EVADING: An Evolutionary Algorithm with Dynamic Niching for Data Classification.

CONFERENCE PAPER · JANUARY 2009
Source: DBLP

READS
28

3 AUTHORS, INCLUDING:

Fani A. Tzima
Aristotle University of Thessaloniki
16 PUBLICATIONS  46 CITATIONS

Pericles A. Mitkas
Aristotle University of Thessaloniki
250 PUBLICATIONS  1,372 CITATIONS

Available from: Pericles A. Mitkas
Retrieved on: 04 February 2016
Abstract—Multimodal optimization problems (MMOPs) have been widely studied in many fields of machine learning, including pattern recognition and data classification. Formulating the process of rule induction for the latter task as a MMOP and inspired by corresponding findings in the field of function optimization, our current work proposes an evolutionary algorithm (EvADiNG) capable of discovering a set of accurate and diverse classification rules. The proposed algorithm uses a dynamic clustering technique as a parallel niching method to maintain rule population diversity and converge to the optimal rules for the attribute-space defined by the target dataset. To demonstrate its applicability and potential, EvADiNG is applied to a series of real-life classification problems and its prediction accuracy is compared to that of other popular non-evolutionary machine learning techniques. Results are encouraging, since EvADiNG manages to achieve the best overall average ranking and performs significantly better (at significance level $\alpha=0.05$) from three out of the eight rival algorithms used in this study. This work is concluded with some insights on the factors affecting the proposed algorithm’s performance, along with the directions of our future research.

Index Terms—Classification, evolutionary rule induction, dynamic niching.

I. INTRODUCTION

During the last decades, multimodal optimization has been the focus of research in many fields of machine learning, including pattern recognition, predictive analytics and classification. These efforts have resulted in a multitude of algorithmic approaches for tackling multimodal optimization problems (MMOPs), with several of them being based on evolutionary computation methods. Among the methods targeting the domain of data classification, genetic algorithms – although proved to be robust and powerful optimization tools – have received only limited attention. Thus, inspired by classical approaches to multimodal function optimization, our present paper proposes a parallel-niching genetic algorithm (GA), capable of discovering a diverse and accurate set of prediction rules to address the task of data classification. The rationale of our choice of an evolutionary approach to rule induction is also supported by the fact that GAs are considered to be ideal in domains that are not well understood, or where a model of the underlying problem is infeasible to construct [1]. Furthermore, due to their global-search nature, they tend to cope with attribute interactions better than most greedy rule-induction algorithms [2].

The formal definition of data classification, as a supervised learning procedure, involves a learner receiving a number of observations (instances) as the basis for its training. Each instance $i$ consists of an attribute vector $x_i$ and a goal attribute to be predicted, referred to as class $c_i$. The aim of the learner is to search the attribute-space for a set of hypotheses that cover the training instances, while being able to adequately generalize to unknown ones [3]. In the case of a rule-induction algorithm, the search of the attribute-space aims at discovering a set of rules that define a mapping from the attribute domain onto the class domain, with the ultimate goal of using the discovered rule set to subsequently predict the class of unknown attribute vectors. Complementing these definitions, our current approach formulates rule induction as a multi-modal optimization problem, in which several disjunctive rules, representing the optima of the attribute-space, are required to build an accurate prediction model.

Regarding the design of a rule induction GA, there are generally two approaches: the Pittsburgh and the Michigan approach. The former (e.g. GIL [4], GAssist [5]) follows the paradigm of a traditional GA, where each individual of the population represents a candidate solution of the problem – in our case a complete set of classification rules – and thus has the advantage of naturally taking into account rule interactions, during the GA’s evaluation phase [6]. However, the complexity of individuals increases dramatically, affecting the performance of the algorithm and requiring the design of task-specific genetic operators. According to the Michigan approach, on the other hand, each individual encodes a single classification rule, thus leading to much simpler individual representations. Nevertheless, the need to discover “cooperative” sets of rules raises the important issue of utilizing special methods to diversify rule population and take into account rule interactions during the evolutionary search process [7].

