Centroid-Based Particle Swarm Optimization Variant for Data Classification

Jamil Al-Sawwa
Department of Computer Science
North Dakota State University
Fargo, ND, USA
jamil.alsawwa@ndsu.edu

Simone A. Ludwig
Department of Computer Science
North Dakota State University
Fargo, ND, USA
simone.ludwig@ndsu.edu

***Abstract*—Recently, data mining has become more attractive for researchers as a technique to analyze and transform raw data into useful information that would help with decision support. Over the last decade, many data mining applications have been proposed in various research areas such as medicine, agriculture, and finance. Data classification is one of the data mining processes, which is a supervised learning task that analyzes the past data to predict future data. Particle swarm optimization (PSO) is one of the most popular swarm intelligence methods that simulates the behavior of bird flocking whereby the best source of food in a certain area is sought. In this paper, a new approach for data classification based on PSO abbreviated as CPSO is proposed. The main idea of CPSO is to find the optimal centroid and the standard deviation for each target label and then use the normal distribution probability density function and the probability of each target label to classify unseen data. The performance of CPSO was tested using ten data sets and was compared to twelve classification algorithms. The experimental results show that the CPSO algorithm is competitive compared to other classification algorithms. In addition, the algorithm can be efficiently used for data classification.**

***Index Terms*—Classification, Particle Swarm Optimization**

I. Introduction

Data mining is a process of discovering hidden patterns and extracting valuable information from raw data sets. It is an interdisciplinary research area which combines different research areas such as databases, machine learning, and statistics. The main goal of data mining is to analyze historical data stored in an information repository such as data warehouse in order to predict the future [1]. There are mainly two methods of carrying out data mining; supervised learning and unsupervised learning. In supervised learning, the input variables X and the output variable Y of the data set besides the algorithm are used to build and train a model. The aim is to discover hidden relationships between the input variables X and the output variable Y that would help to predict an accurate outcome for an unseen data set. In contrast, unsupervised learning is a process of exploring hidden and interrelated structures in an unlabeled data set based on the features/inputs to learn more about the data set [1], [2].

Classification is one of the supervised learning tasks, which was successfully applied in many domains, e.g. agricultural, engineering, biomedical, and finance. The classification task starts with a labeled data set that is split into a training and a testing data set. A classification algorithm is applied on the training data set to discover and extract hidden relationships between the input attributes and the actual output that would possibly predict the correct outcome. The result of that algorithm is a model which is applied on the testing data set. The model analyzes the inputs of each instance in the testing data set to predict an outcome. Finally, the outcomes of the model are compared with the actual outcomes to measure the model performance [1], [2].

Generally, classification problems can be categorized into three categories, binary, multiclass, and multi-label problems. In binary classification, each object belongs to one of two classes, e.g., a tumor could be benign or malignant, whereas in multi-class classification, a target outcome of an object could be one of several class labels, e.g., a vehicle could be a car, SUV, bus or truck. For multi-label classification problems, one or more class labels are assigned to an object, e.g, the text of a document might be relevant to one or more topics. In this paper, we will focus on binary and multiclass classification problems.

Swarm intelligence (SI) is a paradigm that represents the social behavior of a group of individuals (usually insects) and how they interact with each other and with their environment. In the biological system, for example a flock of birds, there is no centralized control to guide the behavior of each bird, a bird somehow behaves randomly within a space and shares information with its neighbors to find the best source of food in a certain area [3], [4]. SI methods are inspired from various biological systems such as ant colony optimization (ACO) [5], particle swarm optimization (PSO) [6], and bee colony optimization (BCO) [7] inspired from ant colonies, bird flocking behavior, and bee colonies, respectively. During the last decade, researchers successfully applied the SI methods in the data mining field to solve data mining problems or to find the optimal initial parameter values for classification algorithms such as support vector machine [8] and neural networks [9]. In this paper, we used one of these methods, particle swarm optimization, to solve the classification problem.

In this paper, a new approach for data classification based on PSO abbreviated as CPSO is proposed to extend the existing work. The main idea of CPSO is to find the optimal centroid and the standard deviation for each target label and then use the normal distribution probability density function and the probability of each target label to classify unseen data.

The rest of the paper is organized as follows. Section II presents the related work in data mining using swarm intelligence methods. Section III describes the basic particle swarm optimization algorithm. Section IV presents the existing approach to solve classification problems. Section V presents our proposed approach. Section VI describes the data sets that are used in the experiments as well as the preprocessing task performed on these data sets. Section VII presents the experiments and the results. Section VIII concludes our work.

II. Related Work

Swarm intelligence methods were proposed mainly to solve optimization problems.However, recently researchers successfully used these methods in the data mining area to solve classification and clustering problems. In this section, we review existing approaches and techniques of using swarm intelligence methods to solve data mining problems.

