A Parallel Fuzzy-Genetic Algorithm for Classification and Prediction

Hassan Abounaser
Computer Engineering Department
Arab Academy for Science, Technology and Maritime Transport (AASTMT)
Cairo, Egypt

Ihab Talkhan
Computer Engineering Department
Cairo University
Cairo, Egypt

Ahmed Fahmy
Computer Engineering Department
Arab Academy for Science, Technology and Maritime Transport (AASTMT)
Cairo, Egypt

***Abstract*—One of the top challenging problems in data mining domain is the distributed data mining (DDM) and mining multi-agent data. In distributed environment, classical techniques require that the distributed data be first collected in a data warehouse which is usually either ineffective or infeasible. Hence, mining over decentralized data sources can overcome such issues. Rule-based classifiers involve sharp cutoffs for continuous attributes. Fuzzy Logic System (FLS) has features that make it an adequate tool for addressing this shortcoming effectively and efficiently. In this paper, a framework for a Parallel Fuzzy-Genetic Algorithm (PFGA) has been developed for classification and prediction over decentralized data sources. The model parameters are evolved using two nested genetic algorithms (GAs). The outer GA evolves the fuzzy sets whereas the inner GA evolves the fuzzy rules. During optimization, best rules are only distributed among agents to construct the overall optimized model. Several experiments have been conducted over many benchmark datasets. The experiment results show that the developed model has good accuracy and more efficient in performance and comprehensibility of linguistic rules compared to some models implemented in KEEL software tool.**

***Keywords*—Fuzzy Classification; Rule-Base; Fuzzy Logic System (FLS); Genetic Algorithm; Distributed Data Mining (DDM)**

I. Introduction

Data mining, generally, can be described as the process of transforming knowledge from data format into some other human understandable format [1]. This knowledge discovery process has many domains and application areas such as in bioinformatics [2], business analytics [3], text analysis [4], web data analysis [5], health care [1] and many other domains where there is scope for hidden information retrieval.

In literature, data mining systems can be categorized according to data type, data model, task/knowledge type, or exploration technique [1][6]. One form of data mining tasks and Machine Learning (ML) techniques is classification. Classification and prediction are forms of data analysis in order to construct models for describing important data classes or predicting future data trends [7][8]. A classifier model is constructed to predict categorical (discrete, unordered) labels while a predictor model is constructed to predict ordered or continuous valued function. These constructed models give better understanding of the data at large. For example, a marketing manager of a car agency may ask to construct a classifier model to predict to what degree a customer will accept buying a particular car, given the car profile.

Data mining has some challenges. One of the top challenging problems in data mining domain is the distributed data mining (DDM) and mining multi-agent data [9]. In distributed environment, classical techniques require that the distributed data be first collected in a data warehouse [6]. This Collection of huge volume of data is usually either ineffective or infeasible for many reasons. For example, this may encounter problems belongs to privacy and sensitivity of data in addition to the costs in storage, communication, and computation. Hence, mining over decentralized data sources can overcome the above issues and help to reach all network-related domains. For distributed environment, numerous techniques have been developed for data classification and prediction in order to discover knowledge from distributed data effectively and efficiently [10][11][12]. However, no single data mining technique has been proven appropriate for every domain and dataset [6].

Rules are one way for representing information or bits of knowledge. Rule-based classifiers use a set of IF-THEN rules for classification [13]. However, rule-based classification systems have the shortcoming that they involve sharp cutoffs for continuous attributes. Fuzzy Logic System (FLS) has attractive features that make it an alternative tool to tackle this issue in designing data mining systems performing rule-based classification effectively and efficiently [6].

In this paper, FLS features are explored in next section. In third section, Genetic Algorithms (GAs) are presented as an example of evolutionary computing algorithms (EAs) for evolving fuzzy rule-base. In fourth section, an optimized Parallel Fuzzy-Genetic Algorithm (PFGA) is developed for classification and prediction over decentralized data sources. In fifth section, results of conducted experiments are provided, analyzed and discussed compared with some classification models implemented in KEEL software tool. Finally, a conclusion is presented.

II. Fuzzy Logic Systems (FLSs)

One highly successful theory in Computational Intelligence (CI) techniques is fuzzy set theory [14]. The design of FLS was one of the largest application areas derived from fuzzy set theory. FLS have demonstrated their superb ability as system identification tools and has enjoyed wide

popularity in computer science and engineering as an advanced Artificial Intelligence (AI) tool and control technique [15] [16]. The strength of FLS lies in its expressive power and flexibility to handle a complex system even if no precise mathematical model of the underlying processes is available [17]. The key issue resolves around designing the required input and output fuzzy sets that define the semantic of the domain. FLS domains are characterized by linguistic labels rather than by numbers. Hence, FLS is frequently considered as computing with words rather than numbers [17]. This descriptive approach, generally, are suitable for handling the issues related to understandability of patterns, incomplete or noisy data, and can provide approximate solutions faster [18].

As graphically shown in Fig. 1, a general-purpose FLS consists of four generic components. It works by encoding an expert's knowledge into a set of IF-THEN fuzzy rules, which are smoothly interpolated, and the resultant is defuzzified to give the desired behavior in terms of crisp output. Each fuzzy rule is specified as either a trapezoid, triangular, logistic, bell shape, or some other functions, and assigned to some range of input variable. Common sense can provide good estimates for fuzzy sets and membership functions to be associated with each linguistic input and output variables. However, it is the task of the human domain expert to define the function that captures the characteristics of the fuzzy set. Since it tolerate imprecision, FLS is an attractive technique for feature classification because a given feature may have partial membership in different classes. Recent work by data mining researchers has shown that the qualitative nature of FLS makes it a formal tool for constructing classifiers that deal with problems characterized by pervasive presence of uncertainty. For example, Fuzzy-based classifier has been applied successfully in data mining for Hepatitis [19], and data mining for intrusion detection [20]. Fuzzy-based classifier, generally, consists of a set of fuzzy linguistic rules as sentences rather than equations. These fuzzy linguistic rules are easier understood than systems of mathematical equations.

A FLS, generally, is known as knowledge-based system. The Knowledge Base (KB) not only has the rule-base but it also has the fuzzy sets and membership functions of the fuzzy partitions associated to the linguistic input and output variables. Therefore, this specifies a clear distinction between the fuzzy model structure and parameters as defined in classical knowledge discovery techniques [21]. Although the above generic components are common features to all fuzzy-based systems, many design options exist based on this structure[15].

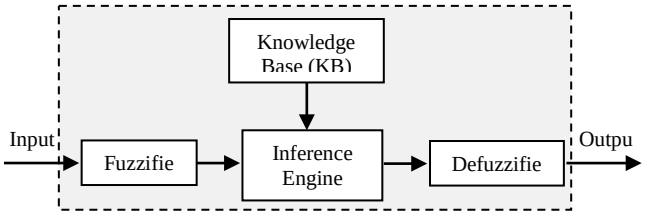

Fig. 1. The structure of a Fuzzy Logic System (FLS) and its components interconnections

Mamdani and Assilian produced the first FLS for control in 1975 [21]. This type of FLS forms the basis for all types of FLS. It is derived directly from available heuristic control strategies mimicking the control knowledge of a human expert. FLS is statically described by linguistic rules [14]. The output fuzzy sets utilized in Mamdani-type FLS are singletons or combinations of singletons, where the combinations are achieved through application of fuzzy set operators. However, FLS generally has the advantage of allowing two design approaches. The first design approach is applied for real-world applications where the human expert knows the appropriate system response, given particular input variable scenarios. In this case, the FLS parameters can be specified, and they are static (e.g., [22]). This contrasts with Artificial Neural Network (ANN) approaches, where the user is in no position to specify weights values, even if the appropriate system response is known.

