2013 12th International Conference on Machine Learning and Applications

An Incremental Parallel Particle Swarm Approach for Classification Rule Discovery from Dynamic Data

Kaveh Hassani and Won-Sook Lee
School of Electrical Engineering and Computer Science
University of Ottawa
Ottawa, Canada
kaveh.hassani@uottawa.ca and wslee@uottawa.ca

***Abstract*— classification is a supervised learning technique that predicts the classes of unobserved data by employing a model built from available data. One of the efficient ways to represent this predictive model is to express it as an optimal set of classification rules to provide comprehensibility and precision, simultaneously. In this paper, we propose a novel incremental parallel Particle Swarm Optimization (PSO) approach for classification rule discovery. Our proposed method separates the training data into a set of data chunks regarding the classes and extracts optimal set of classification rules for each chunk in a parallel manner. In order to extract the rules from data chunks, we introduce an incremental PSO algorithm in which the previously extracted rules are directly employed to initialize the swarm population. Moreover, in each generation of the swarm, a tournament method is employed to substitute the weak individuals with strong extracted knowledge. To support the parallelism, we assign a PSO thread for each data chunk. As soon as all the PSO threads are completed, the extracted rules are integrated into a rule-base to construct a classification model. The evaluation results of the proposed approach on six datasets suggest that the classification precision of our proposed framework is competitive with offline learning methods and is 35% faster than its counterpart offline PSO approach.**

Keywords—classification; rule discovery; incremental learning; parallel computation; particle swarm optimization

I. INTRODUCTION

Data classification is the most applied supervised machine learning approach whose goal is to predict the class of unobserved data. It employs a set of available data samples and their classes to construct a model able to predict the classes of new data objects. Regarding knowledge representation scheme, data classification methods can be categorized as rule based and non-rule based approaches [1]. Rule based methods construct the classification model as a set of explicit rules. C4.5 decision tree [2] and RIPPER rule learner [3] are examples of rule based classifiers. Mostly, the rule based classification methods exploit a set of IF-THEN prediction rules which benefit from semantic symbolic knowledge representation and enhanced comprehensibility [4]. Antecedents of each rule are usually in conjunctive normal form (CNF) with features as literals, and the consequent is the predicted class. Fuzzy classifier is an other example of rule based classifiers that allows modeling of uncertainties and interpolation between rules [5,6]. Non-rule based approaches are black box models that represent the discovered knowledge implicitly. Although these methods tend to have high accuracy, they have low comprehensibility. Support vector machines [7] and Artificial Neural Networks (ANN) [8] are examples of non-rule based classification methods.

Moreover, regarding employed learning scheme, classifiers can be categorized to offline and incremental systems. In offline learning approach (i.e. batch training), it is assumed that the training data is static and completely available in the initial stage of the learning process. On the other hand, in incremental learning (i.e. online or adaptive learning), the system learns the model from dynamic data stream. In offline learning, if the training data changes after the learning process, the constructed model is invalidated and a new model is generated, whereas in incremental learning, the constructed model is modified in response to data changes. In addition to the fact that finding accurate and complete training data in the initial stage is difficult, big training data is not memory resident (i.e. page swapping overload) which leads to slow model construction and storage space problems [9]. In real world applications, the training data can change in three possible ways including: availability of new training data, availability of new features, and availability of new classes [10].

Extracting optimal set of classification rules from a data set is an NP-complete problem [11], and its complexity increases significantly with dynamic training data. There are three possible options for extracting these rules. The first option is to construct a model by employing non-rule based approaches and then extract set of rules from the model. Although this approach provides accurate results, it is not efficient (e.g. generating symbolic rules from a neural network is an NP-hard problem [12]). The second option is to utilize greedy rule based approaches such as decision trees. Although these methods demonstrate high performance, they tend to get trapped in locally optimal solutions. Finally, the third option is to exploit meta-heuristic methods. Although most of these methods suffer from low convergence speed, they are able to reach the globally optimal solutions.