The term “niching” refers to an important class of methods, traditionally used by GAs to allow the formation
and maintenance of diverse solutions, as well as to prevent convergence to a single solution. Fitness sharing [8] and crowding [9], [10] are among the most widely used methods of this category. Resource-sharing [7] is an additional example of a niching technique, specifically targeted to encouraging rule diversity in Michigan-style GAs, during the rule evaluation phase. Similar to resource-sharing, several evolutionary rule induction algorithms incorporate a niching-like technique into the natural selection process. A representative example is REGAL [11] that introduces the “Universal Suffrage” selection operator to “elect” candidate parents of each generation from training instance votes.

An alternative approach to discovering cooperative sets of rules in Michigan-style GAs is “sequential rule induction” [12] (e.g. HIDER [13], ESIA [14], GeSeCo [15]), that is essentially an iterative learning scheme. In each iteration, the GA discovers a single rule, while future search for rules similar to it, either in the genotypic or the phenotypic space, is “discouraged”. The final ruleset is built by combining the rules discovered through all iterations. A similar approach, that only differs in each iteration’s target, is “sequential covering” [16], where the GA iteratively discovers sets of rules with the same class in the rule consequent.

Inspired by work in the field of multimodal function optimization, our current work deviates from the traditional approaches outlined above, and adopts an approach based on the combination of individuals’ clustering with a fitness sharing scheme to maintain rule diversity during their evolutionary induction. This hybrid approach was originally used in the work of Yin and Gernay [17], which employed MacQueen’s adaptive KMEAN clustering algorithm and a modified version of the Goldberg-Richardson sharing function [8]. More recent examples of the synergy between clustering and fitness sharing include the Novel Clustering method, proposed by Yu [18], that takes into account inter- and intra-cluster distances and the Dynamic Niche Clustering technique, proposed by Gan and Warwick [19], that maintains a separate population of overlapping fuzzy niches with variable radius in parallel to the individual population. Finally, a different strategy is adopted by Jiao et. al. [20], where instead of evolving a population of candidate rules, training examples are evolved by clustering them into “organizations” of the same class. The final ruleset is extracted from the evolved organizations, according to the “useful attributes” that their member examples possess.

The remainder of this paper is structured as follows: Sections 2 and 3 describe the EvAdiNG classification algorithm, as well as the search techniques that were used to boost its prediction accuracy. Following a brief description of the evaluation methodology employed to assess algorithm performance in Section 4, we continue with the presentation of our experimental setup and results. This work is concluded in Section 5 with some insights on the factors affecting EvAdiNG performance, along with the directions of our future research.

II. DESIGN ISSUES FOR AN EVOLUTIONARY CLASSIFICATION ALGORITHM

In the following subsections, the aspects of knowledge representation, individual encoding, rule evaluation, as well as the natural selection method and genetic operators employed in the EvAdiNG algorithm are described.

A. Knowledge Representation

The first step in designing a GA is the selection of an appropriate representation for the target search-space. In the case of EvAdiNG, that follows the Michigan approach, discovered knowledge is represented as a set of rules in the traditional production system form of “IF conditions THEN predicted_class”. Each rule comprises two parts: the rule consequent (the THEN part), which determines the predicted class, and the rule antecedent (the IF part), which consists of a conjunction of propositional logic conditions of the form $\langle$Attribute|Operator|Value$\rangle$, wherein (i) numeric attributes are associated with a single discrete or continuous value using the comparison operators “$<$” or “$>$”, and (ii) nominal attributes are associated with one or more symbolic values using the set operator “$\in$”. Examples of conditions are: “Income $\leq 50.000$” and “Salary $\in \{\text{low, medium}\}$”.

It is worth noting that, since we deal with the task of data classification, the goal attribute to be predicted cannot be part of any condition. Furthermore, relational or first-order logic conditions associating prediction attributes (e.g. $\text{Debts} > \text{Income}$) cannot be represented in our current implementation. This stems from the fact that, although much more expressive, such a representation requires a more complex GA and a significantly larger computational effort to handle the much broader search-space [6].

