A Hybrid Fuzzy-Firefly Approach for Rule-Based Classification

M. Baran Pouyan, R. Yousefi, S. Ostadabbas, M. Nourani
Quality of Life Technology Laboratory
The University of Texas at Dallas, Richardson, TX 75080
{mxb112230, r.yousefi, sarahostad, nourani}@utdallas.edu

Abstract

Pattern classification algorithms have been applied in data mining and signal processing to extract the knowledge from data in a wide range of applications. The Fuzzy inference systems have successfully been used to extract rules in rule-based applications. In this paper, a novel hybrid methodology using: (i) fuzzy logic (in form of if−then rules) and (ii) a bio-inspired optimization technique (firefly algorithm) is proposed to improve performance and accuracy of classification task. Experiments are done using nine standard data sets in UCI machine learning repository. The results show that overall the accuracy and performance of our classification are better or very competitive compared to others reported in literature.

1 Introduction

Classification is a predictive technique with the ability to scientists to extract knowledge from a large set of data. Nowadays classification techniques can be seen in every area of science such as finance, biomedical expert systems, data mining, bio informatics, signal processing and so on (Nisbet, Elder IV, and Miner 2009)(Michalski, Bratko, and Bratko 1998). In classification, extracting a predictive model of data is very important. With this model, the data instances can be classified into predefined classes and we can predict class of new instances that may be generated in future. The extracted knowledge from the data model can be shown in the form of conditional statements or if-then prediction rules. This kind of rules have several merits such as concise form, interpretability of extracted rules, obvious knowledge exhibition and better understanding of feature relationship in existing rules for expert people.

Fuzzy logic (Zadeh 1965) provides an empirical approach to interpretable data analysis and robust inference processes. In several fields of science, fuzzy systems have been successfully used (Lee 1990). Fuzzy techniques have been applied for classification problems which interpretability is as important as accuracy for them. The fuzzy logic uses concepts such as membership function and certainty coefficient to enhance the classification performance (Ishibuchi et al. 1995)(Nozaki, Ishibuchi, and Tanaka 1996). The results of fuzzy classifier can be shown as "if-then" rules that make discovered knowledge from database more understandable. Fuzzy if-then rules were obtained from human experts. Various methods have been proposed to automatically produce and intelligently regulate fuzzy if-then rules, without any interference of human experts (Wang and Mendel 1992)(Abe and Lan 1995)(Roubos and Setnes 2001). Authors in (Baran Pouyan et al. 2009) applied a hybrid algorithm of Simulated Annealing (SA) and Genetic Algorithm (GA) for extracting the fuzzy rules. Genetic Algorithm (GA) has been used to extract a set of good fuzzy rules from numerical data (Mansoori, Zolghadri, and Katebi 2008). In reference (Keleş, Keleş, and Yavuz 2011), an expert system for diagnosing of breast cancer using neuro fuzzy rules was developed.

Nature-inspired algorithms are iterative search techniques becoming more efficient in solving complicated optimization problems. For example, references (Xiang and Lee 2008)(Jeong et al. 2010)(Yang 2010a) show some of these applications. The two important components of any metaheuristic algorithms are: selection of the best solution and randomization. The selection of the best ensures that the solutions will converge to the optimality, while randomization eschews the solution being trapped at local optima and of course increase the diversity of the solution. All nature-inspired algorithms apply a balance between randomization and local search (Yang 2010a). There are some probabilistic steps to avoid being trapped in local optima (Abbass, Sarker, and Newton 2002). So, we need a robust strategy to have a better search in problem space and be able to converge to the global optima.

Firefly Optimization Algorithm (FA), introduced by Xin-She Yang (Yang 2009), is among the most recent and rigorous optimization methods. The authors in (Yang 2009) formulated the conduct of the flashing properties of fireflies. The light intensity I at a particular distance r from the source decreases as the distance r increases. These two twisted factors make a swarm communication between fireflies to find the new positions of fireflies with more light intensity (Yang 2010b).