Particle Swarm Optimization (PSO) algorithm introduced by Eberhart and Kennedy [13] is a swarm intelligence (SI) based meta-heuristic that imitates the individual and social behavior of flocks of birds to find the globally optimal solutions within the problem hyperspace. Elbeltagia, Hegazyb, and Griersonb compared the performance of five evolutionary and SI algorithms including genetic algorithm (GA), memetic algorithm (MA), ant colony optimization (ACO), shuffled frog

978-0-7695-5144-9/13 $31.00 © 2013 IEEE
DOI 10.1109/ICMLA.2013.87

leaping algorithm (SFLA), and PSO in solving benchmark problems. They concluded that the PSO method generally outperforms other algorithms in terms of success rate, solution quality and processing time [14]. Moreover, it is shown that the PSO algorithm provides more successful results than the artificial bee colony (ABC) algorithm [15]. Prado, Galan, Exposito and Yuste proved that PSO achieves a faster convergence than the GA [16]. Finally, Akay experimentally showed that the PSO is scalable and its processing time grows at a linear rate with respect to the size of the problem [17].

In this paper, we propose a novel incremental parallel PSO (IPPSO) approach for classification rule discovery based on the methods proposed in [18,19]. Our proposed method employs a separate and conquer approach [20] to separate the training data into a set of data chunks regarding the classes and extracts optimal set of classification rules for each chunk in a parallel manner. To extract the rules from data chunks, we introduce an incremental PSO algorithm in which the previously extracted rules are directly employed to initialize the swarm population. Moreover, in each generation, a tournament selection method is employed to substitute the weak individuals with strong extracted knowledge. In order to support the parallelism, we assign a PSO thread for each data chunk. As soon as all the PSO threads are completed, the extracted rules are integrated into a rule-base to construct a classification model. The paper is organized as follows: in section 2 an overview of related work is presented. In section 3, we describe our proposed incremental parallel PSO-based approach. In section 4, we discuss the evaluations and experimental results. Finally, section 5 concludes the paper.

II. RELATED WORK

Many studies have employed SI techniques for classification rule discovery. Ant-miner [21], ABC-miner [22] and MOPSO-P [23] are examples of classifiers that utilize ACO, ABC and PSO, respectively. Also, many classification rule mining algorithms are constructed based on evolutionary approaches such as GA [24], SFLA [25], MA [26], genetic programming (GP) [27] and gene expression programming (GEP) [28]. GA and PSO are the most frequent biologically inspired algorithms applied for classification rule mining. Regarding individual representation, they are categorized to Michigan and Pittsburgh approaches. In the Michigan approach each individual represents a single rule, whereas in the Pittsburgh approach each individual encodes a set of rules [29]. As far as the authors' knowledge is concerned, proposed methods in literature mostly employ Michigan approach.

Au, Chan, and Yao proposed data mining by evolutionary learning (DMEL) algorithm to handle classification problem. Their proposed method initializes the population by a probabilistic induction technique. The initial population consists of a set of first-order rules which are expanded iteratively [24]. Chan, Chiang, and Fu introduced a two-phase multi-objective evolutionary algorithm which first searches decent rules and then produces the final rule sets regarding rule interactions [1]. Guan and Zhu employed GA as a basic learning algorithm for incremental learning within classifier agents in a multi-agent environment [10]. Bakirli, Birant, and Kut devised an incremental GA for classification. Their proposed approach initializes the population by extracted knowledge from previous training steps [18]. Dehuria, Patnaika, Ghoshb, and Mallc presented an elitist multi-objective GA (EMOGA) for mining classification rules from large databases by emphasizing on predictive accuracy, comprehensibility and interestingness of the rules [30]. Lu, Yang, Li, and Wang proposed a multi-objective evolutionary algorithm called improved niched Pareto genetic algorithm (INPGA) for mining multi-objective rules from large databases [31]. Another multi-objective evolutionary-based method for mining classification rules is proposed in [32].

Although GA is a well-founded and frequently applied algorithm with high exploration capability, it suffers from two pitfalls: low exploitation and convergence speed. On the other hand, PSO is able to reach the globally optimal solution within a few iterations. In literature, it has been experimentally shown that PSO-based classification outperforms ACO (e.g. Ant-Miner), GA (e.g. ESIA and OCEC), estimation of distribution algorithm (EDA), artificial immune systems (AIS), self-organizing map (e.g. 2D-SOM), multi-layer perceptron (MLP), radial basis function ANN (RBF), k-nearest neighbor algorithm (k-NN), k-Means, KStar, Bagging, Multi-BoostAB, naive Bayes tree (NBTree), ripple down rule (Ridor), voting feature interval (VFI), C4.5 decision tree, and Bayes Net classifiers [4, 33-36]. Furthermore, Sousa, Silva, and Neves suggested that discrete PSO (DPSO) outperforms constricted PSO (CPSO), linear decreasing weight PSO (LDWPSO), GA and C4.5 in classification tasks with nominal data [37].

