Contents lists available at SciVerse ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Novel swarm optimization for mining classification rules on thyroid gland data

Wei-Chang Yeh

Integration & Collaboration Laboratory, Advanced Analytics Institute, Faculty of Engineering and Information Technology, University of Technology Sydney, P.O. Box 123, Broadway, NSW 2007, Australia
Department of Industrial Engineering and Engineering Management, National Tsing Hua University, P.O. Box 24-60, Hsinchu 300, Taiwan, ROC

A R T I C L E   I N F O

Article history:
Received 24 June 2010
Received in revised form 31 October 2011
Accepted 10 February 2012
Available online 18 February 2012

Keywords:
Data mining
Simplified swarm optimization (SSO)
Classification rules
Thyroid gland data
Orthogonal array test (OAT)

A B S T R A C T

This work uses a novel rule-based classifier design method, constructed by using improved simplified swarm optimization (SSO), to mine a thyroid gland dataset from UCI databases. An elite concept is added to the proposed method to improve solution quality, close interval encoding (CIE) is added to efficiently represent the rule structure, and the orthogonal array test (OAT) is added to powerfully prune rules to avoid over-fitting the training dataset. To evaluate the classification performance of the proposed improved SSO, computer simulations are performed on well-known thyroid gland data. Computational results compare favorably with those obtained using existing algorithms such as conventional classifiers, including Bayes classifier, k-NN, k-Means, and 2D-SOM, and soft computing based methods such as the simple SSO, immune-estimation of distribution algorithms (IEDA), and genetic algorithm (GA).

1. Introduction

Data mining is an efficient approach for analyzing and discovering knowledge from a large complex dataset of heterogeneous quality, for which a variety of data mining tools have been developed. The rule-based classifier is one such important tool [1–7,12,13,17,20–22,26–28,30–40] which mines a small set of IF-THEN rules (e.g., IF condition THEN conclusion) from training data for classification with predicted classes, and then uses this rule set to predict new data instances [25]. The rule-based classifier has the advantage of generating high-level symbolic-knowledge representation, which increases the comprehensibility of discovered knowledge [1–7,12,13,17,20–22,26–28,30–40], and it has been extensively applied to many real-world problems in medicine, social sciences, management, and engineering [1–7,12,13,17,20–22,26–28,30–40].

Many conventional algorithms have been proposed for classification such as the Bayes classifier [18,19], k-NN, k-Means, and 2D-SOM (self-organizing map) [6,11,15,23]. Soft computing methods (SCs) have been utilized to find optimal or good quality solutions to complex optimization problems in a number of fields [4,5,8–10,13–22,24,30–40]. Consequently, many new data mining techniques have been based on SC, such as the genetic algorithm (GA, a biology-inspired SC) [5,14,31] and particle swarm optimization (PSO, a swarm-intelligence SC) [1,2,8,9,13,17,21,22,26,28,30,32–40].

Swarm-intelligence is an artificial intelligence, primarily inspired by the social behavior patterns of self-organized systems, that considers the interactions among large groups of individuals [1,2,8,9,13,17,21,22,26,28,30,32–40]. The simplified swarm optimization (SSO) proposed by Yeh is a population-based stochastic optimization technique that belongs to the swarm-intelligence category [37–39] and is also an evolutionary computational method inspired by PSO [37]. Also known as discrete PSO (DPSO), SSO was originally proposed to overcome the drawbacks of PSO for discrete-type optimization

E-mail address: yeh@ieee.org

0020-0255/$ - see front matter © 2012 Elsevier Inc. All rights reserved.
doi:10.1016/j.ins.2012.02.009

problems (e.g., the multi-level redundancy allocation in series systems [38]). Although simple SSO has been successfully applied to classify breast-cancer data [39], it is only valid for discrete data; hence, there is a need to extend simple SSO to efficiently and effectively solve this problem with more complex data.

This work has two principal goals. First, a modification is introduced to simple SSO to deal with non-discrete data by integrating close interval encoding (CIE), the elite concept, and the orthogonal array test (OAT) to establish the classification rules. To demonstrate its efficiency, the proposed algorithm is tested on thyroid gland data, a famous dataset in the UCI database [3,4,7,12,31].

This paper is organized as follows. Section 2 provides a description of simple SSO. The proposed CIE, the elite concept, and OAT are described in Sections 3–5, respectively. The proposed improved SSO combining CIE, the elite concept, and OAT are discussed in Section 6. Three comparisons based on two experiments on the UCI thyroid gland dataset demonstrate the effectiveness of the proposed improved SSO in Section 7. Finally, the conclusion is presented in Section 8.

2. Introduction to SSO

The advantages of SSO are its simplicity, efficiency, and flexibility [37–39]. In its early development, SSO was introduced by the author to solve the redundancy allocation problem by simplifying the traditional PSO procedure to overcome the drawbacks of PSO for discrete variables [37]. Since SSO was only valid for discrete data, it was initially called discrete particle swarm optimization (DPSO) but was later renamed SSO due to its simplicity of use and the introduction of a different schematic to update the solutions (called ‘particles’ in PSO). The basics of SSO are introduced in this section before discussing the proposed improved SSO.

Like most SCs such as GA, SA, ACO, PSO, or ABC, SSO is also initialized with a population of random solutions inside the problem space and it then searches for optimal solutions by updating generations. Let DIM, POP, GEN, and REP represent the number of attributes, populations, generations, and independent replications, while parameters c_w, c_p, c_g, and c_r represent the probabilities of the new variable value generated from the current solution, *pBest*, *gBest* and a random number in SSO, respectively, where $c_w + c_p + c_g + c_r = 1$. In the update mechanism of SSO, the *i*th solution at generation $t+1 X_i^{t+1} = \left(x_{i1}^{t+1}, x_{i2}^{t+1}, \ldots, x_{i,\mathrm{DIM}}^{t+1}\right)$ is a compromise of the current *i*th solution at generation t $X_i^t = \left(x_{i1}^t, x_{i2}^t, \ldots, x_{i,\mathrm{DIM}}^t\right)$; the personal best of the current *i*th solution (*pBest*) $P_i = (p_{i1}, p_{i2}, \ldots p_{i,\mathrm{DIM}}) \in \left\{X_i^1, X_i^2, \ldots, X_i^t\right\}$; the global best value of the whole swarm (*gBest*) $G = (g_1, g_2, \ldots, g_{\mathrm{DIM}}) \in \{P_1, P_2, \ldots, P_{\mathrm{POP}}\}$; and a random movement after $C_w = c_w$, $C_p = C_w + c_p$ and $C_g = C_p + c_g$ is given [37–39] as follows:

$$x_{ij}^{t+1} = \begin{cases} x_{ij}^t & \text{if } \rho \in [0, C_w) \\ p_{ij} & \text{if } \rho \in [C_w, C_p) \\ g_i & \text{if } \rho \in [C_p, C_g) \\ x & \text{if } \rho \in [C_g, 1) \end{cases} \tag{1}$$

where x_{ij}^t is the *j*th variable of the *i*th solution at generation *t*; ρ represents a random numbers uniformly distributed in [0, 1]; x represents random numbers uniformly distributed in $[l_j, u_j]$; and l_i, u_i are the lowest and greatest values of the *i*th attribute, respectively.