The second approach in FLS design is applied for real-world applications where the system response is not known. In this case the FLS parameters can be optimized through learning using some CI techniques such as EAs or ANN strategies, until the overall FLS response matches the desired behavior [23]. Since it is the most natural approach, the use of EAs strategies in designing FLSs is currently a hot topic in classification area and has been largely extended in the last few years to face the tradeoff between interpretability and accuracy as both requirements are clearly in conflict [21]. The developed framework in this paper adopts the second approach in design and utilizes Pittsburgh approach in learning. However, a structure of nested GAs is developed for encoding the whole KB definition such that the performance of the optimized model proposed fits the desired efficiency and accuracy.

III. EVOLUTIONARY ALGORITHMS (EAS)

Optimization is a classical problem in several domain such as in Economy and Biology among others [24]. Analytical techniques were popularly used to solve optimization problems efficiently. However, alternative and competitive methods in CI techniques have appeared such as EAs. EAs have been applied to a wide range of problem areas such as control, function optimization, regression, classification and clustering [14]. One of the most common EAs strategies is the GA. GA generally combines adaptive heuristic search along with mathematical analysis to find approximate or even true solutions for optimization problems. The technique of GAs not only provides alternative methods for solving optimization problems, but it also consistently outperforms other classical methods in most of these problems [25].

The basic concepts in designing GAs follow the principle of survival of the fittest. This principle is inspired by natural selection and natural genetics which is first laid down by Charles Darwin. Although the steps of designing GAs are simple to understand and not difficult in coding, designing a suitable GA for a real-world task is a nontrivial exercise and almost an art [14][25]. However, design a GA has the exploration-exploitation tradeoff due to the interactions between the representation, selection, reproduction, mutation, and replacement operations. The high consumption of computational resources besides this tradeoff has been the

source of active research directions to discover alternative EA strategies [14].

From coding mechanism's perspective, genetic rule-based algorithms in DM can be categorized into three approaches. The first approach is known as "Pittsburgh" approach where each individual encodes a rule set and the best individual in final population represent the rule-based classifier. The second approach is known as "Michigan" approach, where each individual encodes a single rule and the final population represents the rule-based classifier. The third approach is known as Iterative Rule Learning (IRL) approach, where each individual also encodes a single rule but each rule of the rule-based classifier is obtained from each run [8]. The developed framework in this paper utilizes Pittsburgh approach in learning, as mentioned earlier.

IV. Proposed Parallel Fuzzy-Genetic Algorithm (PFGA) for Classification and Prediction

A. Fuzzy Logic Classifier & Predictor

The behavior of the FLS is determined by a number of parameters such as the fuzzy sets, the membership functions, and the structure and entries in the Fuzzy Associative Memory (FAM) matrix or fuzzy rules. In addition, some FLSs include parameters that assign a weight for each fuzzy rule to indicate its relative importance in the overall FLS behavior. All of these parameters are possible candidates for optimization using EA strategies, for example. However, it's a daunting task for code developers to design a FLS that optimizes all of these parameters. In this research, the fuzzy sets and the fuzzy rule-base are both the candidates for optimization since they have the most influence in determining the FLS behavior.

A classifier or predictor model is a decision rule that assign a class to every data point in attribute/feature space. Expert knowledge, in classification and prediction domain, can be effectively used to design a FLS as a set of flexible overlapping fuzzy rules that can be evolved in order to construct an adaptive model that approximate human reasoning in this domain. The fuzzy rules actually represent direct linguistic description of the particular relationships between the given attributes/features and their assigned class. Equation (1) represents the general form of a fuzzy rule:

$$R_i: IF\ (x_1 = A_1) AND\ \dots\ AND (x_n = A_n)\ THEN\ (y = C) \quad (1)$$

Where: R_i is fuzzy rule label number i in rule-base

n is the number of attributes in dataset

$x_1,\dots, x_n$ are input linguistic variables

y is the class linguistic variable

$A_1,\dots, A_n$ are terms in input domains

C is the assigned class term in output domain

For example, one of the evolved fuzzy rules that predict the customer acceptability degree for buying a particular car, given the car profile, specify *"IF (buying=MED) AND (safety=HIGH) THEN (acceptability=ACC)*". In this research, the input attributes/features of a dataset are assumed to be of equal importance and independent of one another. However, the types and the number of membership functions defining the fuzzy sets utilized for particular attribute is attribute dependent. For continuous attributes, triangular and trapezoid membership functions are selected with symmetry and initial overlap degree criteria of 25% [15] since they are computationally efficient in real-time FLS [6]. Moreover, a range of 3 fuzzy sets is utilized for input variables whereas a range of 5 output rules is utilized as this provides adequate resolution without excessive computational cost. For discrete variables, a singleton membership function is selected in representing discrete domains.

Simplicity of design has been imposed, so the fuzzy rules are assumed to be of equal importance. Hence, all fuzzy rules in the rule-base are used with equal weighting. However, the size of the rule-base is controlled by the size of a dataset. For datasets having more than 1000 tuples, the rule-base is limited to minimum 10 fuzzy rules and maximum 25 fuzzy per agent. Otherwise, the rule-base is limited to minimum 5 fuzzy rules and maximum 15 fuzzy rules per agent. This control approach is necessary in order to avoid ignorance or explosion in case of too small rule-base size or too large rule-base size, respectively, which may results in undesired rules degrading both the accuracy and the interpretability. However, a rule-base size for particular dataset can be computed as a function of its attributes and tuples sizes.

For a low cost in storing fuzzy rules, virtual multidimensional FAM matrix approach is used in this research alternatively to multidimensional FAM matrix of fixed dimension. In this approach, higher-dimensional spaces are separated into two dimensions matrices and only the actually used entries are stored. Hence, no need to allocate the full multidimensional FAM matrix since a fuzzy rule is represented by index information that specifies its location in the virtual matrix. By using this approach, flexible FAM matrix structuring is allowed since rule entries are stored consecutively with variable number of inputs and the storing order becomes insignificant. In addition, huge benefits are also allowed when using this approach such as flexibility, implementation simplicity, ability to handle FAM matrices of arbitrary dimension, and ability to handle multiple lower-dimensional FAM matrices simultaneously. Furthermore, this approach allows for a very compact implementation for the defuzzification process. The firing strength for a FAM entry is computed using the rule of minimum membership degree values as:

$$w_i = min\{\mu_{A_1}(x_1), \dots, \mu_{A_n}(x_n)\} \quad (2)$$

Where: n is the number of attributes in dataset

$x_1,\dots, x_n$ are input linguistic variables

i is fuzzy rule number in rule-base

μ_{A_i} is the membership function of fuzzy set A_i

w_i is the firing strength of i^{th} fuzzy rule

However, if a membership function for each output fuzzy set is defined as:

$$\mu_{C_i}(y_i) = output\ membership\ set \quad (3)$$

Where: i is fuzzy rule number in rule-base

y_i is the class linguistic variable in i^{th} fuzzy rule

μ_{C_i} is the membership function of fuzzy set C_i

Thereafter, the overall FLS response is computed from the defuzzification process for n output membership fuzzy sets as:

$$output = \frac{\sum_{i=1}^{n}\left(w_i \times \mu_{C_i}(y_i)\right)}{\sum_{i=1}^{n} w_i} \quad (4)$$

Where: n is the number of total current active rules

$\left(w_i \times \mu_{C_i}\right)$ is the height defuzzification

The height defuzzification method is generally referred to as "clipped center of gravity". This method is computationally efficient in real-time FLS compared to other methods such as the "center of gravity" or "centroid" defuzzification method, where the output is a combination of centroids for each overlapping fuzzy membership function. Moreover, it is highly recommended since it utilizes better use of the information in the output distribution and generates unique fuzzy centroid [26].