Some research works have employed multi-objective PSO (MOPSO) for classification rule mining to satisfy incommensurable and conflicting criteria such as predictive accuracy and comprehensibility [33, 38-39]. MOPSO-P [23] and MOPSO-RL [40] are examples of MOPSO classifiers. Connolly, Granger, and Sabourin proposed an incremental learning strategy based on aggregated dynamical niching PSO (ADNPSO) to evolve heterogeneous classifier ensembles in response to new reference data. They applied their proposed framework for real time video face recognition [41]. Zhao, Zeng, Gao, and Yang employed a PSO algorithm to extract an optimal group of fuzzy classification rules [42]. Chen and Ludwig devised a PSO-based discrete implementation with a local search strategy (DPSO-LS) to find the best possible classification model [43]. Moreover, some research works have introduced hybrid PSO methods for classification rule mining. Hybrid PSO-ACO [44], immune-based PSO [9] and rough set-based PSO [45] are examples of hybrid classification rule discovery methods introduced in literature.

III. PROPOSED CLASSIFICATION APPROACH

Our proposed framework for incremental and parallel classification rule discovery is shown in Fig. 1. This framework consistently receives training data chunks and modifies its classification model in a way that it can address both old and new data. When a new training data chunk arrives, a fast preprocessing step refines the data and splits it into smaller chunks regarding the class each data object belongs to. Then, for each split data sub-chunk, a thread is assigned whose core function implements an IPPSO algorithm. By employing this method, we concurrently extract

classification rules for all classes. Once all threads are stopped, the resulted rules are directly extracted from fittest individuals, and then are integrated as a classification model. During the rule extraction process, the extracted rules from previous data chunks are directly injected into IPPSO algorithm. This mechanism supports the individuals with a compact knowledge that represents the data seen so far, and let them to modify it in a way that it can represent both old and new data. Finally, the extracted rules are pruned regarding their support count (i.e. those rules whose support count is less than minimum support count are removed). Furthermore, the old and new data are accumulated in a repository and the accuracy of the extracted model is tested by classifying data samples from that repository. In other words, we construct the model by new data and test its performance by both old and new data.

A. Individual Representation

We employ Michigan approach for the individual representation where each individual represents a single rule. Furthermore, we utilize bitmap indexing approach for rule encoding, which assigns a bit for each possible value of the feature. In this encoding, if all the bits of a feature equal to 0 or 1, the feature will be considered as a "do not care". As an example, suppose that training data consists of two features: A_1 and A_2. The first feature consists of three possible values $\{a_{11}, a_{12}, a_{13}\}$, and the second feature has two distinct values $\{a_{21}, a_{22}\}$. In this case, each individual will consist of five bits where the first three bits represent the first feature and the last two bits encode the second feature. Thus, a binary string such as 10101 will represent the rule shown in (1).

$$IF\ (A_1=a_{11}\ OR\ A_1=a_{13})\ AND\ (A_2=a_{22})\ THEN\ Class=C_i \qquad (1)$$

B. IPPSO Algorithm

Our proposed PSO algorithm, shown in Fig. 2, is a binary PSO. The first step in this algorithm is to initialize the swarm. We have adopted the method introduced in [18] to populate the initial swarm. In this method, instead of initializing the swarm randomly, we employ the extracted rules to generate individuals. To do so, we apply a roulette wheel to select which rules to be copied to the individuals. Those rules that have higher support count (i.e. number of instances they cover) have better chances of being overwritten in initial population. After initializing the swarm, the optimization loop begins. The first method within the loop evaluates the particles. The particle's fitness equals to well-known F_β measure as shown in (2).

$$F_\beta = \frac{(1+\beta^2)\times precision \times recall}{\beta^2 \times precision + recall} \qquad (2)$$

where β determines the precision and recall weights. Classification precision and recall (i.e. sensitivity) criteria are computed by (3).

$$precision = \frac{TP}{TP+FP} \quad , \quad recall = \frac{TP}{TP+FN} \qquad (3)$$

