PATREC 3554
27 October 2004 Disk Used
ARTICLE IN PRESS
No. of Pages 11, DTD = 5.0.1

ELSEVIER
Available online at www.sciencedirect.com
SCIENCE DIRECT®
Pattern Recognition Letters xxx (2004) xxx–xxx
Pattern Recognition Letters
www.elsevier.com/locate/patrec

Stratification for scaling up evolutionary prototype selection

José Ramón Cano [a], Francisco Herrera [b,*], Manuel Lozano [b]

[a] Department of Computer Science, University of Jaén, 23700 Linares, Jaén, Spain
[b] Department of Computer Science and Artificial Intelligence, E.T.S.I. Infomatica-Dpto. Ciencias, de la Computacion e I.A., University of Granada, Avenida de Andalucia 38, 18071 Granada, Spain

Received 8 September 2004

Abstract

Evolutionary algorithms has been recently used for prototype selection showing good results. An important problem that we can find is the Scaling Up problem that appears evaluating the Evolutionary Prototype Selection algorithms in large size data sets. In this paper, we offer a proposal to solve the drawbacks introduced by the evaluation of large size data sets using evolutionary prototype selection algorithms. In order to do this we have proposed a combination of stratified strategy and CHC as representative evolutionary algorithm model. This study includes a comparison between our proposal and other non-evolutionary prototype selection algorithms combined with the stratified strategy. The results show that stratified evolutionary prototype selection consistently outperforms the non-evolutionary ones, the main advantages being: better instance reduction rates, higher classification accuracy and reduction in resources consumption.

Keywords: Stratification; Scaling up; Evolutionary algorithms; Prototype selection

1. Introduction

A machine learning model presents a training set which is a collection of training examples called prototypes or instances. The machine learning algorithm is tasked by generating a decision procedure called a "classifier" used to predict the outcome class of unseen test instances on the basis of observing training instances. After the learning process, the learning model is presented with additional input vectors, and the model must generalize deciding what the output value should be for the new test instance. The generalization is done, in a large number of machine learning algorithms, by evaluation of the distance between the input vector and the stored exemplars. Exemplar-based

* Corresponding author. Tel.: +34 958 240598; fax: +34 958 243317.
E-mail addresses: jrcano@decsai.ugr.es (J.R. Cano), herrera@decsai.ugr.es (F. Herrera), lozano@decsai.ugr.es (M. Lozano).

0167-8655/$ - see front matter © 2004 Published by Elsevier B.V.
doi:10.1016/j.patrec.2004.09.043

learning models (Aha et al., 1991; Kibbler and Aha, 1987; Wilson et al., 2000) must often decide what exemplars to store for use during generalization, in order to avoid excessive storage and time complexity. In Prototype Selection (PS) we intend to select the most promising examples to avoid these drawbacks (see Fig. 1).

In the literature we can find several approaches to PS, see Wilson et al. (2000) for a recent review. Evolutionary Algorithms (EAs) (Back et al., 1997; Goldberg, 1989) have been used to solve the PS problem with promising results (Cano et al., in press; Kuncheva, 1995; Nakashina and Ishibuchi, 1998; Ravindra and Narasimha, 2001; Shinn-Ying et al., 2002).

EAs are adaptive methods based on natural evolution that may be used for search and optimization. We introduce CHC (Eshelman, 1991) as representative and efficient EA model for PS (see Cano et al., in press).

The issue of scalability and the effect of increasing the size of data are always present in PS. The Scaling up problem, due to large size data sets, produces excessive storage requirement, increases times complexity and affects to generalization accuracy, introducing noise (Angluin and Laird, 1987) and over fitting. In EAs we have to add to these drawbacks the ones produced by the chromosome's size (Forrest and Mitchell, 1993) associated to the representation of the PS solution. Large chromosome's size increases the storage requirement and time execution and reduces significantly the convergence capabilities of the algorithm.

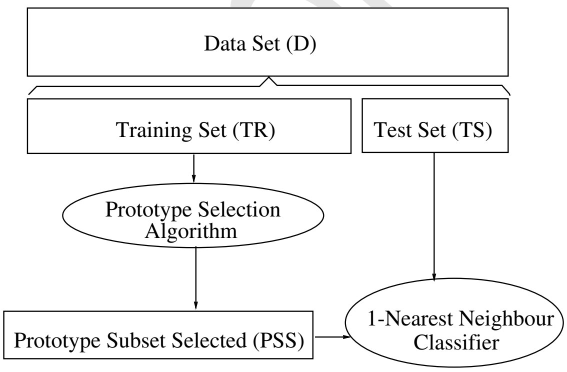

Fig. 1. Prototype Selection.

To avoid these drawbacks we propose a combination of EAs and the stratified strategy. In large size data sets we cannot evaluate the algorithms over the complete data set so the stratification is a possible way to carry out the executions. Combining the subset selected per strata we can obtain the subset selected for the whole initial data set. The stratification reduces the data set size for algorithm runs, while EAs select the best local training subset.

The aim of this paper is to study the combination of stratification and EAs applied to large data sets. Our proposal is compared with non-evolutionary prototype selection algorithms following the stratified strategy. To address this, we have carried out a number of experiments with increasing complexity and size of data sets.

In order to do this, this paper is set out as follows. In Section 2, we introduce the Scaling up problem and its effect on PS algorithms. Section 3 is dedicated to the combination of stratified strategy and evolutionary PS algorithm, giving details of how EAs can be applied to the PS problem in large size data sets. In Section 4 we explain the methodology used in the experimentation. Section 5 deals with the results and their analysis. Finally, in Section 6, we point out our conclusions.