B. Evolved Fuzzy Logic Classifier & Predictor

Normally, a FLS is not adaptive since it has no learning ability in itself. In addition, the design of FLS for certain real-world application is considered as knowledge-intensive and time-consuming. This is due to the rapid increase in the possible combinations of FLS parameters in the KB such as the fuzzy rules, fuzzy rule structure, the number of input and output variable dimensions, fuzzy membership types, number of fuzzy sets per variable, and so on. Hence, FLS is a suitable domain to apply EA or ANN learning strategies to form a hybrid system in order to find an optimum set of fuzzy rules to finally get the desired behavior (e.g., [27]). The real-world application addressed in this research is how to automate the process of designing FLS parameters for a dataset classification and prediction algorithm in distributed environment. Therefore, a framework for a Parallel Fuzzy-Genetic Algorithm (PFGA) has been developed in this paper to evolve the parameters of the FLS that is designed for classification and prediction over multi-agent dataset in decentralized data sources.

The general structure for a FGA agent process is graphically shown briefly in Fig. 2. As shown, the main role for each FGA agent is to construct its local model from the input dataset. In order to construct its local model, the FGA agent uses 70% of the dataset tuples along with their associated class as training data while 30% of the dataset tuples are kept separately independent and used as testing data for validation. The k-fold cross-validation is not used since it has some limitations [28]. The local model parameters of the FGA agent are evolved using two nested GAs. The outer GA evolves the fuzzy sets whereas the inner GA evolves the fuzzy rules. Hence, the chromosome of the outer GA encodes the fuzzy sets whereas the chromosome of the inner GA encodes the fuzzy rules. The global collaborative objective to be solved by the FGA agents is to interact to get the best formation of fuzzy sets and fuzzy rules that best describe and classify the dataset in a more efficient manner than one single FGA agent could. Fig. 3 shows graphically the nested GAs architecture developed for a FGA agent. In this figure, outer GA has population of N_1 chromosomes whereas inner GA has population of N_2 chromosomes. Given a set of fuzzy rules $R_i=\{R_1,\ldots,R_n\}$, a pseudo-code to design the inner GA is given below:

1) Initialize population of N_2 chromosomes.
2) **for** i = 1 to maxGeneration **do**
   a) Evaluate fitness of all chromosomes
   b) Select $N_2/2$ parents for reproduction using roulette te wheel selection
   c) Perform crossover operation on the selected pairs at some random point along each chromosome with probability P_c
   d) Perform mutation operation randomly with small probability P_m
   e) Replace the old population P_i with the new population P_{i+1} using elitist strategy

end for

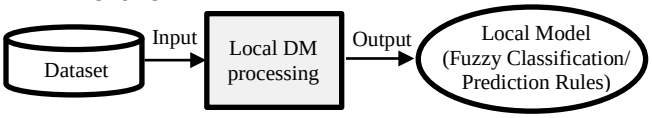

Fig. 2. The brief structure of a FGA agent constructing its local model from the dataset

Since the inner GA evolves fuzzy rules, the inner GA chromosome is designed such that it encodes a rule-base. The encoding scheme here represents each fuzzy rule as an integer array of fixed length equal to size of dataset tuple along with its assigned class. The integer elements in this array represent key values indexing the fuzzy sets utilized in the fuzzy rule, in order. The integer representations of all fuzzy rules are then strung together to form a single and variable-length array of fuzzy rules indices that constitute the inner GA chromosome encoding the rule-base. However, the approach mentioned earlier for controlling the rule-base size has been imposed. Fig. 4 graphically illustrates an example for designing an inner GA chromosome for a dataset having 2 attributes along with its associated class. In this example, the inner GA chromosome is assumed encoding a rule-base having 3 fuzzy rules, with respect to fuzzy sets keys shown in figure. Each inner GA chromosome, or complete FAM matrix, is then evaluated against the fitness function and the normal operations of selection, crossover, mutation, and replacement are applied.

However, in order to evaluate candidate solutions in inner GA, a chromosome from the outer GA must be utilized since it encodes the fuzzy sets definitions required in evaluating the rule-base encoded to obtain the accuracy of classification or prediction. In this case, the fitness function can be simply defined as the testing error:

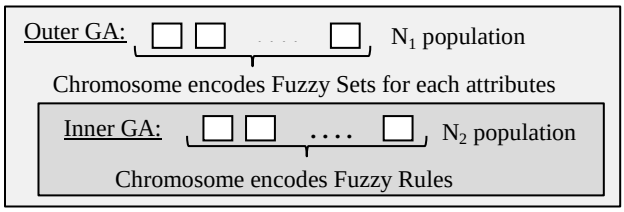

Fig. 3. The structure of nested GAs that evolves local model parameters of FGA agent

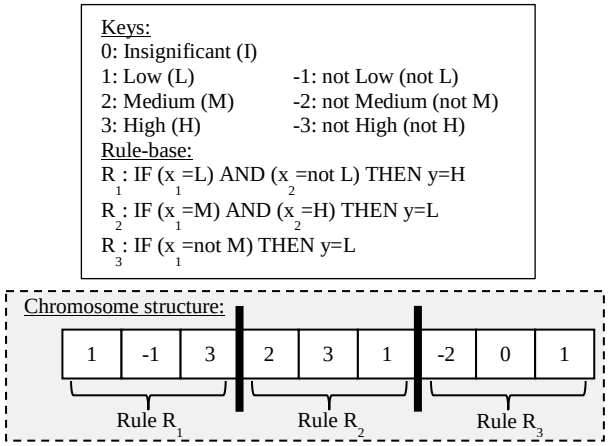

Fig. 4. Example of designing inner GA chromosome encoding rule-base of 3 fuzzy rules

$$f = \frac{1}{e} \quad (5)$$

Where: e is the classification error

However, in case of prediction, the testing error fitness function can be defined as:

$$f = 1 - E \quad (6)$$

Where: E is the root mean squared error (%)

For selecting potential inner GA chromosomes for reproduction, the strategy of roulette wheel technique is utilized due to its implementation simplicity [25]. In this strategy, the probability of selecting particular individual chromosome is proportional to its fitness. When selecting parents for reproduction, a single-point crossover operation is applied. In this operation, all fuzzy rules beyond that point in either individual chromosome are swapped. The crossover point on either individual is selected randomly and independently between encoded fuzzy rules borders. The resulting individuals are the offsprings. However, once upon again, the control approach for the size of a rule-base encoded in inner GA chromosome must be preserved for the reasons mentioned earlier. Fig. 5 graphically shows an example of single-point crossover operation between two inner GA parent individuals P_1 and P_2 having chromosome sizes of 3 and 5 fuzzy rules, respectively. In this example, the crossover point for parent P_1 occurred between fuzzy rules R_1 and R_2 whereas the crossover point for parent P_2 occurred between fuzzy rules R'_2 and R'_3. As shown in figure, the reproduction between P_1 and P_2 results in two offsprings O_1 and O_2 having equal size of 4 fuzzy rules.