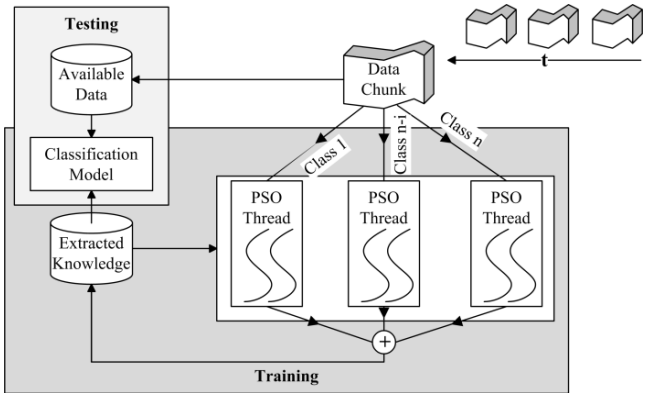

Figure 1. Our proposed framework for classification rule discovery by employing incremental parallel PSO algorithm.

where *TP* refers to the number of positive data objects that are correctly classified, *FP* is the number of negative instances that are incorrectly classified, and *FN* refers to positive objects that are mislabeled as negative.

As soon as the evaluation process is completed, the particles' velocity and position vectors are updated. We applied binary PSO algorithm introduced in [19] for this purpose which exploits a probabilistic updating model. It computes the probability of changing a bit in the position vector based on its corresponding value in personal and global best particles. Intuitively, the probability of changing a bit to 0 is high, if the corresponding bit in both global and personal bests is 0.

We have added a new function to the conventional PSO algorithm to reinforce the swarm. This function utilizes a tournament selection method to select one candidate particle from swarm and one candidate rule from classification model constructed previously. In order to select the candidate particle, it randomly selects *m* particles (i.e. *m* is fairly smaller than swarm size) and determines the weakest (i.e. lowest fitness) particle among selected particles as the candidate. On the other hand, it uses same method to select the candidate rule. However, it defines the candidate rule as the rule with greatest support count among the selected rules. Finally, it overwrites the selected rule on the selected particle to reinforce the swarm.

Stochastic characteristic of the tournament selection prevents the algorithm from getting trapped in local optimums and enhances the convergence speed and accuracy of the classification model. When the reinforcement phase is completed, the best personal and global vectors are updated.

```
1.  Initialize swarm (Extracted Rules)
2.  While (NOT satisfactory fitness)
3.  {
4.      Evaluate Swarm ( )
5.      Update Velocity ( )
6.      Update Position ( )
7.      Reinforce Swarm ( )
8.      Update Personal Bests ( )
9.      Update Global Best ( )
10. }
11. Return Global Best ( )
```

Figure 2. Proposed PSO algorithm for incremental learning.

IV. EXPERIMENTAL RESULTS

In order to evaluate the proposed method, we conducted two experiments. The goal of the first experiment is to evaluate the accuracy of classifier constructed by PSO algorithm in offline training mode and comparing it to the well-known classifiers. To do so, six datasets are utilized from UCI machine learning repository. The characteristics of datasets are shown in Table I. These datasets present a proper distribution of different characteristics. We employed WEKA data mining software for data preprocessing (i.e. removing noise, replacing missing values, data normalizing, and discretizing-- five bins with equal frequency). After preprocessing the datasets, we used our PSO algorithm implemented in visual C#.Net to extract the classification rules from each of the datasets. The PSO parameters are set based on expert knowledge as follows: the social and personal parameters (c_1 and c_2) are set to 2, number of particles is set to 10, number of iterations is set to 1000, and the inertia boundaries are set to 0.1 and 0.9. Finally, min support count is set to 3% of the dataset size, and the β element is set to 2. We employed six well-known classifies from WEKA software to classify the same datasets. The parameters of these classifiers are set on default values of the WEKA software. In the second experiment, we employed our proposed parallel framework to incrementally construct the classification models for the selected datasets. We implemented two programs in visual C#.Net for incremental parallel classification rule discovery. In the first program, IPPSO-I, we implemented the system without swarm reinforcing step (i.e. Step 7 in Fig. 2 is removed from the algorithm). IPPSO-I is the PSO-based version of the incremental genetic algorithm proposed in [18]. In the second program, IPPSO-II, we implemented our proposed method. The values of the parameters except for the iteration number are the same as those in offline implementation. In IPPSO-I and IPPSO-II, we set the number of iterations to 100, and the number of selected candidates for tournament selection (m) to 3. In order to provide these two programs with training data chunks, a data streaming simulation program is implemented in visual C#.Net. This program splits the dataset to a few data chunks considering the predetermined window size by using a stratified sampling method. Due to stratified sampling, the distributions of classes in data chunks are similar and thus there is no concept drift in the stream. The data stream simulator sends the training data chunks with predefined frequency to the IPPSO-I and IPPSO-II programs via a socket. We employed IPPSO-I and IPPSO-II to extract classification rules from the benchmark datasets with four different data window sizes: 5%, 10%, 20%, and 30% of the dataset size. We repeated this process for 10 times. The average classification precision for Tic-Tac-Toe dataset extracted by IPPSO-I and IPPSO-II programs are shown in Fig. 3. As shown, our proposed method is approximately 20% more accurate than the incremental approach introduce in [18]. Furthermore, as shown in Fig. 3, increasing the window size leads to an increase in precision. Finally, to evaluate the performance of the applied parallelism in our proposed approach, we computed the average processing time for both incremental and offline learning on a computer with a 3.9GHz 64-bit processor and 8GB3 dual channel DDR3 SDRAM at 1600MHz memory. Comparison made between the comprehensibility, precision and average processing time of the offline PSO-based classifier, IPPSO and other classifiers are shown in Tables II, III and IV, respectively. The acquired results verify the experimental results reported in [4, 34-38]. A comparison between average processing time for parallel incremental method and serial offline method suggests that our parallel incremental approach is 35% faster than serial approach. Finally, it can be concluded that our proposed method is competitive with offline learning in terms of precision.