The authors in [10] proposed three versions of a PSO-based classification algorithm according to the fitness function used to evaluate the particles. The idea of their work is to find the optimal centroids for each class label in an N-dimensional search space and then assign each data instance in a testing data set to the closest centroid. The fitness function of the first version computes the percentage of misclassifications on a training data set after each data instance is assigned to the closest class centroid. In the second version, the fitness function computes the sum of all training data based on the Euclidean distance between the centroid of class label c_j and the data instance that belongs to class label c_j according to the training data set. For the last version, the fitness function is a linear combination of the fitness functions. The performance of three versions of PSO were validated using thirteen benchmark data sets and compared to nine well-known classification algorithms. From the experimental results, the authors deduced that the third version of PSO outperformed the other two versions as well as five out of nine classification algorithms in terms of the classification error rate.

In [11], the authors introduced two algorithms, a gbest PSO and a HybridPSO, for data clustering. The idea of the gbest PSO is to use the basic PSO algorithm to find the optimal centroids for a predefined number of clusters, while in the HybridPSO, the authors used the result of the k-means algorithm as the initial position of one of the particles before running the gbest PSO algorithm. According to the inter-cluster distance, intra-cluster distance, and quantization errors, the authors concluded that the performance of the gbest PSO and HybridPSO algorithms are comparable to the k-means clustering performance.

The PSO clustering algorithm was efficiently used to solve an image classification problem in [12]. In this work, the PSO clustering algorithm was applied to MRI (magnetic resonance imaging) and satellite images. The experimental results showed that the performance of the PSO algorithm outperforms the performance of the four clustering algorithms which are K-means, Fuzzy C-means, K-Harmonic means and Genetic algorithms. Another work was proposed in [13] for document clustering using the PSO clustering algorithm. The authors applied the PSO and Hybrid PSO algorithms on four different document data sets and compared the results of both algorithms with the k-means algorithm's result. The authors concluded that the hybrid PSO is better than PSO and k-means since it is able to generate higher compact clusters.

In [14], the authors used a PSO-based clustering algorithm to investigate the performance among four types of clusters, validity index, Euclidean distance based PBM index, the kernel function induced measure, the point symmetry distance-based index, and the manifold distance induced index. According to that, four versions of PSO clustering were proposed based on the validity index that is used to compute the fitness of a particle. Comprehensive experiments were performed on real and synthetic data sets to evaluate the performance of each version. The results revealed that the PSO clustering, which uses the manifold distance induced index as the fitness function achieved better accuracy and robustness than the other versions.

Another work for data clustering can be found in [15]. In this work, an algorithm (CGSO) is proposed for data clustering using a glowworm swarm optimization approach. Glowworm swarm optimization is one of the newer swarm intelligence methods that simulates the behavior of the lighting worms. Three variants of the CGSO algorithm were proposed based on three fitness functions to achieve high-quality clustering. The performance of the three variants of CGSO were verified using seven data sets and were compared with the performance of other clustering algorithms. In terms of purity and entropy of the clustering, the first variant of CGSO obtained the best result compared to the other versions and other clustering algorithms.

A new approach for data classification using BCO is proposed in [16]. The BCO method was introduced to imitate the foraging behavior of honey bees [7]. The main idea of the work is to find the optimal centroid of each class label by minimizing the sum of all training data based on the Euclidean distance between the centroid of class label c_j and the data instance that belongs to c_j. The experimental results showed that the proposed approach can efficiently be used for data classification.

Another work in classification using BCO is [17]. The authors presented a bee-colony based classification rule algorithm (ABC-Miner) to discover and extract classification rules from data sets, where each rule consists of an antecedent and consequent clause. Using three benchmark data sets, they concluded that the average accuracy of ABC-miner and the average number of extracted rules using ABC-miner are comparable and competitive with the C4.5 decision tree and PSO algorithms.

Ant colony optimization (ACO) is one of the SI techniques that imitates the behavior of ants seeking the optimal path between their nest and a food source, which was proposed to mainly solve "shortest path" type of problems [5]. In [18], a new algorithm for data classification using ACO named Ant-miner was proposed. The goal of Ant-miner is to extract simple rules from the data set in the IF-THEN form. The authors performed an experiment on six data sets to evaluate the performance of Ant-miner. From the experimental results, the authors observed that the rules extracted by the Ant-miner are simpler than those extracted by a CN2 [19] classifier (rule-induction classifier). In addition, the overall accuracy of Ant-miner is better than the CN2 classifier. Other ACO approaches in data mining are applied to [20] are data clustering, [21] features selection, [22] web page classification, and [23] data classification.

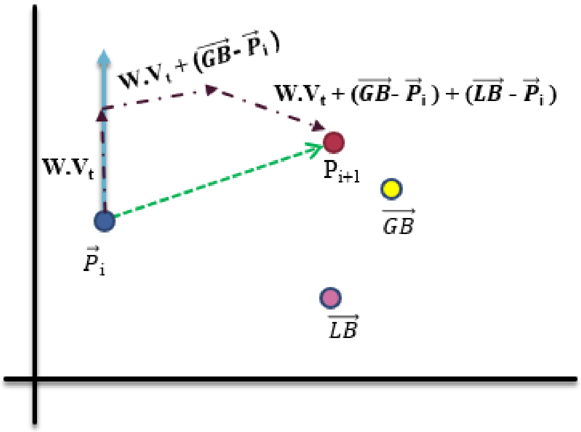

Fig. 1. Illustration of velocity and position updates.