B. Individual Encoding

GAs traditionally use binary strings to represent candidate solutions of the target search-space. While binary encoding is ideal for symbolic domains, it is restrictive when dealing with continuous values. In the latter case, a high-level encoding is more natural and gives more flexibility to genetic operations. Since EvAdiNG is capable of handling both numeric and nominal attributes, a hybrid encoding, exploiting the advantages of the two aforementioned approaches, has been employed (Figure 1). A positional encoding of the attribute–value pair is implicitly defined, meaning that each attribute has a fixed position inside the individual and therefore only its value (and the associated operator for numeric conditions) needs to be encoded.

As already mentioned, conditions entailing numeric attributes follow a value–operator format, using two genes. The first gene is real-valued and encodes the value associated with the corresponding attribute, while the second is binary and encodes the available operators (“$<$” and “$>$”). Instead of letting real-valued genes take any value in $(-\infty, \infty)$, we perform an analysis of the training dataset, prior to the evolutionary process, to locate the maximum and minimum observed values of each numeric attribute. These values are
then used as the upper and lower limit for the values of the corresponding genes.

Conditions entailing nominal attributes are encoded using a bit-string, with length equal to the size of the concept set of the respective attribute. Within that bit-string, a value of “1” enables the corresponding concept, while a value of “0” disables it. For example, for a nominal attribute Color whose values may be drawn from the set \{Black, Brown, Green, Blue\}, a bit-string of length four would be used and the bit-string “0101” would encode the condition Color ∈ \{Brown, Blue\}. Given the selected encoding for nominal attributes, it is evident that, if all concepts in a condition are active (or they are all inactive), the condition is omitted.

For each condition, regardless of the kind of attribute it entails, an activation bit is also encoded into the individuals. This bit determines whether the corresponding condition is considered active (“1”) or not (“0”) and provides a convenient way of specializing/generalizing a rule via condition insertions/removals (see also Section II-D).

Finally, it is worth mentioning that the rule consequent (the predicted class) is not directly encoded into the individual, but rather dynamically determined during the individuals’ evaluation phase as follows: each rule’s predicted class is set to the majority class of the instances it covers.

C. Rule Evaluation

The current implementation of EvADiNG focuses solely on discovering rules with high prediction accuracy. During preliminary testing, however, it was observed that the algorithm could not easily locate and maintain rules predicting classes with low frequency of appearance in the training dataset. This shortcoming had a major impact on the total prediction accuracy of the algorithm in datasets with non-uniform class distributions. An analysis of the corresponding fitness landscapes, revealed that candidate rules predicting low frequency classes had smaller fitness values than those predicting prevalent classes. Consequently, selection pressure would tend to favor the latter rules during the evolutionary procedure.

To circumvent this problem, a weighted individuals’ evaluation scheme, based on their predicted class, was employed: prior to the evolutionary procedure, training instances are assigned weights reversely proportional to their class’s frequency of appearance. More specifically:

1) The training dataset is analyzed and the frequency \(f_j\) of each available class \(j\) is calculated.
2) For each class, a weight factor \(w_j\) is calculated according to \(w_j = f_m / f_j\), where \(f_m\) is the frequency of appearance of the majority class.
3) Each instance \(i\) with class \(j\) is assigned the corresponding weight factor \(w_j\).

Before defining the fitness function used to evaluate rules’ prediction accuracy, we need to introduce two additional weighted metrics:

- **Weighted true positives** \((wTP)\) — equals to the total weight of instances that are covered by a rule and their class coincides with the rule’s predicted class.
- **Weighted false positives** \((wFP)\) — equals to the total weight of instances that are covered by a rule and their class does not coincide with the rule’s predicted class.

Given the above metrics, the fitness function, in its general form, is given by:

\[ f(x) = \alpha \cdot (N - \beta \cdot wFP) + \gamma \cdot wTP \]

where \(\alpha, \beta, \gamma\) are non-negative constants and \(N\) is the size of the training dataset. The default values for the constant parameters are \(\alpha = 2, \beta = 1\) and \(\gamma = 1\).