V. CONCLUSION

In this paper, we introduced a novel incremental parallel PSO-based classification rule discovery framework. Our proposed framework supports parallelism by assigning a PSO thread for each class in the training data. Also, it supports the incremental learning by guided initialization and reinforcing swarm in each generation by employing available knowledge. The results suggest that the classification precision of our proposed framework is competitive with offline learning methods and is 35% faster than them.

TABLE I. CHARACTERISTICS OF THE BENCHMARK DATASETS

Characteristics	**Dataset**					
	Nursery	***Tic-Tac-Toe***	***Breast Cancer***	***Iris***	***Statlog (Heart)***	***Thyroid***
#Instances	12960	958	569	150	270	3163
#Classes	5	2	2	3	2	2
#Features	8	9	31	4	12	25

TABLE II. A COMPARISON BETWEEN PSO AND OTHER CLASSIFIERS REGARDING NUMBER OF RULES

Classifier	**Dataset**					
	Nursery	***Tic-Tac-Toe***	***Breast Cancer***	***Iris***	***Statlog (Heart)***	***Thyroid***
J. 48 (C4.5) decision tree	359	95	29	13	24	38
JRIP rule learner (RIPPER)	127	9	12	**5**	6	**2**
Binary PSO	**104**	**7**	**9**	8	**5**	**2**
IPPSO	106	9	13	10	8	5

TABLE III. A COMPARISON BETWEEN PSO AND OTHER CLASSIFIERS REGARDING CLASSIFICATION PERCISION

Classifier	Dataset					
	Nursery	*Tic-Tac-Toe*	*Breast Cancer*	*Iris*	*Statlog (Heart)*	*Thyroid*
J. 48 (C4.5) decision tree	0.970	0.849	0.915	0.940	0.778	0.974
Bagging	0.972	0.921	0.941	0.913	0.782	0.979
SMO (SVM)	0.931	0.984	0.939	0.927	0.815	0.977
MLP	**0.998**	0.967	0.944	0.888	0.792	0.976
Naïve Bayesian	0.906	0.682	0.947	0.913	**0.844**	0.980
JRIP rule learner (RIPPER)	0.970	0.978	0.891	0.915	0.777	**0.979**
Binary PSO	0.970	**0.995**	**0.956**	**0.952**	0.800	0.976
IPPSO	0.963	0.983	0.944	0.932	0.768	0.968

TABLE IV. A COMPARISON BETWEEN PSO AND OTHER CLASSIFIERS REGARDING RUN TIME (IN MILLISECONDS)

Classifier	Dataset					
	Nursery	*Tic-Tac-Toe*	*Breast Cancer*	*Iris*	*Statlog (Heart)*	*Thyroid*
J. 48 (C4.5) decision tree	30	10	**10**	50	70	10
Bagging	260	40	50	6	20	90
SMO (SVM)	25790	230	170	20	50	310
MLP	92060	5680	86850	610	3950	63920
Naïve Bayesian	**10**	**6**	**10**	**4**	**6**	**7**
JRIP rule learner (RIPPER)	27220	40	50	8	10	50
Binary PSO	45120	6540	74150	5110	62140	53480
IPPSO	29140	4180	49260	3220	40430	33520