In Eq. (1), both concepts of *pBest* and *gBest* are adopted directly from the traditional PSO. Let $F(\bullet)$ be the fitness (function) value for $\bullet$. Then, P_i is a local best such that $F(P_i) \geqslant F\left(X_i^j\right)$ for $j = 1, 2, \ldots, t$, to push X_i^t to climb the local optimum (i.e., local exploitation), and G is a global best such that $F(G) \geqslant F(P_i)$ for $i = 1, 2, \ldots$, POP to guide the search towards unexplored regions to find the global optimizer (i.e., global exploration). To maintain population diversity and enhance the capacity of escaping from a local optimum, a random movement is added to the update mechanism of SSO [37–39].

The above update mechanism considers the interaction between individuals and maintains the diversity between them. It is much simpler than other major SC techniques such as PSO, which needs to calculate both velocity and position functions; GA, which requires genetic operations such as cross-over and mutation; estimation of distribution algorithm (EDA), which proves difficult and complicated for building an appropriate probability model; and IMA, which does not consider the interaction of variables [4]. The overall steps of SSO are listed as follows:

PROCEDURE SSO

```
STEP S0. Generate $X_i$ randomly, let $t = 1$, $P_i = X_i$, and $G = P_j$, where $F(P_j) = \underset{i}{Max}\{F(P_i)\}$ for $i = 1, 2, \ldots$, POP.
STEP S1. Let $i = 1$.
STEP S2. Update $X_i$ based on Eq. (1) and calculate $F(X_i)$.
STEP S3. If $F(X_i) > F(P_i)$, let $P_i = X_i$; else go to STEP S5.
STEP S4. If $F(P_i) > F(G)$, then let $G = P_i$.
STEP S5. If $i <$ POP, then let $i = i + 1$ and go to STEP S2.
STEP S6. If $t =$ GEN, then $G$ is the final solution and stop; otherwise let $t = t + 1$ and go to STEP S1.
```

A simple example of the procedure for updating a solution using SSO is illustrated below according to [37]. Let = (2.3,3.5,5.6,4.2,7.8), P_4 = (6.6,7.7,4.7,2.5,8), G = (5.4,3.1,5.2,4.5,8), (l_1,l_2,l_3,l_4,l_5) = (2,2,4,2.5,4), (u_1,u_2,u_3,u_4,u_5) = (7.5,8,7,5,8), ρ = (.94,.58,.37,.11,.79), and (C_w,C_p,C_g) = (.15,.40,.75). Since

Table 1
Summarize of the example for the update procedure in SSO.

i	1	2	3	4	5
l_i	2	2	4	2.5	4
u_i	7.5	8	7	5	8
$x_{4,i}^{20}$	2.3	3.5	5.6	4.2	7.8
p_{4i}	6.6	7.7	4.7	2.5	8
g_i	5.4	3.1	5.2	4.5	8
ρ_i	.94	.58	.37	.11	.79
$x_{4,i}^{21}$	7.1	3.1	4.7	4.2	8
Remark	$C_g < \rho_1$	$C_p < \rho_2 < C_g$	$C_w < \rho_3 < C_p$	$\rho_4 < C_w$	$C_p < \rho_5 < C_g$

$$C_g = .75 < \rho_1 = .94, \tag{2}$$
$$C_p = .40 < \rho_2 = .58 < C_g = .75, \tag{3}$$
$$C_w = .15 < \rho_3 = .37 < C_p = .40, \tag{4}$$
$$\rho_4 = .11 < C_w = .15, \tag{5}$$
$$C_p = .4 < \rho_5 = .79 < C_g = .75, \tag{6}$$

let represent random numbers uniformly distributed in [2,7.5], say 7.1, $x_{4,2}^{21} = g_2\ (= 3.1)$; $x_{4,3}^{21} = p_{43}\ (= 4.7)$, $x_{4,4}^{21} = x_{4,4}^{20}\ (= 4.2)$, and $x_{4,5}^{21} = g_5\ (= 8)$. Hence, $X_4^{21} = (7.1, 3.1, 4.7, 4.2, 8)$. The above procedure is shown in Table 1.

3. Close interval encoding and majority first search

A rule represents the range of each attribute that contains a related predicated class. Each rule is a "feasible solution" in the proposed improved SSO. The exhaustive rule-based ordering scheme is used, so that each instance is covered by at least one rule.

In the common rule-based classifier [4,5,38], each attribute is encoded by three cells (storage spaces). The first cell represents the selected attribute; the second cell represents the value of the selected attribute "equal to" (denoted by 0), "less than" (denoted by 1), or "greater than" (denoted by 2) the threshold of the selected attribute; and the last cell represents the threshold of the selected attribute. For example, the first, second, and third cells are "k", "1", and "4.5", if $a_k < 4.5$; where a_k is the value of the kth attribute. However, the first cell is redundant, since one can easily identify an attribute based on its position, and the above encoding can only represent half the range of an attribute value and is not valid for a constant value (e.g. $a_k = 4.5$) or limited range (e.g., $-1.2 \leqslant a_k \leqslant 4.5$).

CIE is a matrix of size $2 \times$ DIM and the first and second elements in each row, say the ith row, represent the lower bound and upper bound respectively of the range of the ith attribute in rules. Hence, the length of each solution equals double the total number of attributes used in a dataset. Additionally, each attribute can be a discrete, floating-point or characteristic type. Compared with the SC-based classifiers proposed in [4,5,38], the storage space of the proposed improved SSO is reduced from $3 \cdot \text{DIM} + 1$ to only $2 \cdot \text{DIM}$ and all data can be mined, not only the discrete data. For instance, Table 2 shows the difference between the encoding examples used in [4,5,38] and CIE.

Without loss of generality, the range of values in each interval is assumed to be ordered from the smallest to the largest value. Additionally, the left endpoint always equals or is less than the right endpoint in each interval; otherwise, both endpoints switch automatically and immediately. For example, let X_i represent the ith solution. Table 3 lists some instances in the thyroid gland dataset adopted from the UCI database.

$$X_i = \begin{bmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \\ \vdots & \vdots \\ x_{\text{DIM},1} & x_{\text{DIM},2} \end{bmatrix} = \begin{bmatrix} 92 & u_1 \\ 7.2 & 10.8 \\ l_3 & 2.3 \\ 1.0 & 1.9 \\ 3.3 & 4.8 \end{bmatrix} \tag{7}$$

As a result, only instances 145–147 fall into the range denoted by X_i (Table 3). The predicted class in each solution is set to the class with the largest number in the remaining instances in the training set, i.e., MFS. For example, the class of the ith instance y_i is 1, since class 1 comprises the majority of instances 145–147.

