (which is the best one) $s \in C_0(s)$ is selected and memorized from $C_0(s)$ for the purpose of replacing the current circle center (μ_0). Before the selection process, it must be made sure that the candidate solutions are inside the search spaces (Eq. (5)).

$$s_k^i = \begin{cases} rand.\left(upperlimit^i - lowerlimit^i\right) + lowerlimit^i, s_k^i < lowerlimit^i \\ lowerlimit^i \le s_k^i \le upperlimit^i \\ rand.\left(upperlimit^i - lowerlimit^i\right) + lowerlimit^i, s_k^i > upperlimit^i \end{cases} \quad (5)$$

where $k = 1, 2, \ldots, n$ and $i = 1, 2, \ldots, d$, and rand indicates a random number that is distributed uniformly. VSA uses s as a new center, and reduces the vortex size using Eq. (3) to select the next solutions. Thus, the new set of solutions $C_1(s)$ can be generated. If the chosen solution is better than the best solution, it can be determined as the new best solution and was memorized.

3.4 The radius decrement process

In the VSA, the inverse incomplete gamma function is applied for the purpose of decreasing the radius value during each iteration pass. The incomplete gamma function provided in Eq. (6) often arises in probability theory, especially in applications that involve the chi-square distribution.

$$\gamma(x,a) = \int_0^x e^{-t}t^{a-1}dta > 0 \quad (6)$$

where a > 0 is the shape parameter while x ≥ 0 is a random variable. Similar to the incomplete gamma function, its complementary $\Gamma(x,a)$ is usually also introduced (Eq. (7)). In Eq. (7), $\Gamma(a)$ is a (1).

$$\Gamma(x,a) = \int_0^\infty e^{-t}t^{a-1}dta > 0 \quad (7)$$

Table 2 describes pseudocode of VSA algorithm.

4 Proposed methods

In this section, the hybrid form of VSA and CMs will be explained. The simple shape of the VSA consists of important keys that can be center and radius. First, the center is a current position from which the VSA may be evaluated

Table 2 A description of the VSA algorithm

Initializing step
- Algorithm parameters: Input the population size, the lower and upper bounds
- Fitness of best solution
- Center of the circle(μ_0), Eq. (1)
- Radius of the circle (σ_0), Eq. (2)

Repeat
- Create candidate solutions within the circle by Eq. (3)
- If Exceeded, then shift values into the boundaries by Eq. (5)
- Select best solution to replace the current center
- Decrease the standard deviation (radius) for the next iteration by Eq. (7)

End

Output
- Best solution found so far S_{best}

based on the problem search space where iterations skip. With respect to exploration for a premier solution that has been carried out up to now, VSA used this function to identify the 'center' with the purpose of replacing a new position of the populations. Secondly, 'radius' is a method that is utilized to simplify the issues-creating a massive-radius problem to grow to be a small-radius problem. In extra, a Gaussian distribution is a VSA which is used to stability the exploration and exploitation at every iteration skip. However, the VSA used best a single center this is referred to as the single strategy to generate candidate solutions around the current great answer. However, the disadvantages of the VSA can be not noted from the local factor whilst it suffers from the issues that have numerous neighborhood minimal values. in the equal time, the radius used to update the pleasant solution have been capable of decrease the new release skip by way of the usage of a Gaussian distribution, making it less complicated to trap the VSA in neighborhood optima. This explains some drawbacks of the VSA. The presented have a look at specializes in hybridizing the VSA with the CMs. This hybridization is referred to as the chaotic VSA which 10 CMs have been used. These 10 maps have been used in three different locations of the VSA [74]. Figure 1 shows flowchart of proposed method. In the first step is done the initialization of the parameters. In the second step, the VSA Eqs. (9, 10, and 11) are optimized based on the chaotic maps in order to FS. In the third step, the samples are classified and at the end, the accuracy percentage is displayed.

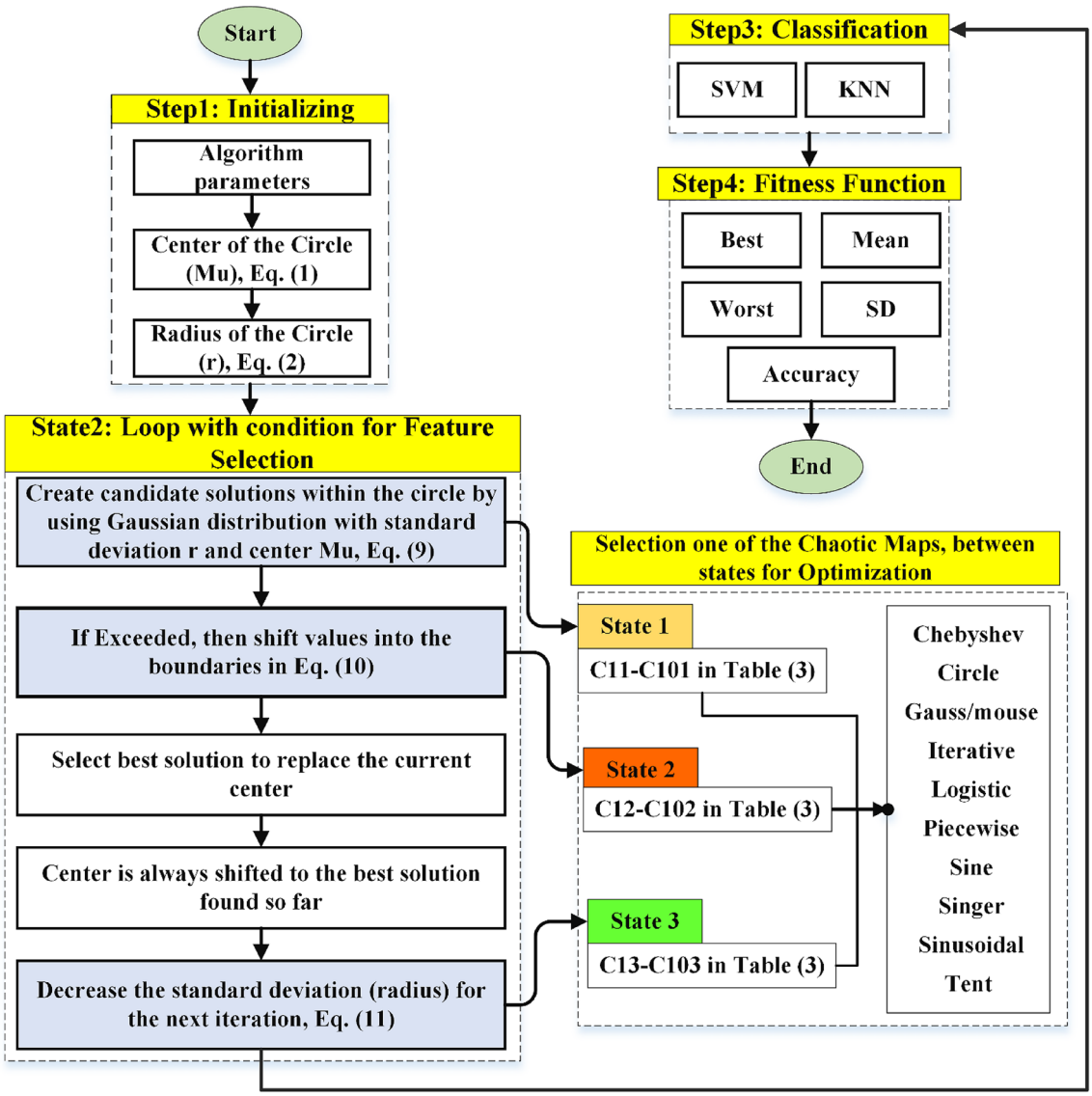

Fig. 1 Flowchart of Proposed Method

In the proposed model, we combine the formulas of CMs based on Table 3 with Eqs. (3), (5), and (6). The goal is to find the best CMs to optimize VSA. These places can be expressed as follows:

State 1 the production of candidate solutions inside the search circle [Eq. (9)].
State 2 If the solution is out of range, these mappings are used to move to the desired range. (Eq. 10).
State 3 Reduced search radius using reverse gamma function and CMs [Eq. (11)].

In Table 3, the CMs formulas and methods are shown. The optimization of the VSA based on three methods (State1, State2, and State3) has been done. In each method have been used 10 CMs. So, in each run, there are 30 different modes for a given dataset.

Chaos is described as a phenomenon. Any exchange of its preliminary scenario might also purpose non-linear change for future behavior. Chaos optimization is one of the optimization models for search algorithms. The primary idea behind it is too seriously change parameters/variables from the chaos to the solution area. It relies upon for looking out of the global optimum on chaotic motion properties including ergodicity, regularity, and stochastic properties. The major advantages of chaos are speedy convergence and functionality for warding off local minima. CMs have a form of determinate in which no random factors are applied. In this paper, 10 distinguished non-invertible unidimensional maps were adopted to attain chaotic sets. The adopted CMs have been defined in Table 3, where q denotes the index of the chaotic sequence p, and p_q is the q^{th} number in the chaotic sequence. The remaining parameters including d, c, and μ are considered as the control parameters, determining the chaotic behavior of the dynamic system. The initial point p_0 was set to 0.7 for all CMs, as the initial values for CMs may have a great influence of fluctuation patterns on CMs. In this paper, ten different CMs were applied for the optimization process. These maps are Chebyshev, circle, gauss/mouse, iterative, logistic, piecewise, sine, singer, sinusoidal, and tent [74].

Descriptions of State 1, State 2, and State 3 are as follows:

State 1 VSA generates candidate solutions using just a single 'center' (μ). The generation of 'center' is then transformed to new center when iterations pass through the limitation of upper and lower bound of problems. This mechanism has some problems. One of such problems is that VSA tends to be trapped in local minima when suffering from a local point of minimum problems. To overcome this, the CMs of candidate solution VSA was proposed.

In this method, chaos maps are used to generate candidate solutions. Several neighbor solutions $C_t(s)$, (t indicates the iteration index and is t=0 at initial stages) were generated randomly around the initial center μ_0 in the d-dimensional

Table 3 CMs and proposed methods

ID	Map	Definition	Range	State1	State2	State3
1	Chebyshev map	$p_{q+1} = \cos\left(qcos^{-1}\left(p_q\right)\right)$	(−1,1)	C11	C12	C13
2	Circle map	$p_{q+1} = mod\left(p_q + d - \left(\frac{c}{2\pi}\right)\sin\left(2\pi p_q\right), 1\right), c = 0.5\ and\ d = 0.2$	(0,1)	C21	C22	C23
3	Gauss/mouse map	$p_{q+1} = \begin{cases} 1, & p_q = 0 \\ \frac{1}{mod(p_q,1)}, & otherwise \end{cases}$	(0,1)	C31	C32	C33
4	Iterative map	$p_{q+1} = \sin\left(\frac{c\pi}{p_q}\right), c = 0.7$	(−1,1)	C41	C42	C43
5	Logistic map	$p_{q+1} = cp_q\left(1 - p_q\right), c = 4$	(0,1)	C51	C52	C53
6	Piecewise map	$p_{q+1} = \begin{cases} \frac{p_q}{l}, 0 \le p_q < l \\ \frac{p_q - 1}{0.5 - l}, l \le p_q < 0.5 \\ \frac{1 - l - p_q}{0.5 - l}, 0.5 \le p_q < 1 - l, l = 0.4 \\ \frac{1 - p_q}{l}, 1 - l \le p_q < 1 \end{cases}$	(0,1)	C61	C62	C63
7	Sine map	$p_{q+1} = \frac{c}{4}\sin\left(\pi p_q\right), c = 4$	(0,1)	C71	C72	C73
8	Singer map	$p_{q+1} = \mu(7.86p_q - 23.31p_q^2 + 28.75p_q^3 - 13.302875p_q^4, \mu = 1.07$	(0,1)	C81	C82	C83
9	Sinusoidal map	$p_{q+1} = cp_q^2\sin\left(\pi p_q\right), c = 2.3$	(0,1)	C91	C92	C93
10	Tent map	$p_{q+1} = \begin{cases} \frac{p_q}{0.7}, p_q < 0.7 \\ \frac{10}{3}\left(1 - p_q\right), p_q \ge 0.7 \end{cases}$	(0,1)	C101	C102	C103

space by using a Gaussian distribution and CMs. Here, $C_0(s) = \{s_1, s_2, \ldots, s_m\} m = 1, 2, 3, \ldots, n$ represents the solutions, and n represents the total number of candidate solutions. In Eq. (9), the formula of the proposed method is given.

$$p(x|\mu) = \frac{1}{\sqrt{2\pi^d}} exp\left\{ -\frac{1}{2}(cm - \mu)^T \sum^{-1} (cm - \mu) \right\} \quad (9)$$

where d represents the dimension, cm is the $d \times 1$ vector of a CMs variable, μ is the d × 1 vector of sample mean (center), and Σ is the covariance matrix.

State 2 If the solution is out of range, these mappings are used to move to the desired range. During the selection phase, a solution (i.e., the best one), $s \in C_0(s)$ is selected and memorized from C_0(s) for the purpose of replacing the current circle *center* μ_0. Before the selection phase, it must be made sure that the candidate solutions are inside the search boundaries. To attain this goal, the solutions that exceed the boundaries are shifted into the boundaries, as in Eq. (10). The VSA combined with chaotic sequences is described in Eq. (10). In Eq. (10), $Cm(i)$ is the obtained value of chaotic map at j^{th} iteration.

$$s_k^i = \begin{cases} Cm_i * \left(\text{upperlimit}^i - \text{lowerlimit}^i\right) + \text{lowerlimit}^i, s_k^i < \text{lowerlimit}^i \\ \text{lowerlimit}^i \le s_k^i \le \text{upperlimit}^i \\ Cm_i * \left(\text{upperlimit}^i - \text{lowerlimit}^i\right) + \text{lowerlimit}^i, s_k^i > \text{upperlimit}^i \end{cases} \quad (10)$$

State 3 Reduced search radius using reverse gamma function and CMs. In the VSA, the inverse incomplete gamma function is used for the purpose of decreasing the value of the radius during each iteration pass. The incomplete gamma function has been given in Eq. (11).

$$\gamma(x.a) = \int_0^{cm} e^{-t} t^{a-1} dta > 0 \quad (11)$$

where a > 0 is known as the shape parameter and cm ≥ 0 is a CMs variable.

In the current study, the chaotic VSA has been implemented as an FS algorithm based on the wrapper method. In VSA, a chaotic sequence is embedded in the search iterations, and the optimal feature subset that describes the dataset is selected using VSA. The FS strategy is aimed at improving the classification efficiency, reducing the length of feature subset, and reducing the computational costs.

4.1 Fitness function

At each iteration, every point position is evaluated the use of a special fitness function fit. The data are randomly divided into extraordinary components, especially training and testing datasets by using the m-fold techniques. Goal standards are used for assessment, which are classification accuracy and the number of selected features. The followed fitness function equation hybrids the two standards into one by means of setting a weight factor as in Eq. (12). a is the class accuracy calculated with the aid of dividing the variety of efficiently labeled instances over the full variety of instances. K-nearest neighbor (KNN) [80] is the used classifier in which k equals to three with suggesting absolute distance. KNN is one in every of supervised learning algorithms which rely on classifying new instance based totally on distance from the new sample to the training samples. The KNN classifier predicts the class of the testing sample through calculating and sorting the distances between the testing sample and each one of the training samples. Such a process is repeated until each datum in the dataset has been selected once as the testing sample. What is meant by the classification accuracy of a feature subset is the ratio of the number of samples that have been predicted correctly to that of all the samples. In this paper, KNN has been used for determining the fitness of the selected features. The selection of K and distance method was decided based on trial and error. L_s is the length of the selected feature subset, L_n is the total number of features, and β is the weighted factor which has value in [0, 1]. β is used to control the importance of classification accuracy and the number of selected features. Since improving accuracy is the primary goal for any classifier, the weight factor is usually set to values near 1 [81]. In this paper, β was set to 0.8. The best solution is maximizes the classification accuracy and minimizes the number of selected features [81].

$$Fit = maximize\left(a + \beta \times \left(1 - \frac{L_n}{L_s}\right)\right). \quad (12)$$

5 Result and discussion

In this section, first a summary of the main characteristics of the implemented datasets will be discussed. Second, the proposed methods (State1, State2 and State3) using different CMs will be investigated. Third, comparisons

will be made between VSA and the proposed method based on FS. Finally, to emphasize the advantages of the proposed method compared to other algorithms, different experiments will be described and the obtained results will thoroughly be discussed.