Algorithm 1 PSO Algorithm

```
for each particle do
    randomly initialize particle's position and velocity
end for
repeat
    for each particle do
        compute fitness value ($FV$)
        if $FV$ is better than personal best fitness value ($FV\_PBest$) then
            $FV\_PBest$ = $FV$
            take current particle position as $PBest$
        end if
    end for
    take the position of particle whose best $FV$ value as $GBest$
    for each particle do
        update particle velocity using Eq. (1)
        update particle position using Eq. (3)
    end for
    update the inertia weight using on Eq. (2)
until stopping criterion is satisfied
```

In [24], a PSO-based classifier (FCM) was proposed based on c-mean fuzzy clustering, which consist of two phases, unsupervised clustering and supervised classification. In this work, PSO was used to optimize the centroid of the clusters and the parameters of the membership function. The FCM performance was evaluated using eight data sets and outperformed well-known classifiers such as support vector machine and k-nearest neighbors in terms of classification error rates.

Fuzzy c-means clustering is one of the fuzzy clustering algorithms, which divides the objects into groups where the intra-distance among objects within the group is minimized while the inter-distance among objects that belong to a different group is maximized. However, fuzzy c-means clustering suffers from two drawbacks; the first one is that the number of clusters must be predefined before the algorithm is run, and the second drawback is that most of the objects in overlapping areas are incorrectly assigned. In [25], a new approach using PSO to overcome the aforementioned drawbacks is proposed. The main goal of that approach is to automatically find the optimal number of clusters. The approach was tested using synthetic and real data sets and the experimental results revealed that the approach can correctly identify the number of clusters, however, it needs more than one run to achieve that.

III. Basic Particle Swarm Optimization Algorithm

PSO is one of the SI methods which was proposed by Eberhart and Kennedy [6] in 1995. The main idea of PSO is that it is inspired from the motions of bird flocking, when these are searching to find the best source of food in a certain area. In PSO, a particle is the basic component of the PSO algorithm which represents a potential solution in the solution space. The multiple particles that form a swarm are placed randomly in a search space, then each particle flies through the search space with its velocity and shares information with its neighbors until an optimal solution is found.

The PSO algorithm starts by initializing each particle's position vector $p = \{p_1, p_2, p_3, \ldots, p_n\}$ randomly in an N-dimensional search space with a random velocity vector $v = \{v_1, v_2, v_3, \ldots, v_n\}$. At each iteration of the PSO, each particle's position is evaluated using a fitness function to determine how close it is to achieve the goal. An outcome of that function is used to determine the $PBest$ position, which is the best position that has been found so far by each particle, and the $GBest$ position, which is the best position that has been found so far by any particle in the swarm. $PBest$ and $GBest$ are used to update the current velocity vector of each particle as follows:

$$\begin{aligned} v_j^{t+1} &= w \cdot v_j^t + r_1 \cdot c_1 \cdot \left(PBest - p_j^t\right) \\ &\quad + r_2 \cdot c_2 \cdot \left(GBest - p_j^t\right) \end{aligned} \tag{1}$$

here, p_j^t and v_j^t are the current position vector and the current velocity vector of particle j at iteration t, respectively; r_1 and r_2 are random vectors, c_1 and c_2 are constant coefficients for cognitive and social factors, respectively, which are specified by the user. The inertia weight value w is computed using Eq. 2 and starts from w_{max}, and then its value is linearly decremented as the number of iterations increase until it reaches w_{min} [26]. The v_j^{t+1} is a new velocity vector of particle j, where the new value at each dimension of the velocity vector must be within the range of $[v_{min}, v_{max}]$.

$$w(t) = w_{max} - \left((w_{max} - w_{min}) \frac{t}{T_{max}} \right) \quad (2)$$

After the particle's velocity vector is updated, the new position of the particle is updated as follows:

$$p_j^{(t+1)} = p_j^t + v_j^{t+1} \quad (3)$$

where p_j^t is the current position vector of particle j, v_j^{t+1} is the new velocity vector of particle j, and $p_j^{(t+1)}$ is the new position vector of particle j.

Fig. 1 illustrates the velocity and position updates of particle P_i at iteration t in the two-dimensional space. All previous operations are repeated until a stopping criterion is satisfied. The pseudo code of the PSO algorithm is given in Algorithm 1 [10].