4. The elite concept

The entire dataset is divided randomly into a training set and a test set during data mining. For example, during the k-fold cross-validation method, which is used in this work, the dataset is divided into k mutually exclusive partitions of approximately equal size. One of the k folds, say T_f, is used as the test set and the other k-1 folds are combined to form a training set,

Table 2
The CIE vs. the encoding method listed in [2,3,29].

Description	The encoding method listed in [2,3,29]			CIE
	The (3i – 2)th cell	The (3i – 1)th cell	The (3i)th cell	The ith row
$x_i = \alpha$	i	0	α	$[\alpha,\alpha]$
$x_i \leqslant \alpha$	i	1	α	$[l_i,\alpha]$
$x_i \geqslant \alpha$	i	2	α	$[\alpha,u_i]$
$x_i \in [\beta,\alpha]$	N/A			$[\beta,\alpha]$
a_i is not in the rule				$[l_i,u_i]$

Table 3
The example of CIE and MFS.

Instances i	a_{1i}	a_{2i}	a_{3i}	a_{4i}	a_{5i}	y_i
144	**118**	12.2	**1.5**	**1.0**	2.3	1
145	**94**	**7.5**	**1.2**	**1.3**	**4.4**	**1**
146	**126**	**10.4**	**1.7**	**1.2**	**3.5**	**1**
147	**114**	**7.5**	**1.1**	**1.6**	**4.4**	**1**
148	**111**	11.9	**2.3**	0.9	**3.8**	1
149	**104**	6.1	**1.8**	0.5	0.8	1
150	**102**	6.6	**1.2**	**1.4**	1.3	1
151	**139**	16.4	3.8	**1.1**	−0.2	2
152	**111**	16	**2.1**	0.9	−0.1	2
153	**113**	17.2	**1.8**	**1.0**	0	2

Each bold value indicated that it is fallen into the related range denoted Eq. (7).

say S_f. After finding a related IF-THEN rule set in the ith replication, say R_{if}, to S_f, the following equation is implemented to calculate the accuracy for classifying T_f based on R_{if}.

$$r_{if} = \frac{|S(T_f, R_{if})|}{|T_f|}, \quad \text{where } f = 1, 2, \ldots, k. \tag{8}$$

The process is repeated until each fold has served as a test set, i.e., the process is repeated k times. The rule accuracy of the classifier in the qth replication is then calculated as follows:

$$r_q = \frac{\sum_{f=1}^{k} r_{if}|T_f|}{\sum_{f=1}^{k}|T_f|}. \tag{9}$$

The general process in all SC-based classifiers for each training set is described as follows:

STEP G0. Let $S_f = S - T_f$ and $Best = 1$, where $S = \bigcup_f T_f$ and $f = 1, 2, \ldots, k$.
STEP G1. Let $q = f = 1$.
STEP G2. Let the ordered IF-THEN rule set $R_{qf} = \varnothing$ and unlabel all records in S_f.
STEP G3. Perform a SC-based classifier, e.g., GA, PSO, the simple SSO, and the proposed improved SSO a certain number of generations to unlabeled records in S_f, e.g., 100 generations.
STEP G4. Add the best solution (i.e., the rule with the best fitness value), say X, in R_{qf}.
STEP G5. Label the records (in S_f) that fall within the range defined in X obtained in STEP G4.
STEP G6. Return to STEP G3 until no more rules can be added.
STEP G7. Calculate r_{qf} based on Eq. (8).
STEP G8. If $f < k$, then let $f = f + 1$ and return to STEP G2.
STEP G9. Calculate r_q based on Eq. (9) and let $Best = q$ and $R_f^* = R_{qf}$ for $f = 1, 2, \ldots, k$, if $r_q > r_{best}$.
STEP G10. If $q <$ REP, then let $q = q + 1$, $f = 1$, and return to STEP G2. Otherwise, r_{best} is the final accuracy and R_f^* is the associated rule set w.r.t., the fth fold for $f = 1, 2, \ldots, k$.

The elite concept simply selects the rule set that has the best accuracy in each disjointed fold rather than all records among all replications, and then unifies these rules as the final rule set. Thus, the final accuracy based on the elite concept is

$$r = \frac{\sum_{f=1}^{k} r_{Best_f f}|T_f|}{\sum_{f=1}^{k}|T_f|}, \tag{10}$$

where $r_{Best_f f} \geqslant r_{qf}$ for all $Best_f \in \{1,2,\ldots,\text{REP}\}$, $f = 1, 2, \ldots, k$, and $q = 1, 2, \ldots,$ REP.

The proposed elite concept can easily be integrated into the above general process by rewriting STEPs G0 and G6, deleting STEPs G8 and G9, and a new STEP G8 as follows:

STEP G0. Let $S_f = S - T_f$ and $Best_f = 1$, where $S = \bigcup_f T_f$ and $f = 1, 2, \ldots, k$.
STEP G1. Let $q = f = 1$.
STEP G2. Let the ordered set $R_{qf} = \emptyset$ and unlabel all records in S_f.
STEP G3. Perform a SC-based classifier (e.g., GA, PSO, the simple SSO, and the proposed improved SSO) a certain generations to unlabeled records in S_f (e.g., 100 generations).
STEP G4. Add the best solution (i.e., the rule with best fitness value), say X, in R_{qf}.
STEP G5. Label the records (in S_f) that fall within the range defined in X obtained in STEP G4.
STEP G6. Return to STEP G3 until no more rules can be added.
STEP G7. Calculate r_{qf} based on Eq. (8) and let $Best_f = q$ if $r_{qf} > r_{Best_f,f}$
STEP G8. If $f < k$, then let $f = f + 1$ and return to STEP G2.
STEP G9. If $q <$ REP, then let $q = q + 1$, $f = 1$, and return to STEP G2. Otherwise, calculate the accuracy based on Eq. (10) by letting $\lambda_f = Best_f$ for $f = 1, 2, \ldots, k$.

5. Orthogonal-array-testing rule pruning algorithm

The number of rules and attributes in each rule in a classification is an important issue, as a simple rule with few attributes can make the interpretation of mining results unchallenging. Rule pruning reduces the number of rules in mining results and/or attributes in each rule, and can avoid over-fitting the training dataset [1]. The general process for rule pruning for each rule is as follows:

STEP P0. Get a copy for current rule, say X.
STEP P1. Remove one attribute from X and let the new rule be X^*.
STEP P2. If $F(X) < F(X^*)$, then let $X = X^*$.
STEP P3. Go to STEP P1 until no more attribute can be removed.

After obtaining each rule in the proposed SSO, rule pruning is performed immediately to reduce the number of attributes. This work proposes orthogonal array testing (OAT, often called the Taguchi Method) as a rule pruning algorithm. The OAT is a special statistical design of experiments based on orthogonal array (OA) that studies the effects of several factors simultaneously to efficiently determine the best combination of factor levels to use in design problems rather than exploring all possible combinations of assignments. An OA is an array of numbers arranged in rows (tests/combinations) and columns (factors/variables) such that each column is statistically independent of other columns, *i.e.*, orthogonality. Algorithms for constructing OAs with various levels are found in [29]. The details of OAT are given as follows.