Firefly optimization has recently been applied in many problems that need complicated optimization process. A novel approach to determine the feasible optimal solution

of the Economic Dispatch (ED) problems is presented in (Yang, Sadat Hosseini, and Gandomi 2012). Authors in (Senthilnath, Omkar, and Mani 2011) have applied FA for clustering on benchmark problems. Their results and comparison with other clustering methods indicate that FA is very efficient for clustering problems. Reference (Yang 2009) compared bio-inspired algorithms in solving some hard problems and concluded that FA can be potentially more powerful in solving NP-hard problems.

Utilization of FA to extract a set of high accuracy and interpretable fuzzy classifier rules is our main contribution. In this paper, we propose and implement an efficient fuzzy classifier to better explore of the huge problem search space. FA in this work is the search strategy and is applied to find the optimum set of fuzzy rules as classifier rules.

The subsequent sections of this paper are organized as follows. In Section II, we review the background on firefly optimization and discuss on fuzzy-rule based model for classification. In Section III, we present our Fuzzy - Firefly hybrid system. Experimental results are shown in Section IV. Finally, Section V is devoted to concluding remarks.

2 Background

2.1 Firefly Optimization

The firefly optimization algorithm operates based on the shining characteristic of fireflies in nature. In this algorithm, we have three main rules: (i) Fireflies are considered unisex, so that a firefly is attracted to another firefly regardless of its sex; (ii) Attractiveness is proportional to their brightness which means the less shining one tends to approach the brighter one. The brightness of two considered fireflies decreases as their distance increases. If a firefly cannot find the brighter firefly, it will move in a random direction; (iii) The last rule states that brightness of each firefly is determined by the the landscape of the objective function (Yang 2010b).

This method works like other bio-natured algorithms such as genetic algorithm or simulated annealing. It first launches a random initial state which means some random fireflies (solutions) are generated in the problem space. The brightness I of a firefly at location x, has a direct relationship with problem objective function. The attractiveness coefficient β will be altered with the distance between two fireflies. So, β should be sensed between any two fireflies. Since β is changed by distance, it can typically be defined by Eqn. (1) (Yang 2010b):

$$\beta(r) = \beta_0 e^{-\gamma r^m} \tag{1}$$

where r is the distance between two fireflies and β_0 is attractiveness in $r = 0$. Parameter γ characterizes the variation of the attractiveness and m is a constant often considered as 2 in most optimization applications of firefly optimization. In general, $\beta(r)$ can be any monotonically decreasing function (Yang 2010b). The movement of firefly FF_i toward firefly FF_j can be determined by Eqn. (2):

$$\mathbf{X_i(t+1)} = \mathbf{X_i(t)} + \frac{\beta_0}{1+\gamma r^m}[\mathbf{X_j(t)} - \mathbf{X_i(t)}] + \mathbf{R}(\alpha) \tag{2}$$

where $\mathbf{X_i(t)}$ is the present position of firefly FF_i, $\mathbf{X_i(t+1)}$ is the next position of FF_i after moving toward FF_j. In this equation, firefly attraction rule is incorporated in the second term. $\mathbf{R}(\alpha)$ is the randomization step that directly depends on α (randomization parameter). Diversity of the solutions and ability to escape from local optima traps is increased with the third term that is the randomization phase. More specifically, $\mathbf{R}(\alpha)$ indicates our strategy for the local search. Randomization step can be implemented in several ways. For instance, authors in (Łukasik and Żak 2009) applied a random step according to the uniform distribution.

2.2 Fuzzy System for Classification

The goal is to classify m instances each of which considered in a vector model as $\mathbf{x_p}$=(x_{p1}, x_{p2}, ..., x_{pn}), $p = 1, 2, 3, ..., m$, in a problem with c classes. At first, we change the search space in our problem and normalize any feature value to be between [0,1]. On the other hand, our pattern space will be revised to $[0, 1]^n$ by:

$$AV_{Norm} = \frac{AV - AV_{min}}{AV_{max} - AV_{min}} \tag{3}$$

where AV denotes *attribute value*. Consider the following rule template as the general form of fuzzy "if-then" rule:

$$Rule\ \ R_j : if(x_1\ is\ A_{j1})\ and\ (x_2\ is\ A_{j2}) \cdots (x_n\ is\ A_{jn}),\ Class\ is\ C_j\ with\ certainty\ CF_j \tag{4}$$

where R_j is the label of the jth fuzzy if-then rule; $A_{j1}, , A_{jn}$ values are antecedent fuzzy sets, C_j is the consequent class, and CF_j is the certainty grade of related fuzzy if-then rule and is a real value in (0,1) range. By using CF_j, the covering area of each rule in problem space can be regulated. Domain range of each feature is partitioned based on the membership function representing the linguistic similarity value. C_j and CF_j for each rule R_j can be detected by using the training set. Reference (Ishibuchi, Nozaki, and Tanaka 1992) proposed a simple procedure using four steps to ascertain C_j and CF_j of rules in training phase:

Step 1: Compatibility of each training pattern $\mathbf{x_p}$=($x_{p1}, x_{p2}, , x_{pn}$) with rule R_j is calculated as:

$$\mu_j(\mathbf{x_p}) = \mu_{j1}(x_{p1}) \times ... \times \mu_{jn}(x_{pn}) \qquad p = 1, ..., m \tag{5}$$

where $\mu_{ji}(x_{pi})$ is the membership function of A_{ji} and m is the number of patterns in the territory of that rule.

Step 2: Relative sum of the compatibility grades of the training instances with the R_j is computed using:

$$\beta_{Class\ h}(R_j) = \sum_{\mathbf{x_p} \in Class\ h} \frac{\mu_j(\mathbf{x_p})}{N_{Class\ h}} \qquad h = 1, 2, .., c \tag{6}$$

where $\beta_{Classh}(R_j)$ is the sum of the compatibility grades of the training patterns in Class h with the fuzzy if-then rule R_j. $\beta_{Class\ h}(R_j)$ is the number of instances of $Class\ h$ in our training set.

Step 3: A class $\hat{h}$ with the maximum value of $\beta_{Class\ h}(R_j)$ is chosen for R_j:

$$\hat{h} = \arg\max_{1 \le k \le c} \{\beta_{Class\ k}(R_j)\} \tag{7}$$

where c is the number of classes.

Step 4: Each rule in fuzzy classifier system must have a certainty grade, which implies the strength of that fuzzy rule and clarifies the size of the covering territory of that rule. So, we have:

$$CF_j = \frac{\beta_{Class\,\hat{h_j}}(R_j) - \bar{\beta}}{\sum_{h=1}^{c} \beta_{Class\,h}(R_j)} \tag{8}$$

where

$$\bar{\beta} = \sum_{h \neq \hat{h_j}} \frac{\beta_{Class\,h}(R_j)}{c-1}. \tag{9}$$

When an input pattern vector $\mathbf{x_p}$=(x_{p1}, x_{p2}, ..., x_{pn}) is being classified by a rule set S_{rule}, a single winner rule R_{j^*} must have the maximum product of the compatibility and the certainty grade CF_j, where:

$$j^* = \arg\max_{R_j \in S_{rule}} \{\mu_j(\mathbf{x_p}).CF_j\} \tag{10}$$

meaning that, the winner rule (R_{j^*}) has the highest value of the product certainty grade and compatibility ($\mu_j(\mathbf{x_p}).CF_j$). If all $\mu_j(\mathbf{x_p})$ values in Eqn. (10) become 0, in this situation the classification of $\mathbf{x_p}$ is, per definition, declined.

3 Hybrid Fuzzy - Firefly Classifier

A set of triangular fuzzy sets have been chosen for fuzzy membership function in this work. Triangular fuzzy sets are simple and in comparison with the other membership functions such as gaussian, can effectively reduce the computation complexity of the algorithm. In computer simulation, a typical set of linguistic values shown in Figure. 1 (Mansoori, Zolghadri, and Katebi 2008), is used.

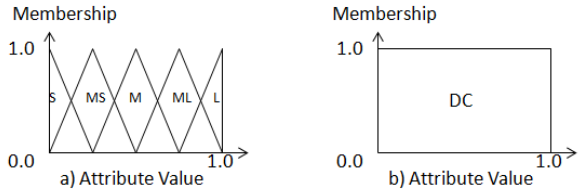

Figure 1: The antecedent fuzzy sets. (a) 1: small (S), 2: medium small (MS), 3: medium (M), 4: medium large (ML), 5: large (L). (b) 6: don't-care (DC).