IV. Existing Approach

In [10], the authors proposed a new approach for data classification using PSO. In their work, each particle flies through an N-dimensional space to find the optimal centroids for all target classes $\{c_1, c_2, \ldots, c_i\}$ in a data set. Each particle j's position and velocity are encoded as a vector as follows:

$$\vec{p_j} = \{p_j^{c_1}, p_j^{c_2}, \ldots, p_j^{c_i}\} \quad (4)$$

$$\vec{v_j} = \{v_j^{c_1}, v_j^{c_2}, \ldots, v_j^{c_i}\} \quad (5)$$

where $p_j^{c_i}$ and $v_j^{c_i}$ are the position and the velocity vector for particle j 's class label c_i, respectively, which are represented in an N-dimensional space as follows:

$$p_j^{c_i} = \{d_1, d_2, \ldots, d_n\} \quad (6)$$

$$v_j^{c_i} = \{v_1, v_2, \ldots, v_n\} \quad (7)$$

where d_n and v_n are the real values of position and velocity at dimension n, respectively.

Furthermore, each particle contains the following attributes:

- Current Fitness Value (FV)
- Best Centroids Vector that has been found so far during the journey of the particle ($PBest$)
- Best fitness value that has been found so far during the journey of the particle (FV_PBest)

In addition, the entire swarm keeps track of the best centroid vector ($GBest$) that has been found so far by any particle and the best group fitness value found so far by any particle.

For the particle's fitness evaluation, three fitness functions are used to evaluate the fitness of each particle. The first fitness function ($F1$) computes the fitness value of each particle in two steps. In the first step, each data instance in a training data set is assigned a class label whose position is closest to that data instance. In the second step, the function $F1$ computes the misclassification rate. In mathematical terms, the fitness value of particle j is calculated as follows:

$$F1\,(j) = \frac{1}{D_T} \sum_{i=1}^{D_T} \delta(\vec{x}_i) \quad (8)$$

$$\delta(\vec{x}_i) = \begin{cases} 1 & if c_{predicted}(\vec{x}_i) \neq c_{actual}(\vec{x}_i) \\ 0 & otherwise \end{cases} \quad (9)$$

where D_T is the number of data instances in a training data set, c_{actual} is the actual outcome of $\vec{x}_i$, and $c_{predicted}$ is the predicted outcome of $\vec{x}_i$.

For the second fitness function ($F2$), the fitness value of each particle is computed by taking the average of the sum of all Euclidean distances (Eq. 11) between the current position of class label c_i and the data instances that belongs to class label c_i according to the training data set. In mathematical terms, the fitness value of particle j is given by:

$$F2(j) = \frac{1}{D_T} \sum_{i=1}^{D_T} d\left(\vec{x}_i, \vec{p_j}^{c_i}\right) \quad (10)$$

where D_T is the number of data instances in a training data set, $\vec{x}_j$ is a data instance vector that belongs to class label c_i according to a training data set, and $\vec{p_j}^{c_i}$ is the current centroid vector for class label c_i.

$$d\left(\vec{x}, \vec{y}\right) = \sqrt{\sum_{i=1}^{d} (x_i - y_i)^2} \quad (11)$$

The last fitness function $F3$ is a linear combination of $F1$ and $F2$. The fitness value of particle j is computed as follow:

$$F3(j) = \frac{F1(j) + F2(j)}{2} \quad (12)$$

The algorithm starts by initializing a predefined number of particles randomly in a search space. At each iteration of CPSO, the fitness value of each particle is computed using one of the above fitness functions. After computing the fitness value and finding $PBest$ and $GBest$, the particle updates its current velocity vector and current centroid vector using Eq. 1 and Eq. 3, respectively.

The above operations are repeated until a stopping criterion is met. Finally, the optimal centroid vector of each class label is used to classify unseen data based on distance.

V. Proposed Approach

In this paper, we propose a new approach that extends the existing work to improve the performance of the classification model. The new approach is further referred to as CPSO. In CPSO, the optimal centroid vector of a class label that has been found by PSO, is used to compute the class label's standard deviation vector, which is the average of the squared differences between the class label's centroid vector and the data instances that belong to that class label. The standard deviation at dimension i is calculated as follows:

TABLE I
PROPERTIES OF DATA SETS

Data set	I	T	D	C	L
Cancer-Int	699	Real	9	2	No
Credit	690	Categorical, Integer, Real	14	2	No
Diabetes	768	Integer, Real	8	2	No
Heart-Satlog	270	Categorical, Real	13	2	No
Hepatitis	155	Categorical, Integer, Real	19	2	Yes
Balance	625	Categorical	4	3	No
Iris	150	Real	4	3	Yes
Thyroid	215	Categorical, Real	5	3	Yes
Wine	178	Integer, Real	13	3	Yes
E. Coli	327	Real	5	5	Yes

$$\sigma_i^{c_i} = \sqrt{\frac{1}{N}\sum_{j=1}^{N}\left(x_{ji}^{c_i} - p_i^{c_i}\right)^2} \tag{13}$$

where $\sigma_i^{c_i}$ is the standard deviation of class label c_i at dimension i, $p_i^{c_i}$ is the value of dimension i for class label c_i's centroid, and $x_{ji}^{c_i}$ is the value for the i dimension for the data instance j with class label c_i.