Since only five attributes exist and each attribute uses one interval in each instance in this work, *i.e.*, $\leqslant$10 factros, the class of the two-level OA $L_4(2^3)$, $L_8(2^7)$, and/or $L_{16}(2^{15})$ are employed in this work. Tables 4–6 present an example OA $L_4(2^3)$, $L_8(2^7)$, and/or $L_{16}(2^{15})$, respectively.

Table 4
OA $L_4(2^3)$.

h	α_{1h}	α_{2h}	α_{3h}
1	1	1	1
2	1	2	2
3	2	1	2
4	2	2	1

Table 5
OA $L_8(2^7)$.

h	α_{1h}	α_{2h}	α_{3h}	α_{4h}	α_{5h}	α_{6h}	α_{7h}
1	1	1	1	1	1	1	1
2	1	1	1	2	2	2	2
3	1	2	2	1	1	2	2
4	1	2	2	2	2	1	1
5	2	1	2	1	2	1	2
6	2	1	2	2	1	2	1
7	2	2	1	1	2	2	1
8	2	2	1	2	1	1	2

Table 6
OA $L_{16}(2^{15})$.

h	α_{1h}	α_{2h}	α_{3h}	α_{4h}	α_{5h}	α_{6h}	α_{7h}	α_{8h}	α_{9h}	$\alpha_{10,h}$	$\alpha_{11,h}$	$\alpha_{12,h}$	$\alpha_{13,h}$	$\alpha_{14,h}$	$\alpha_{15,h}$
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	1	1	1	1	1	1	1	2	2	2	2	2	2	2	2
3	1	1	1	2	2	2	2	1	1	1	1	2	2	2	2
4	1	1	1	2	2	2	2	2	2	2	2	1	1	1	1
5	1	2	2	1	1	2	2	1	1	2	2	1	1	2	2
6	1	2	2	1	1	2	2	2	2	1	1	2	2	1	1
7	1	2	2	2	2	1	1	1	1	2	2	2	2	1	1
8	1	2	2	2	2	1	1	2	2	1	1	1	1	2	2
9	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2
10	2	1	2	1	2	1	2	2	1	2	1	2	1	2	1
11	2	1	2	2	1	2	1	1	2	1	2	2	1	2	1
12	2	1	2	2	1	2	1	2	1	2	1	1	2	1	2
13	2	2	1	1	2	2	1	1	2	2	1	1	2	2	1
14	2	2	1	1	2	2	1	2	1	1	2	2	1	1	2
15	2	2	1	2	1	1	2	1	2	2	1	2	1	1	2
16	2	2	1	2	1	1	2	2	1	1	2	1	2	2	1

The OAT is given below to reduce each obtained solution (rule), say

$$X = \begin{bmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \\ \vdots & \vdots \\ x_{\mathrm{DIM},1} & x_{\mathrm{DIM},2} \end{bmatrix}. \tag{11}$$

PROCEDURE OAT

STEP A0. Let $i = j = 1$.
STEP A1. If $x_{j,1} > l_j$, let $J(i) = j$ and $i = i + 1$.
STEP A2. If $j <$ DIM, let $j = j + 1$ and go to STEP A1. Otherwise, let $j_1 = i$ and $j = 1$.
STEP A3. If $x_{j,2} < u_j$, let $J(i) = j$ and $i = i + 1$.
STEP A4. If $j <$ DIM, let $j = j + 1$ and go to STEP A3. Otherwise, let $c = i$ and $j_2 = c - j_1$.
STEP A5. Construct OA $L_H(2^F)$, where $H = 4$ and $F = 3$ if $c \leqslant 3$, $H = 8$ and $F = 7$ if $4 \leqslant c \leqslant 7$, or $H = 16$ and $F = 15$ if $8 \leqslant c \leqslant 15$, respectively.
STEP A6. Let $x_{j\beta,h} = x_{j\beta}$ for $j = 1, 2, \ldots,$ DIM, $\beta = 1,2$, and $h = 1, 2, \ldots, H + 1$.
STEP A7. If $\alpha_{jh} = 2$ and $j \in \{1,2,\ldots,j_1\}$, let $x_{J(j),1,h} = l_{J(j)}$, where $h = 1, 2, \ldots, H$.
STEP A8. If $\alpha_{jh} = 2$ and $j \in \{j_1+1,j_1+2,\ldots,c\}$, let $x_{J(j),2,h} = u_{J(j)}$, where $h = 1, 2, \ldots, H$.
STEP A9. Let

$$X_{(h)} = \begin{bmatrix} x_{11,h} & x_{12,h} \\ x_{21,h} & x_{22,h} \\ \vdots & \vdots \\ x_{\mathrm{DIM},1,h} & x_{\mathrm{DIM},2,h} \end{bmatrix}, \tag{12}$$

and calculate $F(X_{(h)})$.
STEP A10. Calculate $S_{j\beta} = \sum_{h\in H^*} F(X_{(h)})$, where $j = 1, 2, \ldots, c, \beta = 1,2$, and $H^* = \{h|\alpha_{jh} = \beta\}$.
STEP A11. If $S_{j2} > S_{j1}$ and $j \in \{1,2,\ldots,j_1\}$, let $x_{J(j),1,H+1} = l_{J(j)}$.
STEP A12. If $S_{j2} > S_{j1}$ and $j \in \{j_1+1,j_1+2,\ldots,c\}$, let $x_{J(j),1,H+1} = u_{J(j)}$.
STEP A13. Let.

$$X_{(H+1)} = \begin{bmatrix} x_{11,H+1} & x_{12,H+1} \\ x_{21,H+1} & x_{22,H+1} \\ \vdots & \vdots \\ x_{\mathrm{DIM},1,H+1} & x_{\mathrm{DIM},2,H+1} \end{bmatrix}, \tag{13}$$

and calculate $F(X_{(H+1)})$.
STEP A14. Let $X = X^*$, where $F(X^*)$ is the best one in $\{F(X_{(h)})|h = 1,2,\ldots,H + 1\}$. Ties are broken arbitrarily.