To preserve and introduce genetic diversity, mutation operation is applied through generations. In this operation, inner GA chromosome is modified per key entry with small probability. To maintain useful genetic diversity and to improve inner GA performance, elitist strategy for replacement is applied. In this strategy, the fittest individuals are selected to replace the old population.

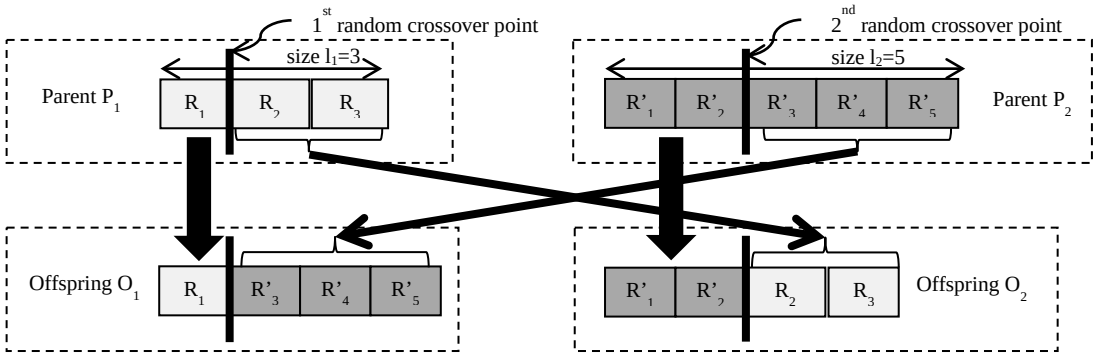

Fig. 5. Example of single-point crossover operation in inner GA where crossover points can be different positions

Given a set of n attributes $x_i=\{x_1,\ldots, x_n\}$ along with its associated class attribute y of a particular dataset, a pseudo-code to design the inner GA is very similar to the pseudo-code of inner GA mentioned above. However, the structure of outer GA chromosomes is dissimilar to the structure of inner GA chromosomes since outer GA evolves fuzzy sets whereas inner GA evolves fuzzy rules. Hence, outer GA chromosome is designed such that it encodes the fuzzy sets utilized in dataset attributes along with its associated class attribute. The encoding scheme here represents each attribute as an array of features defining the membership functions selected for the fuzzy sets utilized. Fig. 6 illustrates an example designing a structure encoding 3 fuzzy sets named "LOW" (L), "MEDIUM" (M) and "HIGH" (H) utilized for a continuous input attribute. In this example, triangular membership functions are selected to represent the fuzzy sets utilized, where fuzzy set L, M, and H are represented by left, regular, and right triangular membership functions, respectively. As shown in figure, the structure of the fuzzy sets utilized is represented as an array of the features defining their membership functions selected. These membership function representations for each attribute are then strung together to form a single and fixed-length array of features that constitute the outer GA chromosome encoding the overall fuzzy sets. Fig. 7 graphically shows the general structure of outer GA chromosome. However, fuzzy sets utilized are attributed dependent, as mentioned earlier.

The difficulty with outer GA is how to evaluate the fitness of candidate solutions. The fitness of each outer GA chromosome cannot be computed in isolation from the inner GA chromosomes since fuzzy sets encoded in outer GA chromosome are contributing in the evaluation of the rule-base

encoded in inner GA chromosomes. A simple solution to this problem is to construct generational fitness regions in inner GA by arranging the fitness of all inner GA chromosomes in equal size regions each generation. The fitness function for an outer GA individual is then can be defined as the average fitness of the majority region. Hence, each outer GA chromosome is evaluated against the fitness function and the operations of selection, crossover, mutation, and replacement are typically applied.

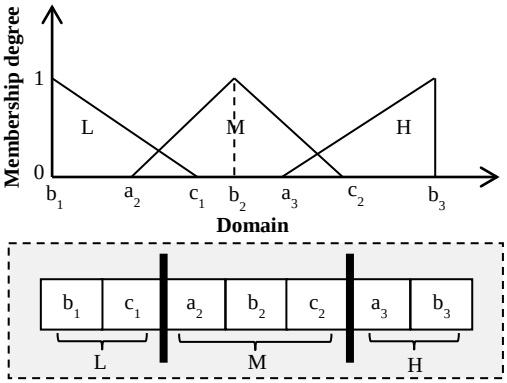

Fig. 6. Example of designing a structure encoding 3 fuzzy sets defined by triangular membership functions utilized for continuous input attribute

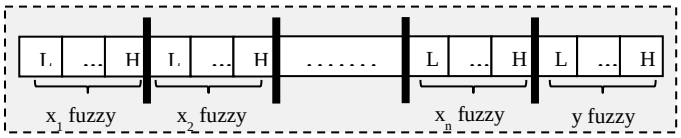

Fig. 7. The general structure of outer GA chromosome encoding fuzzy sets utilized in all dataset attributes along with its class attribute y

Similarly to inner GA, the outer GA utilizes the strategy of roulette wheel technique in selecting potential chromosomes for reproduction. In addition, outer GA applies single-point crossover operation in reproduction. However, in contrast to inner GA, the randomly selected crossover point in outer GA, on either individual, must have identical position between encoded attributes borders to preserve encoded fuzzy sets from being distorted. Fig. 8 graphically shows the single-point crossover operation in outer GA between two parent individuals P_1 and P_2 having fixed chromosome size of n attributes along with its associated class attribute. In this figure, the encoded fuzzy sets $FS_i=\{FS_1,\ldots, FS_{n+1}\}$ define the corresponding n input attributes along with the associated class attribute, respectively. As shown in figure, the crossover points for parents P_1 and P_2 have identical position and the reproduction results in two offsprings O_1 and O_2 having permanently identical size to their parents. Moreover, similarly to inner GA, the outer GA applies mutation and replacement operations through generations. Hence, outer GA chromosome is modified per feature entry with small probability and old population is replaced using elitist strategy. However, limits of fuzzy sets in domain must be preserved during mutation operation.

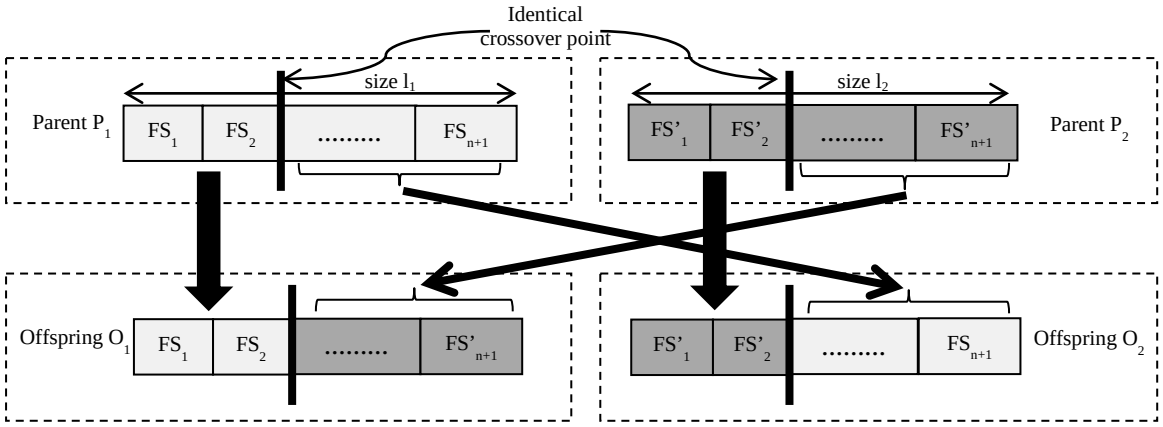

Fig. 8. Single-point crossover operation in outer GA where crossover points must have identical position