As shown in Fig. 1 (a), each value domain is divided to 5 overlapping regions: *Small (S), Medium-Small (MS), Medium (M), Medium-Large (ML)* and *Large (L)*. In addition, we consider the feasible sixth value (don't-care) with linguistic value 6 (Fig. 1 (b)) enable our classifier to mask some attribute in its search space. According to these values in any classification problem with n attributes, the possible number of fuzzy if-then rules will be 6^n. Search in this very huge space to find a small set of fuzzy classifier rules with most accuracy exhaustively is not practical. So, we need a robust strategy to learn from training samples in order to reduce the searching complexity. Assume, we have a set of fuzzy rules generated randomly in our linguistic value set as a first solution. The fuzzy rules are coded as a string of 6 possible linguistic value mentioned earlier. For example, the string "513624" is a code representing rule R_j that means: If A1 is *large* and A2 is *small* and A3 is *medium-small* and A4 is *don't-care* and A5 is *medium-small* and A6 is *medium*, then class is C_j with certainty CF_j.

The main goal is finding the best (or one of the best) rule set S^*_{rule} in problem space to get better fuzzy learning model for our data set. Since the problem space is huge, we employ the Firefly optimization that has shown a good performance with low running time for combinatorial optimization problems. By using the fuzzy classification system, the accuracy rate of each rule in fuzzy rule set S_{rule} is determined. The firefly optimization improves the performance of classification. The proposed hybrid fuzzy-firefly classifier runs in four steps as explained next.

3.1 Step I: Generate the Initial Firefly Population

At first, an empty roster of learned (chosen) rules is considered, i. e. S_{rule}={}. Let SOF be the firefly set with the size N_{ff}. The rule R_i is generated by a firefly. To avoid overfitting issue, the initial rules (fireflies) are generated randomly but we consider the probability of each linguistic value in pertinent class. The highest probability of all linguistic values in each class is considered for *don't-care* value in this work. The firefly parameters (β_0, γ, α) are adjusted in this phase. The class and certainty degree for each firefly (rule) are calculated by the procedure mentioned in the section II. B. Powerful and weak rules should be detected by a fitness function for each firefly as follows:

$$fit(FF_j) = N_c(FF_j) - N_{nc}(FF_j) \tag{11}$$

where $fit(FF_j)$ is the fitness value for FF_j that can be computed according to fuzzy classification rules. Similarly, $N_c(FF_j)$ implies the number of correctly classified training patterns by FF_j. $N_{nc}(FF_j)$ is the total number of incorrectly classified and not-classified patterns.

3.2 Step II: Firefly Movement

In Firefly optimization, a firefly is inclined to move toward brighter firefly. It means a brighter firefly has better fitness. By the second part of Eqn.(2), firefly FF_i must be closer to firefly FF_j. In other words, their distance must be decreased after this step. The distance between two fireflies must be described in the problem space. In our problem, each firefly is a fuzzy rule. Let us consider two fireflies FF_i and FF_j in a pattern classification problem with k attributes. If FF_i is shinier than FF_j, it does not change its position. But, if FF_j is brighter than FF_i, it means FF_j has better fitness in comparison with FF_i and consequently, FF_i has to move toward FF_j. The following two steps (attraction and randomization) show this movement:

II.1. **Attraction Step:** Consider two fireflies $FF_i = \{ff_{i,1}, ..., ff_{i,k}\}$ and $FF_j = \{ff_{j,1}, ..., ff_{j,k}\}$. The $distance$ between FF_i and FF_j is defined as follow:

$$r[FF_i, FF_j] = \sum_{l=1}^{k} |ff_{i,l} - ff_{j,l}| \tag{12}$$

where r is distance between two rules (fireflies). In our work, since the attribute values are discrete (1 to 6), we define β as:

$$\beta(r) = \frac{\beta_0}{1 + \gamma r}. \quad (13)$$

We can show by this definition that fireflies can move closer to each other according to the firefly attractiveness. When $ff_{i,l} = ff_{j,l}$, we don't have any change in that position. When they are not equal, according to Eqn. (13) and with the probability β, $ff_{i,l}$ will be randomly updated in their difference range. By a uniform probability, we assign a random number in the range between $ff_{i,l}$ and $ff_{j,l}$ to modify $ff_{i,l}$.