Table 7
An example for the OAT.[a]

h	α_{1h}	α_{2h}	α_{3h}	α_{4h}	α_{5h}	α_{6h}	α_{7h}	α_{8h}	$x_{11,h}$	$x_{21,h}$	$x_{41,h}$	$x_{51,h}$	$x_{22,h}$	$x_{32,h}$	$x_{42,h}$	$x_{52,h}$	$F(X_{(h)})$
1	1	1	1	1	1	1	1	1	92	7.2	1.0	3.3	10.8	2.3	1.9	4.8	3.0
2	1	1	1	1	1	1	1	2	92	7.2	1.0	3.3	10.8	2.3	1.9		3.0
3	1	1	1	2	2	2	2	1	92	7.2	1.0					4.8	0.0
4	1	1	1	2	2	2	2	2	92	7.2	1.0						0.0
5	1	2	2	1	1	2	2	1	92			3.3	10.8			4.8	3.0
6	1	2	2	1	1	2	2	2	92			3.3	10.8				3.0
7	1	2	2	2	2	1	1	1	92					2.3	1.9	4.8	3.0
8	1	2	2	2	2	1	1	2	92					2.3	1.9		3.0
9	2	1	2	1	2	1	2	1		7.2		3.3		2.3		4.8	4.0
10	2	1	2	1	2	1	2	2		7.2		3.3		2.3			4.0
11	2	1	2	2	1	2	1	1		7.2			10.8		1.9	4.8	3.0
12	2	1	2	2	1	2	1	2		7.2			10.8		1.9		3.0
13	2	2	1	1	2	2	1	1			1.0	3.3			1.9	4.8	3.0
14	2	2	1	1	2	2	1	2			1.0	3.3			1.9		3.0
15	2	2	1	2	1	1	2	1			1.0		10.8	2.3		4.8	4.0
16	2	2	1	2	1	1	2	2			1.0		10.8	2.3			4.0
S_{j1}	18	20	20	26	26	28	24	23									
S_{j2}	37	35	35	29	29	27	31	32									
17	2	2	2	2	2	1	2	2						2.3			3.0

Each bold value indicated that it is fallen into the related range denoted Eq. (7).
[a] The empty cell in each column denotes the corresponding value of the variable is equal to its upper- or lower-bound.

If OAT improves its current fitness function value, then the related change is performed and the current fitness function value is updated in STEP A14; otherwise, the solutions remain the same. The proposed OAT increases the quality of the obtained solution under reasonable execution time. For example, let X_i as define in Eq. (7). The analytical result for implementing the above procedure for X_i is as follows, where $F(X_{(h)})$ is defined in Section 7.4.

Since $F(X_{(10)}) = F(X_{(16)}) = 4.0 \geqslant F(X_{(h)})$ in Table 7 for $h = 1, 2, \ldots, 17$, and both $X_{(10)}$ and $X_{(16)}$ also include the less number of variables (*i.e.*, 3), X_i is updated to $X_{(10)}$ or $X_{(16)}$, say $X_{(10)}$, after pruning using the OAT, *i.e.*,

$$X_i = \begin{bmatrix} l_1 & u_1 \\ l_2 & 10.8 \\ l_3 & 2.3 \\ 1.0 & u_4 \\ l_5 & u_5 \end{bmatrix} \tag{14}$$

Notice that X_i has 8 variables (*i.e.*, x_{11},x_{21},x_{22},x_{32},x_{41},x_{42},x_{51}, and x_{52}) at 2 different settings (*i.e.*, 1 and 2) in Eq. (7). Hence, the OAT can reduce the number of tests from $2^8 = 256$ to $16 + 1 = 17$ in this example.

6. The proposed improved SSO

This section discusses other main components of the proposed improved SSO implemented for data mining. The details of the major changes, including how to encode the solution (i.e., CIE) and the proposed method for pruning based on OAT, are also described.

6.1. Initial population

All rules except the first are generated randomly in the initial population. Using a high-quality initial population reduces running time, hence each attribute of the first rule is set to the smallest value in the lower bound and the largest value in the upper bound level to guarantee that at least one rule has a non-zero fitness value to accelerate the convergence process in the proposed improved SSO.

6.2. The update procedure

The procedure for updating a rule is the most important part of SC. The proposed updated procedure has a significant role in exchanging information among rules and results in an effective combination of partial solutions in other rules and accelerates the search procedure as follows.

Set X_h and calculate $F(X_h)$, for $h = 1, 2, \ldots$, DIM, as follows:

$$x_{ij,\beta}^{t+1} = \begin{cases} x_{ij,\beta}^{t} & \text{if } \rho \in [0, C_w) \\ p_{ij,\beta} & \text{if } \rho \in [C_w, C_p) \\ g_{i,\beta} & \text{if } \rho \in [C_p, C_g) \\ x & \text{if } \rho \in [C_g, 1) \end{cases} \tag{15}$$

where $\beta = 1, 2$.

6.3. The stop criterion

The most commonly used stop criterion for SC techniques and data mining methods is to specify a maximum number of generations; this work adapts this criterion to terminate the proposed improved SSO. In the proposed improved SSO, the final solution (i.e., *gBest*) after the GENth generation is recorded as a rule and any instance not in the range of this rule is removed regardless of its class. The proposed improved SSO is then continued until all instances are in one class.

7. The thyroid gland data set and numerical examples

To evaluate its quality and performance for data mining, the proposed improved SSO is applied to a widely referenced real-world dataset called the thyroid gland data set, adopted from the UCI database [3,4,7,12,31]. This famous benchmark dataset has 215 instances, each of which has five features and one class (Classes 1, 2, and 3). All attributes are continuous and no values are missing. In total, 150 (69%), 35 (16.3%), and 30 (13.9%) instances belong to the normal (euthyroidism) class (Class 1), hyperthyroid class (Class 2), and hypothyroid class (Class 3) respectively in this data set. The problem is to determine whether a patient's thyroid belongs to the euthyroidism, hypothyroidism, or hyperthyroidism class. Table 8 shows the corresponding data.

The proposed improved SSO is implemented in *C* programming language on an Intel Pentium 2.6 GHz PC with 256 MB memory. The unit of running time is a CPU second.

In this work, three comparisons are based on two experiments (Ex1 and Ex2). In Ex1, which is based on a 3-fold cross-validation method used in [11], the roles of the proposed improved OAT and the elite concept are tested in the first comparison. The computational result of Ex1 also provides the comparison between the proposed improved SSO and the best-known conventional classifier [11] in the literature regarding the same problem, such as the Bayes classifier, k-NN, k-Means, and 2D-SOM [11] (Section 7.2) in the second comparison.

Ex2 is combined with the third comparison, which is based on a 10-fold cross-validation method as implemented in [4], and compares the proposed improved SSO with the best-known SC-based classifier, including IEDA and GA recently proposed in [4], where IEDA is a hybrid algorithm combining the main concepts of EDAs and immune system algorithms (ISAs).

7.1. The first comparison: test of the role of OAT and/or elite

Two SSO-based algorithms, SSO (the simple SSO proposed in [37–39]) and ISO (the proposed improved SSO without integration of the elite concept), are implemented. The SSO and ISO are renamed eSSO and eISO respectively if the elite concept is implemented. The notation $\bullet_{\mathrm{a,b,c}}$ denotes that POP = a, GEN = b, and REP = c for method •; e.g., $\mathrm{eISO}_{100,100,100}$, where method • can be SSO, eSSO, ISO, or eISO. In these experimental results, subscripts—max, min, avg, and std— represent the maximum, minimum, average, and standard deviations respectively; e.g., $R_{\max}$. Notation $\bullet(\alpha)$ represents the measure α of method •,

Table 8
The features of thyroid gland dataset [1,2,7,12].

