The Generic Feature Extraction Model Using LEM (ID3)

M.Elemam.Shehab[1] , K.Badran[1] and Gouda Ismail Salama[2]

[1] Information System Department, Egyptian Armed Forces, [2]Egyptian Armed Forces

melemam@hotmail.com,dr_gouda80@yahoo.com

Abstract

In this paper, we presents a new classification algorithm called LEM(ID3), which is based on the techniques from the learnable evolution models (LEM) to improve convergence and accuracy of the algorithm and use of ID3 in order to construct the tree used in classification. We converted LEM from optimization domain to classification domain and then examine the feature extraction problems and show that learning evolutional can significantly enhance the performance of pattern recognition systems with simple classifiers. We have applied this model to real world datasets from the UCI Machine Learning databases to verify our approach and compare our proposed approach with other reported results. We conclude that our algorithm is able to produce classifiers of superior (or equivalent) performance to the conventional classifiers examined.

Keywords: *Feature Extraction, Pattern Recognition, learnable Evolution Model, Dynamic threshold classifier.*

1. Introduction

The Learnable Evolution Model (LEM, [1]) was introduced in 2000, as a highly generalized hybrid approach to *optimization*, in which the overall idea is to run repeated phases of evolution and learning in series. Each 'evolution' period is informed in some way by the previous 'learning'period. In the learning periods, the general idea is to use a machine learning technique to infer relationships between gene values and fitness. For example, we may start by running an evolutionary algorithm for 10 generations; then we halt the evolutionary algorithm and do some learning (perhaps a neural network, or an AQ rule learner – as in the original LEM – and so on). The result of the learning phase is then used in the next period of evolution. The way in which learning influences evolution is not restricted by (our view of) the LEM framework. E.g. the learned model could be used to predict the fitness (or fitness category) of children before they are evaluated, and the evolution phase discards, without evaluation, children that are predicted to be particularly unfit. Or, the learned model may be used to constrain genetic operators in a beneficial way. Or, the learned model may be used to 'repair' children that are otherwise generated by standard operators. Evolution then continues for another few generations, resulting in new data for the learning method (chromosomes and their evaluated fitnesses), and so it continues. The learning method in most LEM work [1] is AQ15 [2], and the reported results tend to be very promising in optimization domain, with improvements in solution quality and dramatic speedup when compared to the 'without learning' equivalent EA.In application-oriented work, a multiobjective LEM-based approach, using C4.5 as the learning method, was found to significantly speed up and improve solution quality for large-scale problems in water distribution networks [3]. The developers of the LEM framework are continually updating the"AQ15" version and continue to report impressive results, albeit on a limited suite of test functions. Meanwhile, of course, Estimation of Distribution Algorithms (EDAs) [4] can also be viewed as learning/evolution hybrids, with the emphasis on building and maintaining models of fit chromosomes. While EDAs focus on modeling (i.e. search is guided closely by statistical models, with new sample points generated directly from the model), in LEM the evolutionary component is responsible for the search (i.e. new points are sampled mainly in the usual way by using genetic operators), with guidance from learning. Recent results using LEM3 compare EDAs and LEM3 [4], and report better quality results than a good EDA on two hard functions, with between 15 and 230 fold speedup of LEM3 over the EDA. Also, of course, hybrids of EDA and GAs (e.g. [5], [6]) are also successful optimizers. LEM[13] is similar in style to a hybrid of EDA and EA. The design and application of LEM is clearly worth considerably more research. The speedup reported in several papers that apply LEM – that is, the reduction in the number of fitness evaluations needed to reach high quality results, is of particular interest for many important applications in which fitness evaluation is costly.. We look of classification problem as searching for the optimum features in optimization problem.

This paper is structured as follows: Section 2 explains classification with dynamic threshold where Section 3 illustrates LEM then Section 4 presents used datasets where Section 5 introduces a performance study of LEM-ID3 and finally section 6 contains conclusion.