D. Natural Selection & Genetic Operators

EvADiNG uses probabilistic tournament selection (with a tournament size of 2 and a probability of the fittest individual winning the tournament equal to \(p_t=0.9\)) and conventional operators for genetic crossover and mutation. There is a probability \(p_c\) (\(p_c=0.9\) in our experiments) for applying traditional single-point crossover to a pair of candidate parents. There is also the option to apply uniform crossover with probability \(p_u\), in parallel to the single-point crossover, in order to balance the positional bias\(^1\) that the latter operator suffers from. However, this option was not used in our final experiments, as preliminary findings indicated that the use of the uniform crossover operator does not produce significantly better results, while it also increases computational cost.

The mutation operator randomly mutates the value of a gene, with a given probability \(p_m\). The value of \(p_m\) differs for real-valued and binary genes. We have set \(p_m=0.15\) for the first case and \(p_m=0.05\) for the second. Given the high values of crossover and mutation probabilities chosen, we have also used a high percent of elitist individuals \(e=20\%\), as a guarantee that good candidate rules would not disappear during reproduction.

In addition to crossover and mutation operators, EvADiNG also employs a generalization operator, generalizing rules either by condition removals or by increasing the interval of values satisfying a condition. Condition removal is implemented by deactivating an active condition (switching the corresponding activation bit to “0”). The increase of the interval of values satisfying a condition, on the other

\(^1\)The probability of swapping a pair of genes during single-point crossover depends on their position in the individual’s encoding.
hand, may either be achieved by (i) dropping an active concept, in case the condition entails a nominal attribute or (ii) mutating the value part of the condition, in case it entails a numeric attribute. Mutation of the value part of a condition results in either its increase or its decrease, depending on the (corresponding) comparison operator used. It is important to note that each rule is generalized only once per generation.

III. THE EVADIING ALGORITHM

EVADIING uses a dynamic niching technique, originally proposed in [21], that is based on two assumptions: (i) that the number of optima (peaks), \( q \), is known or can be estimated and (ii) that the optima are all at a minimum distance \( 2 \cdot \sigma_{sh} \) from each other. The procedure is based on the observation that selection pressure, during the evolution phase of a traditional GA, gradually assembles individuals around the peak of the search-space. Having defined data classification as a MMOP, we need a mechanism to mitigate this undesirable effect: dynamic niching constrains selection pressure in each generation via clustering of the population in groups of similar individuals and applying a fitness sharing method based on the discovered clusters (peak areas). Dynamic niching can be, thus, used as a parallel-niching method, allowing a rule-induction algorithm to simultaneously locate and maintain a set of rules that hopefully represent all peak areas of the search-space.

A. Clustering Method

Before any clustering method can be applied, though, an appropriate similarity/dissimilarity measure on the search-space needs to be defined. For the purpose of this work, we have introduced a weighted phenotypic distance, based on the definition of the regular phenotypic distance. More specifically:

- The phenotypic distance of two rules \( r_i \) and \( r_j \) is equal to the number of instances, whose attribute vector either satisfies the conditions of \( r_i \) or those of \( r_j \), but not both simultaneously.
- The weighted phenotypic distance (wPhd) of two rules \( r_i \) and \( r_j \) is equal to the sum of weights of the instances, whose attribute vector either satisfies the conditions of \( r_i \) or those of \( r_j \), but not both simultaneously.

Having defined the distance measure, we can proceed with the presentation of the clustering procedure used in EVADIING (Algorithm 1). In each generation, the fittest individual of each of the \( q \) peak areas is dynamically located and assigned as the center of a cluster, thus ultimately creating a dynamic peak set (DPS). The rest of the individuals are subsequently distributed to the DPS according to a minimum distance criterion. As a result of this procedure, each individual is either assigned to its “nearest” cluster of the DPS or to the non-peak set, if its distance from all cluster centers is greater than \( \sigma_{sh} \).

After identifying the peak areas (clusters) of the search-space, fitness sharing is applied among both the members of the DPS and the non-peak set, using a slightly different approach in each case. We start by defining the dynamic niche count \( m_{dsh,i} \) as:

\[
m_{dsh,i} = \begin{cases} n_j, & \text{if individual } i \text{ belongs to cluster } j \\ m_i, & \text{if individual } i \text{ belongs to the non-peak set} \end{cases}
\]

where \( n_j \) is the population size of the \( j \)th cluster.