Feature	Range	Notation
1. T3-resin uptake test	[65,144]	f_1
2. Total Serum thyroxin	[0.5,25.3]	f_2
3. Total serum triiodothyronine	[0.2,10]	f_3
4. Basal thyroid-stimulating hormone (TSH)	[0.1,56.4]	f_4
5. Maximal absolute difference of TSH value after injection of 200 *lg* of thyrotropin-releasing hormone	[−0.7,56.3]	f_5
Class[a]	{1,2,3}	y
Class		
1. The normal class	[65,144]	f_1
2. The hyperthyroid class	[0.5,25.3]	f_2
3. The hypothyroid class	[0.2,10]	f_3

[a] Class 1: normal, Class 2: hyperthyroid, Class 3: hypothyroid.

Table 9
Computational results for SSO and ISO.[a]

		The 1st fold				The 2nd fold				The 3rd fold				Overall	Elite
		Time	Rule	Tra	Tst	Time	Rule	Tra	Tst	Time	Rule	Tra	Tst		
SSO	Max	0.093	16	100.0	95.8	0.078	17	100.0	100.0	0.109	16	100.0	97.2	94.4	97.7
POP = 10	Min	0.000	5	96.5	79.2	0.000	6	96.5	84.7	0.000	5	97.2	81.7	86.0	
GEN = 100	Avg	0.022	8.55	98.6	87.8	0.029	10.41	98.3	93.6	0.026	9.01	98.8	90.0	90.5	
REP = 10^3	Std	0.013	2.06	0.7	2.9	0.015	2.48	0.7	3.0	0.013	2.37	0.8	3.5	1.8	
SSO	Max	0.078	13	100.0	94.4	0.094	13	99.3	100.0	0.078	12	100.0	97.2	94.4	97.2
POP = 10	Min	0.015	5	97.9	79.2	0.015	6	97.2	86.1	0.015	5	97.9	78.9	85.1	
GEN = 200	Avg	0.031	7.21	99.0	87.5	0.041	9.21	98.7	93.7	0.033	7.40	99.2	89.8	90.4	
REP = 10^3	Std	0.012	1.42	0.5	3.0	0.015	1.80	0.5	2.9	0.013	1.80	0.5	3.5	1.7	
SSO	Max	0.093	11	100.0	94.4	0.140	13	100.0	100.0	0.094	10	100.0	98.6	95.3	97.7
POP = 10	Min	0.015	4	97.9	80.6	0.016	5	97.2	86.1	0.015	5	97.9	78.9	84.2	
GEN = 300	Avg	0.040	6.65	99.1	87.3	0.050	8.07	98.9	93.9	0.042	6.72	99.3	89.4	90.2	
REP = 10^3	Std	0.012	1.32	0.4	3.0	0.019	1.73	0.6	2.7	0.013	1.28	0.5	3.7	1.9	
ISO	Max	0.327	9	100.0	98.6	0.327	10	100.0	100.0	0.281	9	100.0	98.6	97.7	99.1
POP = 10	Min	0.140	4	95.1	81.9	0.140	4	95.1	84.7	0.140	4	95.8	81.7	87.0	
GEN = 100	Avg	0.186	5.58	98.2	91.6	0.195	6.14	97.7	94.4	0.195	5.85	98.4	91.6	92.5	
REP = 10^3	Std	0.026	1.11	0.9	3.0	0.023	1.06	0.9	2.6	0.020	1.02	0.7	3.1	1.7	
ISO	Max	0.468	8	100.0	100.0	0.531	10	100.0	100.0	0.468	8	100.0	98.6	97.7	99.5
POP = 10	Min	0.280	4	95.8	80.6	0.281	4	95.8	80.6	0.281	4	97.2	81.7	87.9	
GEN = 200	Avg	0.349	4.99	98.7	92.4	0.368	5.67	98.3	94.3	0.373	5.32	98.9	92.0	92.9	
REP = 10^3	Std	0.032	0.86	0.7	2.7	0.034	0.90	0.7	2.4	0.031	0.78	0.7	3.1	1.6	
ISO	Max	0.468	8	100.0	98.6	0.499	8	100.0	98.6	0.484	8	100.0	98.6	97.2	98.6
POP = 10	Min	0.296	4	97.2	79.2	0.280	4	96.5	86.1	0.281	4	96.5	78.9	87.9	
GEN = 300	Avg	0.357	4.72	98.9	92.3	0.367	5.34	98.5	94.1	0.379	5.06	99.1	92.5	93.0	
REP = 10^3	Std	0.032	0.77	0.6	2.6	0.034	0.76	0.6	2.3	0.029	0.60	0.6	2.9	1.5	
ISO	Max	0.468	8	100.00	100.0	0.484	9	100.0	100.0	0.468	8	100.0	100.0	97.2	100.0
POP = 30	Min	0.280	4	96.50	83.3	0.280	4	96.5	84.7	0.281	4	96.5	80.3	87.0	
GEN = 100	Avg	0.350	4.83	98.77	92.2	0.363	5.52	98.4	94.2	0.374	5.20	99.0	92.2	92.9	
REP = 10^3	Std	0.030	0.78	0.67	2.6	0.031	0.82	0.7	2.4	0.030	0.74	0.7	3.2	1.6	
ISO	Max	1.404	7	100.00	97.2	1.357	6	100.0	98.6	1.466	6	100.0	98.6	96.3	98.1
POP = 100	Min	0.982	4	97.90	86.1	0.952	4	97.2	88.9	0.983	4	97.9	85.9	88.4	
GEN = 100	Avg	1.142	4.40	99.06	92.4	1.136	4.80	98.7	93.7	1.219	4.84	99.4	93.2	93.1	
REP = 100	Std	0.087	0.59	0.49	2.4	0.113	0.65	0.6	2.3	0.108	0.47	0.4	2.8	1.4	

[a] Time: the average running time; Rule: the number of obtained rules; Tra$_n$: The accuracy for the related training set; Tst$_n$: The accuracy for the related test set.

where method • can be SSO, eSSO, ISO, or eISO and measure α can be A (the accuracy), T (the running time), or N (the rule number), e.g., SSO ($R_{\max}$).

Moreover, $\alpha \prec (\approx) \beta$ means that method (factor) β is better than (no difference to) method (factor) α; *e.g.*, SSO $\prec$ eSSO $\prec$ ISO $\prec$ eISO, and POP $\approx$ GEN $\prec$ REP. For a fair comparison with the best-known convention classifiers [11], the 3-fold cross-validation method is used in Ex1.

From the above observations, SSO $\prec$ eSSO $\prec$ ISO $\prec$ eISO increase accuracy. Notably, after implementing the elite concept (see the computational result in the last column in Table 9), the accuracies of SSO and ISO increase by 2.83% and 1.53% respectively on average. Thus, both OAT and the elite concept improve accuracy at the accepted cost of increased run time.