For example, suppose we have two fireflies with 8 attributes, FF_i=[1,4,3,5,5,4,6,2] and FF_j=[3,6,3,2,6,4,3,5]. Using Eqn. (12), $r[FF_i, FF_j]$=14. With β_0= 1 and $\gamma = 0.5$, we get $\beta = 0.125$. Since we apply absolute distance, the amount of their common elements have to increase. First, the new common attribute of FF_i with FF_j will be: FF_i=[-,-,3,-,-,4,-,-]. In the second step, β is applied to get the probability to change the value of each undecided element in different range $ff_{i,l}$ and $ff_{j,l}$. If no element changes, one element is selected randomly and will be changed in the related range. If in this example, only the fourth attribute of FF_i can be altered, FF_i=[1,4,3,3,5,4,6,2]. So, the distance between FF_j and new location of FF_i becomes 12. Thereby, the two fireflies get closer.

II.2. **Randomization Step:** After fireflies movement by attractiveness property, there is a randomization step. In this step according to a predefined value of the α, we have a random operation. α represents a fraction of rule elements which should be altered randomly. In other words, $\mathbf{R}(\alpha)$ changes the chosen element value by uniform probability. This process is repeated for all rules in each class. If there is a rule (firefly) that has not changed its position (the strongest rule in each class), this rule will alter through changing a small number of its elements (one or more) randomly. The problem space can be very huge and modifying too many rule component values can cause getting farther from global optimum solution.

3.3 Step III: Firefly Set Modification

After any iteration, the following perturbation operation is applied on the current fireflies set. First, a firefly FF^* is selected using the following probabilistic equation:

$$P(FF^*) = \frac{\max\limits_{1 \le i \le N_{ff}} \{fit(FF_i)\} - fit(FF^*)}{\sum_{i=1}^{N_{ff}} [fit(FF_i) - \min\limits_{1 \le i \le N_{ff}} \{fit(FF_i)\}]} \quad (14)$$

where $fit(FF_i)$ is the fitness value of firefly FF_i. This is obvious that the weak rules have more chance to be chosen for changing. Randomly swapping and mutation operations are applied between some elements of the selected firefly and to generate a new firefly to substitute the parent rule. The resulting class of the new firefly (rule) must be updated. This generated firefly (rule) can help its class subset of rules to escape from the local optima. After T_{sel} (a predefined number) iterations, the best firefly is selected. The best firefly generates a rule R_{j^*} according to its elements and adds R_{j^*} to S^*_{rule}. The correct classified instances in training set by R_{j^*} are eliminated from the training set. Afterwards, the fitness values of all fireflies are calculated.

3.4 Step IV: Termination and Evaluation

Steps I through III have to be done for each class separately. In most bio-natured optimization algorithms, a predefined number of iteration is considered as termination condition. Using the number of iterations may cause rule set escape from optimum status. Therefore, in our work, this condition is adjusted. We stipulate a condition related to the an initial number of iteration and the number of uncovered instances for each class. After the algorithm passes this primitive iterations, when the number of uncovered instances in two consecutive iteration doesn't change, the algorithm will be terminated. Afterwards, the final rule set can be applied for our pattern classification problem. We also use this final set for getting accuracy rate on training and test set. To evaluate this final set, the accuracy metric, $A(S^*_{rule})$, is defined:

$$A(S^*_{rule}) = (1/m) * [m - \sum_{j=1}^{N_r} N_c(R_j)] \quad (15)$$

where N_r is the number of rules in final learned list ($N_r = |S^*_{rule}|$). m is the number of training set instances and the second term in Eqn. (15), denotes the number of all correctly classified by S^*_{rule}. Fig. 2 shows the pseudocode of four key steps in our proposed approach.

```
For each class in training set {
  Generate N_ff of fireflies
  Preset α, γ, β_0, T_sel
  S*_rule={}
  While (fireflies moving) {
    for k=1 to T_sel {
      for all i and j (1 ≤ i, j ≤ N_ff) {
        Decide on moving FF_i toward FF_j by fuzzy rules
        If not moving, force a slight random move
      }
    }
    Select the best rule R_j* corresponding to the firefly FF*
    S*_rule = S*_rule ∪ {R_j*}
    Remove correctly classified instances from training set
  }
}
```

Figure 2: Pseudocode of Hybrid Fuzzy-Firefly Classifier.