The variable \( m_i \) is given by \( m_i = \sum_{j=1}^{N} sh(d_{i,j}) \), where \( d_{i,j} \) is the distance of individuals \( i \) and \( j \) and \( sh() \) is the sharing function as defined in [8]:

\[
sh(d_{i,j}) = \begin{cases} 1 - \frac{d_{i,j}}{\sigma_{sh}}, & \text{if } d_{i,j} < \sigma_{sh} \\ 0, & \text{otherwise} \end{cases}
\]

where the parameter \( \alpha_{sh} \) is a constant commonly set to 1.

The shared fitness \( f_{dsh,i} \) of each individual is, finally, calculated by dividing its original fitness value by the dynamic niche count: \( f_{dsh,i} = f_i / m_{dsh,i} \).

B. Intra-cluster Local Search

In parallel to the probabilistic tournament selection method, the algorithm also uses a restricted mating scheme, usually referred to as speciation [22]. The latter approach aims at complementing the clustering procedure that is essentially a global search method for the identification of regions where the prevalent solutions are situated, but fails to locate the exact optimal solutions of the corresponding areas. Aiming at promoting intra-cluster local search, speciation utilizes the information of a cluster and forbids the genetic crossover of two chromosomes, unless they belong to the same peak area. This constraint achieves two goals:

1) **Local search of the solution-space.** Highly similar individuals are probably situated in neighboring regions of the search-space and, thus, their genetic crossover would probably create individuals situated in the same

---

**Algorithm 1 Dynamic Peak Set (DPS) Identification**

**Input:**
- \( Pop \) Population of individuals
- \( k \) Population size
- \( q \) max niches to identify
- \( \sigma_{sh} \) niche radius

**Output:**
- \( DPS \)

**Sort \( Pop \) in decreasing fitness order**

\[ i \leftarrow 0 \]

\[ NumPeaks \leftarrow 0 \]

\[ DPS \leftarrow \emptyset \]

**while** \( i < k \) **AND** \( NumPeaks ! = q \) **do**

**if** \( Pop[i] \) **not within** \( \sigma_{sh} \) of a peak in \( DPS \) **then**

**insert** \( Pop[i] \) **into** \( DPS \)

\[ NumPeaks \leftarrow NumPeaks + 1 \]

**end if**

\[ i \leftarrow i + 1 \]

**end while**

**return** \( DPS \)
area and having slightly different genetic values from their parents.

2) Containment of low quality solutions. Individuals with small degree of similarity are probably situated in different regions of the search-space. Their genetic crossover would probably create individuals with little chance of representing good potential solutions and, thus, little potential for future evolution.

Given the nature of the speciation scheme, the following strategy was adopted to maximize the overall efficiency of the search procedure (global and local) in EVADiNG. At the beginning of the evolutionary process, priority is given to the global search mechanism by allowing individuals to freely mate with any other individual of the population. After a percentage $t_s$ of the total number of generations has passed, at which point we can assume that the peak areas have been identified by the clustering method, speciation is activated at which point we can assume that the peak areas have been identified. In our experiments, we have set $t_s=25\%$ for a total number of generations $N=400$.

C. An Algorithmic Description of EVADiNG

The total procedure of EVADiNG is presented in Algorithm 2. In each generation, after the evaluation of the population, EVADiNG uses the clustering method to dynamically identify the regions that enclose the peaks of the attribute-space. By grouping together individuals belonging to the same peak region and applying a fitness sharing procedure, the algorithm constrains selection pressure and locates the prevalent rules of the attribute-space.

Regarding the extraction of the final ruleset from the last evolutionary generation, it is important to note that we consider a cluster as a group of individuals with similar characteristics that encode a common rule as a whole. The fittest individual of each cluster may be thought of as representing the rest of the individuals as the prevalent individual of this subset. Under this assumption, each cluster represents a partial solution to the target problem, with the combination of all clusters composing the solution as a whole. Thus, the final set of classification rules consists of the rules encoded by the fittest individuals of each cluster of the last evolutionary generation, while the individuals of the non-peak set are ignored. Of course, under normal circumstances and if the two original assumptions of the evolutionary process were satisfied, the non-peak set will be empty and the whole population will be distributed in the DPS.