Fig. 9 graphically illustrates the detailed structure of a FGA agent showing its components and the interconnections between these components. As shown in figure, the FGA agent consists of two components namely, FLS and nested GAs. In this figure, the FLS uses the nested GAs in learning its KB to get the desired response. As can be seen, the FGA agent uses the structure developed to evolve best parameters in constructing its local model that best describe and classify the local input dataset. However, for a dataset distributed over decentralized sources, a Parallel Fuzzy-Genetic Algorithm (PFGA) framework has been developed. In this framework, all FGA agents are contributing and cooperating in parallel for constructing the final model that best describe and classify the overall distributed dataset. Fig. 10 graphically shows the structure of PFGA. As can be seen, all FGA agents are first allowed to construct their local models, as mentioned above. During the construction of local models, all FGA agents are allowed to exchange only their best fuzzy rules evolved each particular number of inner GA generations. For simplicity, the distribution strategy for fuzzy rules among FGA agents is applied sequentially and anticlockwise. By using this simple coordination strategy, changes to FGA agents are not only driven through genetic recombination and mutation but also driven through learning from FGA agent peers. In addition, the search process is not only guided by the fitness function but also guided by the interaction among peers.

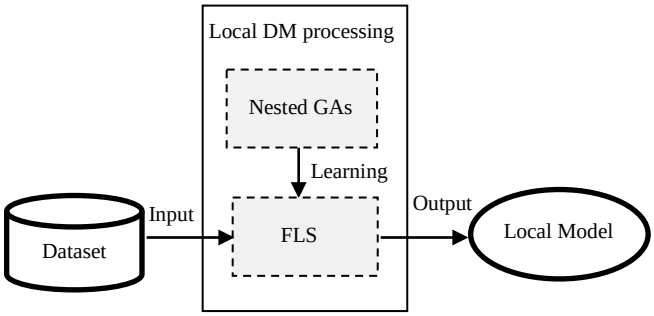

Fig. 9. The detailed structure of a FGA agent constructing its local model from the dataset

As can be also seen, local models of all FGA agents are finally aggregated in order to construct the final model. The adopted aggregation strategy keeps the most common fuzzy rules evolved in local models that give best accuracy and performance whereas eliminates redundant, conflicting and badly-defined fuzzy rules that perturbs the accuracy and performance. To summarize sequence of operations required for these tasks, a pseudo- code for the PFGA is given below:

1) Initialize nested GAs environments in all FGA agents

2) In parallel, build local model per FGA agent:

a) Run through outer & inner GAs populations of N_1 & N_2 individuals respectively and assign each pair of inner-outer GAs individuals to a FAM representing candidate solution.

b) Input a FAM matrix to classify or predict the training dataset.

c) Evaluate current individual of N_2 but evaluate current individual of N_1 each complete generation of N_2.

d) Each particular number of inner GA generations, exchange best fuzzy rules evolved among FGA agents.

e) If a prespecified termination condition is satisfied, stop algorithm execution. Otherwise, apply genetic operations to inner & outer GAs individuals, respectively, then return to step (2-a). In experiments conducted, total number of inner GA generations (N_g) is used as a termination condition.

3) Aggregate most common and good fuzzy rules evolved in all FGA agents to construct the final model.

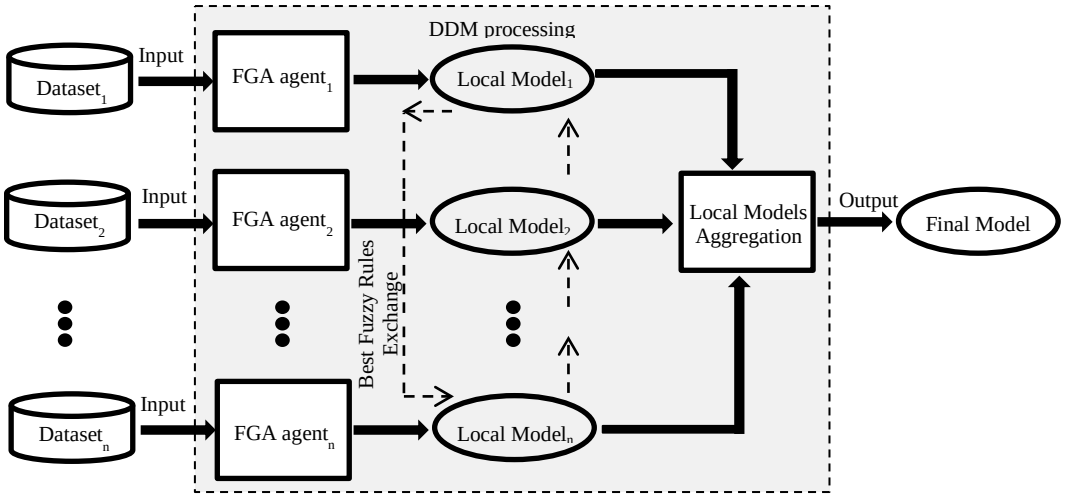

Fig. 10. The structure of PFGA that accepts a datasets distributed over decentralized data sources and construct the final model from these cooperative local models of FGA agents

By using the developed framework, storage cost exists in classical knowledge discovery techniques is avoided since distributed datasets are not required to be collected in a data warehouse. In addition, since fuzzy rules are the only thing allowed to be exchanged, many issues such as communication cost, privacy and sensitivity of data are tackled effectively and efficiently. Furthermore, the parallelism and cooperation exist in this framework allows FGA agents to construct the final model in a more efficient manner than one single FGA agent could.

V. Experimental Results and Discussion

The following results were obtained from a series of experiments conducted using the developed PFGA framework described above, to evolve a model that best describe and classify a distributed datasets. The series of trial runs were performed on i5-3.2 GHz (× 4) system running 32-bit Windows 7 ultimate and having 4.00 GB ram. The performance of the proposed algorithm is measured using two different population sizes, two different numbers of FGA agents and three different policies for best fuzzy rules exchange among these agents. Five different benchmark datasets, listed in Table1, were employed in trial runs which were available from the dataset repository of Knowledge Extraction based on Evolutionary Learning (KEEL) [29]. The trial runs involve using the proposed algorithm against two evolutionary fuzzy rule learning algorithms implemented in KEEL software tool version 3.0 with P_c=0.5 and P_m=0.1 [30]. In classification, the proposed algorithm is used against the Fuzzy Hybrid Genetics-Based Machine Learning (FH-GBML) algorithm [31]. In prediction, the proposed algorithm is used against the Genetic-Base Fuzzy Rule Base Construction and Membership Function Tuning (GFS-RB-MF) algorithm [32].

In the first set of experiments, a population size N_1=N_2=20 is used. In the second set of experiments, a population size N_1=N_2=40 is used. For each population size, six experiments

are conducted per dataset then repeated five times to study how the population size (N_1, N_2), number of agents (N_a), and the exchange policy for best fuzzy rules among these agents affect the system behavior.