5.1 Datasets description

Twenty-four benchmark datasets from different types including medical/biology and business were used in the experiments. Four datasets (21, 22, 23, and 24) were related to the identification and classification of the text author. The datasets were collected from the UCI machine learning repository [82]. A short description of each one of the adopted datasets has been presented in Table 4. As it can be observed, the used datasets involve missing values in some records. In the current study, all such missing values were replaced by the median value of all known values of a given feature class. The mathematical definition of the median method has been defined in Eq. (13). $S_{i,j}$ parameter is the missing value for j^{th} feature of a given i^{th} class W. For missing categorical values, the most appeared value for a feature given class is replaced with the missing value.

$$\bar{s}_{i,j} = median_{i:s_{i,j} \in W_r S_{i,j}} \tag{13}$$

Four different statistical measurements—including the worst, the best, the mean fitness value, and the standard deviation (SD) were adopted. In the current study, this test was used to evaluate the performance of each CM and determine the best one. The worst, the best, the mean fitness value, and the SD are mathematically defined as follows:

$$Best = max_{i=1}^{tMax} BS_i \tag{14}$$

$$Worst = min_{i=1}^{tMax} BS_i \tag{15}$$

$$Mean = \frac{1}{tMax} \sum_{i=1}^{tMax} BS_i \tag{16}$$

$$SD = \sqrt{\frac{\sum_{i=1}^{M} (BS_i - \mu)^2}{tMax}} \tag{17}$$

BS is the best score gained so far for each iteration.

Table 4 Dataset description

ID	Dataset	Missing values	No. of features	No. of classes	No. of instances	Type
Dataset1	Chess	No	36	2	3196	Game
Dataset2	Poker hand	No	10	10	25,010	Game
Dataset3	Germen credit	No	24	2	1000	Business
Dataset4	Credit approval	Yes	15	2	690	Business
Dataset5	Cylinder bands	Yes	40	2	512	Physical
Dataset6	Abalone	No	8	29	4177	Life
Dataset7	Glass identification	No	10	6	214	Physical
Dataset8	Letter recognition	No	17	26	20,000	Computer
Dataset9	Waveform	No	21	3	5000	Physical
Dataset10	Zoo	No	18	2	101	Life
Dataset11	Wisconsin Diagnosis Breast Cancer (WBCD)	No	32	2	596	Clinical
Dataset12	Mice Protein Expression Data set (MPED)	Yes	82	8	1080	Clinical
Dataset13	Parkinson's Disease Detection Data set (PDD)	No	23	2	197	Clinical
Dataset14	Cardiotocography	No	23	3	2126	Clinical
Dataset15	Hepatitis	Yes	19	2	155	Clinical
Dataset16	Lung cancer	Yes	56	3	32	Clinical
Dataset17	Single proton emission computed tomography (SPECT)	No	44	2	267	Clinical
Dataset18	Thoracic surgery	No	17	2	470	Clinical
Dataset19	Statlog (heart)	No	13	2	270	Clinical
Dataset20	Indian Liver Patient Dataset	No	10	2	583	Clinical
Dataset21	WebKB	No	2350	4	4199	Computer
Dataset22	Cade12	No	5340	12	40,983	Computer
Dataset23	Reuters 21,578 – R8	No	3343	8	7674	Computer
Dataset24	Reuter_50_50	No	2340	50	5000	Computer

5.2 Analysis and discussion

For evaluation of methods on different datasets of four criteria (worst, best, mean and SD) have been used. In Table 5, 30 modes and the VSA with the mentioned criteria are investigated. Proposed Method (State1) is equal to VSAC11 to VSAC101 modes, Proposed Method (State2) is equal to VSAC12 to VSAC102 modes, and Proposed Method (State3) is equal to VSAC13 to VSAC103 modes. With regard to the results, it can be stated that Proposed Method (State2) has better results. Proposed Method (State2) with VSAC101 mode offers best result than other modes using Tent map. The main target of this test is to evaluate the efficiency of VSA with different chaotic maps and define the optimal chaotic map (Tables 6, 7, 8).

5.3 Comparisons between VSA and proposed method based on FS

In Table 9 and Fig. 2, the results of the VSA and the Proposed Method are shown based on the FS. We chose the Proposed Method in order to FS because it had a high percentage of accuracy. Based on the results, it can be said that the Proposed Method in 19 datasets is better than the VSA.

5.4 Comparison and evaluation

Comparison of the Proposed Method with GA, PSO, ABC, BOA, IWO, FPA, and FA algorithms has been performed to evaluate the efficiency. In Table 10, the control parameters of the algorithms are expressed.

The comparison of the Proposed Method with the PSO, ABC, BOA, IWO, GA, FA, FPA, and VSA was performed according to the worst criteria. According to Table 11 and Fig. 3, it is clear that the results of other algorithms are worse than the Proposed Method.

In Table 12, the comparison of the Proposed Method with PSO, ABC, BOA, IWO, GA, FA, FPA, and VSA was performed based on the best criteria. According to Table 12 and Fig. 3, it is clear that the results of the Proposed Method are better than other algorithms.

In Table 13, the comparison of the Proposed Method with PSO, ABC, BOA, IWO, GA, FA, FPA, and VSA was performed based on the mean criteria. According to Table 13 and Fig. 3, it is clear that the results of the Proposed Method are better than other algorithms.

To sum up, the results and discussion of this paper demonstrate that integrating CMs to the VSA is definitely beneficial. The reason why that the Proposed Method outperforms all the other algorithms is that the Tent chaotic map assists this algorithm to highly emphasize exploration in the initial steps of optimization and reduced search radius.

6 Real application: author identification

Author identification, is a *stylometric* problem that tries to identify a copied text belonging to an original author [85, 86]. With ever-increasing volume of documents uploaded to the internet, new methods for analyzing and extracting data and knowledge are needed. In order to prevent plagiarism and copying copyrighted materials, the best solution is to use authorship identification. Every writer has his/her own writing style in manuscripts that he/she writes, and the writer's style can be identified in other papers [87]. Authorship identification is one of the up-to-date problems in the field of natural language processing. Author identification, is an effort to show the writer's personal characteristics, based on a piece of linguistic information [88] such that various manuscripts written by various authors can be distinguished. Humans possess certain writing patterns for using a language in their writings, which act like figure prints of the writer (writer print); these patterns are specific to the writers [89].

Authors in [90] have proposed an approach known as the *stylometric* approach to deal with the problem of Author Identification. There are four different steps in this approach:

- Calculation of word frequencies to find the most frequent words in the entire corpus.
- Calculation of normalized frequency. This is done by dividing the frequency of the most frequent word in that document to the total number of words in entire corpus.
- Using Z-score method.
- Calculation of distance table by finding distance between two matrices.

Therefore, since the text is converted into numeric representation (feature extraction), classification, and clustering techniques of machine learning can be implemented on it. The Reuter_50_50 data set is used for experiments. There are 50 authors and 50 documents per each author in this dataset. Thus, both training corpus and test corpus contains 2500 texts. These corpuses do not overlap with each other. By applying *stylometry* approach and n-gram features to the author identification problem an accuracy of about 85% of that of SVM classifier is achieved which is a higher accuracy in comparison to Delta and KNN classifier.

Dissimilarity Counter Method (DCM), DCM-Voting, and DCM-Classifier have been applied in [91] to the problem of Author Identification. Once the representation spaces are selected, similarity measures such as Euclidean distance, correlation coefficient, and Cosine can be used to compare the documents and then, the document author can be identified using one of the above-mentioned approaches (DCM, DCM-voting, or DCM-Classifier). DCM only uses

Table 5 Comparison of results of methods with VSA

Methods	Dataset1					Dataset2					Dataset3				
	Modes	Worst	Best	Mean	SD	Modes	Worst	Best	Mean	SD	Modes	Worst	Best	Mean	SD
VSA	VSA	**1.0976**	1.1776	1.3368	0.0394	VSA	0.8055	1.1195	**1.0782**	0.0793	VSA	0.9830	**1.4678**	1.3695	0.0993
Proposed method (state1)	VSAC11	1.0393	1.3076	1.3048	0.0658	VSAC11	0.7894	1.0633	0.9814	**0.0548**	VSAC11	0.9766	1.4506	1.3541	0.0945
	VSAC21	1.0889	**1.4099**	1.2423	0.0711	VSAC21	0.7765	1.0660	1.0233	0.0676	VSAC21	1.0311	1.4668	1.3387	0.0728
	VSAC31	1.0593	1.2748	1.3172	0.0529	VSAC31	0.7732	1.0566	1.0201	0.0659	VSAC31	0.9761	1.4632	1.3333	0.0960
	VSAC41	1.0116	1.3581	1.2463	0.0844	VSAC41	0.7901	1.1190	1.0133	0.0771	VSAC41	0.9957	1.4124	1.3662	0.0765
	VSAC51	1.0452	1.3552	1.2530	0.0690	VSAC51	0.7989	1.0896	0.9964	0.0612	VSAC51	1.0332	1.4214	1.3989	0.0679
	VSAC61	1.0631	1.3285	1.2346	0.0499	VSAC61	0.7839	1.0976	0.9859	0.0698	VSAC61	1.0205	1.4478	1.3981	0.0808
	VSAC71	1.0792	1.3860	1.2341	0.0653	VSAC71	0.7733	1.1161	1.0379	0.0867	VSAC71	1.0017	1.4035	1.3894	0.0762
	VSAC81	1.0559	1.2231	1.3089	0.0451	VSAC81	0.7896	1.0840	1.0707	0.0757	VSAC81	0.9730	1.4676	1.3427	0.1000
	VSAC91	1.0044	1.2229	1.3067	0.0674	VSAC91	0.7710	1.0815	0.9993	0.0713	VSAC91	0.9725	1.4393	1.3897	0.0994
	VSAC101	1.0876	1.2742	1.2768	0.0286	VSAC101	0.8021	1.1128	1.0435	0.0732	VSAC101	1.0290	1.4433	1.3449	0.0667
Proposed method (state2)	VSAC12	1.0529	1.3255	1.3045	0.0638	VSAC12	0.8056	1.0884	0.9876	0.0571	VSAC12	0.9721	1.4164	1.3905	0.0936
	VSAC22	1.0143	1.2019	1.2335	0.0368	VSAC22	0.7981	1.0509	1.0391	0.0565	VSAC22	0.9730	1.4143	1.3357	0.0822
	VSAC32	1.0957	1.2719	1.2366	**0.0161**	VSAC32	0.7941	1.0908	0.9889	0.0631	VSAC32	1.0060	1.4569	1.3820	0.0873
	VSAC42	1.0308	1.1706	1.2720	0.0389	VSAC42	0.7932	1.0744	1.0606	0.0694	VSAC42	0.9840	1.4357	1.3943	0.0939
	VSAC52	1.0021	1.1868	1.3132	0.0677	VSAC52	0.7975	1.0971	1.0658	0.0744	VSAC52	1.0301	1.4388	1.3899	0.0723
	VSAC62	1.0717	1.3209	1.2547	0.0454	VSAC62	0.7822	**1.1272**	0.9832	0.0815	VSAC62	1.0105	1.4248	1.3607	0.0721
	VSAC72	1.0866	1.2734	1.3173	0.0400	VSAC72	0.8063	1.1066	1.0752	0.0747	VSAC72	0.9759	1.4203	1.3721	0.0891
	VSAC82	1.0315	1.3233	1.2861	0.0697	VSAC82	0.7939	1.0659	1.0045	0.0565	VSAC82	0.9845	1.4172	1.3819	0.0862
	VSAC92	1.0091	1.2543	1.2962	0.0666	VSAC92	0.7982	1.1023	0.9993	0.0663	VSAC92	0.9728	1.4507	1.3435	0.0947
	VSAC102	1.0389	1.2108	1.2746	0.0395	VSAC102	0.7977	1.1233	1.0319	0.0771	VSAC102	0.9941	1.4224	1.3632	0.0795
Proposed method (state3)	VSAC13	1.0100	1.2235	1.2112	0.0379	VSAC13	0.8058	1.0891	1.0382	0.0633	VSAC13	**1.0375**	1.4241	1.3660	**0.0602**
	VSAC23	1.0807	1.3786	1.2443	0.0618	VSAC23	0.7862	1.1007	0.9722	0.0691	VSAC23	0.9806	1.4586	1.3366	0.0928
	VSAC33	1.0246	1.2044	1.2127	0.0268	VSAC33	0.7987	1.0708	1.0361	0.0609	VSAC33	0.9746	1.4617	1.3871	0.1042
	VSAC43	1.0654	1.4037	1.3171	0.0835	VSAC43	0.8038	1.0880	1.0316	0.0629	VSAC43	0.9925	1.4351	1.3494	0.0817
	VSAC53	1.0581	1.3064	1.2760	0.0506	VSAC53	**0.8100**	1.0656	1.0628	0.0598	VSAC53	0.9870	1.4028	1.3497	0.0748
	VSAC63	1.0844	1.2271	**1.3428**	0.0457	VSAC63	0.7987	1.0646	1.0441	0.0608	VSAC63	0.9880	1.4462	1.3796	0.0921
	VSAC73	1.0181	1.2147	1.2440	0.0403	VSAC73	0.7880	1.1148	1.0455	0.0806	VSAC73	1.0125	1.4290	1.3646	0.0730
	VSAC83	1.0375	1.3182	1.2657	0.0618	VSAC83	0.7752	1.1201	0.9816	0.0817	VSAC83	0.9941	1.4543	1.4039	0.0961
	VSAC93	1.0261	1.2238	1.2300	0.0347	VSAC93	0.8083	1.0643	1.0528	0.0581	VSAC93	1.0280	1.4174	**1.4057**	0.0709
	VSAC103	1.0195	1.1869	1.2116	0.0254	VSAC103	0.7738	1.0790	1.0458	0.0767	VSAC103	0.9906	1.4521	1.3675	0.0906

Methods	Dataset4					Dataset5					Dataset6				
	Modes	Worst	Best	Mean	SD	Modes	Worst	Best	Mean	SD	Modes	Worst	Best	Mean	SD
VSA	VSA	1.0587	1.3144	1.2876	0.0548	VSA	1.0114	1.3799	1.3384	0.0748	VSA	0.5416	0.6599	0.8384	0.0420

Table 5 (continued)

Methods	Dataset4					Dataset5					Dataset6				
	Modes	Worst	Best	Mean	SD	Modes	Worst	Best	Mean	SD	Modes	Worst	Best	Mean	SD
Proposed Method (State1)	VSAC11	1.0880	1.3770	1.2946	0.0616	VSAC11	1.0138	1.3721	1.3292	0.0698	VSAC11	0.4663	0.6720	0.8419	0.0736
	VSAC21	1.0712	1.3161	1.3080	0.0536	VSAC21	1.0112	1.3921	1.3328	0.0773	VSAC21	0.5123	0.8086	0.8081	0.0595
	VSAC31	1.1361	1.3954	1.2624	0.0459	VSAC31	1.0169	1.4002	1.3189	0.0749	VSAC31	0.5026	0.7927	0.8480	0.0715
	VSAC41	1.0834	1.2983	1.2900	0.0394	VSAC41	1.0137	1.3826	1.3106	0.0697	VSAC41	0.4878	0.5965	0.8528	0.0730
	VSAC51	1.0942	1.3998	1.3078	0.0680	VSAC51	1.0137	1.4198	1.3033	0.0807	VSAC51	0.5451	0.6770	0.8684	0.0527
	VSAC61	1.0741	1.3500	1.2901	0.0585	VSAC61	1.0170	1.3781	**1.3388**	0.0718	VSAC61	0.5194	0.7853	0.8184	0.0538
	VSAC71	1.1019	1.3126	1.3021	0.0370	VSAC71	1.0142	1.3869	1.3070	0.0702	VSAC71	0.4786	0.7875	**0.8698**	0.0884
	VSAC81	1.0968	1.3949	1.3050	0.0648	VSAC81	1.0189	1.3773	1.3061	0.0649	VSAC81	**0.5494**	0.8144	0.8144	0.0449
	VSAC91	**1.1381**	1.3435	1.2693	0.0249	VSAC91	1.0120	1.4163	1.3212	0.0826	VSAC91	0.5451	0.7079	0.8144	**0.0307**
	VSAC101	1.1095	1.3773	**1.3092**	0.0537	VSAC101	1.0112	1.4444	1.3252	0.0927	VSAC101	0.5253	0.6853	0.8626	0.0578
Proposed Method (State2)	VSAC12	1.1262	1.3242	1.2510	0.0218	VSAC12	1.0135	1.3774	1.3370	0.0728	VSAC12	0.5178	**0.8738**	0.8303	0.0786
	VSAC22	1.0578	1.3752	1.3054	0.0762	VSAC22	1.0194	1.4010	1.3332	0.0762	VSAC22	0.4797	0.8377	0.7928	0.0792
	VSAC32	1.0881	1.3501	1.2761	0.0503	VSAC32	1.0106	1.3900	1.3131	0.0738	VSAC32	0.5330	0.7060	0.8127	0.0353
	VSAC42	1.0537	1.2853	1.2986	0.0524	VSAC42	**1.0194**	1.4036	1.3164	0.0744	VSAC42	0.5134	0.6942	0.8297	0.0496
	VSAC52	1.1042	**1.3998**	1.2805	0.0614	VSAC52	1.0175	1.3704	1.3133	**0.0647**	VSAC52	0.4875	0.7062	0.8177	0.0571
	VSAC62	1.1158	1.3116	1.2496	0.0217	VSAC62	1.0162	1.4367	1.3220	0.0875	VSAC62	0.4753	0.8537	0.8308	0.0932
	VSAC72	1.1114	1.3633	1.2641	0.0436	VSAC72	1.0159	1.3933	1.3083	0.0717	VSAC72	0.5008	0.8408	0.8633	0.0858
	VSAC82	1.0555	1.2932	1.2426	0.0422	VSAC82	1.0183	1.4367	1.3218	0.0865	VSAC82	0.4679	0.7445	0.7921	0.0630
	VSAC92	1.1109	1.3618	1.2823	0.0447	VSAC92	1.0106	1.4437	1.3042	0.0905	VSAC92	0.4674	0.6364	0.7936	0.0532
	VSAC102	1.0776	1.3493	1.2812	0.0554	VSAC102	1.0187	1.3993	1.3371	0.0767	VSAC102	0.4943	0.6033	0.8131	0.0523
Proposed Method (State3)	VSAC13	1.0813	1.2764	1.2555	0.0274	VSAC13	1.0124	1.4059	1.3146	0.0781	VSAC13	0.4681	0.6582	0.8086	0.0593
	VSAC23	1.1154	1.2914	1.2469	**0.0147**	VSAC23	1.0136	1.3882	1.3077	0.0710	VSAC23	0.4696	0.6789	0.8111	0.0606
	VSAC33	1.0596	1.2843	1.2631	0.0413	VSAC33	1.0169	1.4456	1.3272	0.0907	VSAC33	0.5189	0.5296	0.8083	0.0540
	VSAC43	1.0727	1.2788	1.2980	0.0420	VSAC43	1.0188	**1.4463**	1.3186	0.0891	VSAC43	0.4603	0.6987	0.8460	0.0789
	VSAC53	1.0895	1.3832	1.2876	0.0623	VSAC53	1.0104	1.4034	1.3054	0.0770	VSAC53	0.4699	0.7904	0.7909	0.0712
	VSAC63	1.1028	1.3547	1.2910	0.0469	VSAC63	1.0118	1.3891	1.3186	0.0738	VSAC63	0.5173	0.8254	0.7994	0.0596
	VSAC73	1.0776	1.3072	1.2938	0.0452	VSAC73	1.0185	1.4352	1.3083	0.0844	VSAC73	0.4795	0.8182	0.7962	0.0747
	VSAC83	1.0892	1.3187	1.2608	0.0375	VSAC83	1.0161	1.4071	1.3166	0.0771	VSAC83	0.5205	0.7337	0.8562	0.0587
	VSAC93	1.0779	1.2916	1.2883	0.0400	VSAC93	1.0154	1.4041	1.3047	0.1049	VSAC93	0.5359	0.6846	0.8695	0.0564
	VSAC103	1.1324	1.3713	1.2418	0.0376	VSAC103	1.0126	1.4016	1.3223	0.1078	VSAC103	0.4834	0.5521	0.7961	0.0542