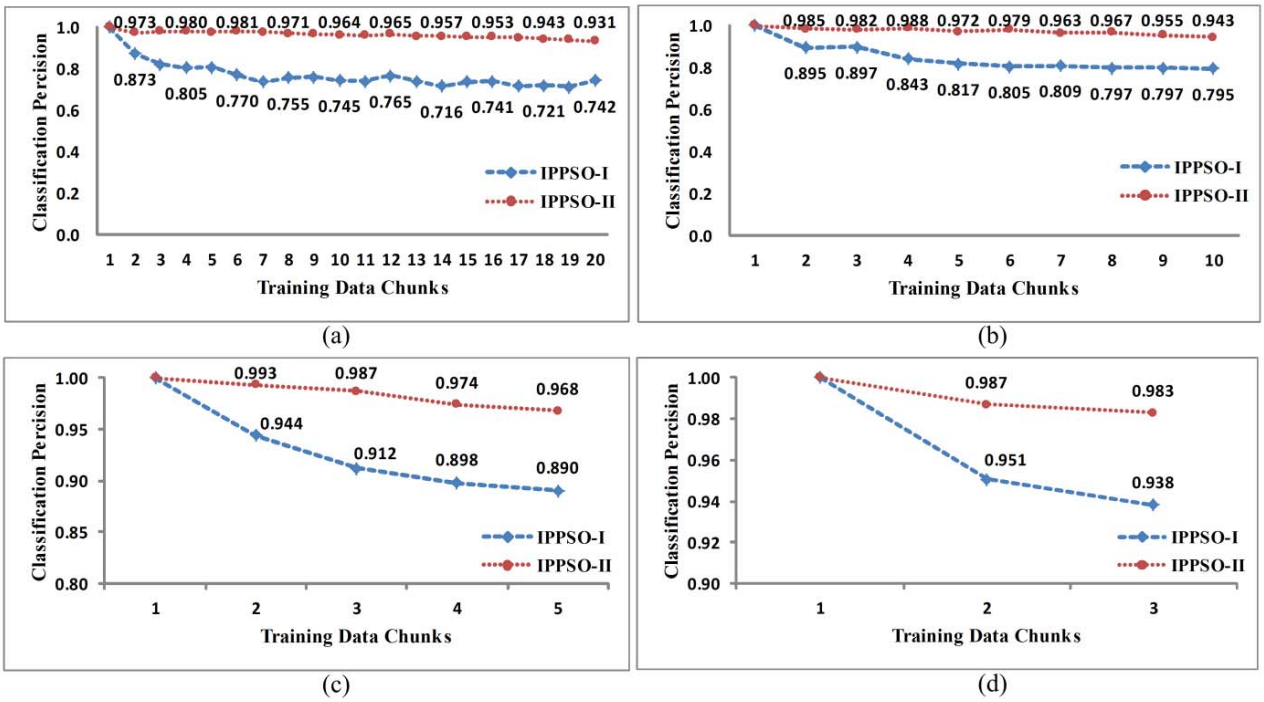

Figure 3. A comparison between the precision of the two introduced incremental parallel PSO classifiers with different windows sizes (Tic-Tac-Toe Dataset). (a) window size = 5% of dataset size, (b) window size = 10% of dataset size, (c) window size = 20% of dataset size, and (d) window size = 30% of dataset size.

REFERENCES

[1] Y. Chan, T. Chiang, and L. Fu, "A two-phase evolutionary algorithm for multiobjective mining of classification rules", in *2010 IEEE Cong. Evolutionary Computation*, 2010, pp. 1–7.

[2] R. Quinlan, *C4.5: Programs for Machine Learning*. San Francisco, CA: Morgan Kaufmann, 1993.

[3] W. Cohen, "Fast effective rule induction", in *Proc. Int. Con. on Machine Learning*, 1995, pp. 115–123.

[4] Z. Wang, X. Sun, and D. Zhang, "A PSO-based classification rule mining algorithm", in *Proc. 3rd Int. Con. Intelligent Computing*, 2007, pp. 377–384

[5] E. Lughofer, *Evolving Fuzzy Systems --- Methodologies, Advanced Concepts and Applications*. Berlin Heidelberg: Springer, 2011.