2. Classification using Dynamic Threshold

Over the years much effort has been expended in the pattern recognition community on finding a best classifier (e.g. [7, 8] rocket), the conclusion of which is that there is no single classifier which is best for every problem. In binary classification problems a feature extractor used to map multi-dimensional input patterns into a one-dimensional decision space, as shown in Figure 1 Using a fixed threshold combines the feature extraction stage and the classification stage. A dynamic threshold is therefore needed to minimize the misclassification rate during training.

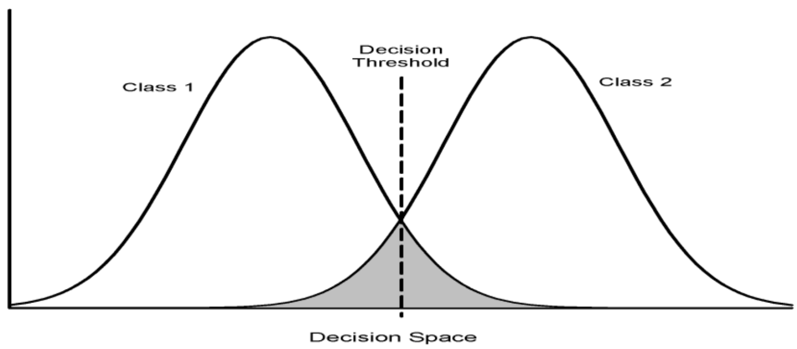

Fig.1 two class label with fixed threshold

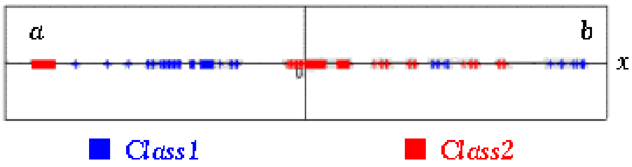

Figure 2 Output of binary genetic programming tree classifier

Golden Section search is used to search for this optimal threshold as the misclassification error represents a unimodal function, *f* over the interval *[a…b]*, where *a*, *b* are the extremes of the mapped real values, which means *f(x)* has only one minimum in *[a…b]*. Iteratively, golden search algorithm tries to identify the point with the minimum misclassification error. Golden Section search is terminated when there is no further improvement can be achieved. This method finds the dynamic threshold in an efficient way.

2.1Classifier Evaluation

Besides classification accuracy of a classifier, other factors should be taken into consideration such as [9]:

- The training/testing time with respect to the scale of the application.
- The interpretability of the results
- The ability of the classifier to embed different misclassification costs.

Training error cannot be used to compare the performance of two classifiers since a more complex classifier with more free parameters would have a better training error but will be likely to generalise worse on unseen patterns. Therefore, data are commonly partitioned into training and validation datasets to judge the generalisation performance of the classifier. In many cases obtaining datasets which are large enough to be split into statistically-meaningful parts is difficult. Therefore, experiments are repeated several time to average over the random fluctuations which occur while splitting the data. Then, some statistical test is performed to accept or reject the null hypothesis that there is a significant difference between the test error rates of the two classifiers at a specific confidence level.

When statistical significance between results is reported in the literature, the typical approach is to perform *k*-fold cross-validation where data are split into *k* (maybe 10) partitions and the experiment is repeated *k* times. In each experiment *k-1* partitions of the data are merged to form the training dataset while the last partition is used as the test dataset. Then a paired *t*-test is performed on the results of the *k*-fold cross-validation.

Dietterich [10] has pointed-out this test is unsound due to the violation of the implicit assumptions about independence. Any two training sets will share *k-2* partitions of the original data. Thus the paired *t*-test suffers from high type error I, explained in Table 3.II, leading to differences being declared statistically significant more frequently than they should. Dietterich has proposed an empirical cross-validation test named the *5 × 2 cv t*-test which splits the dataset into two folds and repeats this for five different splitting. For each splitting, one of the datasets is used as a training set and the other as the validation data; the experiment is then repeated, interchanging the roles of the datasets.