Bold values is to show the best-obtained value in the comparisons

Table 6 Comparison of results of methods with VSA (continuance)

Methods	Dataset7					Dataset8					Dataset9				
	Modes	Worst	Best	Mean	SD	Modes	Worst	Best	Mean	SD	Modes	Worst	Best	Mean	SD
VSA	VSA	1.0327	1.7105	1.5810	0.0938	VSA	1.0813	1.3577	1.2658	0.0549	VSA	1.0725	1.3744	1.2617	0.0446
Proposed Method (State1)	VSAC11	1.0622	1.7039	1.5924	0.0799	VSAC11	1.0842	1.3458	1.2703	0.0499	VSAC11	1.0763	1.3764	1.2678	0.0441
	VSAC21	1.1233	1.7201	1.5845	**0.0554**	VSAC21	1.0974	1.3345	1.2662	0.0397	VSAC21	1.0600	1.3357	1.2679	0.0373
	VSAC31	1.0912	1.7133	1.5996	0.0704	VSAC31	1.1245	1.3479	1.2598	0.0319	VSAC31	1.0716	1.3087	1.3014	0.0301
	VSAC41	1.0180	**1.7291**	1.6070	0.1105	VSAC41	1.1043	1.3529	1.2628	0.0428	VSAC41	1.0726	1.3331	1.3121	0.0382
	VSAC51	1.0277	1.7234	1.6277	0.1079	VSAC51	1.1059	1.3438	1.2547	0.0381	VSAC51	1.0629	1.3375	**1.3166**	0.0448
	VSAC61	1.0476	1.7064	1.6000	0.0888	VSAC61	1.1266	1.3489	1.2777	0.0327	VSAC61	1.0785	1.3285	1.3007	0.0319
	VSAC71	1.1099	1.7213	1.5909	0.0629	VSAC71	1.1106	1.3372	1.2722	0.0353	VSAC71	1.0553	1.3474	1.2740	0.0441
	VSAC81	1.0484	1.7283	1.5891	0.0932	VSAC81	1.0897	1.3369	1.2594	0.0432	VSAC81	**1.0789**	1.3591	1.2761	0.0375
	VSAC91	1.1205	1.7090	1.6188	0.0588	VSAC91	1.0907	1.3496	1.2670	0.0480	VSAC91	1.0720	1.3313	1.2798	0.0321
	VSAC101	1.0627	1.7109	1.5934	0.0820	VSAC101	1.1240	1.3310	1.2703	0.0269	VSAC101	1.0709	1.3294	1.3060	0.0367
Proposed Method (State2)	VSAC12	1.0749	1.7106	1.5961	0.0766	VSAC12	1.1094	1.3493	**1.2786**	0.0407	VSAC12	1.0601	1.3193	1.2913	0.0361
	VSAC22	1.0502	1.7035	1.5814	0.0836	VSAC22	1.0901	1.3542	1.2707	0.0502	VSAC22	1.0612	1.3613	1.3001	0.0495
	VSAC32	1.0779	1.7251	1.6094	0.0818	VSAC32	1.1174	1.3573	1.2672	0.0389	VSAC32	1.0586	1.3569	1.2616	0.0444
	VSAC42	1.1263	1.7100	1.6274	0.0579	VSAC42	1.1006	1.3506	1.2615	0.0434	VSAC42	1.0717	**1.3782**	1.2670	0.0467
	VSAC52	1.0769	1.7185	1.6142	0.0811	VSAC52	**1.1282**	1.3312	1.2727	**0.0254**	VSAC52	1.0501	1.3567	1.2594	0.0479
	VSAC62	1.0740	1.7092	1.6143	0.0798	VSAC62	1.0959	1.3408	1.2547	0.0414	VSAC62	1.0710	1.3517	1.2908	0.0406
	VSAC72	**1.1288**	1.7209	1.6054	0.0563	VSAC72	1.0887	1.3538	1.2535	0.0493	VSAC72	1.0650	1.3280	1.2680	0.0326
	VSAC82	1.0588	1.7105	1.6272	0.0896	VSAC82	1.0950	1.3500	1.2522	0.0450	VSAC82	1.0760	1.3314	1.3151	0.0367
	VSAC92	1.0938	1.7192	1.6253	0.0754	VSAC92	1.1124	1.3497	1.2631	0.0380	VSAC92	1.0531	1.3195	1.2529	0.0332
	VSAC102	1.0309	1.7026	1.6180	0.0987	VSAC102	1.1055	1.3332	1.2709	0.0360	VSAC102	1.0729	1.3093	1.2719	**0.0238**
Proposed Method (State3)	VSAC13	1.0201	1.7183	1.5909	0.1036	VSAC13	1.1158	1.3449	1.2569	0.0344	VSAC13	1.0780	1.3084	1.3018	0.0271
	VSAC23	1.0979	1.7097	1.5866	0.0642	VSAC23	1.1167	1.3431	1.2648	0.0339	VSAC23	1.0690	1.3738	1.2685	0.0464
	VSAC33	1.0945	1.7074	1.5988	0.0670	VSAC33	1.0918	1.3363	1.2616	0.0423	VSAC33	1.0503	1.3379	1.2865	0.0452
	VSAC43	1.1189	1.7240	1.6134	0.0631	VSAC43	1.1008	1.3495	1.2594	0.0428	VSAC43	1.0736	1.3764	1.2590	0.0446
	VSAC53	1.0429	1.7097	1.6164	0.0948	VSAC53	1.1031	1.3534	1.2775	0.0448	VSAC53	1.0714	1.3163	1.2974	0.0313
	VSAC63	1.0439	1.7063	**1.6284**	0.0956	VSAC63	1.0805	1.3353	1.2586	0.0467	VSAC63	1.0649	1.3548	1.2807	0.0430
	VSAC73	1.0555	1.7082	1.5865	0.0834	VSAC73	1.0915	**1.3577**	1.2710	0.0508	VSAC73	1.0611	1.3612	1.3028	0.0499
	VSAC83	1.0849	1.7247	1.6176	0.0798	VSAC83	1.1007	1.3575	1.2730	0.0469	VSAC83	1.0682	1.3586	1.3106	0.0471
	VSAC93	1.0543	1.7269	1.5807	0.0888	VSAC93	1.0905	1.3507	1.2705	0.0488	VSAC93	1.0596	1.3686	1.3034	0.0530
	VSAC103	1.0679	1.7055	1.6236	0.0832	VSAC103	1.1026	1.3343	1.2715	0.0379	VSAC103	1.0761	1.3645	1.2945	0.0428

Methods	Dataset4					Dataset11					Dataset12				
	Modes	Worst	Best	Mean	SD	Modes	Worst	Best	Mean	SD	Modes	Worst	Best	Mean	SD
VSA	VSA	1.1862	1.5481	1.3965	0.0884	VSA	1.2308	1.6919	1.6400	0.0762	VSA	1.2996	1.7581	1.6363	0.0839

Table 6 (continued)

Methods	Dataset4					Dataset11					Dataset12				
	Modes	Worst	Best	Mean	SD	Modes	Worst	Best	Mean	SD	Modes	Worst	Best	Mean	SD
Proposed Method (State1)	VSAC11	**1.1992**	1.5405	1.4010	0.0801	VSAC11	1.2294	1.6963	1.5527	0.0653	VSAC11	1.2860	1.7431	1.6426	0.0861
	VSAC21	1.1862	**1.5494**	1.3920	0.0887	VSAC21	1.1812	**1.6990**	1.5806	0.0915	VSAC21	1.2751	1.7532	1.5976	0.0891
	VSAC31	1.1948	1.5347	1.4500	0.0845	VSAC31	1.2143	1.6811	1.5910	0.0722	VSAC31	1.2730	1.7468	1.6656	0.0969
	VSAC41	1.1829	1.5137	1.4000	0.0773	VSAC41	1.1760	1.6916	1.5470	0.0872	VSAC41	1.2848	1.7570	1.6617	0.0939
	VSAC51	1.1891	1.5379	1.4067	0.0838	VSAC51	1.2027	1.6881	1.6216	0.0848	VSAC51	1.2748	1.7562	1.6665	0.0990
	VSAC61	1.1908	1.5153	1.4068	**0.0749**	VSAC61	1.2408	1.6897	1.6102	0.0656	VSAC61	1.2830	1.7360	1.5938	0.0792
	VSAC71	1.1969	1.5441	1.4278	0.0843	VSAC71	**1.2456**	1.6821	1.5549	**0.0533**	VSAC71	1.2768	1.7524	1.6126	0.0896
	VSAC81	1.1949	1.5376	1.4331	0.0834	VSAC81	1.1655	1.6963	1.5902	0.0993	VSAC81	1.2906	1.7450	1.6614	0.0875
	VSAC91	1.1922	1.5333	1.4525	0.0855	VSAC91	1.1947	1.6981	1.5973	0.0875	VSAC91	1.2968	1.7443	1.6201	0.0786
	VSAC101	1.1807	1.5416	1.4281	0.0907	VSAC101	1.1607	1.6989	1.6173	0.1068	VSAC101	1.2792	**1.7594**	1.6602	0.0970
Proposed Method (State2)	VSAC12	1.1896	1.5193	1.4497	0.0819	VSAC12	1.2071	1.6990	1.5745	0.0788	VSAC12	1.2777	1.7337	1.6396	0.0865
	VSAC22	1.1976	1.5403	1.4394	0.0838	VSAC22	1.1616	1.6863	1.5449	0.0917	VSAC22	1.2881	1.7462	1.5954	0.0806
	VSAC32	1.1817	1.5146	1.4334	0.0817	VSAC32	1.1985	1.6905	1.6434	0.0917	VSAC32	1.2896	1.7306	1.6276	0.0784
	VSAC42	1.1909	1.5381	1.3953	0.0825	VSAC42	1.2291	1.6815	1.6307	0.0723	VSAC42	1.2984	1.7429	1.6667	0.0841
	VSAC52	1.1819	1.5424	1.4465	0.0924	VSAC52	1.2395	1.6918	1.5861	0.0632	VSAC52	1.2888	1.7338	1.6233	0.0792
	VSAC62	1.1942	1.5210	1.4141	0.0761	VSAC62	1.2383	1.6826	1.5341	0.0547	VSAC62	1.2890	1.7528	1.6634	0.0909
	VSAC72	1.1844	1.5117	1.4324	0.0794	VSAC72	1.1832	1.6858	1.6093	0.0911	VSAC72	1.2915	1.7565	**1.6725**	0.0923
	VSAC82	1.1989	1.5139	1.4465	0.0754	VSAC82	1.2376	1.6812	1.6132	0.0651	VSAC82	1.2726	1.7555	1.6607	0.0989
	VSAC92	1.1957	1.5220	**1.4578**	0.0811	VSAC92	1.2332	1.6904	1.6416	0.0750	VSAC92	1.2802	1.7582	1.6326	0.0923
	VSAC102	1.1845	1.5175	1.4039	0.0782	VSAC102	1.2143	1.6913	1.6421	0.0842	VSAC102	1.2872	1.7540	1.6200	0.0862
Proposed Method (State3)	VSAC13	1.1861	1.5153	1.4381	0.0806	VSAC13	1.2198	1.6898	1.5747	0.0700	VSAC13	1.2812	1.7340	1.6387	0.0849
	VSAC23	1.1818	1.5145	1.4077	0.0787	VSAC23	1.2087	1.6895	1.5412	0.0710	VSAC23	1.2820	1.7369	1.6606	0.0889
	VSAC33	1.1878	1.5408	1.4187	0.0864	VSAC33	1.2407	1.6906	1.6048	0.0650	VSAC33	1.2706	1.7320	1.6602	0.0927
	VSAC43	1.1891	1.5249	1.4392	0.0825	VSAC43	1.2347	1.6808	1.6047	0.0649	VSAC43	1.2771	1.7576	1.5999	0.0900
	VSAC53	1.1849	1.5402	1.4029	0.0863	VSAC53	1.1923	1.6829	1.6446	0.0928	VSAC53	1.2784	1.7579	1.6529	0.0958
	VSAC63	1.1825	1.5323	1.3975	0.0841	VSAC63	1.1791	1.6939	1.5991	0.0937	VSAC63	1.2736	1.7307	1.6575	0.0905
	VSAC73	1.1821	1.5187	1.4187	0.0811	VSAC73	1.2165	1.6856	1.5374	0.0658	VSAC73	**1.2997**	1.7443	1.5993	**0.0752**
	VSAC83	1.1829	1.5214	1.4029	0.0802	VSAC83	1.2224	1.6829	**1.6467**	0.0791	VSAC83	1.2829	1.7364	1.6692	0.0899
	VSAC93	1.1863	1.5402	1.4211	0.0870	VSAC93	1.1692	1.6938	1.6277	0.1033	VSAC93	1.2709	1.7507	1.6201	0.0926
	VSAC103	1.1918	1.5205	1.4030	0.0760	VSAC103	1.1848	1.6969	1.5420	0.0844	VSAC103	1.2840	1.7564	1.6505	0.0924

Bold values is to show the best-obtained value in the comparisons

Table 7 Comparison of results of methods with VSA (continuance)