[6] E. Lughofer, O. Buchtala, "Reliable all-pairs evolving fuzzy classifiers", IEEE Trans. Fuzzy Syst., vol. 21 (4), pp. 625–641, Aug. 2013.

[7] B. Waske and J. Benediktsson, "Fusion of support vector machines for classification of multisensor data", *IEEE Trans. Geosci. Remote Sens.*, vol. 45, pp. 3858–3866, Dec. 2007.

[8] G. Zhang, "Neural networks for classification: a survey", *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 30, pp. 1094–6977, Nov 2000.

[9] X. Zhou, J. Yuyu, R. Qi, F. Qian, and Z. Wang, "IPSO: An immune based PSO supervised learning system for incremental learning", in *8th Cong. on Intelligent Control and Automation*, 2010, pp. 2707–2711.

[10] S. Guan and F. Zhu, "An incremental approach to genetic-algorithms-based classification", *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 35, pp. 227–239, April 2005.

[11] T. Andersen and T. Martinez, "NP-completeness of minimum rule sets", in *Proc. 10th Int. Symp. Computer and Information Sciences*, 1995, pp. 411–418.

[12] M. Golea, "On the complexity of rule extraction from neural networks and network querying", in *Proc. Rule Extraction from Trained Artificial Neural Networks Workshop*, 1996, pp.51–59.

[13] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory", in *Proc. 6th Symp. Micro Machine and Human Science*, 1995, pp. 29–43.

[14] E. Elbeltagia, T. Hegazyb, and D. Griersonb, "Comparison among five evolutionary-based optimization algorithms", *Adv. Eng. Inform.*, vol. 19, pp. 43–53, Jan. 2005.

[15] P. Civicioglu and E. Besdok, "A conceptual comparison of the Cuckoo-search, particle swarm optimization, differential evolution and artificial bee colony algorithms", *AI Rev.*, vol. 39, pp. 315–346, April 2013.

[16] R. Prado, S. Galan, J. Exposito and A. Yuste, "Knowledge acquisition in fuzzy-rule-based systems with particle-swarm optimization", *IEEE T. Fuzzy Systems*, vol. 18, pp. 1083–1097, Dec. 2010.

[17] B. Akay, "A study on particle swarm optimization and artificial bee colony algorithms for multilevel thresholding", *Appl. Soft. Comput.*, vol. 13, pp. 3066–3091, June 2013.

[18] G. Bakirli, D. Birant, and A. Kut, "An incremental genetic algorithm for classification and sensitivity analysis of its parameters", *Expert Syst. Appl.*, vol. 38, pp. 2609–2620, March 2011.

[19] M. Khanesar, M. Teshnehlab, and M. Shoorehdeli, "A novel binary particle swarm optimization", in *Mediterranean Con. on Control & Automation*, 2007, pp. 1–6.

[20] F. Stahl and M. Bramer, "Computationally efficient induction of classification rules with the PMCRI and J-PMCRI frameworks", *Knowl. Based. Syst.*, vol. 35, pp. 49–63, Nov. 2012.

[21] K.Salama, A.Abdelbar, and A. Freitas, "Multiple pheromone types and other extensions to the Ant-Miner classification rule discovery algorithm", *Swarm Intell.*, vol. 5, pp. 149–182, June 2011.

[22] M. Celik, D. Karaboga, and F. Koylu, "Artificial bee colony data miner (ABC-Miner)", in *Int. Symp. Innovations in Intelligent Systems and Applications*, 2011, pp. 96–100.

[23] A. Carvalho and A. Pozo, "Mining rules: A parallel multiobjective particle swarm optimization approach", in Swarm Intelligence for Multi-objective Problems in Data Mining, C. Coello, S. Dehuri, and S. Ghosh, Eds. New York: Springer, 2009, pp. 179–198.

[24] W. Au, K. Chan, and X. Yao "A novel evolutionary data mining algorithm with applications to churn prediction", *IEEE Trans. Evol. Comput.*, vol. 7, pp. 532–545, Dec. 2003.

[25] H. Yin, F. Cheng, and C. Zhou, "An efficient SFL-based classification rule mining algorithm", in *IEEE Int. Symp. IT in Medicine and Education*, 2008, pp. 969–972.