4 Experimental Results

The key parameters of our algorithm are shown in Table 1. We ran our algorithm with several number of fireflies N_{ff}. When N_{ff} gets bigger, the algorithm needs more time in training step. There is a tradeoff between number of fireflies (N_{ff}) and the running time. While we experimented using various N_{ff}, we empirically found 60 fireflies provide an excellent tradeoff. The proposed pattern classification technique has been well evaluated by extensive testing. Nine various data sets of the University of California at Irvine

Parameters	Symbol	Value
attractiveness in r = 0	β_0	1
absorbtion coefficient	γ	0.5
perturbation factor	α	0.1
max number of fireflies	N_{ff}	60

Table 1: Parameter Setting in Computer Simulation.

Name	No. Instances	No. Features	No. Class
wdbc	569	30	2
pima	768	8	2
wine	178	13	3
cra	690	15	2
ion	351	34	2
iris	150	4	3
bswd	625	4	3
lab	57	16	2
gid	214	9	6

Table 2: UCI Data set Properties.

(UCI) machine learning repository (Asuncion and Newman 2007) are used to evaluate performance of this algorithm. These data sets are: (i) Wisconsin diagnosis breast cancer (wdbc), (ii) pima Indians diabetes (pima), (iii) wine recognition data (wine), (iv) credit approval, (v) John Hopkins University ionosphere database (ion), (vi) iris plants database (iris), (vii) balance scale weight and distance (bswd), (viii) final settlements in labor negotiation in Canadian industry (lab), and (xi) glass identification database (gid).

Table 2 summarizes the key properties of these data sets. Algorithm performance in this work is measured based on the accuracy of train and test sets and is computed and compared against several classification machine learning methods. $LIBSVM$, XCS, KNN, $NaiveBayes(NB)$, $C4.5$, $GAssist$, $Part$ (all reported in (Bacardit 2004)) that are among prominent classification techniques and another fuzzy based technique *SA-GA* (reported in (Baran Pouyan et al. 2009)) are considered. To have diversity in the experiments, ten-fold cross-validation (10-CV) technique, sometimes called rotation estimation, is used to have a better evaluation of our classification algorithm. The data sets are divided into 10 subsets of the same size randomly. Nine subsets are used as *training patterns*, and the tenth subset is used as *test*. The comparative results with different algorithms on training and test sets are presented in Tables 3 and 4.

The performance of our algorithm (i. e. accuracy as shown in Eqn. (15)) on the train data set like the other algorithms is expectedly very good. As shown in Table. 3, XCS has the best performance on training sets. Since all approaches apply training set in the learning phase, it is not enough to evaluate a classifier only for training set. The main goal of any classifier is the good performance on test data set. In Table 4, the performance of our approach on data sets in test phase are shown. The performance of our algorithm on $wdbc$, $pima$ and gid specifically is the best. The classification performance of proposed method on $pima$ set is in particular salient. Benchmark $pima$ attracted a lot of attentions due to its inherent difficulty of classification for diagnosis of diabetic disease. For example, reference (Kahramanli and Allahverdi 2008) summarized 43 algorithms that

wdbc	pima	wine	cra	ion	iris	bswd	lab	gid
483	462	217	456	502	137	145	344	154

Table 5: Training time of our algorithm (Seconds).

achieved accuracy of 59.5% to 84.2%. Our proposed algorithm shows accuracy of 85.30% which is the best reported for $pima$ in literature. Also, our proposed classifier performed superior for the data set gid, that is one of the most difficult data sets for all of the classifiers. The performance of our algorithm is low for lab. We believe this is due to insufficient number of instances (57) with respect to number of features (16) in comparison with other data sets. If more data is provided, like other examples, our algorithm will perform much better. As shown in Table 4, our proposed classification algorithm has the best average accuracy of nine cases.

The training time of our algorithm on all data sets using a $3GHz$ CPU running on a desktop reported in Table 5. Note that running time for test sets are less than 0.1 second. We do not have access to the running time for $GAssist$ and XCS. For the other algorithms, with our simulation by $weka$, the training time is lower than 10 seconds. Note that for most data mining applications, the training is done only once a priori. Therefore, fast execution during training is not considered a major advantage. The real advantage of our method is its superior accuracy as reflected in Tables 3 and 4.