3. LEM (AQ) and the LEM (ID3) frameworks

In LEM(AQ), an initial population is divided into high-performance (H-group) and low performance (L-group), groups are classified according to their fitness; these two groups are saved as high and low training examples for AQ learning algorithm. The output of learning process is a set of rules which predict a class label (i.e. H-group or L-group). LEM (AQ) then proceeds with an otherwise normal EA, except that the operators generate new individuals only with gene values within the ranges reasonable by the recently learned rules. LEM (AQ) then continues for a specific amount of generations, and then stop for more learning based on the current population. This feeds into the next stage of evolution, and so on. LEM (AQ) has many additional details that mediate the transitions between learning and evolution, and we refer readers to [1],[12] for more details. LEM(AQ) is one instantiation of the wider LEM framework, which allows for creativity in the choices of learning method, and the way in which learning and evolution interact. In this paper, we continue to investigate the LEM framework, and focus on an approach in which the learning mechanism is ID3 because is an easy and effective algorithm.

3.1The LEM (ID3) algorithm

We assume readers are familiar with the ID3 decision tree learning algorithm [7]. We note only that standard ID3 requires discrete, nominal data (rather than real values), and within LEM(ID3) it is always treats a real-valued range as a set of discrete equal-width intervals. As we will see, this is initially set to 2 intervals for each gene, but adapts during the search. In LEM(ID3), ID3 is employed to learn from a population of evaluated chromosomes. Each chromosome is labelled as either high-performance or low-performance (see below), and ID3 learns a tree that predicts this label from the gene values. Further details are given next. LEM(ID3) contains two main components: evolution and learning. In the evolution component, a standard evolutionary algorithm is applied. In the learning component, ID3 is used, in a way detailed below.LEM(ID3) divides the current population into a high performance (H-group) and low-performance (L-group) groups according to their fitness values and a given threshold (say, 30% - that is, the fittest 30% from the H-group and the worst 30% from the L-group). ID3 then uses the H-group and L-group as the training data to construct the decision tree, which is then transformed into a set of rules. These sets of rules are the hypotheses that differentiate between the two groups. New individuals are generated by instantiating these hypotheses, or by evolution, or are randomly generated. The learning mode continues until there is no better individual generated for a certain number of generations, or the diversity of the population is too small. The evolution mode begins when the learning mode is finished, offering the opportunity to escape from local optima and also preserve diversity, which is crucial for success in the subsequent learning phase. Evolution continues for a certain number of generations, before the learning phase begins again. The overall pseudo-code of LEM(ID3) is set out here as Algorithm 1, with some components elaborated further later in the paper.

1) The Learning Mode: In the learning mode, there are three main steps. First, select training examples. Second, learn and generate hypotheses. Third, instantiate hypotheses and generate new individuals.

• To select the training examples, we use 'population based selection' ([1]), in which we specify that a given percentage of the population will be in the H-group and a given percentage will be in the L-group. We use 30% in both cases – i.e., after sorting the individuals by fitness value, the top 30% are placed into the Hgroup and the lowest 30% are put in the L-group.

An alternative discussed in [1], but which is more problematic to implement, is based on specifying fitness value thresholds.

• Learn and generate hypotheses: Given the training examples, in LEM(ID3) we use ID3 to construct a decision tree. The construction procedure is straightforward, as discussed above. The resulting tree can be transformed into a set of rules, which can then be seen as hypotheses discriminating H-group and L-group individuals. An example decision tree produced during a LEM (ID3) .

4. UCI Datasets.

The datasets used in the current work are real world datasets from the UCI Machine Learning databases:

1) **Glass – 163** instances with nine attributes -This dataset has been converted to a two-class problem by seeking to distinguish between float glass and non-float glass.
2) **BUPA Liver Disorders (BUPA)** Prediction of whether a patient has a liver disorder. There are two classes, six numerical attributes and 345 records.
3) **Wisconsin Diagnostic Breast Cancer (WDBC)** This dataset has been discussed before by Mangasarian et al. [35]. 569 examples with thirty numerical attributes.
4) **Pima Indians Diabetes (PID)** Records with missing attributes were removed. This dataset comprises 532 complete examples with seven attributes.

Table 1: Details of UCI Datasets Used in This Work

Name	No of features	Size	Distribution
Glass	9	163	87 (Float) + 76 (Non-float)
BUPA	6	345	200 (Normal) + 145 (Diseased)
WDBC	30	569	357 (Benign) + 212 (Malignant)
PID	7	532	355 (Normal) + 177 (Diabetic)

5. Testing methodology

In the proposed model we divide the methodology for testing into four consecutive steps in order to achieve the classification and show the results.