In each of these six experiments, the system is allowed to run first for N_g=500 then for N_g=1000. For each of these inner GA generations numbers (N_g), computational experiments are performed to examine various specifications of FGA agents numbers (N_a) along with different exchange policies for best fuzzy rules among these agents in different intervals of N_g. More specifically, the examined numbers of FGA agents N_a= 5, 10 whereas the examined exchange policies for best fuzzy rules among these agents are: "No Exchange" (Pol_1), "Exchange each 5 generations" (Pol_2), and "Exchange each 10 generations" (Pol_3). The best fitness parameter is recorded for each generation. The average of best fitness parameter is then computed for the total of twelve experiments conducted per dataset, resulting in 500 and 1000 values when using N_g=500 and N_g=1000, respectively. A time series of this averaged parameter is then plotted using the number of inner GA generations (N_g) for time axis. Specifically, the average fitness is computed for the twelve experiments conducted using the PFGA framework and the average fitness is computed for the experiments conducted using the FH-GBML and GFS-RB-MF algorithms.

TABLE I. LIST OF DATASETS EMPLOYED IN EXPERIMENTS

Name of Dataset	Number of instances	Number of attributes (Real, Integer, Nominal)	Number of classes
Banana	5300	2 (2/0/0)	2
haberman	306	3 (0/3/0)	2
saheart	462	9 (5/3/1)	2
car	1728	6 (0/0/6)	4
plastic	1650	2 (2/0/0)	-

Fig. 11 shows the results when a population size N_1=N_2=20 is used in classifying "banana" dataset for N_g=500 and N_a=5. As can be seen, the best fitness almost reached the same level when using the three exchange policies for best fuzzy rules in PFGA framework. However, it can be also seen that Pol_2 policy outperforms other exchange policies for best fuzzy rules in short run. The reason is that the FGA agents do more exploitation for best fuzzy rules in short interval of times. Moreover, the same figure shows that the convergence time for PFGA framework is slower than the FH-GBML algorithm. This is due to the slow exploration in real attributes. On the other hand, Fig. 12 shows the results when a population size N_1=N_2=40 is used in classifying "banana" dataset for N_g=1000 and N_a=5. As can be seen, the Pol_2 policy performs better in long run and almost reached best fitness value of 0.78. The reason is that using larger population size increases the diversity and, consequently, the exploration for better solutions by FGA agents that exploit best fuzzy rules in short interval of times.

Fig. 13 shows the results when a population size N_1=N_2=20 is used in classifying "haberman" dataset for N_g= and N_a=10. As can be seen, the best fitness almost reached a value of 0.74 after 20 generations in PFGA framework when using policies that exchange best fuzzy rules among FGA agents whereas it reached a same value after 80 generations in a policy that doesn't exchange best fuzzy rules among FGA agents. This highlights the importance of exchanging best fuzzy rules among FGA agents. As can be seen in Fig. 14, the best fitness using PFGA framework almost reached the same best fitness value of 0.78 using FH-GBML algorithm when classifying "haberman" dataset using N_1=N_2=40 for N_g=1000 and N_a=10. The reason is that the exploration in integer attributes is faster than in exploration in real attributes which results in fast fuzzy rules evolving.

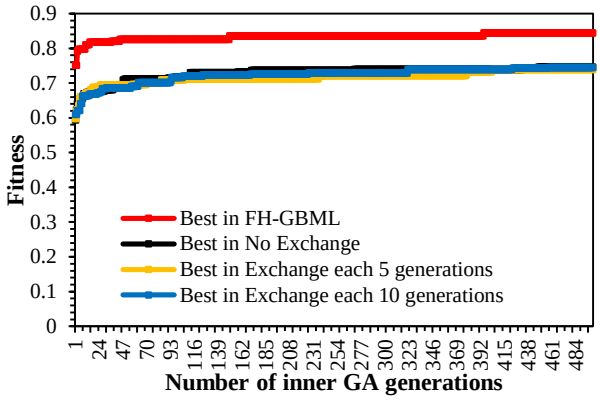

Fig. 11. Example of best fitness data for N_1=N_2=20 using "banana" dataset for N_g=500 and N_a=5

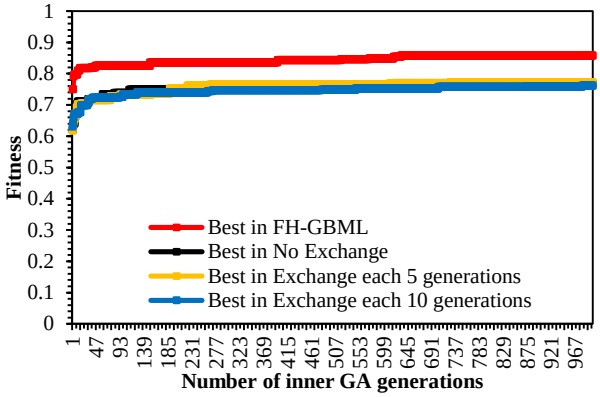

Fig. 12. Example of best fitness data for N_1=N_2=40 using "banana" dataset for N_g=1000 and N_a=5

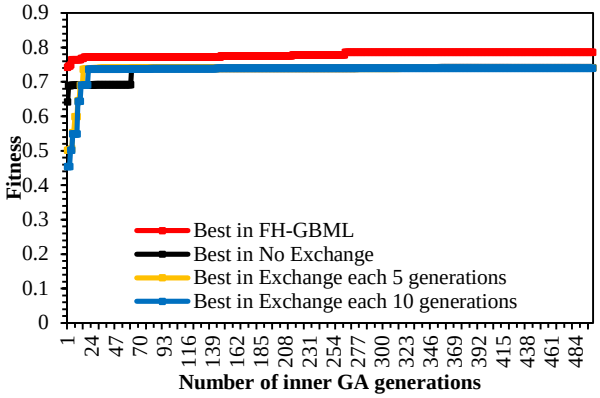

Fig. 13. Example of best fitness data for N_1=N_2=20 using "haberman" dataset for N_g=500 and N_a=10

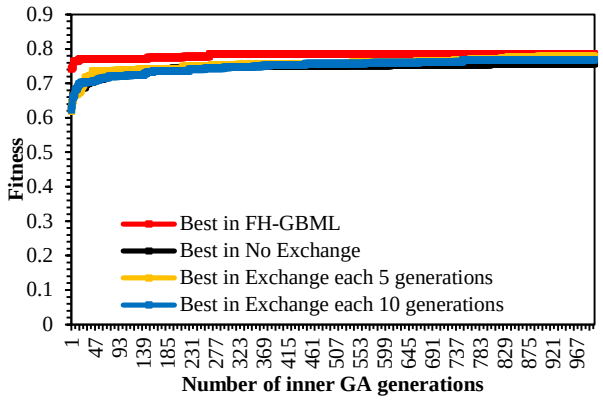

Fig. 14. Example of best fitness data for N_1=N_2=40 using "haberman" dataset for N_g=1000 and N_a=10

Fig. 15 shows the results when a population size N_1=N_2=40 is used in classifying "saheart" dataset for N_g=1000 and N_a=10. As can be seen, the best fitness curve when using Pol_2 policy converges faster than other policies in PFGA framework. However, using the same configuration in Fig. 15, Fig. 16 shows that the FH-GBML algorithm outperforms the PFGA framework. The reason is that the exploration takes longer time when increasing number of attributes in dataset especially when it has real attributes.