[26] X. Sun and Z. Wang, "An efficient MA-based classification rule mining algorithm", in *Int. Con. Computer Science and Software Engineering*, 2008, pp. 702–705.

[27] I. Falco, A. Cioppab, and E. Tarantinoa, "Discovering interesting classification rules with genetic programming", *Appl. Soft. Comput.*, vol. 1, pp. 257–269, May 2002.

[28] S. Dehuri and S. Cho, "Multi-objective classification rule mining using gene expression programming", in *3rd Int. Con. Convergence and Hybrid Information Technology*, 2008, pp. 754–760.

[29] A. Freitas, "A survey of evolutionary algorithms for data mining and knowledge discovery", in Advances in Evolutionary Computation, A. Ghosh, S. Tsutsui, Eds. New York: Springer, 2001, pp. 819–845.

[30] S. Dehuria, S. Patnaika, A. Ghoshb, and R. Mallc, "Application of elitist multi-objective genetic algorithm for classification rule generation", *Appl. Soft. Comput.*, vol. 8, pp. 477–487, Jan. 2008.

[31] J. Lu, F. Yang, M. Li, and L. Wang, "Multi-objective rule discovery using the improved Niched Pareto genetic algorithm", in *Proc. 3rd Int. Con. Measuring Technology and Mechatronics Automation*, 2011, pp. 657–661.

[32] K. Kshetrapalapuram and M. Kirley, "Mining classification rules using evolutionary multi-objective algorithms", in *Proc. 9th Int. Con. Knowledge-Based Intelligent Information and Engineering Systems*, 2005, pp 959–965.

[33] S. Li, C. Chen, and J. Li, "A multi-objective particle swarm optimization algorithm for rule discovery", in *Proc. 3rd Int. Con. on Information Hiding and Multimedia Signal Processing*, 2007, pp. 597–600.

[34] Z. Wang, X. Sun, and D. Zhang, "Classification rule mining based on particle swarm optimization", in *Proc. 1st Int. con. on Rough Sets and Knowledge Technology*, 2006, pp. 436–441.

[35] I. Falco, A. Cioppab, and E. Tarantinoa, "Facing classification problems with Particle Swarm Optimization", *Appl. Soft. Comput.*, vol. 7, pp. 652–658, June 2007.

[36] W. Yeh, "Novel swarm optimization for mining classification rules on thyroid", *Inf. Sci.*, vol. 197, pp. 65–76, Aug. 2012.

[37] T. Sousa, A. Silva, and A. Neves, "Particle swarm based data mining algorithms for classification tasks", *Parallel Comput.*, vol. 30, pp. 767–783, May 2004.

[38] A. Toracio and A. Pozo, "Multiple objective particle swarm for classification-rule discovery", in *IEEE Cong. Evolutionary Computation*, 2007, pp. 684–691.

[39] B. Alatas and E. Arkin, "Multi-objective rule mining using a chaotic particle swarm optimization algorithm", *Knowl. Based. Syst.*, vol. 22, pp. 455–460, Aug. 2009.

[40] A. Toracio, "Multiobjective particle swarm optimization in classification rule learning", Swarm Intelligence for Multi-objective Problems in Data Mining, in C. Coello, S. Dehuri, S. Ghosh, Eds. New York: Springer, 2009, pp. 37–64.

[41] J. Connolly, E. Granger, and R. Sabourin, "Dynamic multi-objective evolution of classifier ensembles for video face recognition", , *Appl. Soft. Comput.*, vol. 13, pp. 3149–3166, June 2013.

[42] X. Zhao, J. Zeng, Y. Gao, and Y. Yang, "A particle swarm algorithm for classification rules generation", in *6th Int. Con. Intelligent Systems Design and Applications*, 2006, pp. 957–962.

[43] M. Chen and S. Ludwig, "Discrete particle swarm optimization with local search strategy for rule classification", in *4th World Cong. Nature and Biologically Inspired Computing*, 2012, pp. 162–167.

[44] N. Holden and A. Freitas, "A hybrid PSO-ACO algorithm for discovering classification rules in datamining", *J. of Artific. Evl. Appl.*, vol. 2008, No. 2, Jan. 2008.

[45] K. Huang, "A hybrid particle swarm optimization approach for clustering and classification of datasets", *Knowl. Based. Syst.*, vol. 24, pp. 420–426, April 2011.