A final remark about the nature of the evolved ruleset is that it is used to classify instances in a non-hierarchical way. Simply put, rules are neither ordered inside the ruleset nor dependent on other rules preceding them. This however does not eliminate rule overlapping or rule redundancy. Thus, in case of instances covered by more than one rules, a priority criterion (usually rule fitness) is used to select the rule that will actually fire. Finally, in case of instances whose attribute vector does not satisfy any rule, the majority class of the training dataset is selected as the predicted class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N$</td>
<td>Max generations</td>
<td>400</td>
</tr>
<tr>
<td>$P$</td>
<td>Population size</td>
<td>800</td>
</tr>
<tr>
<td>$p_c$</td>
<td>Mutation probability</td>
<td>0.9</td>
</tr>
<tr>
<td>$p_m$</td>
<td>Generalization probability</td>
<td>0.15 (Nom.) &amp; 0.05 (Nom.)</td>
</tr>
<tr>
<td>$p_g$</td>
<td>Tournament probability</td>
<td>0.9</td>
</tr>
<tr>
<td>$N_t$</td>
<td>Tournament size</td>
<td>2</td>
</tr>
<tr>
<td>$e$</td>
<td>Elitism</td>
<td>0.2</td>
</tr>
<tr>
<td>$t_s$</td>
<td>Speciation flag</td>
<td>0.25</td>
</tr>
</tbody>
</table>

IV. EXPERIMENTS AND RESULTS

The main aim of our experimental methodology is the comparison of EVADiNG with eight other well-known algorithmic approaches for data classification, implemented in the machine learning tool WEKA [23]. All algorithms have been applied to 13 benchmark datasets, obtained from the UCI machine learning repository [24], with their prediction accuracy serving as the basis for their comparison.

Table I provides a synopsis of the basic parameters of EVADiNG kept constant throughout all reported experiments. Table II reports the parameters for the clustering procedure, which were specifically picked for each target dataset through a modest parameter-tuning phase. Therein, the value of $\sigma_{sh}$ is expressed as a percentage of the total weight of the training instances, in accordance to the weighted nature of the distance measure. As far as the other classification algorithms are concerned, their parameters were kept constant to their default values, as given by WEKA.

To estimate the prediction accuracy of an algorithm for a given dataset, we performed 10 stratified 2-fold cross-validation runs and calculated the mean prediction accuracy.
The corresponding results are summarized in Table IV, along with algorithm rankings for each dataset (in parenthesis). The average algorithm rank (last row of Table IV) provides a clear indication of the studied algorithms’ relative performance, with EvADiNG receiving an average rank of 3.154 and SMO being the next to follow with an average rank of 3.846.

Based on the measured accuracy results, our approach outperforms rival algorithms on three out of the thirteen classification problems used in this study. Moreover, it consistently achieves better ranks than all its rivals in more than half of the target datasets (Table III). Finally, it is worth noting that EvADiNG displays robust behavior throughout the 10 cross-validation runs, with acceptable variance of the measured accuracy for most of the datasets. This is more accurately presented in Figure 2 that depicts the mean prediction accuracy per dataset, along with the corresponding error bars. Therein, the relatively high variance observed for the c.lenses and labor datasets is justified by their limited number of instances (24 and 57 respectively).

TABLE III

<table>
<thead>
<tr>
<th>Datasets</th>
<th>q</th>
<th>( \sigma_{sh} ) (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>br. cancer</td>
<td>14</td>
<td>14</td>
</tr>
<tr>
<td>BUPA</td>
<td>16</td>
<td>26</td>
</tr>
<tr>
<td>c. lenses</td>
<td>12</td>
<td>29</td>
</tr>
<tr>
<td>glass</td>
<td>26</td>
<td>17</td>
</tr>
<tr>
<td>heart</td>
<td>24</td>
<td>17</td>
</tr>
<tr>
<td>hepatitis</td>
<td>14</td>
<td>11</td>
</tr>
<tr>
<td>iris</td>
<td>12</td>
<td>17</td>
</tr>
<tr>
<td>labor</td>
<td>24</td>
<td>35</td>
</tr>
<tr>
<td>pima</td>
<td>14</td>
<td>20</td>
</tr>
<tr>
<td>sonar</td>
<td>16</td>
<td>11</td>
</tr>
<tr>
<td>voting</td>
<td>16</td>
<td>29</td>
</tr>
<tr>
<td>wine</td>
<td>10</td>
<td>20</td>
</tr>
<tr>
<td>zoo</td>
<td>16</td>
<td>23</td>
</tr>
</tbody>
</table>