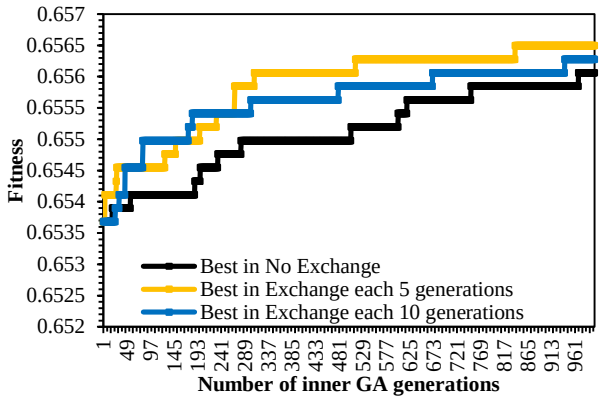

Fig. 15. Example of best fitness data for N_1=N_2=40 using "saheart" dataset for N_g=1000 and N_a=10

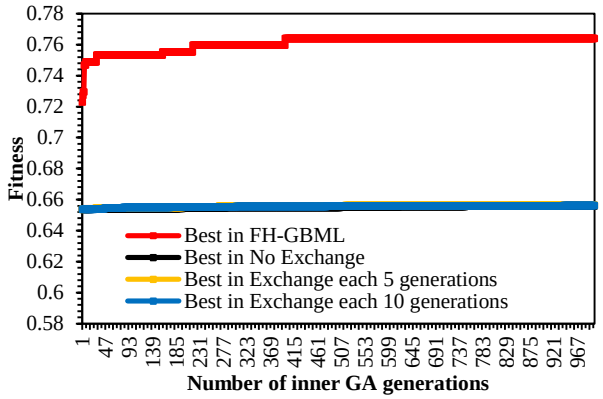

Fig. 16. Example of best fitness data for N_1=N_2=40 using "saheart" dataset for N_g=1000 and N_a=10

Fig. 17 shows the results when a population size N_1=N_2=40 is used in classifying "car" dataset for N_g=1000 and N_a=5. As can be seen, the best fitness curve when using Pol_2 policy converges slower than Pol_3 policy in PFGA framework whereas it converges faster in Fig. 18. The reason is that using short interval of times for exchanging best fuzzy rules among small number of FGA agents increases the exploitation whereas using larger number of FGA agents increases the exploration. Comparing with results from "saheart" dataset, it can be also seen in Fig. 18 that PFGA framework performs better when number of attributes decreases where it reached almost to fitness value of 0.716 instead of 0.656.

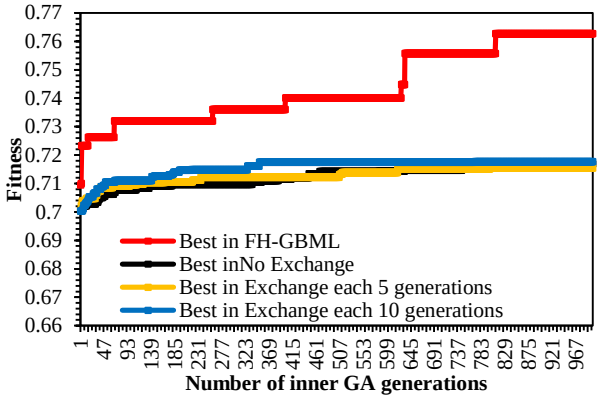

Fig. 17. Example of best fitness data for N_1=N_2=40 using "car" dataset for N_g=1000 and N_a=5

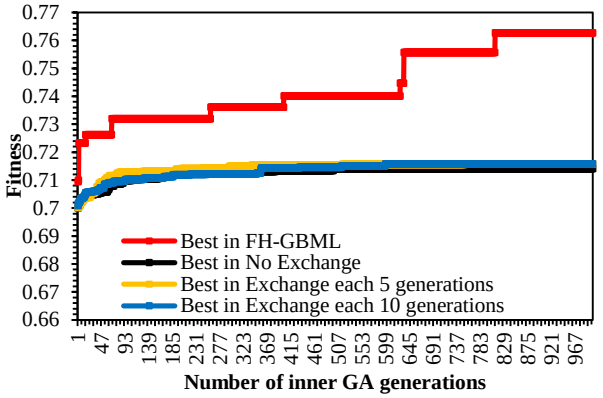

Fig. 18. Example of best fitness data for N_1=N_2=40 using "car" dataset for N_g=1000 and N_a=10

Fig. 19 shows the results when a population size N_1=N_2=40 is used in predicting "plastic" dataset for N_g=1000 and N_a=10. As can be seen, the GFS-RB-MF algorithm outperforms the PFGA framework. The reason is that predicting continuous valued function takes longer time than predicting categorical labels in FLSs. Table 2 shows the summarized average results for the experiments conducted when a population size N_1=N_2=40 is used for N_g=1000 and N_a=10. As shown in the last column, the average computation time were long for some dataset compared with other algorithms (e.g., more than 40 minutes for plastic dataset). However, the first column shows that the average number of fuzzy rules evolved for some dataset has less size along with good accuracy compared with other algorithms (e.g., around 6 fuzzy rules for "haberman" dataset).

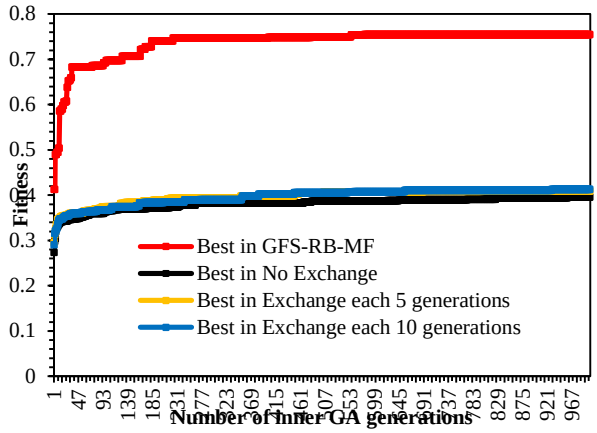

Fig. 19. Example of best fitness data for $N_1=N_2=40$ using "plastic" dataset for $N_g=1000$ and $N_a=10$

VI. CONCLUSION

A new framework for classification and prediction is proposed as a main contribution to the scientific community. The developed Parallel Fuzzy-Genetic Algorithm (PFGA) framework provides flexible mechanism for processing distributed data and offers significant advantage over classical techniques which help to reach all network-related business. Several experiment have been conducted with various specification of population sizes, numbers of FGA agents along with different exchange policies for best fuzzy rules among these agents in different intervals of generations. Small population size does not provide sufficient diversity in individuals for the optimum fuzzy rules to be evolved. Using small interval of times when exchanging best fuzzy rules among FGA agents incorporates more exploitation over exploration whereas using larger number of FGA agents compromises between them. However, using the developed framework with decreasing number of attributes has been shown that it has good accuracy and more efficient in performance and comprehensibility of linguistic rules compared to FH-GBML and GFS-RB-MF models implemented in KEEL software tool.