Furthermore, the probability (P) of each class label is calculated by the following equation:

$$P(c_i) = \frac{\#\ instances\ belonging\ to\ class\ label\ c_i}{\#records} \tag{14}$$

where $P(c_i)$ is the probability of class label c_i, and $\#records$ is the total number of instances in the data set.

The outcome of the previous operations is a CPSO model which is applied on a testing data set. The CPSO model assigns each data instance to the class label c_i based on the best value of the CLASSIFY function (Eq. 16).

$$dnorm\left(x_i, p_i, \sigma_i\right) = \frac{1}{\sigma_i\sqrt{2\pi}} \cdot e^{-(x_i-p_i)^2/2\sigma_i{}^2} \tag{15}$$

$$CLASSIFY\left(c_i, \vec{x}, p^{\vec{c_i}}, \sigma^{\vec{c_i}}\right) = P\left(c_i\right) \cdot \prod_{i=1}^{D} dnorm\left(x_i, p_i^{c_i}, \sigma_i^{c_i}\right) \tag{16}$$

In the CLASSIFY function, x_i is the value of dimension i of data instance x and $p_i^{c_i}$, $\sigma_i^{c_i}$ are the values of dimension i in the centroid vector $p^{\vec{c_i}}$ and the standard deviation vector $\sigma^{\vec{c_i}}$, respectively, for class label c_i.

VI. DATA SETS AND PREPROCESSING

Two types of data sets, binary data sets and multi-class data sets are used to evaluate the performance of the classification algorithms. In this section, we present the data sets and its preprocessing.

A. Data sets

The two types of data sets were taken from the UCI Machine Learning Repository [27], which is one of the most popular database repository to evaluate machine learning algorithms. Table I shows the data sets and their characteristics: the total number of instances (I), the number of features (D), the type of features (T), the number of class labels (C), and whether it is sorted by class label or not (L). The following is a brief description of these data sets:

- **Cancer-Int:** contains the diagnosis of breast cancer of patients with two outcomes: benign or malignant. It contains 699 samples and 9 features.
- **Credit:** is a mixed type data set for estimating a customer's credit card application based on 14 features: 6 real features and 8 discrete features. The instances in the data set are classified into 2 class labels: approval (+) and disapproval (-).
- **Diabetes:** was collected by the National Institute of Diabetes and Digestive and Kidney Diseases. The data samples show whether a patient has diabetes or not based on diagnostic measurements included in the data set.
- **Heart-Satlog:** contains samples of 270 patients, where each sample has 13 features that are used to predict whether heart disease exists or not for each patient.
- **Hepatitis:** contains data of patients who have the hepatitis disease and whether they are alive or died.
- **Balance:** is the result of a model psychological experiment. It has 625 instances that are classified into three class labels: tip to the right (R), tip to the left (L), and balance (B) based on four features included in the data set.
- **Iris:** contains 150 samples for three of the iris flower species: versicolor, setosa, and virginica.
- **Thyroid:** contains 5 features that are used to predict a patient's thyroid condition whether they have hypothyroidism, hyperthyroidism, or a normal thyroid.
- **Wine:** contains 13 features of a chemical analysis result of three wine cultivators that were grown in the same region in Italy.
- **E. Coli:** contains 8 features beside the class label and 327 instances that are distributed into eight class labels, but three of the eight class labels represent only up to 5 instances in the data set, so we removed them from the data set.

B. Data Preprocessing

Data preprocessing is one of the data mining processing steps that prepares a raw data set for further processing. In this stage, all data sets in Table I are normalized using mean normalization (Eq. 17) to rescale the features to the range [0,1]. In addition, we used the "ReplaceMissingValues" filter in the WEKA tool [28] to replace the missing values in some data sets.

$$e_i' = \frac{e_i - min(e)}{max(e) - min(e)} \tag{17}$$

TABLE II
MISCLASSIFICATION RATE (MCR) FOR THREE VERSIONS OF CPSO BASED ON FITNESS FUNCTIONS

Data set	**CPSO-F1**	**CPSO-F2**	**CPSO-F3**
Cancer-Int	4.58	**3.91**	4.00
Credit	**17.00**	**17.00**	17.72
Diabetes	30.64	**25.78**	26.22
Heart-Satlog	22.83	**16.04**	21.11
Hepatitis	17.42	16.13	**15.05**
Balance	16.26	**9.65**	11.68
Iris	6.22	**4.66**	5.33
Thyroid	15.81	**3.56**	7.90
Wine	4.11	2.80	**2.24**
E. Coli	19.57	**11.52**	17.94
Average	*15.44*	*11.11*	*12.92*

We have also noticed that the instances of some of the data sets are sorted by class label, so we decided to shuffle them to make sure that all class labels appear in both the training and testing phases.

VII. EXPERIMENTS AND RESULTS

In our work, we performed three experiments to evaluate the performance of CPSO compared to the performance of other classification algorithms. The validation method for the evaluation of the classifiers in three experiments are different. In the first and second experiments, we used K-fold cross-validation. For this validation method, the data set is split into k number of approximately equal sized subsets. Next, each subset is used once to test a model and the remaining $k-1$ sub sets are used to train the model; this process is repeated k times. In the second experiment, we used the same validation method as described in [10], which is the Hold-Out method.