Since ISO$_{30,100,1000}$ $\approx$ ISO$_{10,300,1000}$and eISO$_{30,100,1000}$ $\prec$ eISO$_{10,300,1000}$ increase accuracy, we have GEN $\approx$ POP when the elite concept is not used and GEN $\prec$ POP when the elite concept is used. Moreover, ISO$_{100,100,100}$ $\prec$ ISO$_{10,100,1000}$ and eISO$_{100,100,100}$ $\prec$ eISO$_{10,100,1000}$ increase accuracy; i.e., POP $\prec$ REP with or without the elite concept increase accuracy. Hence, we have POP $\approx$ GEN $\prec$ REP when the elite concept is not implemented and GEN $\prec$ POP $\prec$ REP when the elite concept is implemented.

The running time is < 0.5 s on average for each run which is a significant improvement in effectiveness (solution quality) without impairment of ISO efficiency. The high REP value is better than that of POP in reducing the running time of the classifier, since ISO$_{100,100,100}$ $\prec$ ISO$_{10,100,1000}$ in running time. No significant difference exists between the high values of GEN and POP in reducing the running time of the classifier, since ISO$_{30,100,100}$ $\approx$ ISO$_{10,300,1000}$. Hence, it is evident that GEN $\approx$ POP $\prec$ REP reduces the running time of the classifier.

The smallest number of rules is best, and high accuracy reduces the number of rules from Table 9. Thus, we have SSO $\prec$ eSSO $\prec$ ISO $\prec$ eISO, and POP $\approx$ GEN $\prec$ REP, if the elite concept is not implemented, and GEN $\prec$ POP $\prec$ REP if the elite concept is implemented in reducing the number of rules. Moreover, the maximal number of rules obtained from each ISO is <10, meaning that the number of rules can only be reduced to <10.

The fact that the values of the maximal, minimal, average, and standard deviations of the running times increase for increased numbers of GEN is trivial. However, the values of the maximal, minimal, average, and standard deviations of the rule

Table 10
3-Fold cross-validation method performance results of the test subsamples Correct%.

Method	Fold 1	Fold 2	Fold 3	Overall	With the elite concept
ISO	98.61	100.00	98.59	97.70	99.06
SSO	95.83	100.00	97.18	94.41	97.67
Bayes classifier	95.77	98.61	95.83	96.74	
k-NN	91.55	94.44	91.67	92.55	
k-Means	84.00	91.00	72.00	82.33	
2D-SOM	92.95	87.50	84.72	88.40	

Table 11
The average (%) type I and type II error of IEDA and GAs.

	Method	Max	Min	Avg	Std	Elite
Part 1	$ISO_{10,100,1000}$	99.46	94.35	97.81	1.32	100
	IEDA	98.39	93.55	97.78	1.86	
	GAs	95.16	83.87	90.80	4.24	
Part 2	$ISO_{10,100,1000}$	99.44	94.78	97.91	0.99	100
	IEDA	98.33	93.33	96.78	1.86	
	GAs	95.00	90.00	90.80	4.24	

number slightly decrease accordingly, but overall accuracy is not significantly improved; therefore, GEN = 100, POP = 10, and REP = 1000 are adapted in the remaining comparisons.

7.2. The second comparison: the proposed improved SSO vs. conventional classifiers

The second comparison based on the first experiment compares the proposed improved SSO and conventional classifiers. To the author's knowledge, the best-known conventional classifier is the Bayes classifier discussed in [11], and performance of the proposed improved SSO is compared with that of the Bayes classifier together with k-NN, k-Means, and 2D-SOM under the settings: GEN = 100, POP = 10, and REP = 1000.

Table 10 lists the classification accuracy of each method; all computational results for the Bayes classifier, k-NN, k-Means, and 2D-SOM are obtained directly from [11]. The proposed improved SSO also adopts 3-fold cross-validation as in [11]; however, the data in each fold of the proposed improved SSO are selected randomly and may differ from those in [11].

According to experimental results, the accuracy of the proposed improved SSO without applying the elite concept exceeds 97.7%, which is the best among all other methods listed in Table 10. Moreover, the accuracy of the proposed improved SSO increases to 99.06% after implementing the elite concept, demonstrating that the proposed improved SSO is generally more effective than the Bayes classifier, k-NN, k-Means, and 2D-SOM. Even the simple SSO proposed in [39] is better than the best-known conventional classifier; i.e., the Bayes classifier in Table 10 after implementing the elite concept. These experimental results demonstrate the validity of the SSO-based method.

7.3. The third comparison: the proposed improved SSO vs. the best-known SC-based classifiers

The third comparison based on the second experiment compares the SC-based classifier and the proposed improved SSO classifier (i.e., ISO) in terms of prediction accuracy. Using the same settings as in [4], the second experiment has two parts. The first part finds the classification rules after removing all hypothyroid data and the second part finds the classification rules after removing all hyperthyroid data.

Table 11 shows the computational results of the first and second parts. In both parts, the best, maximal, minimal, and standard deviations of accuracy of classification by ISO are all better than those of GAs and IEDA. Additionally, the accuracy by ISO integrated with the elite concept is 100% for both parts.

The results of the above experiments show that the proposed improved SSO is superior to the conventional approaches and SC-based classifiers in terms of prediction accuracy.

Computational results compare favorably with those of previously developed algorithms in the literature, including the conventional classifiers such as the Bayes classifier, k-NN, k-Means, and 2D-SOM classifiers [11], and the SC-based methods such as the simple SSO [37], IEDA and GA [4].

8. Conclusions and future research

Classification is an important task in data mining. This work describes a new IF-THEN rules-based classifier design method called the improved SSO. The proposed improved SSO is a modification of the simple SSO, which is only valid for discrete data, and uses new rule structures called CIE to reduce storage space for any data type, the OAT to prune rules, and the elite method to improve solution quality.

Three comparisons based on two experiments on the UCI thyroid gland dataset demonstrate the effectiveness of the proposed method. These comparisons reveal the efficacy of the proposed method in classifying real-world highly dimensional data. Computational results obtained by the new classifier in the first experiment are better than those of conventional classifiers in the literature such as the Bayes classifier, k-NN, k-Means, and 2D-SOM [11]. Moreover, the new classifier outperforms SC-based classifiers, including the simple SSO [37], GA, and IEDA [4] (a new hybrid method combining the main concepts of AIS [10] and EDA [24]) in the second experiment.

Although integration of the OAT, the elite concept, and different update mechanisms into the proposed improved SSO scheme in various ways is an ongoing process, preliminary results are promising. The thyroid gland data from the UCI databases includes only 215 instances and each instance has five features and one class (Classes 1, 2, and 3), which is too few to demonstrate the efficiency of the proposed algorithm. As a result, the limitations of our conclusions are obvious and there are several issues that merit future research. Future research will focus on strengthening the performance of the proposed improved SSO, extending the experimental datasets, and providing more solid experimental results and validating them through appropriate statistical tests.