TABLE II. RESULTS OF PFGA FRAMEWORK VERSUS FH-GBML AND GF-RB-MF ALGORITHMS WHEN $N_1=N_2=40$, $N_G=1000$, AND $N_A=10$

Name of Dataset	Algorithm	Number of fuzzy rules	Accuracy in testing (%)	Time (minutes)
banana	PFGA-Pol$_1$	7.2	75.69	83.25
	PFGA-Pol$_2$	10.6	77.85	85.5
	PFGA-Pol$_3$	10.1	76.86	77.5
	FH-GBML	30	85.83	47
haberman	PFGA-Pol$_1$	6.2	74.14	9.5
	PFGA-Pol$_2$	5.6	74.18	10.75
	PFGA-Pol$_3$	7.1	74.15	10
	FH-GBML	27	77.54	3
saheart	PFGA-Pol$_1$	3.8	65.61	18.5
	PFGA-Pol$_2$	5	65.65	18
	PFGA-Pol$_3$	3.5	65.63	17.5
	FH-GBML	27	76.41	7
car	PFGA-Pol$_1$	6.7	71.41	32.5
	PFGA-Pol$_2$	7.4	71.59	32.5
	PFGA-Pol$_3$	7.3	71.59	32.75
	FH-GBML	29	76.27	8.15
plastic	PFGA-Pol$_1$	17.7	39.58	41.75
	PFGA-Pol$_2$	19.3	40.93	43.75
	PFGA-Pol$_3$	20.4	41.24	42.5
	GFS-RB-MF	9	75.49	1.5

This work can be extended in several directions. For example, part of the dataset can be exchanged along with best fuzzy rules. Furthermore, other exchange policy schemes can be employed in addition to parallelizing data level along with algorithm level per data source such that FLS to evolve to dynamic optimum number of fuzzy rules.

REFERENCES

[1] M. Sharma, "Data mining: a literature survey", in International Journal of Emerging Research in Management & Technology, 3(2), 2014.

[2] X. Hu, "Data mining and its applications in bioinformatics: Techniques and methods", in 2011 IEEE International Conference on Granular Computing (GrC), Nov. 8-10, IEEE Xplore Press, Kaohsiung, pp. 3-3, 2011.

[3] G. Shmueli, , N. R. Patel, and P. C. Bruce, Data Mining for Business Analytics: Concepts, Techniques, and Applications in XLMiner, 3rd Ed., John Wiley & Sons. New Jersy, 2016.

[4] T. Nasukawa, and T. Nagano, "Text analysis and knowledge mining system", in IBM Systems Journal, 40(4), pp. 967-984, 2001.

[5] S. Jaideep, R. Cooley, M. Deshpande, and P. Tan, "Web usage mining: discovery and applications of usage patterns from web data", in SIGKDD Explorations Newsletter, 1(2), pp. 12-23, 2000.

[6] J. Han, J. Pei, and M. Kamber, Data mining: concepts and techniques, 3rd Ed., Elsevier, 2011.

[7] N. Jain, and V. Srivastava, "Data Mining techniques: a survey paper", in IJRET: International Journal of Research in Engineering and Technology, 2(11), pp. 2319-1163, 2013.

[8] A. Fernández, S. García, J. Luengo, E. Bernadó-Mansilla, and F. Herrera, "Genetics-based machine learning for rule induction: taxonomy, experimental study and state of the art", in IEEE Transactions on Evolutionary Computation, 14(6), pp. 913-941, 2010.

[9] Q. Yang, and X. Wu, "10 challenging problems in data mining research", in International Journal of Information Technology & Decision Making, 5(04), pp. 597-604, 2006.

[10] A. Fariz, J. Abouchabaka, and N. Rafalia, "Using multi-agents systems in distributed data mining: a survey", in Journal of Theoretical & Applied Information Technology, 73(3), 2015.

[11] S. V. S. G. Devi, "A survey on distributed data mining and its trends", in IMPACT: International Journal of Research in Engineering & Technology, 2(3), pp. 107-120, 2014.

[12] D. Khan, "CAKE – Cassifying, associating and knowledge discovery - an approach for distributed data mining (DDM) using parallel data mining agents (PADMAs)", in WI-IAT '08. IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, Dec. 9-12, IEEE Xplore Press, Sydney, NSW, pp. 596-601, 2008.

[13] M. Kantardzic, Data mining: concepts, models, methods, and algorithms, 2nd Ed., John Wiley & Sons. New Jersy, 2011.

[14] A. P. Engelbrecht, Computational intelligence: an introduction, 2nd Ed., John Wiley & Sons, Chichester. England, 2007.

[15] D. Driankov, H. Hellendoorn, and M. Reinfrank. An Introduction to Fuzzy Control, 2nd Ed, Springer-Verlag. Berlin, 2013.

[16] R. G. Hercock, Applied Evolutionary Algorithms in Java, Springer-Verlag Ltd. London, 2013.

[17] D. Driankov and A. Saffiotti (eds.), Fuzzy Logic Techniques for Autonomous Vehicle Navigation, Physica-Verlag. Berlin, 2013.

[18] I. Baturone, , A. Barriga , S. S. Solano, C. J. Fernandez, and D. R. Lopez, Microelectronic design of fuzzy logic-based systems, CRC Inc. Press, Boca Raton, FL, USA. 2000.

[19] S. Madhusudhanan, M. Karnan and K. Rajivgandhi, "Fuzzy based ant miner algorithm in datamining for hepatitis", in ICSAP '10. International Conference on Signal Acquisition and Processing, FEB. 9-10, IEEE Xplore Press, Bangalore, pp. 229-232, 2010.

[20] J. Gomez, and D. Dasgupta, "Evolving fuzzy classifiers for intrusion detection", in Proceedings of the 2002 IEEE Workshop on Information Assurance, JUN. 17, IEEE Computer Press, New York, 6(3), pp. 321-323, 2002.

[21] O. Cordón, "A historical review of evolutionary learning methods for mamdani-type fuzzy rule-based systems: designing interpretable genetic fuzzy systems", in International Journal of Approximate Reasoning, 52(6), pp. 894-913, 2011.

[22] P. Singhala, D. Shah, and B. Patel, "Temperature control using fuzzy logic", in International Journal of Instrumentation and Control Systems (IJICS), 4(1), 2014.

[23] A. Mittal and M. Singh, "Automatic fuzzy rule base generation from numerical data: cuckoo search algorithm", in International Journal of Computer Applications, 133(3), pp. 7-12, 2016.

[24] D. Corne, et al. (eds.), New ideas in optimization, McGraw-Hill Ltd. Maidenhead, UK, England, 1999.

[25] D. A. Coley, An Introduction to Genetic Algorithms for Scientists and Engineers, World Scientific Publishing Co. Pte. Ltd, Singapore, 2001.

[26] S. T. Welstead, Neural Network and Fuzzy Logic Applications in C/C++. 1st Ed., John Wiley & Sons Inc. USA, 1994.

[27] O. Nasaroui, F. Gonzalez, and D. Dasgupta, "The fuzzy artificial immune system: motivations, basic concepts, and application to clustering and web profiling", in Proceedings of the IEEE International Conference on Fuzzy Systems, May 12-17, IEEE Xplore Press, Honolulu, HI, USA, pp. 711-716, 2002.

[28] K. H. Esbensen, and P. Geladi, "Principles of proper validation: use and abuse of re-sampling for validation", in Journal of Chemometrics, 24(3-4), pp. 168-187, 2010.

[29] J. Alcalá-Fdez, et al., "KEEL data-mining software tool: data set repository, integration of algorithms and experimental analysis framework", in Journal of Multiple-Valued Logic and Soft Computing, 17(2-3), pp. 255-287, 2011.

[30] J. Alcalá-Fdez, et al., "KEEL: A software tool to assess evolutionary algorithms to data mining problems", in Soft Computing 13(3), pp. 307-318, 2009.

[31] H. Ishibuchi, T. Yamamoto and T. Nakashima, "Hybridization of fuzzy GBML approaches for pattern classification problems", in IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 35(2), pp. 359-365, 2005.

[32] A. Homaifar, and E. McCormick, "Simultaneous design of membership functions and rule sets for fuzzy controllers using genetic algorithms", in IEEE Transactions on Fuzzy Systems, 3(2), pp. 129-139, 1995.