1- Selecting chromosome Representation

Firstly we represent the chromosome by taking the real values of attributes as a weights of the attributes taking 0 as a fixed threshold between two class labels as shown in fig1After that we replace fixed threshold technique with dynamic threshold (Golden search) depends on the values of the attributes and then put the adaptive threshold to effectively find the boundary value between two class which enhanced the efficiency of our classifier.

2- Selecting Learning algorithm

With the original LEM, in which the learning mechanism was AQ and the evolution/learning interface was more sophisticated. It is surprising and interesting to see more Algorithms such as KNN,C4.5and ID3 are clearly recommended to explore for large-scale tasks in which savings in evaluation time are necessary.In our work we use ID3 as a learning algorithm

3- Apply to Different Datasets and Analyze the Results

6.Results

In this section we address our guiding issue of producing a generic methodology by examining performance across a wide range of classification problems from the UCI Machine Learning [8] dataset [9] databases., we suggest that our methodology is inventing a near-optimal classifier for every dataset to which it is applied; in some instances these evolved classifiers may be similar to existing classifiers and in other cases, quite unlike any known classifier paradigm. The key issue is that the generation of the feature extraction stage is being driven by the notion of optimality. We consider an extensive set of comparisons across learning problems. For each dataset we make a statistical comparison of the classification performance between our algorithm and a range of established classifiers. If our conjecture about the generic power of our method is supported, perform at least as well as any other classifier (and in a number of cases, better).In addition, we compare where possible with previously reported evolutionary feature extraction techniques.

1.Glass

The first dataset used in our work in order to assessment the performance of our classifier is Glass which has 163 instances with nine attributes ,This dataset has been converted to a two-class problem by seeking to distinguish between float glass and non-float glass.figure 4 shows the using of learning only and evolution only and the use of LEM as a hybrid approach between learning and evolution,x axis represent the number of evolution rounds and y axis represent minimum error.

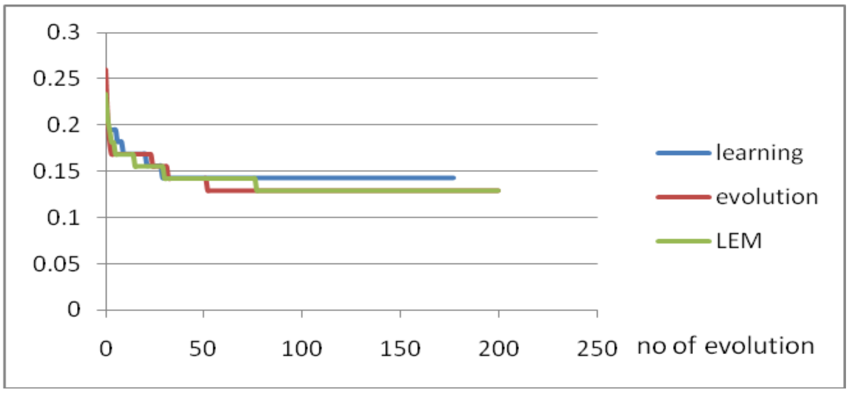

Figure 4: Comparison among LEM and both evolution only and learning only

at figure 4 we conclude that LEM(ID3) go to the minimum error faster than the using of evolution only or learning only and expecting that speed will be more noticable in large size datasets.