Methods	Dataset13					Dataset14					Dataset15				
	Modes	Worst	Best	Mean	SD	Modes	Worst	Best	Mean	SD	Modes	Worst	Best	Mean	SD
VSA	VSA	1.1441	1.6251	1.4886	0.0924	VSA	1.1812	1.5273	1.5207	0.0716	VSA	1.1460	1.7857	1.6785	0.0997
Proposed method (state1)	VSAC11	1.0892	1.5685	1.4332	0.0918	VSAC11	1.1704	1.5441	1.5232	0.0814	VSAC11	1.1630	1.7513	1.6713	0.0805
	VSAC21	1.1195	1.5472	1.4824	0.0782	VSAC21	1.1874	1.5341	1.4535	0.0582	VSAC21	1.1570	1.7418	1.6000	0.0690
	VSAC31	1.1091	1.5705	1.5483	0.1025	VSAC31	1.1790	1.5162	**1.5298**	0.0722	VSAC31	1.1493	1.7299	1.6540	0.0777
	VSAC41	1.0870	1.5579	1.5286	0.1054	VSAC41	1.1727	1.5294	1.5127	0.0743	VSAC41	1.1584	1.7621	1.6856	0.0884
	VSAC51	1.1146	1.5887	1.5776	0.1110	VSAC51	1.1763	1.5168	1.4998	0.0666	VSAC51	1.1406	1.7398	1.5210	0.0676
	VSAC61	1.1436	1.6068	1.5190	0.0909	VSAC61	1.1893	1.5752	1.4953	0.0763	VSAC61	1.1758	1.7855	1.5634	0.0720
	VSAC71	1.1567	1.5226	1.5409	0.0669	VSAC71	**1.1895**	1.5488	1.4515	0.0617	VSAC71	1.1680	**1.7983**	1.5230	0.0780
	VSAC81	1.1112	1.5242	1.5405	0.0887	VSAC81	1.1812	1.5464	1.4671	0.0668	VSAC81	1.1648	1.7897	1.5283	0.0762
	VSAC91	1.1511	1.5526	1.5002	0.0682	VSAC91	1.1747	1.5269	1.4733	0.0650	VSAC91	1.1611	1.7312	1.6098	0.0652
	VSAC101	1.0727	1.5530	1.5720	0.1210	VSAC101	1.1700	1.5292	1.4665	0.0667	VSAC101	1.1711	1.7363	1.5221	**0.0530**
Proposed method (state2)	VSAC12	1.1623	1.5626	1.5780	0.0825	VSAC12	1.1849	1.5092	1.5295	0.0679	VSAC12	1.1672	1.7850	1.5450	0.0743
	VSAC22	1.0926	1.6297	1.4349	0.1120	VSAC22	1.1723	1.5311	1.4913	0.0706	VSAC22	1.1709	1.7863	1.6375	0.0821
	VSAC32	1.0783	1.5767	1.5770	0.1250	VSAC32	1.1890	1.5308	1.4701	0.0589	VSAC32	1.1791	1.7460	1.5282	0.0535
	VSAC42	1.1287	1.5378	1.5334	0.0818	VSAC42	1.1834	1.5098	1.5199	0.0663	VSAC42	**1.1794**	1.7680	1.6707	0.0776
	VSAC52	1.1125	1.6148	1.4404	0.0982	VSAC52	1.1754	1.5187	1.5000	0.0676	VSAC52	1.1486	1.7252	1.6701	0.0798
	VSAC62	1.1348	1.5408	1.5033	0.0732	VSAC62	1.1800	1.5200	1.4650	0.0590	VSAC62	1.1793	1.7619	1.6899	0.0794
	VSAC72	1.0734	1.6152	1.4171	0.1139	VSAC72	1.1791	1.5708	1.5295	0.0857	VSAC72	1.1439	1.7678	1.6385	0.0889
	VSAC82	1.1081	1.5829	1.4854	0.0947	VSAC82	1.1855	1.5637	1.5266	0.0802	VSAC82	1.1721	1.7848	1.6272	0.0798
	VSAC92	1.0951	1.6040	1.5084	0.1108	VSAC92	1.1875	1.5707	1.4886	0.0747	VSAC92	1.1556	1.7730	1.6695	0.0900
	VSAC102	1.1343	1.5986	1.5142	0.0919	VSAC102	1.1803	1.5127	1.5118	0.0665	VSAC102	1.1591	1.7647	1.5690	0.0723
Proposed method (state3)	VSAC13	**1.1636**	1.5286	1.5780	0.0748	VSAC13	1.1725	1.5495	1.4963	0.0766	VSAC13	1.1517	1.7890	1.5272	0.0815
	VSAC23	1.1256	**1.6364**	1.4539	0.1013	VSAC23	1.1772	1.5437	1.5003	0.0735	VSAC23	1.1659	1.7216	1.6352	0.0642
	VSAC33	1.0866	1.5536	1.4532	0.0907	VSAC33	1.1810	1.5741	1.4962	0.0799	VSAC33	1.1496	1.7648	1.5559	0.0754
	VSAC43	1.1054	1.5949	1.4492	0.0952	VSAC43	1.1873	**1.5780**	1.4658	0.0743	VSAC43	1.1598	1.7357	1.5758	0.0627
	VSAC53	1.1309	1.5694	1.4590	0.0762	VSAC53	1.1872	1.5031	1.4708	**0.0519**	VSAC53	1.1494	1.7346	1.6714	0.0822
	VSAC63	1.1253	1.6004	1.5405	0.1013	VSAC63	1.1703	1.5673	1.4430	0.0758	VSAC63	1.1655	1.7815	1.5669	0.0753
	VSAC73	1.1628	1.5362	**1.5881**	0.0794	VSAC73	1.1706	1.5053	1.4890	0.0641	VSAC73	1.1430	1.7487	1.5400	0.0712
	VSAC83	1.1032	1.5374	1.5235	0.0915	VSAC83	1.1814	1.5611	1.4473	0.0691	VSAC83	1.1440	1.7794	**1.6899**	0.1008
	VSAC93	1.1575	1.5742	1.4197	**0.0620**	VSAC93	1.1749	1.5740	1.4582	0.0776	VSAC93	1.1579	1.7901	1.6727	0.0945
	VSAC103	1.1055	1.6348	1.4488	0.1092	VSAC103	1.1788	1.5262	1.4693	0.0621	VSAC103	1.1592	1.7674	1.6270	0.0800

Methods	Dataset4					Dataset17					Dataset18				
	Modes	Worst	Best	Mean	SD	Modes	Worst	Best	Mean	SD	Modes	Worst	Best	Mean	SD
VSA	VSA	0.7088	1.8566	1.6167	0.1044	VSA	1.0689	1.7354	1.6695	0.0899	VSA	1.1120	1.6863	1.6872	0.1109

Table 7 (continued)

Methods	Dataset4					Dataset17					Dataset18				
	Modes	Worst	Best	Mean	SD	Modes	Worst	Best	Mean	SD	Modes	Worst	Best	Mean	SD
Proposed method (state1)	VSAC11	0.6902	1.8539	1.5967	0.1091	VSAC11	1.0864	1.7759	1.6687	0.0930	VSAC11	1.1723	1.7033	1.6068	0.0710
	VSAC21	0.8725	1.8518	1.5915	0.0242	VSAC21	1.0813	1.7860	1.6392	0.0936	VSAC21	1.1095	**1.7584**	1.5863	0.1145
	VSAC31	0.8720	1.8505	1.6129	0.0267	VSAC31	1.1189	1.7389	1.6394	0.0619	VSAC31	1.1338	1.7174	1.6306	0.0971
	VSAC41	0.8873	1.8525	1.5597	**0.0141**	VSAC41	1.0986	**1.8086**	1.5121	0.0812	VSAC41	1.1430	1.7571	1.5692	0.0969
	VSAC51	0.7766	1.8590	1.5251	0.0626	VSAC51	1.0884	1.7982	1.6546	0.0964	VSAC51	1.1540	1.6816	1.5372	0.0626
	VSAC61	0.8194	1.8598	1.5882	0.0506	VSAC61	1.0562	1.7333	1.5936	0.0819	VSAC61	1.1528	1.6619	1.5289	0.0556
	VSAC71	0.8537	1.8548	1.5435	0.0283	VSAC71	1.0740	1.7528	1.6620	0.0909	VSAC71	1.1517	1.6910	1.5192	0.0649
	VSAC81	0.8656	1.8514	1.5442	0.0219	VSAC81	1.0919	1.7855	1.5151	0.0754	VSAC81	1.1644	1.6820	1.5290	0.0571
	VSAC91	0.8687	1.8551	1.5206	0.0196	VSAC91	1.0547	1.7438	1.5134	0.0764	VSAC91	1.0928	1.7002	**1.6988**	0.1260
	VSAC101	0.7156	**1.8599**	1.6146	0.1019	VSAC101	1.0918	1.7454	1.6640	0.0808	VSAC101	1.1239	1.7315	1.6920	0.1176
Proposed method (state2)	VSAC12	0.7711	1.8576	1.6005	0.0736	VSAC12	1.1128	1.7373	1.5244	**0.0492**	VSAC12	1.1395	1.6854	1.5345	0.0702
	VSAC22	0.7731	1.8597	1.5485	0.0669	VSAC22	1.0669	1.7586	1.6877	0.1007	VSAC22	1.1702	1.6906	1.6156	0.0697
	VSAC32	0.7652	1.8593	1.5492	0.0704	VSAC32	1.0431	1.7597	1.5528	0.0912	VSAC32	1.1796	1.7157	1.6159	0.0728
	VSAC42	0.8710	1.8524	1.5246	0.0179	VSAC42	1.1024	1.8005	1.5231	0.0770	VSAC42	1.1012	1.7181	1.6372	0.1138
	VSAC52	0.7723	1.8510	1.5716	0.0671	VSAC52	1.1161	1.7365	1.6164	0.0586	VSAC52	1.1537	1.7449	1.5215	0.0837
	VSAC62	0.7172	1.8541	1.5670	0.0927	VSAC62	1.1157	1.7601	1.6432	0.0703	VSAC62	1.1446	1.7060	1.5646	0.0784
	VSAC72	0.8708	1.8513	1.5962	0.0254	VSAC72	**1.1381**	1.7894	1.6688	0.0729	VSAC72	1.1835	1.7404	1.5216	0.0691
	VSAC82	0.7680	1.8575	1.5988	0.0748	VSAC82	1.0417	1.7339	1.5291	0.0803	VSAC82	1.1049	1.6851	1.5991	0.0956
	VSAC92	0.7155	1.8535	1.5680	0.0934	VSAC92	1.0597	1.8063	1.5647	0.1010	VSAC92	1.1621	1.6884	1.6959	0.0899
	VSAC102	0.8549	1.8545	1.5769	0.0313	VSAC102	1.0837	1.8083	1.5419	0.0893	VSAC102	1.0988	1.6896	1.5441	0.0913
Proposed method (state3)	VSAC13	0.7810	1.8520	1.5818	0.0648	VSAC13	1.1011	1.8030	1.6219	0.0875	VSAC13	1.1840	1.6864	1.6454	0.0678
	VSAC23	0.8029	1.8574	1.5613	0.0541	VSAC23	1.0747	1.7686	1.6550	0.0939	VSAC23	1.1284	1.6943	1.5352	0.0783
	VSAC33	0.7254	1.8520	1.6354	0.0981	VSAC33	1.0406	1.7397	**1.6881**	0.1081	VSAC33	1.1336	1.6709	1.5366	0.0683
	VSAC43	**0.8986**	1.8577	1.6172	0.0174	VSAC43	1.0686	1.7778	1.6737	0.1027	VSAC43	1.1232	1.7109	1.5954	0.0942
	VSAC53	0.7473	1.8524	1.5286	0.0739	VSAC53	1.1298	1.7651	1.6274	0.0629	VSAC53	1.1565	1.7258	1.6131	0.0862
	VSAC63	0.7880	1.8561	1.5700	0.0615	VSAC63	1.0953	1.7621	1.6328	0.0788	VSAC63	**1.1862**	1.6637	1.5385	**0.0422**
	VSAC73	0.8155	1.8535	**1.6382**	0.0573	VSAC73	1.0707	1.8040	1.5938	0.0983	VSAC73	1.1614	1.7475	1.6339	0.0938
	VSAC83	0.7385	1.8572	1.6048	0.0891	VSAC83	1.1067	1.7787	1.6165	0.0763	VSAC83	1.0969	1.7391	1.6547	0.1250
	VSAC93	0.8227	1.8512	1.6346	0.0527	VSAC93	1.0478	1.7816	1.5119	0.0930	VSAC93	1.1100	1.7445	1.6903	0.1272
	VSAC103	0.8314	1.8518	1.5446	0.0374	VSAC103	1.0768	1.7904	1.6597	0.1002	VSAC103	1.1569	1.6792	1.6744	0.0851

Bold values is to show the best-obtained value in the comparisons

Table 8 Comparison of Results of methods with VSA (continuance)

Methods	Dataset19					Dataset20					Dataset21				
	Modes	Worst	Best	Mean	SD	Modes	Worst	Best	Mean	SD	Modes	Worst	Best	Mean	SD
VSA	VSA	0.9269	1.4273	1.3141	0.0743	VSA	0.9335	1.3484	1.3985	0.0784	VSA	1.0266	1.3506	1.2821	0.0594
Proposed method (state1)	VSAC11	0.9138	1.3929	1.3797	0.0828	VSAC11	0.9271	1.3953	1.3572	0.0823	VSAC11	1.0539	1.3512	1.2407	**0.0427**
	VSAC21	0.9192	1.4345	1.3057	0.0790	VSAC21	0.9327	1.3936	1.3533	0.0785	VSAC21	1.0210	1.3840	1.2459	0.0696
	VSAC31	0.9218	1.4036	1.3280	0.0716	VSAC31	0.9293	1.3813	1.3138	0.0691	VSAC31	1.0013	1.3876	1.2364	0.0789
	VSAC41	0.9331	1.4302	1.3624	0.0801	VSAC41	0.9343	1.3927	1.3533	0.0774	VSAC41	1.0586	1.3805	1.2804	0.0545
	VSAC51	0.9117	1.3751	1.3125	0.0653	VSAC51	0.9185	**1.4397**	1.3324	0.0947	VSAC51	1.0258	1.3886	1.2647	0.0706
	VSAC61	0.9198	1.4048	1.3046	0.0691	VSAC61	**0.9368**	1.3669	1.3234	0.0633	VSAC61	1.0251	1.3697	1.2523	0.0631
	VSAC71	0.9354	1.3796	1.3286	0.0585	VSAC71	0.8805	1.4118	1.3516	0.1075	VSAC71	1.0254	1.3722	1.2542	0.0639
	VSAC81	0.9280	1.3706	1.3393	0.0617	VSAC81	0.8644	1.3749	1.2917	0.0936	VSAC81	1.0009	1.3761	1.2723	0.0782
	VSAC91	0.9411	1.3743	1.3765	0.0647	VSAC91	0.9223	1.4006	1.3328	0.0813	VSAC91	1.0429	**1.3889**	1.2431	0.0618
	VSAC101	0.9398	1.4117	1.3012	0.0616	VSAC101	0.8920	1.3797	1.2962	0.0830	VSAC101	1.0517	1.3861	1.2671	0.0584
Proposed method (state2)	VSAC12	0.9206	1.4400	1.3166	0.0816	VSAC12	0.9051	1.3541	1.3658	0.0845	VSAC12	1.0380	1.3510	1.2309	0.0490
	VSAC22	0.9374	**1.4449**	1.3079	0.0744	VSAC22	0.9261	1.3785	1.2897	0.0657	VSAC22	**1.0620**	1.3730	1.2777	0.0501
	VSAC32	0.9331	1.4191	1.3284	0.0710	VSAC32	0.9244	1.4068	1.3796	0.0913	VSAC32	1.0449	1.3792	1.2432	0.0572
	VSAC42	0.9201	1.4053	1.3627	0.0794	VSAC42	0.9249	1.3893	1.3939	0.0900	VSAC42	1.0150	1.3561	1.2393	0.0615
	VSAC52	0.9359	1.3762	1.3241	0.0565	VSAC52	0.8656	1.3387	1.3980	0.1082	VSAC52	1.0486	1.3520	1.2623	0.0472
	VSAC62	0.9321	1.3753	1.3651	0.0665	VSAC62	0.9008	1.3425	1.3884	0.0898	VSAC62	1.0189	1.3714	1.2493	0.0662
	VSAC72	0.9486	1.3840	1.3346	0.0546	VSAC72	0.8763	1.3433	1.2977	0.0802	VSAC72	1.0248	1.3516	1.2422	0.0558
	VSAC82	0.9209	1.4353	1.3635	0.0875	VSAC82	0.9252	1.3333	1.2750	**0.0502**	VSAC82	1.0112	1.3700	1.2759	0.0719
	VSAC92	0.9251	1.4186	1.3434	0.0771	VSAC92	0.9218	1.3572	1.3824	0.0815	VSAC92	1.0336	1.3584	1.2449	0.0546
	VSAC102	0.9450	1.3770	1.3235	0.0523	VSAC102	0.8720	1.4383	1.3011	0.1112	VSAC102	1.0433	1.3690	1.2348	0.0537
Proposed method (state3)	VSAC13	0.9414	1.4073	1.3230	0.0627	VSAC13	0.9005	1.3308	**1.4063**	0.0928	VSAC13	1.0267	1.3762	1.2303	0.0633
	VSAC23	0.9377	1.4139	1.3537	0.0717	VSAC23	0.8611	1.3899	1.3941	0.1202	VSAC23	1.0277	1.3520	1.2370	0.0542
	VSAC33	0.9488	1.4272	1.3755	0.0744	VSAC33	0.9262	1.3827	1.3839	0.0855	VSAC33	1.0325	1.3843	1.2598	0.0657
	VSAC43	0.9419	1.4002	1.3880	0.0732	VSAC43	0.8640	1.4302	1.2804	0.1096	VSAC43	1.0189	1.3516	**1.2896**	0.0645
	VSAC53	0.9473	1.4355	1.3805	0.0783	VSAC53	0.9094	1.3536	1.3330	0.0747	VSAC53	1.0437	1.3756	1.2891	0.0606
	VSAC63	**0.9500**	1.3758	1.3061	**0.0465**	VSAC63	0.9042	1.3611	1.4056	0.0966	VSAC63	1.0615	1.3703	1.2842	0.0501
	VSAC73	0.9245	1.3929	1.3346	0.0684	VSAC73	0.9112	1.3800	1.2873	0.0727	VSAC73	1.0443	1.3731	1.2428	0.0552
	VSAC83	0.9179	1.3834	1.3330	0.0686	VSAC83	0.9284	1.3581	1.2770	0.0564	VSAC83	1.0132	1.3852	1.2568	0.0742
	VSAC93	0.9342	1.4103	**1.3894**	0.0797	VSAC93	0.8747	1.3644	1.3424	0.0959	VSAC93	1.0072	1.3574	1.2468	0.0662
	VSAC103	0.9395	1.3729	1.3707	0.0638	VSAC103	0.9255	1.3728	1.3494	0.0756	VSAC103	1.0457	1.3581	1.2763	0.0522