5 Conclusion

In this work, a novel hybrid classification approach using fuzzy logic and firefly optimization is presented. First, the data set was modeled using fuzzy membership function. Then, a set of fuzzy if-then rules were generated by predefined fuzzy set and by applying firefly optimization, the best fuzzy if-then rules to classify patterns were constructed. The proposed algorithm shows a very promising and reliable performance when it was tested against various data sets. When applied to classify nine benchmarks in UCI database, our algorithm shows the best average result in terms of accuracy of classification. This includes the best accuracy compared to all reported in literature for the two most challenging data sets, i. e. $pima$ and gid.

References

Abbass, H. A.; Sarker, R. A.; and Newton, C. S. 2002. *Data mining: a heuristic approach*. IGI Global.

Abe, S., and Lan, M.-S. 1995. A method for fuzzy rules extraction directly from numerical data and its application to pattern classification. *Fuzzy Systems, IEEE Transactions on* 3(1):18–28.

Asuncion, A., and Newman, D. 2007. Uci machine learning repository.

Bacardit, J. 2004. *Pittsburgh genetics-based machine learning in the data mining era: representations, generalization, and run-time*. Ph.D. Dissertation, PhD thesis, Ramon Llull University, Barcelona, Catalonia, Spain.

Baran Pouyan, M. B.; Mohamadi, H.; Saniee Abadeh, M.; and Foroughifar, A. 2009. A novel fuzzy genetic annealing classification approach. In *Computer Modeling and Simulation, 2009. EMS'09. Third UKSim European Symposium on*, 87–91. IEEE.

Data set	Fuzzy-Firefly	$LIBSVM^{\dagger}$	$XCS^{\dagger}$	$KNN^{\dagger}$	$NB^{\dagger}$	$C4.5^{\dagger}$	$GAssist^{\dagger}$	$Part^{\dagger}$	$SA-GA^{\dagger}$
wdbc	99.38	97.22	**100**	98.47	95.38	98.87	99.06	99.13	–
pima	91.42	78.27	**98.90**	85.67	77.07	84.43	84.25	78.88	84.89
wine	99.51	99.33	**100**	97.27	98.67	98.86	**100**	99.15	99.95
cra	84.51	55.51	**98.90**	91.05	82.58	90.31	90.61	93.23	91.52
ion	98.56	94.19	**99.86**	90.94	93.00	98.68	98.49	98.39	–
iris	**99.71**	97.11	99.10	96.59	96.67	98.00	99.47	97.75	**99.71**
bswd	**96.43**	91.01	95.19	90.53	91.92	89.93	92.14	93.39	–
lab	95.98	96.04	99.92	98.77	95.92	91.58	**100**	94.21	98.93
gid	**98.71**	61.23	97.62	81.12	57.72	92.91	87.89	93.08	–
Avg	96.02	85.54	**98.83**	92.26	87.65	93.73	94.65	94.13	–

Table 3: Results of global comparative test on UCI data sets - Training accuracy(%).

Data set	Fuzzy-Firefly	$LIBSVM^{\dagger}$	$XCS^{\dagger}$	$KNN^{\dagger}$	$NB^{\dagger}$	$C4.5^{\dagger}$	$GAssist^{\dagger}$	$Part^{\dagger}$	$SA-GA^{\dagger}$
wdbc	**97.38**	96.72	96.00	96.83	94.55	93.32	96.53	94.27	–
pima	**85.30**	77.32	72.40	74.52	75.30	75.44	74.46	74.88	75.16
wine	97.72	**98.10**	95.6	96.61	97.20	92.24	96.33	91.71	97.70
cra	81.63	55.51	**85.6**	84.73	81.07	85.55	85.18	84.23	85.29
ion	91.70	**92.14**	90.1	85.66	91.50	88.97	90.43	90.43	–
iris	94.53	96.22	94.70	94.89	96.22	94.22	94.49	93.78	**96.66**
bswd	91.06	90.90	81.1	86.09	**91.43**	77.66	89.62	83.22	–
lab	81.69	93.35	83.5	95.38	93.76	80.31	97.30	80.81	**97.85**
gid	**72.00**	58.89	71.80	70.36	51.00	68.81	68.28	69.56	–
Avg	**88.11**	84.35	85.64	87.23	85.78	85.13	88.07	84.76	–

† *References (Bacardit 2004) and (Baran Pouyan et al. 2009) have used Weka tool (Witten and Frank 2005) to report these results.*

Table 4: Results of global comparative test on UCI data sets - Test accuracy(%).