2. Pima Indians Diabetes (PID)

This dataset comprises 532 complete examples with seven attributes

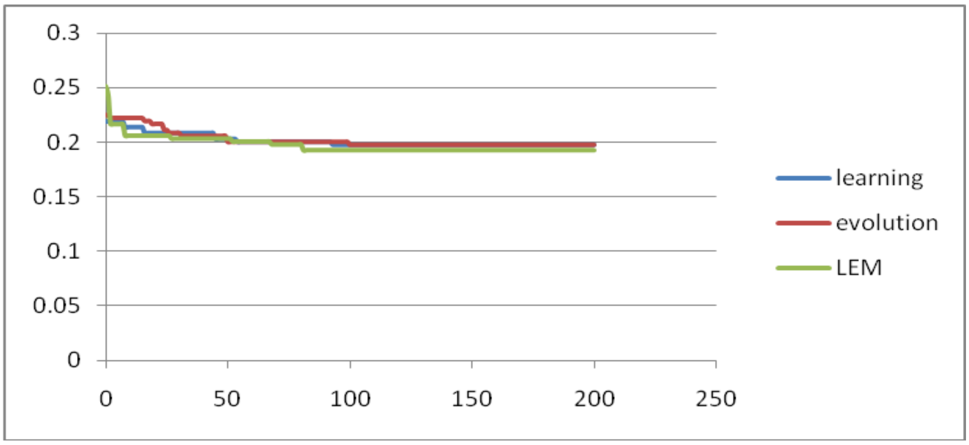

Figure 5: Comparison among LEM and both evolution only and learning only

at figure 5 for PID datasets we see that using learning phase before evolution phase guide the LEM (ID3) fastly to go to the minimum error.

3. **BUPA Liver Disorders (BUPA)** is the third dataset in our study Prediction of whether a patient has a liver disorder. There are two classes, six numerical attributes and 345 records. Figure 6 also show the benefits of using LEM instead of using separate evolution or learning individually.

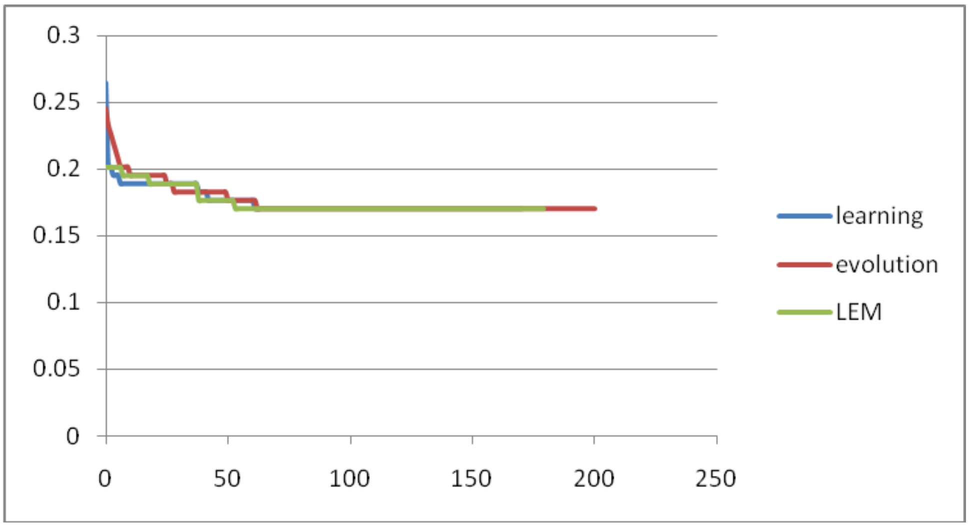

Figure 6: Comparison among LEM and both evolution only and learning only

4. Wisconsin Diagnostic Breast Cancer (WDBC)

The last dataset used to verify the LEM(ID3), This dataset has been discussed before by Mangasarian et al. [11]. 569 examples with thirty numerical attributes. Also show the benefits of using LEM instead of using separate evolution or learning individually.