Methods	Dataset4					Dataset23					Dataset24				
	Modes	Worst	Best	Mean	SD	Modes	Worst	Best	Mean	SD	Modes	Worst	Best	Mean	SD
VSA	VSA	**1.1897**	1.5564	1.5208	0.0851	VSA	1.0158	1.3750	1.3256	0.0690	VSA	0.9870	1.4522	1.3572	0.0407

Table 8 (continued)

Methods	Dataset4					Dataset23					Dataset24				
	Modes	Worst	Best	Mean	SD	Modes	Worst	Best	Mean	SD	Modes	Worst	Best	Mean	SD
Proposed method (state1)	VSAC11	1.1876	1.5374	**1.5285**	0.0829	VSAC11	1.0145	1.3895	1.3149	0.0721	VSAC11	1.0012	1.4425	1.3945	0.0377
	VSAC21	1.1773	1.5394	1.4289	0.0715	VSAC21	1.0175	1.4069	1.3059	0.0750	VSAC21	1.0067	1.4529	1.3599	0.0322
	VSAC31	1.1897	1.5401	1.4896	0.0746	VSAC31	1.0174	1.3841	1.3352	0.0725	VSAC31	1.0213	1.4292	1.3603	0.0183
	VSAC41	1.1712	1.5071	1.4836	0.0731	VSAC41	1.0179	1.4491	1.3330	0.0922	VSAC41	1.0305	1.4663	1.3747	0.0276
	VSAC51	1.1700	1.5216	1.4667	0.0744	VSAC51	1.0142	1.3903	1.3020	0.0706	VSAC51	1.0346	1.4278	1.3573	0.0112
	VSAC61	1.1824	1.5080	1.5095	0.0738	VSAC61	1.0116	1.4418	1.3009	0.0890	VSAC61	0.9807	**1.4672**	1.3742	0.0509
	VSAC71	1.1721	1.5156	1.5276	0.0848	VSAC71	1.0115	1.4461	1.3299	0.0937	VSAC71	1.0164	1.4096	1.3419	0.0116
	VSAC81	1.1782	1.5442	1.4576	0.0762	VSAC81	1.0136	1.4351	1.3286	0.0890	VSAC81	0.9928	1.4460	1.3534	0.0355
	VSAC91	1.1733	1.5612	1.4795	0.0870	VSAC91	1.0101	1.4122	1.3065	0.0802	VSAC91	1.0327	1.4621	1.3757	0.0255
	VSAC101	1.1804	**1.5746**	1.5080	0.0923	VSAC101	1.0102	**1.4492**	1.3079	0.0930	VSAC101	0.9989	1.4285	1.3604	0.0285
Proposed method (state2)	VSAC12	1.1888	1.5573	1.5084	0.0834	VSAC12	1.0132	1.3813	1.3252	0.0720	VSAC12	1.0320	1.4259	1.3587	0.0121
	VSAC22	1.1808	1.5455	1.4537	0.0749	VSAC22	1.0121	1.4121	1.3255	0.0818	VSAC22	1.0225	1.4153	1.3390	**0.0100**
	VSAC32	1.1836	1.5436	1.4478	0.0722	VSAC32	1.0165	1.4380	1.3291	0.0886	VSAC32	1.0110	1.4184	1.3305	0.0151
	VSAC42	1.1718	1.5017	1.4702	0.0686	VSAC42	1.0122	1.4304	1.3296	0.0882	VSAC42	0.9899	1.4146	1.3517	0.0271
	VSAC52	1.1720	1.5058	1.4722	0.0700	VSAC52	1.0195	1.4455	1.3002	0.0868	VSAC52	0.9791	1.4246	1.3695	0.0383
	VSAC62	1.1770	1.5322	1.4853	0.0776	VSAC62	1.0165	1.4336	**1.3381**	0.0884	VSAC62	0.9829	1.4062	1.4061	0.0395
	VSAC72	1.1851	1.5333	1.4687	0.0713	VSAC72	1.0164	1.4128	1.3129	0.0783	VSAC72	1.0237	1.4276	1.4013	0.0245
	VSAC82	1.1853	1.5710	1.4496	0.0810	VSAC82	1.0187	1.4131	1.3199	0.0783	VSAC82	1.0158	1.4184	1.3413	0.0145
	VSAC92	1.1790	1.5481	1.4977	0.0834	VSAC92	1.0102	1.4219	1.3355	0.0873	VSAC92	0.9728	1.4201	1.3367	0.0342
	VSAC102	1.1821	1.5046	1.4437	**0.0599**	VSAC102	1.0158	1.4147	1.3215	0.0804	VSAC102	1.0189	1.4217	1.3563	0.0165
Proposed method (state3)	VSAC13	1.1749	1.5629	1.5217	0.0940	VSAC13	1.0159	1.4236	1.3113	0.0819	VSAC13	1.0292	1.4056	1.3730	0.0103
	VSAC23	1.1821	1.5664	1.4781	0.0843	VSAC23	1.0122	1.4042	1.3016	0.0760	VSAC23	0.9879	1.4021	1.3331	0.0212
	VSAC33	1.1847	1.5522	1.4861	0.0800	VSAC33	1.0157	1.3706	1.3188	**0.0666**	VSAC33	1.0202	1.4470	1.4036	0.0318
	VSAC43	1.1831	1.5304	1.4298	0.0659	VSAC43	1.0142	1.3926	1.3026	0.0714	VSAC43	1.0170	1.4634	1.3791	0.0336
	VSAC53	1.1713	1.5451	1.4829	0.0835	VSAC53	1.0135	1.4211	1.3004	0.0810	VSAC53	**1.0399**	1.4135	1.4087	0.0150
	VSAC63	1.1825	1.5467	1.4260	0.0715	VSAC63	**1.0197**	1.4042	1.3214	0.0752	VSAC63	1.0005	1.4222	1.3626	0.0263
	VSAC73	1.1803	1.5572	1.4855	0.0834	VSAC73	1.0160	1.4196	1.3254	0.0824	VSAC73	1.0089	1.4579	1.3645	0.0334
	VSAC83	1.1852	1.5076	1.5033	0.0710	VSAC83	1.0104	1.4202	1.3364	0.0868	VSAC83	1.0287	1.4363	**1.4090**	0.0261
	VSAC93	1.1794	1.5254	1.4340	0.0664	VSAC93	1.0140	1.4161	1.3062	0.0797	VSAC93	0.9908	1.4042	1.3628	0.0259
	VSAC103	1.1858	1.5717	1.4757	0.0840	VSAC103	1.0107	1.3905	1.3235	0.0755	VSAC103	1.0019	1.4603	1.3812	0.0401

Bold values is to show the best-obtained value in the comparisons

Table 9 Results of the VSA and the proposed method based on the FS

Datasets	VSA			Proposed method		
	Feature count-total	Accuracy (%)	Time(sec)	Feature count-select	Accuracy (%)	Time(sec)
Dataset1	36	44.2472	2.3244	25	**56.5962**	2.105
Dataset2	10	48.4139	1.5936	8	**67.549**	0.7746
Dataset3	24	**91.6334**	1.0279	11	87.8578	2.2247
Dataset4	15	63.7899	1.6667	8	**64.9167**	0.175
Dataset5	36	77.4904	1.7881	19	**92.073**	1.1341
Dataset6	8	60.3178	1.6666	6	**62.4584**	1.191
Dataset7	10	72.4666	1.941	6	**81.1022**	2.4077
Dataset8	16	71.3516	3.5829	8	**96.3686**	2.6418
Dataset9	21	87.4691	1.1463	9	**96.3448**	2.1726
Dataset10	16	82.3352	0.0883	7	**90.1066**	0.456
Dataset11	30	98.4868	2.5427	16	**98.7271**	0.3067
Dataset12	80	36.3746	0.1836	53	**71.7682**	0.9082
Dataset13	22	**54.9418**	2.5736	13	50.01	0.8355
Dataset14	22	35.7599	0.5076	17	**59.9471**	2.2448
Dataset15	20	**88.6538**	2.1595	8	80.2569	1.2598
Dataset16	57	36.5519	2.3174	13	**43.957**	0.5924
Dataset17	45	45.2476	3.066	23	**93.6619**	1.7498
Dataset18	17	**60.1896**	3.4182	6	58.0906	0.0865
Dataset19	13	69.6363	2.7172	7	**92.9925**	2.6428
Dataset20	10	82.1821	1.1213	6	**83.4601**	1.9857
Dataset21	2350	48.185	0.9042	1200	**90.7128**	1.4438
Dataset22	5340	**84.1727**	1.245	2341	83.7111	2.7264
Dataset23	3343	95.4847	3.8613	1845	**99.7029**	0.8341
Dataset24	2340	47.1271	0.6734	800	**50.26194**	1.3153

Bold values is to show the best-obtained value in the comparisons

the similarities between Victoria representations of documents in one space to solve a problem p of P. In the other two DCM-based approaches, it is possible to hybrid different representation spaces. In the case of DCM-voting approach, this is done using a voting technique and as for the DCM-classifier, it can be performed through a supervised learning method which requires the definition of predictive features. During evaluation of the challenge PAN-CLEF 2013, it is observed that DCM-classifier has the best performance only on the Greek corpus with 85%, and the two other approaches i.e. DCM-voting and DCM-classifier obtain the best results or equivalent to the winner of the competition for all evaluation measures (F1, precision and recall) on all the corpora.

The General Impostors Method (GENIM) which took part in the PAN'13 authorship identification competition has been evaluated in [92]. The basis of this model is the comparison made between the given documents and a number of external (impostor) documents, and since there are two stages in their method, the performance had to be measured and parameters needed to be optimized at each step. 25–33 percent of the training documents of each language were used for measuring and optimizing IM, whereas the rest were used for evaluation of GENIM. For the IM evaluation set, 3 or 4 documents were used as seed documents to retrieve the web impostor. The test accuracy is equal to 75.3%.

Blocks containing 140, 280, and 500 characters were investigated. The feature set contains conventional features like syntactic, lexical, application specific features, and some new features that are extracted from n-gram analysis. Moreover, the proposed approach has a mechanism for handling issues related to unbalanced dataset. It also uses Support Vector Machine (SVM) for data classification and uses Information Gain and Mutual Information as a FS strategy. The proposed approach was evaluated experimentally using the Enron email and Twitter corpuses. The results of this evaluation were very promising including an Equal Error Rate (EER) changing between 9.98% and 21.45%, for different block sizes [93].

In [94], by using a cluster-based classification approach, a model is presented for email authorship identification (EAI). Contributions of this paper are as follows: a) Developing a new model for email authorship identification. b) Evaluation

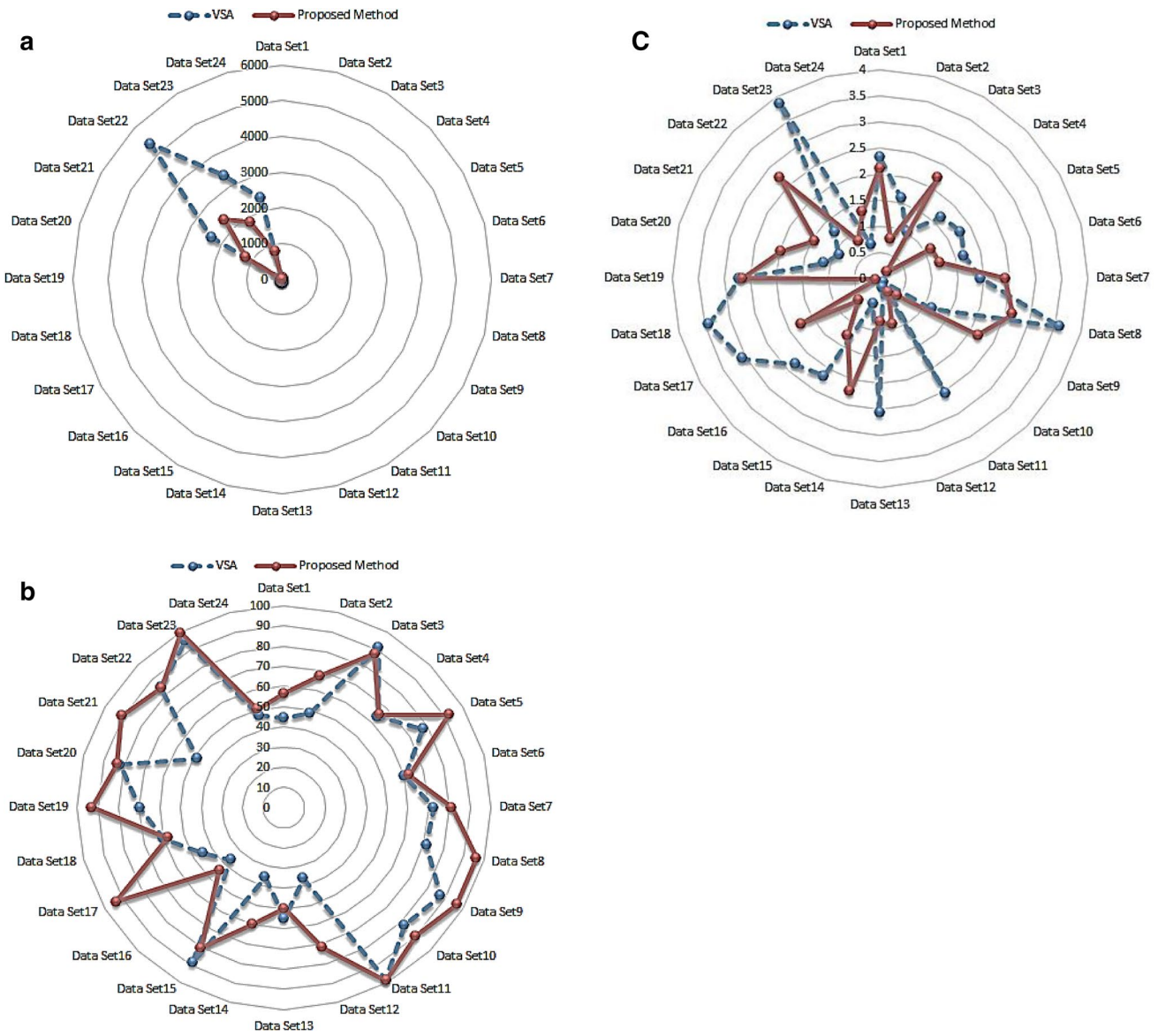

Fig. 2 **a** Feature Count-Total. **b** Accuracy (%). **c** Time (sec)

of using additional features together with basic stylometric features for email authorship identification as well as content features that are based on Info Gain FS. On the Enron dataset, the proposed model achieved accuracies of 94, 89, and 81 percent for 10, 25, and 50 authors, respectively. Whereas, on real email dataset constructed by authors, it attained an accuracy of 89.5%.