Ishibuchi, H.; Nozaki, K.; Yamamoto, N.; and Tanaka, H. 1995. Selecting fuzzy if-then rules for classification problems using genetic algorithms. *Fuzzy Systems, IEEE Transactions on* 3(3):260–270.

Ishibuchi, H.; Nozaki, K.; and Tanaka, H. 1992. Distributed representation of fuzzy rules and its application to pattern classification. *Fuzzy sets and systems* 52(1):21–32.

Jeong, Y.-W.; Park, J.-B.; Jang, S.-H.; and Lee, K. Y. 2010. A new quantum-inspired binary pso: application to unit commitment problems for power systems. *Power Systems, IEEE Transactions on* 25(3):1486–1495.

Kahramanli, H., and Allahverdi, N. 2008. Design of a hybrid system for the diabetes and heart diseases. *Expert Systems with Applications* 35(1):82–89.

Keleş, A.; Keleş, A.; and Yavuz, U. 2011. Expert system based on neuro-fuzzy rules for diagnosis breast cancer. *Expert Systems with Applications* 38(5):5719–5726.

Lee, C.-C. 1990. Fuzzy logic in control systems: fuzzy logic controller. ii. *Systems, Man and Cybernetics, IEEE Transactions on* 20(2):419–435.

Łukasik, S., and Żak, S. 2009. Firefly algorithm for continuous constrained optimization tasks. In *Computational Collective Intelligence. Semantic Web, Social Networks and Multiagent Systems*. Springer. 97–106.

Mansoori, E. G.; Zolghadri, M. J.; and Katebi, S. D. 2008. Sgerd: A steady-state genetic algorithm for extracting fuzzy classification rules from data. *Fuzzy Systems, IEEE Transactions on* 16(4):1061–1071.

Michalski, R. S.; Bratko, I.; and Bratko, A. 1998. *Machine Learning and Data Mining; Methods and Applications*. John Wiley & Sons, Inc.

Nisbet, R.; Elder IV, J.; and Miner, G. 2009. *Handbook of statistical analysis and data mining applications*. Academic Press.

Nozaki, K.; Ishibuchi, H.; and Tanaka, H. 1996. Adaptive fuzzy rule-based classification systems. *Fuzzy Systems, IEEE Transactions on* 4(3):238–250.

Roubos, H., and Setnes, M. 2001. Compact and transparent fuzzy models and classifiers through iterative complexity reduction. *Fuzzy Systems, IEEE Transactions on* 9(4):516–524.

Senthilnath, J.; Omkar, S.; and Mani, V. 2011. Clustering using firefly algorithm: Performance study. *Swarm and Evolutionary Computation* 1(3):164–171.

Wang, L.-X., and Mendel, J. M. 1992. Generating fuzzy rules by learning from examples. *Systems, Man and Cybernetics, IEEE Transactions on* 22(6):1414–1427.

Witten, I. H., and Frank, E. 2005. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann.

Xiang, W., and Lee, H. 2008. Ant colony intelligence in multiagent dynamic manufacturing scheduling. *Engineering Applications of Artificial Intelligence* 21(1):73–85.

Yang, X.-S.; Sadat Hosseini, S. S.; and Gandomi, A. H. 2012. Firefly algorithm for solving non-convex economic dispatch problems with valve loading effect. *Applied Soft Computing* 12(3):1180–1186.

Yang, X.-S. 2009. Firefly algorithms for multimodal optimization. In *Stochastic algorithms: foundations and applications*. Springer. 169–178.

Yang, X.-S. 2010a. *Engineering optimization: an introduction with metaheuristic applications*. John Wiley & Sons.

Yang, X.-S. 2010b. *Nature-inspired metaheuristic algorithms*. Luniver press.

Zadeh, L. A. 1965. Fuzzy sets. *Information and control* 8(3):338–353.