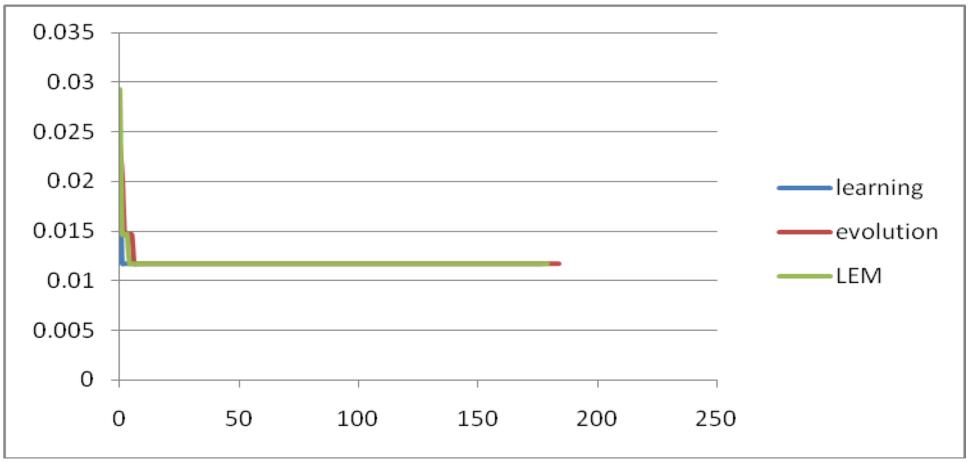

Figure 7: Comparison among LEM and both evolution only and learning only

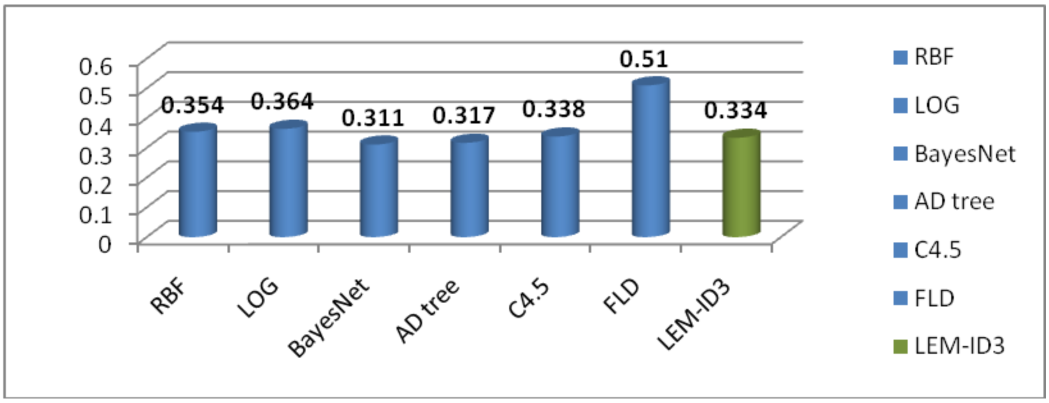

Figure 8: Comparison among LEM (ID3) and convention classifiers for Glass163 dataset