In order to evaluate the statistical significance of the measured differences in algorithm ranks, we have used the procedure suggested by Demsar [25] for robustly comparing classifiers among multiple datasets. This procedure involves the use of the Friedman test to establish the significance of the differences between classifier ranks and, potentially, a post-hoc test to compare classifiers to each other. In our case, the goal was to compare the performance of the rival algorithms to that of our implementation (control classifier). Thus, the Holm procedure was selected as the appropriate post-hoc test.

Having established (through the Friedman test) that the measured average ranks are significantly different, at significance level \( \alpha=0.05 \), we proceeded with the Holm post-hoc test to conclude whether our method performs significantly better than its rival algorithms. Our analysis reveals that at significance level \( \alpha=0.05 \), the performance of EvADiNG is significantly better than that of VFI, Decision Table and JRip.

V. CONCLUSIONS AND FURTHER WORK

In this paper we have presented EvADiNG, an evolutionary classification algorithm that employs a combination of individuals’ clustering with a fitness sharing scheme to maintain rule diversity during the rule induction process. After analyzing the architecture and the underlying mechanisms of EvADiNG, we went on to present an experimental comparison of its performance to that of eight well-known algorithmic approaches to data classification. The comparison was based on the studied algorithms’ prediction accuracy on a number of benchmark datasets, derived from the UCI machine learning repository.

Overall, EvADiNG proved to be robust and accurate, as, contrary to the rival approaches, it displayed consistent behavior and ranked in the first places in the majority of the datasets. More specifically, the algorithm ranked first in three out of the thirteen datasets and received the highest average algorithm rank. Statistical evaluation of results further revealed that, at significance level \( \alpha=0.05 \), EvADiNG performs significantly better than three of its rivals, namely VFI, Decision Table and JRip.

Several issues deserve further investigation to improve the effectiveness of the algorithm. An important consideration concerns the clustering procedure, for which, in the present implementation, two parameters are required: the max number of clusters to be discovered and their radius \( \sigma_{sh} \). Although the algorithm proved to be relatively insensitive to changes in the first parameter’s values, cluster radius \( \sigma_{sh} \) has been crucial to the success of the clustering procedure. Thus, to avoid the need to fine-tune these parameters by hand, we envisage an extension of the algorithm utilizing a variable niche radius and elements of fuzzy logic to allow individuals to belong to more than one clusters simultaneously.
TABLE IV
PREDICTION ACCURACIES OF CLASSIFICATION ALGORITHMS