A large number of researches only focus on enhancing predictive accuracy and do not pay much attention to intrinsic value of the collected evidence. In this paper, a customized associative classification approach, which is a well-known data mining technique, is applied to the authorship attribution problem. This method models the features of writing style which are unique to a person. Then, it measures the associativity level of these features and generates an instinctive classifier. In this research, it is also concluded that a more accurate write print can also be provided by applying modifications on the rule pruning and ranking system described in the popular Classification by Multiple Association Rule (CMAR) algorithm. More convincing evidences can be provided for a court of law by eliminating patterns common amongst different authors since it leads to fairly unique and easy-to-understand write prints. Since this customized abandonment counter method is helpful in solving the problem of the e-mail authorship attribution, it can be used as a powerful tool against cybercrimes. The

Table 10 Control parameters of the algorithms

Algorithms	Description	Parameters	Value
GA [83]	Crossover rate	Pc	0.9
	Generation	Ng	200
	Mutation rate	Pm	0.01
PSO [84]	Acceleration coefficients	C1	1.5
	Acceleration coefficients	C2	1.5
	Random number	R1	0.5
	Random number	R2	0.5
	Inertia weight (linearly decreases)	(w)	0.6 to 0.3
ABC [9]	Employed bees	Ne	50
	Onlooker bees	No	50
	Scout bee	Ns	50
	Random number	ϕ	0.5
BOA [8]	Flight distance	$Step_e$	0.05
	Random number	Rand	0.02
	Linearly decreases	a	2–0
IWO [7]	Minimum number of seeds	s_{min}	0
	Maximum number of seeds	s_{max}	5
	Final value of standard deviation	σfinal	0.001
	Initial value of standard deviation	σinitial	3
FPA [14]	Step size scaling factor	γ	0.01
	Switch probability between Local and Global pollination	P	0.4
	Randomization	α	0.2
	Attractiveness of a firefly	β	1
FA [11]	Absorption coefficient	γ	1

effectiveness of the presented approach is verified by the results obtained through experiments [95].

An effort is made by authors in [96] to identify the author of articles written in Arabic. They introduced a new dataset which is composed of 12 features and 456 samples of 7 authors. Furthermore, to distinguish different authors from each other, powerful classification techniques were hybrids with the proposed dataset in their approach. The obtained results revealed that the proposed dataset was very successful and achieved a classification performance accuracy of 82% in the hold-out tests. They also conducted some experiments with two well-known classifiers namely the SVM and functional trees (FT) in order to show the efficiency of the proposed feature set. The reported an accuracy of 82% with the FT approach and holdout testing which confirmed robustness of the proposed feature set. Moreover, an accuracy of 100% has been achieved in one of the classes. They also conducted some test on FT by using tenfold cross validation and the proposed approach retained its accuracy to some extent.

One of the classifiers which have been extensively used for language processing is the Naive Bayes classifiers. Nevertheless, the event model used which can remarkably affect the classifier performance is not often mentioned. So far, Naive Bayes (NB) classifiers have never been used for authorship attribution in Arabic. Thus, they proposed to apply these classifiers to this problem, taking into consideration various event models such as simple NB, multinomial NB (MNB), multi-variant Bernoulli NB (MBNB), and Multi-variant Poisson NB (MPNB). The MBNB probability estimation is dependent on whether a feature exists or not, whereas MNB and MPNB a probability estimation is dependent on the frequency of the feature. The mean and standard deviation of the features form the basis of probability estimation in the NB model. The performances of these models are evaluated using a large Arabic dataset taken from books written by 10 different authors. Then, they are compared with other methods. The obtained results reveal that MBNB outperforms other techniques and is able to identify the author of a text with an accuracy of 97.43%. In addition, these results show that MNB and MBNB can be considered as a good choice for authorship attribution [97].

In [98], authorship identification methods were applied to messages of Arabic web forum. In this study, syntactic, lexical, structural, and content-specific writing style features were used to identify the authors. Some of the problematic characteristics of Arabic language were addressed in order to present a model with an acceptable degree of classification accuracy for authorship identification. SVM had a better performance than C4.5 and compared to English performance, the overall accuracy for Arabic was lower. These results were in consistence with previous researches. Finally, as future work, the authors proposed to analyze the differences between these two languages by evaluating the key features as determined by decision trees. Highlighting the linguistic differences between English and Arabic languages provides further insight into possible technique for enhancing the performance of authorship discrimination methodologies in an online, multilingual setting. The results showed accuracies of 85.43 and 81.03 for SVM and C4.5, respectively.

In [99], they developed an authorship visualization known as Write prints which can be used for identification of individuals based on their writing style. Unique writing style patterns are created through this visualization. These patterns can be distinguished in a similar way that fingerprint biometric systems work. Write prints provide an approach which is based on component analysis and utilizes a dynamic feature-based sliding window algorithm. This makes them very suitable for visualizing authorship across larger groups of messages. The performance of visualization across messages taken from three different Arabic and English forums was evaluated and compared with the performance of SVM. This comparison indicated that Write prints show

Table 11 Comparison of the proposed method with other algorithms based on the worst criterion

Datasets	PSO	ABC	BOA	IWO	GA	FA	FPA	VSA	Proposed Method
Dataset1	1.1652	1.3169	1.2795	1.2116	1.3078	1.0623	**1.3221**	1.151	1.3021
Dataset2	1.0134	0.8357	**1.0904**	0.7935	0.859	0.9388	0.9091	1.0718	1.0804
Dataset3	**1.1403**	0.9748	1.0289	1.0323	**1.0699**	1.0364	1.0189	1.0692	1.0699
Dataset4	1.2313	1.2398	1.0536	**1.3387**	1.0844	1.076	1.1111	1.1463	**1.3387**
Dataset5	1.1624	**1.1992**	1.0705	1.19	1.0613	1.1965	1.0762	**1.1983**	1.1083
Dataset6	0.5866	0.7323	0.7112	0.472	**0.7894**	0.6187	0.7379	0.502	0.7874
Dataset7	1.1565	1.2648	1.2797	**1.4996**	1.1722	1.4984	1.4836	1.0209	1.4984
Dataset8	1.2758	1.0955	1.0956	1.1194	1.2625	**1.1351**	1.1704	1.1696	1.2758
Dataset9	1.1248	1.0839	1.1775	1.1833	1.0772	**1.2775**	1.264	1.0373	1.2375
Dataset10	1.2866	1.1589	1.167	**1.3825**	1.2842	1.2472	1.2791	1.3497	1.2825
Dataset11	1.243	1.327	1.2921	1.3961	**1.6147**	1.3582	1.3734	1.2358	1.3961
Dataset12	1.4566	1.3633	1.4114	1.2155	1.2255	**1.5031**	1.2839	1.4008	1.2255
Dataset13	1.2436	1.1973	1.0512	1.2454	1.0363	1.2885	**1.4906**	1.3004	1.3204
Data set14	1.2653	1.1884	1.2394	1.3543	1.3115	1.2894	**1.3822**	1.3922	1.1644
Dataset15	1.1821	1.2728	1.3882	1.4679	**1.4981**	1.3427	**1.4981**	1.4827	1.3181
Dataset16	1.4262	1.0759	**1.6764**	1.2535	1.3716	1.5304	1.5123	0.8414	1.3264
Dataset17	**1.2562**	1.1611	1.1055	1.2166	1.031	1.0767	0.9641	1.369	1.0169
Dataset18	1.3983	**1.4124**	1.2653	1.303	1.0773	1.2703	1.0404	1.467	1.1167
Data set19	1.1328	0.9997	0.9467	1.0967	0.9658	0.9974	**1.198**	1.1236	1.1208
Dataset20	0.944	1.0662	**1.2361**	1.1814	0.9685	1.0653	1.1904	1.1242	1.0361
Dataset21	1.6487	1.4398	1.0724	1.2875	1.6143	1.631	1.1727	**1.7802**	1.1802
Dataset22	1.3458	1.6606	**1.7973**	1.1826	1.3905	1.0926	1.8201	1.1224	1.0201
Dataset23	1.154	1.0374	0.9335	**1.9864**	1.0774	0.8664	1.4506	1.2632	1.1206
Dataset24	1.0724	1.1301	1.1549	0.8827	0.7526	1.1974	**1.8704**	0.9771	1.1974

Bold values is to show the best-obtained value in the comparisons

an excellent classification performance and provide better results than SVM in many instances. They also concluded that visualization can be used to identify cyber criminals and can help users authenticate fellow online members to prevent cyber fraud. Accuracies of 68.92 and 87.00 were obtained for Write prints and SVM, respectively.

In [100], they introduced approaches to deal with imbalanced multi-class textual datasets. The main idea behind their approach is to divide the training texts into text samples based on the class size thus, a fairer classification model could be generated. Therefore, it becomes possible to divide majority classes into less and longer samples and minority classes into many shorter samples. They used text sampling techniques to form a training set based on a desirable distribution over the classes. By text sampling, they developed new synthetic data that artificially caused the training size of a class to increase. A series of authorship identification experiments were conducted by these researchers on different multiclass imbalanced cases belonging to two text corpora of two languages; newspaper reportage in Arabic and newswire stories in English. Properties of the presented techniques were revealed by the results obtained through these experiments. They also tested four methods to deal with the problem of class imbalance [100]:

- The first method: To under-sample majority classes based on training texts. The same amount of text which was equal to the base was used. No modification is applied to the length of each text.
- The second method: To Under-sample majority classes based on training text lines. All the training texts for a particular author were merged to form a big text. Assuming that x_{min} represents the size (in text lines) of the shortest big file then, the first x_{min} text lines of each big file were segmented into text samples of length a (in text lines). It is worth noting that there was as least one complete sentence in each text line in both corpora. It was concluded that smaller values (such as 2 or 3) lead to better results.
- The third method: Re-balancing the dataset by text samples of varying length. As was mention earlier in this paper, one big file is generated for each author by concatenation of training texts. In other words, the length of text sample is equal to xi/k (where, k is predefined parameter). Short text samples belong to minority authors and long text samples belong to majority authors. Therefore, a balanced dataset is generated which consists of k text samples per class. Experiments were conducted for $k = 10$, 20, and 50. It is noteworthy that each text line

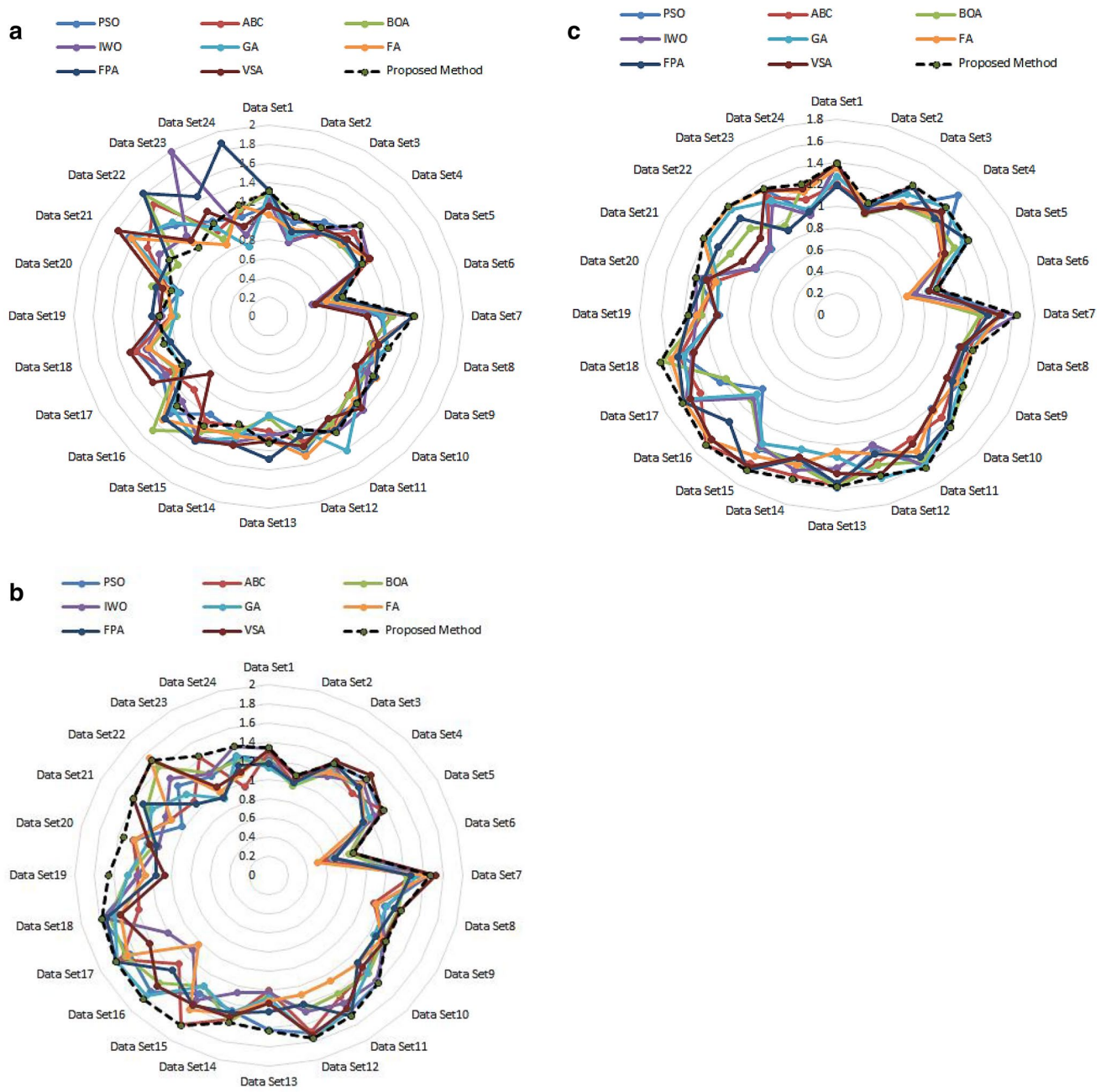

Fig. 3 Result of comparison **a** The worst criterion. **b** The best criterion. **c** The mean criterion

of the training corpus is used exactly once in the text samples.

- The fourth method: Re-balancing the dataset through text re-sampling. A big file is generated for each author once again. Assuming that x represents the text-length (in text lines) of the ith author and xmax is the longest file then, for each author, $k + x_{max}/x_i$ text samples are generated each of which consisting of xi/k. Therefore, based on the length of the big file, a variable number of text samples are generated for each author. Nonetheless, the relationship is inversed now. Longer text samples are generated for the majority classes but a large number of short text samples are generated for the minority classes.

Using a data set extracted from Arabic novels, they the modified this to two sets of words AFW54 and AFW65, with 11 words eliminated [101]. These two sets were used convert several Arabic texts into frequency vectors. They carried out a performance evaluation on these word sets

Table 12 Comparison of the proposed method with other algorithms based on the best criterion

Datasets	PSO	ABC	BOA	IWO	GA	FA	FPA	VSA	Proposed Method
Dataset1	1.2178	1.2977	1.1785	1.3271	1.1248	1.3193	1.1703	1.3125	**1.3425**
Dataset2	1.0049	1.011	0.9677	1.0707	1.0007	1.0669	1.0093	1.0633	**1.0833**
Dataset3	1.3354	1.3319	1.2486	1.2042	1.271	1.2486	**1.3494**	1.3764	**1.3494**
Dataset4	1.4182	1.2193	1.3134	1.3666	1.3154	1.3457	1.3129	**1.4819**	1.4182
Dataset5	1.3251	1.3573	1.2022	1.2551	1.1989	1.1362	1.1206	1.3422	**1.3673**
Dataset6	0.8806	0.6019	0.8504	0.6873	0.6848	0.5194	0.7076	0.9001	**0.9031**
Dataset7	1.6409	1.4838	1.4203	**1.6609**	1.5315	1.6245	1.4666	1.7136	**1.6609**
Dataset8	1.2302	1.1222	1.3725	1.1426	1.2599	1.1481	1.3451	1.4027	**1.4047**
Dataset9	1.3527	1.3663	1.3565	**1.3902**	1.2539	1.3352	1.2761	1.3717	**1.3902**
Dataset10	**1.5953**	1.4125	1.4442	1.5566	1.4284	1.3015	1.2941	1.3567	**1.5966**
Dataset11	1.6458	1.4675	1.4298	1.5415	1.6417	1.276	**1.7019**	1.6149	**1.7019**
Dataset12	1.7133	1.7001	1.4547	1.4802	**1.7619**	1.2872	1.3965	1.7329	1.7610
Dataset13	1.6127	1.208	1.2482	1.2267	1.2807	1.3122	1.4292	1.3327	**1.6227**
Dataset14	1.4605	1.5531	**1.5892**	1.2713	1.4945	1.4966	1.4835	1.5378	**1.5892**
Dataset15	1.4354	1.8054	1.3415	1.5058	1.3346	1.6282	1.5713	1.5591	**1.8154**
Dataset16	1.7259	1.3034	1.5922	1.0983	1.7772	1.0201	1.4102	1.6328	**1.8328**
Dataset17	1.6531	1.7521	1.6948	1.1967	1.7978	1.673	**1.8108**	1.4214	**1.8108**
Dataset18	**1.7742**	1.3852	1.6935	1.7389	1.6448	1.5793	1.7353	1.5856	**1.7742**
Dataset19	1.3372	1.3511	1.4452	1.3507	1.4515	1.2678	1.1654	1.0718	**1.6515**
Dataset20	1.4412	1.4441	1.2213	1.1804	1.2992	1.4351	1.2057	1.2681	**1.5441**
Dataset21	1.0257	1.1663	1.4461	1.2206	1.3912	1.1613	1.4962	**1.6151**	**1.6151**
Dataset22	1.3227	1.0865	1.6166	1.44	1.2056	**1.731**	1.0597	1.7058	1.7031
Dataset23	1.1935	1.426	1.2313	1.2189	0.9236	1.01	0.9433	1.0665	**1.4466**
Dataset24	1.2898	0.9603	1.2427	**1.3963**	1.2943	1.1029	1.1937	1.1191	**1.3963**

Bold values is to show the best-obtained value in the comparisons

through experiments which used a hybridization of an EA and LDA to generate a classifier. Then, they fed unseen data to that classifier in order to test it. The obtained performance was apparently consistent with results of authorship attribution researches performed on other languages. It is arguable that AFW54 is a more suitable choice nevertheless; such a claim cannot be made with any statistical significance. For the cases considered here, only a small number of investigations are reported for evaluating the appropriate 'chunk' size. In real-world applications this will be probably dependent on several factors, but they have identified at least about 1,000 characterization of function word usage for Arabic authors. Through this work, they have confirmed that the concept of function words translates properly into the Arabic language. In other words, various authors use this set of words in various ways, and this enables us to recognize stylistic features of individual authors and use them to distinguish between different authors [101].