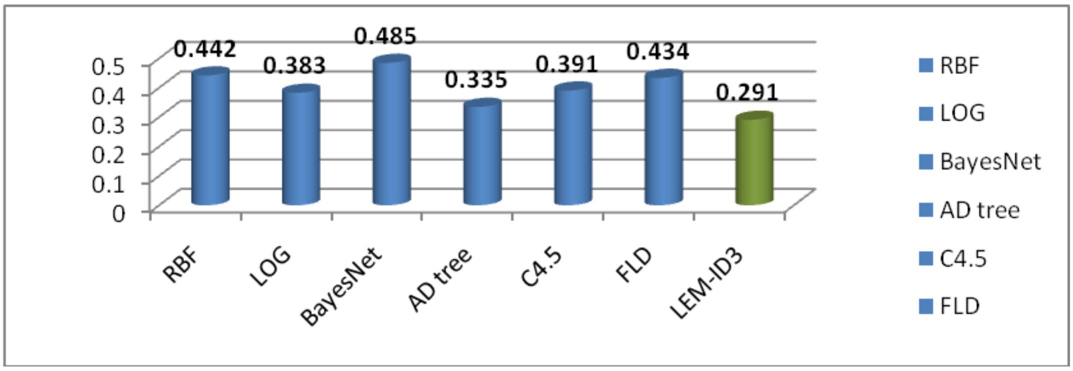

Figure 9: Comparison among LEM (ID3) and convention classifiers for BUPA dataset

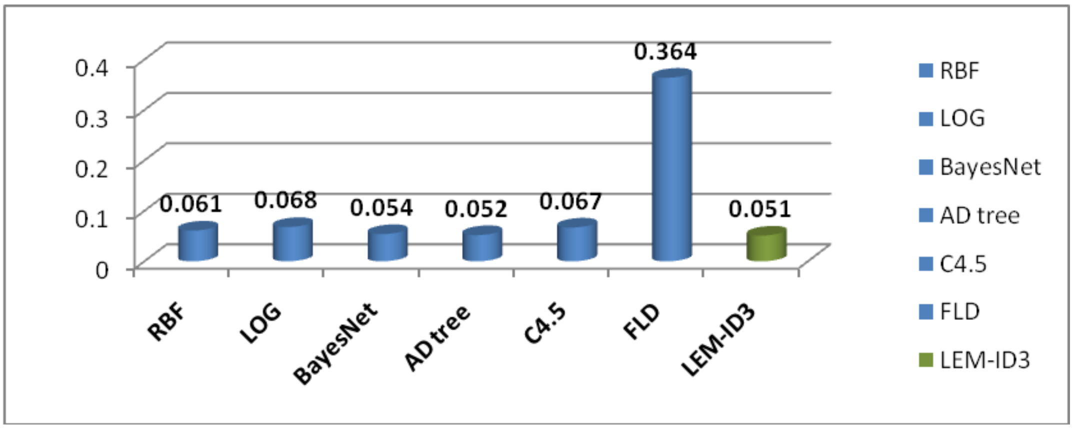

Figure 10: Comparison among LEM (ID3) and convention classifiers for WDBC dataset

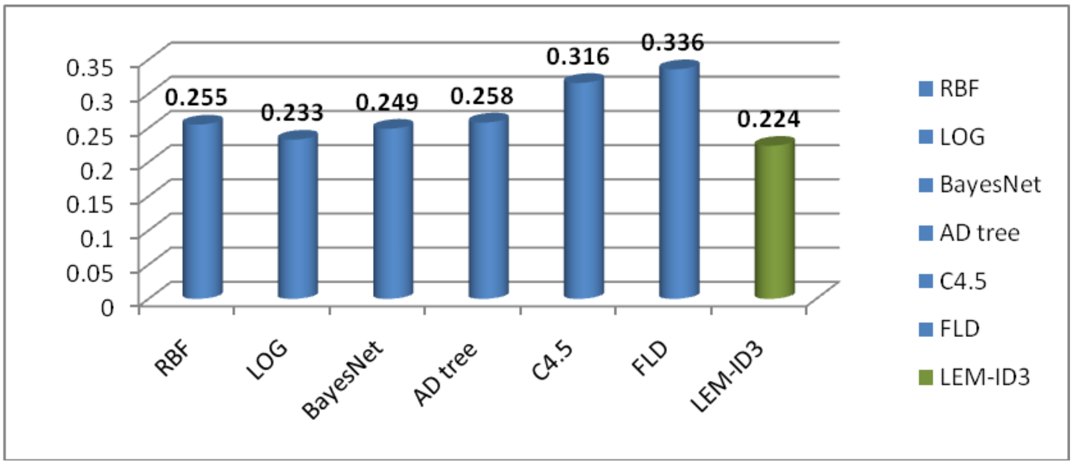

Figure 11: Comparison among LEM (ID3) and convention classifiers for PID dataset

Table2 Mean Error Comparisons of Classifiers on each dataset

Dataset	Classifiers						
	RBF	LOG	BayesNet	AD tree	C4.5	FLD	LEM-ID3
GLASS	.354	.364	.311	.317	.338	.510	**.334**
BUPA	.442	.383	.485	.335	.391	.434	**0.291**
WDBC	.061	.068	.054	.052	.067	.364	**.051**
PID	.255	.233	.249	.258	.316	.336	**0.224**

Table 2 presents the results of comparative tests applied to the LEM (ID3) algorithm in order to measure its accuracy against other algorithms in the classifiers family. The number of iterations for the test was set at 20,000 The algorithm were run 10 times to ensure a reliable average deviation. The results of the applied tests suggest that LEM(ID3) exceed or equal to the accuracy obtained by RBF, LOG,Bayesnet,AD tree and C4.5 in most classification functions performed. Using of LEM (ID3) in the mentioned datasets compared to convention classifiers and we see that LEM (ID3) achieve superior (or equivalent) performance to the conventional classifiers examined. The results of the algorithms were compared in relation to the convergence speed to the minimum error and the number of iterations to reach such solutions. It can be notice that the convergence to the minimum error in the LEM(ID3) algorithm is achieved with a smaller number of iterations. The process of inference rules allows LEM(ID3) to execute qualitative jump towards the optimal error rate, so that optimal results are achieved in an average of 2000 iterations over all test functions, while other algorithms need over 3000 iterations, and even 5000 iterations.