References

[1] A. Abraham, S. Das, S. Roy, Swarm intelligence algorithms for data clustering, Soft Computing for Knowledge Discovery and Data Mining (2008) 279–313.
[2] B. Alatas, E. Akin, Multi-objective rule mining using a chaotic particle swarm optimization algorithm, Knowledge-Based Systems 22 (2009) 455–460.
[3] S. Albayrak, Unsupervised clustering methods for medical data: an application to thyroid gland data, Lecture Notes in Computer Science 2714 (2003) 695–701.
[4] W.-W. Chang, W.-C. Yeh, P.-C. Huang, A hybrid immune-estimation distribution of algorithm for mining thyroid gland data, Expert Systems with Applications 37 (2010) 2066–2071.
[5] T.-C. Chen, T.-C. Hsu, A GAs based approach for mining breast cancer pattern, Expert Systems with Applications 30 (2006) 674–681.
[6] Y. Chen, Y. Zhao, A novel ensemble of classifiers for microarray data classification, Applied Soft Computing 8 (2008) 1664–1669.
[7] D. Coomans, I. Broeckaert, M. Jonckheer, D.L. Massart, Comparison of multivariate discrimination techniques for clinical data—application to the thyroid functional state, Methods of Information in Medicine 22 (1983) 93–101.
[8] S. Das, A. Abraham, A. Konar, Particle swarm optimization and differential evolution algorithms: technical analysis, applications and hybridization perspectives, advances of computational intelligence in industrial systems, in: Y. Liu et al. (Eds.), Studies in Computational Intelligence, Springer Verlag, Berlin, 2008.
[9] S. Das, A. Abraham, A. Konar, Swarm intelligence algorithms in bioinformatics, Computational Intelligence in Bioinformatics 147 (2008) 113–147.
[10] D. Dasgupta, Advances in artificial immune systems, IEEE Computational Intelligence Magazine 1 (2006) 40–49.
[11] B. Diri, S. Albayrak, Visualization and analysis of classifiers performance in multi-class medical data, Expert Systems with Applications 34 (2008) 628–634.
[12] L.A. Gavin, The diagnostic dilemmas of hyperthyroxinemia and hypothyroxinemia, Advances in International Medicine 33 (1988) 185–204.
[13] K.R. Gandhi, M. Karnan, S. Kannan, Classification rule construction using particle swarm optimization algorithm for breast cancer data sets, in: Proceedings of the 2010 International Conference on Signal Acquisition and Processing, 2010, pp. 233–237.
[14] D.E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley, Reading, MA, 1989.
[15] S. Haykin, Neural Networks: A Comprehensive Foundation, Macmillan College Publishing, New York, 1994.
[16] S.J. Ho, S.Y. Ho, L.S. Shu, OSA: Orthogonal simulated annealing algorithm and its application to designing mixed H_2/H_∞ optimal controllers, IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans 34 (2004) 588–600.
[17] N. Holden, A.A. Freitas, A hybrid PSO/ACO algorithm for discovering classification rules in data mining, Journal of Artificial Evolution and Applications (2008), doi:10.1155/2008/316145.
[18] I. Inza, P. Larranaga, B. Sierra, Feature subset selection by Bayesian networks: a comparison with genetic and sequential algorithms, International Journal of Approximate Reasoning 27 (2001) 143–164.
[19] G.H. John, P. Langley, Estimating continuous distributions in Bayesian classifiers, in: Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence, San Mateo, 1995, pp. 338–345.
[20] J. Kennedy, R.C. Eberhart, Neural networks: particle swarm optimization, in: Proceedings of the IEEE International Conference, vol. 27, 1995, pp. 1942–1948.
[21] J. Kennedy, R.C. Eberhart, A discrete binary version of the particle swarm algorithm, in: Proceedings of IEEE International Conference on Systems, Man and Cybernetics, vol. 5, 1997.
[22] J. Kennedy, R.C. Eberhart, Y. Shi, Swarm Intelligence, Morgan Kaufmann, New York, 2001.
[23] T. Kohonen, Self-Organization and Associative Memory, Springer Verlag, Berlin Heidelberg New York, 1989.
[24] P. Larranaga, J.A. Lozano, Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation, Kluwer Academic Publishers, Norwell, MA, 2001.
[25] J. Leski, A fuzzy If-Then rule-based nonlinear classifier, International Journal of Applied Mathematics and Computer Science 13 (2003) 215–223.
[26] Y. Liu, Z. Qin, Z. Shi, J. Chen, Rule discovery with particle swarm optimization, Lecture Notes in Computer Science 3309 (2004) 291–296.
[27] S. Olafsson, X. Li, S. Wu, Operations research and data mining, European Journal of Operational Research 187 (2008) 1429–1448.
[28] Y. Shi, R.C. Eberhart, Evolutionary computation: empirical study of particle swarm optimization, in: Proceedings of the Congress on Evolutionary Computation, 1999, pp. 1945–1950.
[29] Y. Shi, H. Liu, L. Gao, G. Zhang, Cellular Particle Swarm Optimization 181 (20) (2011) 4460–4493.
[30] L. Tao, Y. Feng, C. Jianying, H. Weilin, Acquisition of classification rule based on quantum-behaved particle swarm optimization, Application Research of Computers 26 (2) (2009) 496–499.
[31] <http://www.thyroid.ca/Guides/HG01.html>.
[32] X.Z. Wang, Y.L. He, L.C. Dong, H.Y. Zhao, Particle swarm optimization for determining fuzzy measures from data, Information Sciences 181 (18) (2011) 4230–4252.
[33] Y. Wang, B. Li, T. Weise, J. Wang, B. Yuan, Q. Tian, Self-adaptive learning based particle swarm optimization, Information Sciences 181 (20) (2011) 4515–4538.
[34] Z. Wang, X. Sun, D. Zhang, Classification rule mining based on particle swarm optimization, Lecture Notes in Computer Science 4062 (2006) 436–441.
[35] I.H. Witten, E. Frank, Data Mining: Practical Machine Learning Tools and Techniques, second ed., Morgan Kaufman, San Francisco, 2005.
[36] L. Yan, J. Zeng, Using particle swarm optimization and genetic programming to evolve classification rules, in: Proceedings of the 6th World Congress on Intelligent Control and Automation, WCICA, 2006, pp. 3415–3419.

[37] W.C. Yeh, Study on quickest path networks with dependent components and apply to RAP, 2008/08/01$\sim$2011/07/31, Report, NSC 97-2221-E-007-099-MY3, 2010.
[38] W.C. Yeh, A two-stage discrete particle swarm optimization for the problem of multiple multi-level redundancy allocation in series systems, Expert Systems with Applications 36 (2009) 9192–9200.
[39] W.C. Yeh, W.W. Chang, Y.Y. Chung, A new hybrid approach for mining breast cancer pattern using discrete particle swarm optimization and statistical method, Expert Systems with Applications 36 (2009) 8204–8211.
[40] X. Zhao, J. Zeng, Y. Gao, Y. Yang, Particle swarm algorithm for classification rules generation, in: Proceedings of the 6th International Conference on Intelligent Systems Design and Applications, vol. 2, 2006, pp. 957–962.