High dimensional datasets bring about more computational challenges. One of the problems with high dimensional datasets is that in most cases, all features of data are not crucial for the knowledge implicit in the data [85, 102]. Consequently, in most occasions, reduction in the dimensions of data is a favored subject. Often, many of candidate features for learning are irrelevant and superfluous and degrade the efficiency of the learning algorithm [103, 104]. Learning accuracy and teaching speed may be worsening with superfluous features. Therefore, choosing the corresponding necessary features in preprocessing phase is essentially important. In this section, for identifying the author, at the first stage, the frequency of words is obtained using the method TF-IDF [105]. At the second stage, each feature is weighted [106]. At the third stage, using metaheuristic algorithms, FS is performed. At the fourth stage, classification is performed via KNN [106].

Furthermore, we used the accuracy as the evaluation measure. This accuracy is calculated as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} * 100 \tag{22}$$

In the case that TP represents the number of authors who are in the positive class while, TN indicates the number of authors who are in the negative class. Furthermore, FP is the number of authors falsely was considered as positive class by the model and FN is the number of authors falsely

Table 13 Comparison of the Proposed Method with other algorithms based on the mean criterion

Datasets	PSO	ABC	BOA	IWO	GA	FA	FPA	VSA	Proposed Method
Data set1	1.3916	1.1978	1.3929	1.3897	1.2765	1.3521	1.1941	1.3898	**1.3929**
Data set2	0.9622	1.0635	0.9613	0.9978	**1.0673**	1.0438	1.0089	0.9741	**1.0673**
Data set3	1.1495	1.3674	1.1479	1.1516	1.2789	1.1939	1.3617	1.1515	**1.3717**
Data set4	**1.5543**	1.2753	1.2469	1.2985	1.4039	1.2656	1.2512	1.3383	1.4019
Data set5	1.0945	1.1002	1.23701	1.1344	1.3232	1.1146	**1.3776**	1.1253	**1.3776**
Data set6	0.9357	**0.9499**	0.6947	0.7346	0.8517	0.6554	0.9393	0.8642	0.9393
Data set7	1.5092	1.3172	1.3024	**1.6366**	1.4551	1.4772	1.3768	1.4806	**1.6366**
Data set8	1.1491	1.2448	1.1641	1.1609	1.2675	**1.2762**	1.2003	1.1581	**1.2762**
Data set9	1.2819	1.2329	**1.3252**	1.1997	1.2909	1.2488	1.2102	1.1581	1.3152
Data set10	1.2122	1.3358	**1.4584**	1.4552	1.4227	1.2478	1.4237	1.2327	**1.4584**
Data set11	1.6026	1.3283	1.5601	1.6014	1.5996	1.4536	1.5107	1.3743	**1.6214**
Data set12	1.2846	1.4074	1.4251	1.2414	**1.5571**	1.3214	1.3231	1.5242	1.5271
Data set13	**1.5899**	1.5719	1.5801	1.4036	1.3117	1.2544	1.5519	1.4592	1.5811
Data set14	1.3798	1.5262	1.4221	1.4781	1.2758	1.4304	1.3609	1.3481	**1.5662**
Data set15	1.4311	1.5769	1.4087	1.3936	1.3683	1.4963	**1.6504**	1.6188	**1.6504**
Data set16	0.9593	**1.6964**	1.1033	1.0707	1.0311	1.6315	1.3922	1.6179	**1.6964**
Data set17	1.2304	1.4441	1.1709	1.5863	1.5396	1.6017	**1.6124**	1.5419	**1.6224**
Data set18	1.4723	1.4846	1.6506	1.3603	1.4505	1.5661	1.4996	1.3536	**1.6706**
Data set19	1.0788	1.2331	1.2337	1.2852	1.0904	1.2745	1.3522	1.0931	**1.3622**
Data set20	1.3175	1.1659	1.2635	**1.3272**	1.1213	1.1434	1.2574	1.2245	**1.3272**
Data set21	0.8526	0.8718	1.1197	0.8701	1.3629	1.4002	1.2535	0.9912	**1.4032**
Data set22	0.8526	0.8718	1.1197	0.8701	1.3629	1.4002	1.2535	0.9912	**1.4082**
Data set23	1.3227	1.2502	0.9458	1.1709	1.2078	**1.3314**	0.8971	1.3211	**1.3414**
Data set24	0.9655	1.0916	1.2229	0.9476	0.9983	1.1743	0.9823	1.2059	**1.2429**

Bold values is to show the best-obtained value in the comparisons

was considered as negative class by the model, even though they were positive.

6.1 Reuter_50_50 dataset

In this subsection, the Proposed Method and other algorithms are applied to Reuter_50_50 datasets. The dataset contains 2500 documents and 50 writers (https://archive.ics.uci.edu/ml/datasets/reuter_50_50). The results from the discussed algorithms and the results from other papers are presented in Table 14 and Fig. 4. The results show that the proposed method has a better identification accuracy compared to other algorithms. Moreover, BOA and FPA have also better identification accuracy compared to other algorithms.

Table 14 Comparison of proposed method with other algorithms on Reuter_50_50 datasets

#	Algorithms	Accuracy (%)
1	GA	82
2	PSO	87.91
3	ABC	86
4	BOA	88.13
5	IWO	87.01
6	FPA	88
7	FA	85.6
8	SVA	88.2
9	Delta [90]	67
10	KNN [90]	69
11	SVM [90]	85
12	Proposed method	89.3

6.2 PAN dataset

These datasets consist of scientific documents in Greek, English, and Spanish, and from 2011 until now, a new dataset has been added to the existing ones every year (https://pan.webis.de). The results from the discussed algorithms and the results from other papers on these datasets are evaluated in Table 15 and Fig. 4. Identification accuracy of proposed method for PAN11, PAN12, PAN13, PAN14, PAN15, and PAN16 are 84%, 80.9%, 81.3%, 82.12%, 83.25%, and 81.79%, respectively.

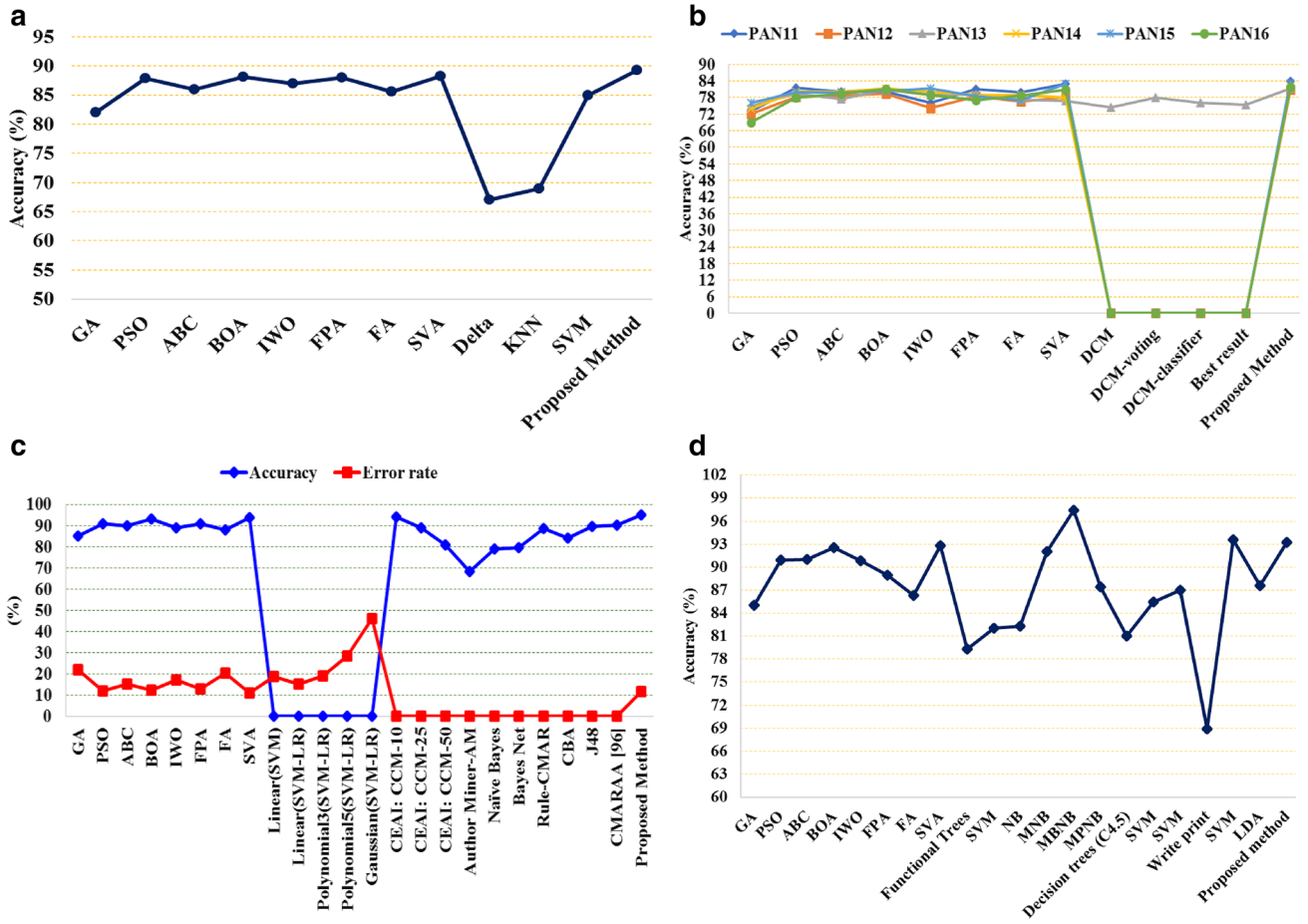

Fig. 4 Result of comparison **a** Reuter_50_50 dataset. **b** PAN Dataset. **c** Enron Email Dataset. **d** Arabic Scripts

Table 15 Comparison of proposed method with other algorithms on PAN datasets

#	Dataset Method	PAN11	PAN12	PAN13	PAN14	PAN15	PAN16
1	GA	73	72	75	74	76	69
2	PSO	81.5	78.3	79.5	80.11	79.78	78.01
3	ABC	80.1	78.8	77.6	80.1	79.9	79.32
4	BOA	80.14	79.35	80.82	81.24	80.3	81
5	IWO	76	74.3	79	80.1	81.2	79
6	FPA	81.01	78.51	79.08	79.03	78.4	77
7	FA	80	76.5	77.3	79	77.02	78.6
8	SVA	83	78.3	76.9	78.1	83	80.9
9	DCM [91]	–	–	74.4	–	–	–
10	DCM-voting [91]	–	–	78.1	–	–	–
11	DCM-classifier [91]	–	–	76	–	–	–
12	Best result [92]	–	–	75.3	–	–	–
13	Proposed method	84	80.9	81.3	82.12	83.25	81.79

Moreover, identification accuracies of DCM models are less than other algorithms. The algorithms BOA, ABC and IWO have better identification accuracies compared to the algorithms GA, PSO, FPA, and FA.

6.3 Enron email dataset

This dataset is collected and prepared by CALO project (a cognitive assistant that learns and organizes). This dataset includes comments of 150 users who are CEOs of Enron (https://www.cs.cmu.edu/~enron/). The results from the proposed method and the results from other papers on Enron Email dataset are presented in Table 16 and Fig. 4. The results show that the accuracy and error rate in proposed method are 95.04 and 11.68, respectively. Accuracy in the algorithms PSO, BOA and FPA are 91.02, 93.01, and 90.78, respectively. The accuracy and error rate in ABC algorithm are 90.02 and 15.2, respectively. Among other models, the model CCM-10 has a better accuracy, and the lowest accuracies are seen in the models Naïve Bayes and Bayes Net.

6.4 Arabic scripts

This dataset consists of 30 documents from 10 authors. The author was chosen from the website, (http://www.alwaraq.net) and their names are: Aljahedh, Alghazali, Alfarabi, Almas3ody, Almeqrezi, Altabary, Altow7edy, Ibnaljawzy, Ibnrshd, and Ibnsena. The results from the proposed method and the results from other papers on these datasets are presented in Table 17 and Fig. 4. The identification accuracy of proposed method model is 93.24%, which is better than other models.

According to the experiments results, it is concluded that the Proposed Method has a better performance than other models in terms of identification accuracy. According to Tables 15, 16, 17 the proposed method in benchmark functions is the closest to minimum compared to the algorithms FPA, IWO, BOA, ABC, PSO, GA, and FA. Moreover, the proposed method has a better accuracy in the author identification problem. The rate of accuracy of ABC, BOA and proposed method is indicated in Table 17. The results revealed that the proposed method outperformed to the other models that is ABC and BOA models. The percentage of proposed method is 93.24%. Consequently, the percentage of ABC is 91.00% and it is 92.51% for BOA model.

Table 16 Comparison of proposed method with other algorithms on enron email dataset

#	Algorithms	Accuracy (%)	Error rate
1	GA	85.06	22
2	PSO	91.02	12
3	ABC	90.02	15.2
4	BOA	93.01	12.3
5	IWO	89.03	17.32
6	FPA	90.78	13
7	FA	88.05	20.3
8	SVA	93.89	10.98
9	Linear(SVM) [93]	–	18.86
10	Linear(SVM-LR) [93]	–	15.34
11	Polynomial3(SVM-LR) [93]	–	19.20
12	Polynomial5(SVM-LR) [93]	–	28.47
13	Gaussian(SVM-LR) [93]	–	46.01
14	CEAI: CCM-10 [94]	94	–
15	CEAI: CCM-25 [94]	89	–
16	CEAI: CCM-50 [94]	81	–
17	Author miner-AM [95]	68.19	–
18	Naïve bayes [95]	79.08	–
19	Bayes net [95]	79.56	–
20	Classification by multiple association rule-CMAR [95]	88.47	–
21	Classification by association-CBA [95]	84.18	–
22	J48 [95]	89.45	–
23	Classification by multiple association rule for authorship attribution-CMARAA [95]	90.08	–
24	Proposed method	95.04	11.68

Table 17 Comparison of proposed method with other algorithms on Arabic scripts

#	Algorithms	Accuracy (%)
1	GA	85
2	PSO	90.9
3	ABC	91
4	BOA	92.51
5	IWO	90.8
6	FPA	89
7	FA	86.3
8	SVA	92.8
9	Functional trees [96]	79.3
10	SVM [96]	82.0
11	NB [97]	82.30
12	MNB [97]	92.03
13	MBNB [97]	97.43
14	MPNB [97]	87.40
15	Decision trees (C4.5) [98]	81.03
16	SVM [98]	85.43
17	SVM [99]	87.00
18	Write print [99]	68.92
19	SVM [100]	93.60
20	LDA [101]	87.63
21	Proposed method	93.24

7 Conclusion and feature works

We proposed three State based on the hybrid of chaotic and VSA in this paper for FS. State2 compared with Method1, Method3, and VSA where it had better values. We also used State2 to FS and text author identification. This paper is accompanied by using 10 CMs to enhance the overall performance and precision of the VSA. VSA is introduced to one of the challenge problems, especially FS. The proposed methods have been evaluated on 24 benchmark datasets. Four precise evaluation standards are followed in this paper. These standards are worst, best, mean, and SD. Similarly, the performance of Proposed Method is compared with the popular and maximum current other algorithms. These algorithms are PSO, ABC, BOA, IWO, GA, FA, FPA, and VSA. The experimental effects show that State2 outperforms the other algorithms in terms of best and mean fitness.

Moreover, the outcomes displayed that the Proposed Method (State2) with Tent map can drastically enhance VSA in terms of classification overall performance, stability exceptional, number of FS, and convergence speed. Moreover, the outcomes showed that Tent map turned into the satisfactory map. Therefore, the following conclusion can be drawn:

- The CMs improve the section of exploration because they change the radius of value search, helping the trapped masses to release themselves from local minima.
- The CMs are permitted to adaptively adjust exploration and exploitation by the proposed method. As it were, the Proposed Method (State1) encourages VSA to transit gradually from the exploration stage to the exploitation stage.