7. Conclusion

This paper presents a new version of the LEM algorithm called LEM (ID3) used in classification domain. The proposed algorithm uses LEM techniques to create a set of rules that allows the inferring of new candidates in the population that emerge not only from the random scan. The amendment allows the new algorithm to perform efficiently in both discrete and continuous functions. The algorithm was subjected to four famous classic datasets and in most cases improved the results against other convention classifiers. It was also concluded following a scalability test that the algorithm maintains its accuracy even in high dimensions. The algorithm also was shown to maintain a higher accuracy than the other algorithms in the number of iterations to go to the minimum error rates.

References

[1] R.S. Michalski, "Learnable Evolutionary Model: Evolutionary Processes Guided by Machine Learning". *Machine Learning*, Vol.38, pp. 9-40, 2000.

[2] Wnek, J., Kaufmann, K., Bloedorn, E., Michalski R.S.(1995) Inductive Learning Sys- tem AQ15c: the method and users guide. Reports of the Machine Learning and Inference Laboratory, MLI95-4, George Mason University, Fairfax, VA, USA.

[3] Jourdan,L., Corne, D., Savic, D., Walters, G. (2005) Hybridising rule induction and multiobjective evolutionary search for optimizing water distribution systems, in Proc of the 4th Hybrid Intelligent Systems conference, published in 2005 by IEEE Com- puter Society Press. pp.434-439, ISBN 0-7695-1916-4.

[4] Larranaga, P., Lozano, J.A. (eds) (2002) Stimulation of Distribution Algorithms: A New Tool for Evolutionary Computation, Kluwer Academic Publishers.

[5] J.Wojtusiak and R.S.Michalski, "The LEM3 Implementation of Learnable Evolution Model and Its Testing on Complex Function Optimization Problems", Proceedings of Genetic and Evolutionary Computation Conference, GECCO 2006, Seattle, WA, July 8-12, 2006.

[6] M. Ebner and A. Zell, Evolving a task specific image operator, in Joint Proceedings of the 1st EuropeanWorkshop on Evolutionary Image Analysis,Signal Processing and Telecommunications (EvoIASP'99 and EuroEcTel'99), Goteborg, Sweden (1999) pp. 74–89

[7] D. Michie, D. J. Spiegelhalter and C. C. Taylor, Machine Learning, Neural and Statistical Classification, Ellis Horwood, Upper Saddle River,NJ, 1994

[8] T. Lim, W. Loh and Y. Shih, A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms, Machine Learning, 40 (2000) 203–228

[9]S. Parsons, "Introduction to Machine Learning by Ethem Alpaydin," *The Knowledge Engineering Review*, vol. 20, no. 4, pp. 432-433, 2005.

[10] J. R. Quinlan, "Induction of Decision Trees " Machine Learning, vol. 1, no. 1, pp. 81-106, 1986.

[11] O. L. Mangasarian, W. N. Street, and W. H. Wolberg, Breast cancer diagnosis and prognosis via linear programming, Operations Research 43 (1995) 570–577

[12] J.Wojtusiak and R.S.Michalski, “The LEM3 Implementation of Learnable Evolution Model and Its Testing on Complex Function Optimization Problems”, Proceedings of Genetic and Evolutionary Computation Conference, GECCO 2006, Seattle, WA, July 8-12, 2006.

[13] L. Jourdan, D. Corne, D. Savic, G. Walters (2005) Hybridising rule induction and multiobjective evolutionary search for optimizing water distribution systems, in Proc of the 4th Hybrid Intelligent Systems conference, published in 2005 by IEEE Computer Society Press. Pp.434-439, ISBN 0-7695-1916-4.