For the three experiments, the MCR measure is used to evaluate the performance of the classifiers, which is computed as follows:

$$MCR = 100 \times \frac{\#\ incorrectly\ classified\ instances}{size\ of\ testing\ data\ set} \tag{18}$$

In this section, we present the three experiments as well as the results.

A. First Experiment

To evaluate the impact of the three fitness functions on the performance of the CPSO algorithm, we ran the three variants of CPSO (CPSO-F1, CPSO-F2, and CPSO-F3) on all data sets as described in Table I with 10-fold cross validation and compared the performance of these variants in terms of MCR. The parameters for the CPSO classifier are the same as in [10], which are:

- maximum number of iterations = 100
- acceleration coefficient constants c_1 and c_2 = 2.0
- swarm size = 50
- velocity range [v_{min} = -0.05, v_{max} = 0.05]
- weight inertia range [w_{min}= 0.4, w_{max}= 0.9]

Table II shows the average MCR results of CPSO that are achieved by each version of CPSO for each data set using 10-fold cross validation. From the results in Table II we can easily see that the performance of CPSO-F2 outperforms the performance of CPSO-F1 and CPSO-F3 in 7 out of 10 data sets in terms of MCR, which achieved the best value on the Cancer-Int, Diabetes, Heart-Satlog, Balance, Iris, Thyroid, and E. Coli data sets, where the MCR results were 3.91%, 25.78%, 16.04%, 9.65%, 4.66%, 3.56%, and 11.52%, respectively. For the Wine and Hepatitis data sets, CPSO-F3 obtained the best MCR with results of 2.24% and 15.05%, respectively. For the Credit data set, the CPSO-F1 and CPSO-F2 variants achieved the best MCR of 17.00%.

Furthermore, the MCR results of each version are averaged over the 10 data sets, as shown in the last row of Table II. From the average MCR result, we can conclude that CPSO-F2 achieved the best performance compared to CPSO-F1 and CPSO-F3.

B. Second Experiment

In this experiment, we compared the performance of the best version of CPSO, which is CPSO-F2, with the performance of eleven well-known classification algorithms. We used the Waikato Environment for Knowledge Analysis (WEKA) tool version 3.6 [28], which contains numerous implemented classification algorithms in order to execute the eleven well-known classification algorithms. From different groups of classification algorithms in WEKA, we chose the following classifiers: BayesNet from package Bayes, Multilayer Perceptron Network (MLP) and Support Vector Machine (SMO) from package Functions, K-nearest-neighbor (IBK) and KStar from package Lazy, Bagging from package Meta, Ridor from package Rules, VFI from package Misc, and RandomForest and NBTree from package Trees. The default setting parameters in WEKA were used for the twelve classification algorithms.

Table III shows the misclassification rate (MCR) obtained by each classifier with 10-fold cross-validation for each data set. The best values of the misclassification rate obtained among all classifiers is marked in bold while the worst value is marked in red italics.

At a glance, we can easily see that the CPSO classifier obtained the best MCR for the Thyroid and E. Coli data sets, where the MCR results were 3.56% and 11.52%, respectively. The Support Vector Machine (SMO) is the only classifier that outperformed CPSO, which obtained the best MCR in 4 out of 10 data sets, where the MCR results were 22.66%, 15.93%, 14.84%, and 0.56% for Diabetes, Heart-Satlog, Hepatitis, and Wine data sets, respectively. For the other classifiers, the Multilayer Perceptron Network (MLP), Bayes Net, RandomForest, and VFI obtained the best MCR in 2,1,1,1 data sets, respectively. In addition to that, another significant conclusion that can be drawn from the results is that the CPSO classifier never obtained the worst MCR on any of the data sets.

TABLE III
MISCLASSIFICATION RATE (MCR) FOR ALGORITHMS ON ALL DATA SETS

	Cancer-Int	Credit	Diabetes	Heart-Satlog	Hepatitis	Balance	Iris	Thyroid	Wine	E. Coli
CPSO-F2	3.91	17.00	25.78	16.04	16.13	9.65	4.66	**3.56**	2.80	**11.52**
Bayes Net	**2.72**	13.91	25.65	18.89	17.42	8.48	*7.33*	4.65	1.68	14.06
SMO	3.00	14.78	**22.66**	**15.93**	**14.84**	10.08	4.00	*11.16*	**0.56**	14.37
MLP	4.58	16.96	24.61	21.85	15.48	**0.48**	**3.33**	6.04	3.37	12.84
KStar	4.58	*21.16*	30.86	*24.81*	18.71	12.16	5.33	5.11	1.12	16.51
IBK	4.72	17.97	29.82	*24.81*	19.35	15.36	4.66	3.72	4.49	17.73
Bagging	4.00	14.35	24.61	18.89	16.77	23.20	4.66	6.04	*6.74*	15.60
VFI	4.29	13.91	*36.07*	20.00	**14.84**	16.00	4.00	10.70	2.80	*19.27*
Ridor	*5.44*	15.51	25.00	21.85	*23.23*	31.04	5.33	9.30	6.18	16.82
NBTree	3.72	14.49	26.43	19.63	18.71	8.48	*7.33*	5.58	2.24	17.74
J48	4.86	14.93	26.17	23.33	19.35	*37.12*	6.00	6.98	6.18	15.90
RandomForest	3.58	**12.46**	25.65	18.14	16.13	20.64	6.00	5.58	1.68	12.84