<table>
<thead>
<tr>
<th>Datasets</th>
<th>EVA/IN G</th>
<th>Decision Table (DT)</th>
<th>J48</th>
<th>JRP</th>
<th>K-Star (KS)</th>
<th>Logistic (LOG)</th>
<th>Naive Bayes (NB)</th>
<th>SMO</th>
<th>VFI</th>
</tr>
</thead>
<tbody>
<tr>
<td>br. cancer</td>
<td>71.86 (2)</td>
<td>60.58 (6)</td>
<td>70.76 (4)</td>
<td>71.08 (4)</td>
<td>71.40 (3)</td>
<td>69.64 (9)</td>
<td>71.89 (1)</td>
<td>68.53 (7)</td>
<td>67.45 (8)</td>
</tr>
<tr>
<td>BUPA</td>
<td>63.15 (2)</td>
<td>57.56 (7)</td>
<td>61.74 (4)</td>
<td>61.86 (3)</td>
<td>61.33 (5)</td>
<td>67.13 (1)</td>
<td>56.03 (8)</td>
<td>58.03 (6)</td>
<td>53.59 (9)</td>
</tr>
<tr>
<td>c. lenses</td>
<td>80.00 (1)</td>
<td>76.67 (3)</td>
<td>76.25 (4)</td>
<td>67.92 (9)</td>
<td>74.58 (5)</td>
<td>69.58 (8)</td>
<td>72.98 (2)</td>
<td>72.08 (7)</td>
<td>74.17 (6)</td>
</tr>
<tr>
<td>glass</td>
<td>51.03 (8)</td>
<td>59.81 (5)</td>
<td>64.25 (2)</td>
<td>60.65 (4)</td>
<td>59.91 (1)</td>
<td>60.84 (3)</td>
<td>49.01 (9)</td>
<td>59.49 (6)</td>
<td>54.25 (7)</td>
</tr>
<tr>
<td>heart</td>
<td>81.02 (3)</td>
<td>75.45 (8)</td>
<td>75.91 (7)</td>
<td>75.98 (6)</td>
<td>74.82 (9)</td>
<td>80.92 (4)</td>
<td>83.70 (1)</td>
<td>82.15 (2)</td>
<td>79.31 (5)</td>
</tr>
<tr>
<td>hepatitis</td>
<td>81.94 (3)</td>
<td>79.36 (6)</td>
<td>79.68 (7)</td>
<td>80.32 (5)</td>
<td>77.87 (9)</td>
<td>78.32 (8)</td>
<td>82.13 (2)</td>
<td>81.68 (4)</td>
<td>83.16 (1)</td>
</tr>
<tr>
<td>iris</td>
<td>95.33 (1)</td>
<td>94.00 (7)</td>
<td>94.13 (6)</td>
<td>90.73 (9)</td>
<td>94.67 (5)</td>
<td>94.87 (4)</td>
<td>95.07 (3)</td>
<td>95.27 (2)</td>
<td>93.87 (8)</td>
</tr>
<tr>
<td>labor</td>
<td>86.84 (4)</td>
<td>77.72 (8)</td>
<td>76.67 (9)</td>
<td>83.33 (6)</td>
<td>88.77 (3)</td>
<td>84.97 (2)</td>
<td>80.65 (1)</td>
<td>85.97 (5)</td>
<td>82.98 (7)</td>
</tr>
<tr>
<td>pima</td>
<td>73.36 (4)</td>
<td>74.66 (3)</td>
<td>72.16 (7)</td>
<td>73.11 (5)</td>
<td>68.72 (8)</td>
<td>76.78 (1)</td>
<td>72.96 (6)</td>
<td>75.72 (2)</td>
<td>58.89 (9)</td>
</tr>
<tr>
<td>sonar</td>
<td>70.14 (3)</td>
<td>67.79 (8)</td>
<td>69.52 (4)</td>
<td>68.08 (6)</td>
<td>79.28 (1)</td>
<td>69.42 (5)</td>
<td>67.93 (7)</td>
<td>74.28 (2)</td>
<td>59.52 (9)</td>
</tr>
<tr>
<td>voting</td>
<td>95.47 (1)</td>
<td>94.48 (5)</td>
<td>95.26 (2)</td>
<td>95.24 (3)</td>
<td>92.10 (8)</td>
<td>92.53 (7)</td>
<td>90.05 (9)</td>
<td>94.94 (4)</td>
<td>92.58 (6)</td>
</tr>
<tr>
<td>wine</td>
<td>90.97 (5)</td>
<td>88.88 (8)</td>
<td>89.61 (7)</td>
<td>88.54 (9)</td>
<td>96.52 (1)</td>
<td>95.17 (4)</td>
<td>96.18 (3)</td>
<td>96.35 (2)</td>
<td>90.34 (6)</td>
</tr>
<tr>
<td>zoo</td>
<td>90.70 (4)</td>
<td>88.92 (8)</td>
<td>90.20 (5)</td>
<td>85.64 (9)</td>
<td>92.87 (3)</td>
<td>93.27 (2)</td>
<td>90.10 (6)</td>
<td>93.47 (1)</td>
<td>89.70 (7)</td>
</tr>
</tbody>
</table>

Avg. Rank: 4.154

REFERENCES