Essential work on the integration of CMs with the addition of other metaheuristic algorithms will be considered. VSA's performance on more problematic science and real-world engineering problems will be applied in future verification.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Gharehchopogh FS, Gholizadeh H (2019) A comprehensive survey: whale optimization algorithm and its applications. Swarm Evol Comput 48:1–24
2. Shayanfar H, Gharehchopogh FS (2018) Farmland fertility: a new metaheuristic algorithm for solving continuous optimization problems. Appl Soft Comput 71:728–746
3. Razmjooy N, Khalilpour M, Ramezani M (2016) A new meta-heuristic optimization algorithm inspired by FIFA world cup competitions: theory and its application in PID designing for AVR system. J Control Autom Electr Syst 27(4):419–440
4. Razmjooy N, Ramezani M (2014) An improved quantum evolutionary algorithm based on invasive weed optimization. Indian J Sci Res 4(2):413–422
5. Gharehchopogh FS, Shayanfar H, Gholizadeh H (2019) A comprehensive survey on symbiotic organisms search algorithms. Artificial Intelligence Review
6. Harrison KR, Engelbrecht AP, Ombuki-Berman BM (2016) Inertia weight control strategies for particle swarm optimization. Swarm Intell 10(4):267–305
7. Xing B, Gao W-J (2014) Invasive Weed Optimization Algorithm. In: Xing B, Gao W-J (eds) Innovative COMPUTATIONAL INTELLIGENCE: A ROUGH GUIDE TO 134 CLEVER ALGORITHms. Springer International Publishing, Cham, pp 177–181
8. Qi X, Zhu Y, Zhang H (2017) A new meta-heuristic butterfly-inspired algorithm. J Comput Sci 23:226–239
9. Karaboga D (2005) An idea based on honeybee swarm for numerical optimization. Technical Report TR06, Erciyes University, Engineering Faculty, Computer Engineering Department
10. Pan W-T (2012) A new fruit fly optimization algorithm: taking the financial distress model as an example. Knowl-Based Syst 26:69–74
11. Yang XS (2008) Nature-Inspired Metaheuristic Algorithms. Luniver Press, United Kingdom
12. Gandomi AH, Alavi AH (2012) Krill herd: A new bio-inspired optimization algorithm. Commun Nonlinear Sci Numer Simul 17(12):4831–4845
13. Storn R, Price K (1996) Minimizing the real functions of the ICEC'96 contest by differential evolution. In: Proceedings of IEEE International Conference on Evolutionary Computation

14. Yang X-S (2012) Flower pollination algorithm for global optimization. In: Unconventional computation and natural computation. Berlin, Heidelberg
15. Navid R et al (2019) A comprehensive survey of new meta-heuristic algorithms. In: Recent advances in hybrid metaheuristics for data clustering, p 1–25
16. Ali N, Mehdi R, Navid R (2016) A New Meta-Heuristic Algorithm for Optimization Based on Variance Reduction of Gaussian distribution. Majlesi J Electr Eng 10(4):49–56
17. Li B, Jiang W (1998) Optimizing complex functions by chaos search. J Cybern Syst 29:409–419
18. Li Y-Y, Wen Q-Y, Li L-X (2009) Modified chaotic ant swarm to function optimization. J China Univ Posts Telecommun 16(1):58–63
19. Yi J, Jian D, Zhenhong S (2017) Pattern synthesis of MIMO radar based on chaotic differential evolution algorithm. Optik 140:794–801
20. He Y et al (2014) A novel chaotic differential evolution algorithm for short-term cascaded hydroelectric system scheduling. Int J Electr Power Energy Syst 61:455–462
21. Wang G-G et al (2014) Chaotic krill herd algorithm. Inf Sci 274:17–34
22. Prasad D, Mukherjee A, Mukherjee V (2017) Application of chaotic krill herd algorithm for optimal power flow with direct current link placement problem. Chaos Solitons Fractals 103:90–100
23. Yousri D et al (2019) Chaotic flower pollination and grey wolf algorithms for parameter extraction of bio-impedance models. Appl Soft Comput 75:750–774
24. Yousefi M et al (2018) Chaotic genetic algorithm and Adaboost ensemble metamodeling approach for optimum resource planning in emergency departments. Artif Intell Med 84:23–33
25. Hong W-C et al (2013) Cyclic electric load forecasting by seasonal SVR with chaotic genetic algorithm. Int J Electr Power Energy Syst 44(1):604–614
26. Chen K, Zhou F, Liu A (2018) Chaotic dynamic weight particle swarm optimization for numerical function optimization. Knowl-Based Syst 139:23–40
27. Chuang L-Y, Hsiao C-J, Yang C-H (2011) Chaotic particle swarm optimization for data clustering. Expert Syst Appl 38(12):14555–14563
28. Liu L et al (2018) Research on ships collision avoidance based on chaotic particle swarm optimization. In: Advances in smart vehicular technology, transportation, communication and applications. Springer International Publishing, Cham
29. Ji J et al (2017) Self-adaptive gravitational search algorithm with a modified chaotic local search. IEEE Access 5:17881–17895
30. García-Ródenas R, Linares LJ, López-Gómez JA (2019) A memetic chaotic gravitational search algorithm for unconstrained global optimization problems. Appl Soft Comput 79:14–29
31. Wang Y et al (2019) A hierarchical gravitational search algorithm with an effective gravitational constant. Swarm Evol Comput 46:118–139
32. Hong W-C et al (2019) Novel chaotic bat algorithm for forecasting complex motion of floating platforms. Appl Math Model 72:425–443
33. Wang H, Tan L, Niu B (2019) Feature selection for classification of microarray gene expression cancers using Bacterial Colony Optimization with multi-dimensional population. Swarm Evol Comput 48:172–181
34. Arora S, Anand P (2019) Binary butterfly optimization approaches for feature selection. Expert Syst Appl 116:147–160
35. Zakeri A, Hokmabadi A (2019) Efficient feature selection method using real-valued grasshopper optimization algorithm. Expert Syst Appl 119:61–72
36. Bolón-Canedo V, Alonso-Betanzos A (2019) Ensembles for feature selection: A review and future trends. Inf Fusion 52:1–12
37. Papa JP et al (2018) Feature selection through binary brain storm optimization. Comput Electr Eng 72:468–481
38. Guvenc U, Duman S, Hinislioglu Y (2017) Chaotic Moth Swarm Algorithm. In: 2017 IEEE International Conference on INnovations in Intelligent SysTems and Applications (INISTA)
39. Wang S et al (2017) Multiple chaotic cuckoo search algorithm. In: Advances in Swarm Intelligence. Springer International Publishing, Cham
40. Rizk-Allah RM, Hassanien AE, Bhattacharyya S (2018) Chaotic crow search algorithm for fractional optimization problems. Appl Soft Comput 71:1161–1175
41. Chahkandi V, Yaghoobi M, Veisi G (2013) CABC–CSA: a new chaotic hybrid algorithm for solving optimization problems. Nonlinear Dyn 73:475–484
42. Zhang Y, Zhou W, Yi J (2016) A novel adaptive chaotic bacterial foraging optimization algorithm. In: 2016 International conference on computational modeling, simulation and applied mathematics (CMSAM 2016), p 1–8
43. Jia D, Zheng G, Khan MK (2011) An effective memetic differential evolution algorithm based on chaotic local search. Inf Sci 181(15):3175–3187
44. Thangaraj R et al (2012) Opposition based Chaotic Differential Evolution algorithm for solving global optimization problems. In 2012 fourth world congress on nature and biologically inspired computing (NaBIC)
45. Du Pengzhen TZ, Yan S (2014) A quantum glowworm swarm optimization algorithm based on chaotic sequence. Optimization **7**(9)
46. Mitić M et al (2015) Chaotic fruit fly optimization algorithm. Knowl-Based Syst 89:446–458
47. Gandomi AH et al (2013) Chaos-enhanced accelerated particle swarm optimization. Commun Nonlinear Sci Numer Simul 18(2):327–340
48. Yao J-F et al (2001) A new optimization approach-chaos genetic algorithm. Syst Eng 1:015
49. Li J-W, Cheng Y-M, Chen K-Z (2014) Chaotic particle swarm optimization algorithm based on adaptive inertia weight. In: Control and Decision Conference (2014 CCDC), The 26th Chinese. IEEE
50. Xu X et al (2018) CS-PSO: chaotic particle swarm optimization algorithm for solving combinatorial optimization problems. Soft Comput 22(3):783–795
51. Sayed GI, Khoriba G, Haggag MH (2018) A novel chaotic salp swarm algorithm for global optimization and feature selection. Appl Intell p 1–20
52. Tuba E et al (2018) Chaotic elephant herding optimization algorithm. In: Applied Machine Intelligence and Informatics (SAMI), 2018 IEEE 16th World Symposium on. IEEE
53. Gandomi AH, Yang X-S (2014) Chaotic bat algorithm. J Comput Sci 5(2):224–232
54. Pan G, Xu Y (2016) Chaotic glowworm swarm optimization algorithm based on Gauss mutation. In: Natural computation, fuzzy systems and knowledge discovery (ICNC-FSKD), 2016 12th International Conference on. IEEE
55. Aslani H, Yaghoobi M, Akbarzadeh-T M-R (2015) Chaotic inertia weight in black hole algorithm for function optimization. In: Technology, Communication and Knowledge (ICTCK), 2015 International Congress on. IEEE
56. Yang X, Niu J, Cai Z (2018) Chaotic Simulated Annealing Particle Swarm Optimization Algorithm. In: 2018 2nd IEEE advanced information management, communicates, electronic and automation control conference (IMCEC). IEEE

57. Aggarwal S et al (2018) A social spider optimization algorithm with chaotic initialization for robust clustering. Proc Comput Sci 143(1):450–457
58. Zhang X, Feng T (2018) Chaotic bean optimization algorithm. Soft Comput 22(1):67–77
59. Boushaki SI, Kamel N, Bendjeghaba O (2018) A new quantum chaotic cuckoo search algorithm for data clustering. Expert Syst Appl 96:358–372
60. Tharwat A, Hassanien AE (2018) Chaotic antlion algorithm for parameter optimization of support vector machine. Appl Intell 48(3):670–686
61. Zhou Y, Su K, Shao L (2018) A new chaotic hybrid cognitive optimization algorithm. Cogn Syst Res 52:537–542
62. Mingjun J, Huanwen T (2004) Application of chaos in simulated annealing. Chaos, Solitons Fractals 21(4):933–941
63. Teng H, Cao A (2011) An novel quantum genetic algorithm with Piecewise Logistic chaotic map. In: Natural Computation (ICNC), 2011 Seventh International Conference on. IEEE
64. Kumar Y, Singh PK (2018) A chaotic teaching learning based optimization algorithm for clustering problems. Appl Intell, p 1–27
65. Yüzgeç U, Eser M (2018) Chaotic based differential evolution algorithm for optimization of baker's yeast drying process. Egypt Inf J
66. Ibrahim RA, Elaziz MA, Lu S (2018) Chaotic opposition-based grey-wolf optimization algorithm based on differential evolution and disruption operator for global optimization. Expert Syst Appl 108:1–27
67. Rahman TA et al (2017) Chaotic fractal search algorithm for global optimization with application to control design. In: Computer applications and industrial electronics (ISCAIE), 2017 IEEE symposium on. IEEE
68. Tuba E, Dolicanin E, Tuba M (2017) Chaotic brain storm optimization algorithm. In International conference on intelligent data engineering and automated learning. Springer, Berlin
69. Hinojosa S et al (2018) Improving multi-criterion optimization with chaos: a novel Multi-Objective Chaotic Crow Search Algorithm. Neural Comput Appl 29(8):319–335
70. Arora S, Anand P (2018) Chaotic grasshopper optimization algorithm for global optimization. Neural Comput Appl p 1–21
71. Saremi S, Mirjalili SM, Mirjalili S (2014) Chaotic Krill Herd Optimization Algorithm. Proc Technol 12:180–185
72. Wang G-G, Hossein Gandomi A, ossein Alavi A, (2013) A chaotic particle-swarm krill herd algorithm for global numerical optimization. Kybernetes 42(6):962–978
73. Zhenyu G et al (2006) Self-adaptive chaos differential evolution. In: International Conference on Natural Computation. Springer, Berlin
74. Gandomi AH et al (2013) Firefly algorithm with chaos. Commun Nonlinear Sci Numer Simul 18(1):89–98
75. dos Santos CL, Mariani VC (2012) Firefly algorithm approach based on chaotic Tinkerbell map applied to multivariable PID controller tuning. Comput Math Appl 64(8):2371–2382
76. Wang L et al (2018) A new chaotic starling particle swarm optimization algorithm for clustering problems. Math Prob Eng **2018**
77. Sayed GI, Hassanien AE, Azar AT (2017) Feature selection via a novel chaotic crow search algorithm. Neural Computing and Applications, p 1–18
78. Kohli M, Arora S (2018) Chaotic grey wolf optimization algorithm for constrained optimization problems. J Comput Des Eng 5(4):458–472
79. Doğan B, Ölmez T (2015) A new metaheuristic for numerical function optimization: Vortex Search algorithm. Inf Sci 293:125–145
80. Martin B (1995) Instance-based learning: nearest neighbour with generalisation. doctoral dissertation, University of Waikato
81. Mafarja M et al (2019) Binary grasshopper optimisation algorithm approaches for feature selection problems. Expert Syst Appl 117:267–286
82. https://archive.ics.uci.edu/ml/index.php, 2019.
83. Holland JH (1975) Adaptation in natural and artificial systems. University of Michigan Press, Ann ArborI
84. Eberhart R, Kennedy J (1995) A new optimizer using particle swarm theory. in MHS'95. In: Proceedings of the Sixth International Symposium on Micro Machine and Human Science
85. Villar-Rodriguez E et al (2016) A feature selection method for author identification in interactive communications based on supervised learning and language typicality. Eng Appl Artif Intell 56:175–184
86. Digamberrao KS, Prasad RS (2018) Author identification using sequential minimal optimization with rule-based decision tree on indian literature in Marathi. Proc Comput Sci 132:1086–1101
87. Bay Y, Çelebi E (2016) Feature selection for enhanced author identification of Turkish Text. In: Information sciences and systems. Springer, Cham
88. Zhang C et al (2014) Authorship identification from unstructured texts. Knowl-Based Syst 66:99–111
89. Zamani H et al (2014) Authorship identification using dynamic selection of features from probabilistic feature set. In: Information Access Evaluation, Multilinguality, multimodality, and interaction. Springer International Publishing, Cham
90. Nirkhi S, Dharaskar RV, Thakre VM (2014) Stylometric approach for author identification of online messages. Int J Comput Sci Inf Technol 5(5):6158–6159
91. Frery J, Largeron C, Juganaru-Mathieu M (2015) Author identification by automatic learning. In: 2015 13th International conference on document analysis and recognition (ICDAR)
92. Seidman S (2013) Authorship verification using the impostors method. In: Notebook for PAN at CLEF, p 13–16
93. Brocardo ML, Traore I, Woungang I (2015) Authorship verification of e-mail and tweet messages applied for continuous authentication. J Comput Syst Sci 81(8):1429–1440
94. Nizamani S, Memon N (2013) CEAI: CCM-based email authorship identification model. Egypt Inf J 14(3):239–249
95. Schmid MR, Iqbal F, Fung BCM (2015) E-mail authorship attribution using customized associative classification. Digit Investig 14:S116–S126
96. Otoom AF et al (2014) Towards author identification of Arabic text articles. In: 2014 5th International conference on information and communication systems (ICICS)
97. Altheneyan AS, Menai MEB (2014) Naïve Bayes classifiers for authorship attribution of Arabic texts. J King Saud Univ Comput Inf Sci 26(4):473–484
98. Abbasi A, Chen H (2005) Applying authorship analysis to arabic web content. In: Intelligence and Security Informatics. Springer, Berlin
99. Abbasi A, Chen H (2006) Visualizing authorship for identification. In: Intelligence and security informatics. Springer, Berlin
100. Stamatatos E (2008) Author identification: Using text sampling to handle the class imbalance problem. Inf Process Manage 44(2):790–799
101. Shaker K, Corne D (2010) Authorship Attribution in Arabic using a hybrid of evolutionary search and linear discriminant analysis. In: 2010 UK Workshop on Computational Intelligence (UKCI)
102. Wang Y, Feng L (2018) Hybrid feature selection using component co-occurrence based feature relevance measurement. Expert Syst Appl 102:83–99
103. Kushwaha N, Pant M (2018) Link based BPSO for feature selection in big data text clustering. Futur Gener Comput Syst 82:190–199

104. Marie-Sainte SL, Alalyani N (2018) Firefly algorithm based feature selection for arabic text classification. J King Saud Univ Comput Inf Sci
105. Uğuz H (2011) A two-stage feature selection method for text categorization by using information gain, principal component analysis and genetic algorithm. Knowl-Based Syst 24(7):1024–1032
106. Trstenjak B, Mikac S, Donko D (2014) KNN with TF-IDF based Framework for Text Categorization. Proc Eng 69:1356–1364