TABLE IV
RANKING OF ALGORITHMS

	MLP	CPSO-F2	SMO	Bayes Net	Random Forest	NBTree	Bagging	KStar	VFI	IBK	Ridor	J48
Average	10.95	11.11	11.14	11.48	12.27	12.44	13.49	14.04	14.19	14.26	15.97	16.08
Rank	1	2	3	4	5	6	7	8	9	10	11	12

TABLE V
MISCLASSIFICATION RATE (MCR) FOR CPSO-F2, PSO-ψ_2, AND PSO-ψ_3

Data set	CPSO-F2	PSO-ψ_2 [10]	PSO-ψ_3 [10]
Balance	**8.97**	25.47	13.12
Cancer-Int	**1.71**	2.87	2.64
Diabetes	22.92	22.50	**21.77**
Iris	**0.00**	2.63	5.26
E. Coli	**7.31**	14.63	13.90
Wine	**2.22**	**2.22**	2.88
Thyroid	**0.00**	5.55	3.88
Average	*6.16*	*10.84*	*9.06*

To summarize the results in Table IV, the MCR results of each classifier are averaged over the 10 data sets, then the classifiers are ranked based on the averaged values as shown in Table III. From this we can see that the CPSO-F2 classifier is ranking second with an average MCR of 11.11%, which is quite close to the best classifier (MLP).

C. Third Experiment

In this experiment, we used Hold-Out validation as described in [10] applied to the data sets to compare the performance of the CPSO-F2 classifier with the performance of the best two versions of PSO (version 2 and 3) as given in [10]. In the Hold-Out validation method, a data set is split into two sub data sets; 75% of the instances is used for training and the remaining for testing. The parameters for the CPSO-F2 classifier are the same as in [10], which are:

- maximum number of iterations = 1000
- acceleration coefficient constants c_1 and c_2 = 2.0
- swarm size = 50
- velocity range [v_{min}= -0.05, v_{max} = 0.05]
- weight inertia range [w_{min} = 0.4, w_{max} = 0.9]

Table V shows the MCR obtained by CPSO-F2, PSO-ψ_2, and PSO-ψ_3 for each data set as well as the MCR average over all data sets. The first conclusion that can be drawn from the results, is that CPSO-F2 outperforms the performance of PSO-ψ_3 and PSO-ψ_2 in 6 out of 7 data sets in terms of MCR, whereas PSO-ψ_3 obtained a somewhat better MCR result for the Diabetes data set. For the Wine data set, CPSO-F2 and PSO-ψ_2 obtained the same MCR. The second conclusion is interesting since CPSO-F2 obtains improved MCR values averaged over all data sets by a factor of 1.75 and 1.5 compared to the average MCR of PSO-ψ_2, PSO-ψ_3, respectively.

VIII. CONCLUSION

In this paper, we proposed the CPSO algorithm for data classification. CPSO uses the basic PSO algorithm to find the optimal centroid for each target class and then computes the standard deviation based on that centroid. Next, the normal distribution density function (PDF) and the probability of each target class is used to classify each instance in a testing data set.

Three experiments were performed to evaluate the performance of CPSO starting by evaluating the performance of CPSO based on the fitness functions and then compared the performance of the best version of CPSO to the performance of the other classification algorithms in terms of MCR. The first experiment results showed that the performance of CPSO-F2 outperformed the performance of the other versions of CPSO (CPSO-F1 and CPSO-F3). In the second experiment, the results showed that the performance of CPSO-F2 outperformed 10 out of 11 classification algorithms and its average MCR is quite close to the average MCR of the best classification algorithm, which is MLP. For the third experiment, the results showed that CPSO-F2 outperformed the performance of PSO-ψ_2 and PSO-ψ_3. From these results, we can conclude that the performance of CPSO-F2 is comparable and competitive with other classification algorithms and that CPSO-F2 can be efficiently used for data classification.

Our future work involves scaling the CPSO algorithm using one of the big data frameworks, such as map-reduce or spark, to work with large data sets and investigate the scalability and the performance of CPSO. Furthermore, improving the average MCR of CPSO using different density estimators such as kernel estimation with Gaussian kernels should be worked on.

References

[1] S. Sayad, Real time data mining. Cambridge, Ontario: Self-Help Publishers, 2011.

[2] M. J. Zaki and W. Meira, Data mining and analysis: fundamental concepts and algorithms. New York: Cambridge University Press, 2014.

[3] C. A. C. Coello, S. Dehuri, and S. Ghosh, Swarm Intelligence for Multiobjective Problems in Data Mining. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009.

[4] K. E. Parsapoulos and M. V. Vrahatis, Particle swarm optimization and intelligence: advances and applications. Hershey: Information Science Reference, 2010.

[5] M. Dorigo and T. Stutzle, Ant colony optimization. Cambridge, Mass.: MIT Press, 2004.

[6] J. Kennedy and R. Eberhart, Particle swarm optimization, Neural Networks, 1995. Proceedings., IEEE International Conference on, Perth, WA, 1995, pp. 1942-1948 vol. 4.

[7] D. KARABOGA, An Idea based on Honey Bee Swarm for Numerical Optimization, Computer Engineering Department, Engineering Faculty, Erciyes University, tech., 2005.

[8] J. Li and B. Li, Parameters Selection for Support Vector Machine Based on Particle Swarm Optimization, Intelligent Computing Theory Lecture Notes in Computer Science, pp. 41-47, 2014.

[9] P. R. Lorenzo, J. Nalepa, M. Kawulok, L. S. Ramos, and J. R. Pastor, Particle swarm optimization for hyper-parameter selection in deep neural networks, Proceedings of the Genetic and Evolutionary Computation Conference on - GECCO, 2017.

[10] I. D. Falco, A. D. Cioppa, and E. Tarantino, Facing classification problems with Particle Swarm Optimization, Applied Soft Computing, vol. 7, no. 3, pp. 652-658, 2007.

[11] D. V. D. Merwe and A. Engelbrecht, Data clustering using particle swarm optimization, The 2003 Congress on Evolutionary Computation, 2003.

[12] M. Omran, A. Engelbrecht, and A. Salman, Particle Swarm Optimization Method For Image Clustering, International Journal of Pattern Recognition and Artificial Intelligence, vol. 19, no. 03, pp. 297-321, 2005.

[13] X. Cui, T. Potok, and P. Palathingal, Document clustering using particle swarm optimization, Proceedings 2005 IEEE Swarm Intelligence Symposium.

[14] R. Liu, X. Sun, L. Jiao, and Y. Li, A comparative study of different cluster validity indexes, Transactions of the Institute of Measurement and Control, vol. 34, no. 7, pp. 876-890, 2011.

[15] I. Aljarah and S. A. Ludwig, A new clustering approach based on Glowworm Swarm Optimization, 2013 IEEE Congress on Evolutionary Computation, 2013.

[16] D. Karaboga and C. Ozturk, A novel clustering approach: Artificial Bee Colony (ABC) algorithm, Applied Soft Computing, vol. 11, no. 1, pp. 652-657, 2011.

[17] M. Celik, D. Karaboga, and F. Koylu, Artificial bee colony data miner (ABC-Miner), 2011 International Symposium on Innovations in Intelligent Systems and Applications, 2011.

[18] R. Parpinelli, H. Lopes and A. Freitas, Data mining with an ant colony optimization algorithm, IEEE Transactions on Evolutionary Computation, vol. 6, no. 4, pp. 321-332, 2002.

[19] P. Clark and T. Niblett, The CN2 induction algorithm, Machine Learning, vol. 3, no. 4, pp. 261283, 1989.

[20] P. Shelokar, V. Jayaraman, and B. Kulkarni, An ant colony approach for clustering, Analytica Chimica Acta, vol. 509, no. 2, pp. 187195, 2004.

[21] K. Robbins, W. Zhang, J. Bertrand and R. Rekaya, The ant colony algorithm for feature selection in high-dimension gene expression data for disease classification, Mathematical Medicine and Biology, vol. 24, no. 4, pp. 413-426, 2007.

[22] N. Holden and A. A. Freitas, Web Page Classification with an Ant Colony Algorithm, Lecture Notes in Computer Science Parallel Problem Solving from Nature - PPSN VIII, pp. 1092-1102, 2004.

[23] A. Chan and A. A. Freitas, A new ant colony algorithm for multi-label classification with applications in bioinfomatics, Proceedings of the 8th annual conference on Genetic and evolutionary computation - GECCO, 2006.

[24] H. Ichihashi, K. Honda, A. Notsu, and K. Ohta, Fuzzy c-means classifier with particle swarm optimization, 2008 IEEE International Conference on Fuzzy Systems (IEEE World Congress on Computational Intelligence), 2008.

[25] M. Chen and S. A. Ludwig, Particle Swarm Optimization Based Fuzzy Clustering Approach to Identify Optimal Number of Clusters, Journal of Artificial Intelligence and Soft Computing Research, vol. 4, no. 1, Jan. 2014.

[26] Y. Shi and R. Eberhart, A modified particle swarm optimizer, 1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No. 98TH8360).

[27] D. Dua, and E. K. Taniskidou, UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science, 2017.

[28] F. Eibe, M. A. Hall, and I. H. Witten. The WEKA Workbench. Online Appendix for Data Mining: Practical Machine Learning Tools and Techniques, Morgan Kaufmann, Fourth Edition, 2016.