2. The scaling up problem

The majority of PS algorithms cannot deal with large data sets. The basic nearest neighbor rule (Cover and Hart, 1967; Wilson, 1972) presents several shortcomings discussed in (Wilson and Martinez, 2000). As main problems we have that it has to store all of the training instances to carry out the classification task, so it has large memory requirements. It must search through all available instances to classify a new input vector, so it is slow during classification. These drawbacks are increased by the size of the data set. In this section we study the effect of the data set size in both groups of algorithms, evolutionary and non-evolutionary. The algorithms are briefly described in Section 4.1.

To test the effect of increasing the data set size, we have evaluated different size data sets. The main difficulties they have to face are the following:

- Efficiency. The efficiency of non-evolutionary PS algorithms evaluated is at least of $O(n^2)$, being n the number of instances in the data set. There are another set of PS algorithms (like Rnn in Gates, 1972; Snn in Ritter et al., 1975; Shrink in Kibbler and Aha, 1987, etc.) but most of them present an efficiency order much greater than $O(n^2)$. Logically, when the size grows, the time needed by each algorithm also increases.
- Resources. Most of the algorithms assessed need to have the complete data set stored in memory to carry out their execution. If the size of the data set was too big, the computer would need to use the disk as swap memory. This loss of resources has an adverse effect on efficiency due to the increased access to the disk.
- Generalization. Algorithms are affected in their generalization capabilities due to the noise and over fitting effect introduced by larger size data sets.
- Representation. EAs are also affected by representation, due to the size of their chromosomes. When the size of these chromosomes is too big, the algorithms experience convergence difficulties, as well as costly computational time.

These drawbacks produce a considerable degradation in the behavior of PS algorithms. There is a group of them that cannot be applied due to its efficiency (the case of Snn in Ritter et al., 1975 with $O(n^3)$).

Algorithms evaluated directly to the whole larger data sets are unefficacy and unefficient.

3. Combination of stratified strategy and evolutionary algorithms

To avoid the drawbacks associated to Scaling Up we led our study towards the hybrid algorithm between stratified strategy and EA.

3.1. Evolutionary algorithms applied to prototype selection

EAs have been applied to the PS problem, because it can be considered as a search problem (Cano et al., in press; Kuncheva, 1995; Nakashina and Ishibuchi, 1998; Ravindra and Narasimha, 2001; Shinn-Ying et al., 2002).

The application of EAs to PS is accomplished by tackling two important issues: the specification of the representation of the solutions and the definition of the fitness function.

3.1.1. Representation

Let's assume a data set denoted TR with n instances. The search space associated with the instance selection is constituted by all the subsets of TR. Then, the chromosomes should represent subsets of TR. This is accomplished by using a binary representation. A chromosome consists on the sequence of n genes (one for each instance in TR) with two possible states: 0 and 1. If the gene is 1, then its associated instance is included in the subset of TR represented by the chromosome. If it is 0, then this does not occur.

3.1.2. Fitness function

Let PSS be a subset (see Fig. 1) of instances of TR to evaluate and be coded by a chromosome. We define a fitness function that combines two values: the classification performance (clas_per) associated with PSS and the percentage of reduction (perc_red) of instances of PSS with regards to TR:

$$\mathtt{Fitness(PSS)} = \alpha \cdot \mathtt{clas\_per} + (1-\alpha) \cdot \mathtt{perc\_red}. \quad (1)$$

The 1-NN classifier is used for measuring the classification rate, clas_per, associated with PSS. It denotes the percentage of correctly classified objects from TR using only PSS to find the nearest neighbour. For each object y in TR, the nearest neighbour is searched for amongst those in the set $\mathtt{PSS}\backslash\{y\}$. Whereas, perc_red is defined as:

$$\mathtt{perc\_red} = 100 \cdot (|\mathtt{TR}| - |\mathtt{PSS}|)/|\mathtt{TR}|. \quad (2)$$

PATREC 3554
27 October 2004 Disk Used
ARTICLE IN PRESS
No. of Pages 11, DTD = 5.0.1

The goal of the EAs is to maximize the fitness function defined, i.e., maximize the classification performance and minimize the number of instances obtained. In the experiments presented in this paper, we have considered the value $\alpha = 0.5$ in the fitness function, due to a previous experiment in which we found the best trade-off between precision and reduction with this value, it was also used in (Cano et al., in press).

3.2. Stratified strategy and prototype selection

The stratified strategy divides the initial data set into disjoint strata with equal class distribution. The prototypes are independent one of each other, so the distribution of the data into strata will not degrade their representation capabilities.

The number of strata will determine the size of them. Using the proper number of strata we can reduce significantly the training set size. This situation allows us to avoid the drawbacks suggested in Section 2.

Following the stratified strategy, initial data set D is divided into t disjoint sets D_j, strata of equal size, $D_1, D_2, \ldots$, and D_t. We maintain class distribution within each set in the partitioning process.

The test set TS will be the TR complementary one in D.

$$TR = \bigcup_{j \in J} D_j, \; J \subset \{1, 2, \ldots, t\} \tag{3}$$

$$TS = D \setminus TR \tag{4}$$

PS algorithms (classical or evolutionary ones) are applied to each D_j obtaining a subset selected DS_j. The prototype selected set is obtained using DS_j (see Eq. (5)) and it is called Stratified Prototype Subset Selected (SPSS).

$$\mathrm{SPSS} = \bigcup_{j \in J} DS_j, \; J \subset \{1, 2, \ldots, t\} \tag{5}$$

The last phase, where the DS_j are being reunited, is not time-consuming, as it does not present any kind of additional processing. The time needed for the stratified execution is the one associated to the instance selection algorithm's execution in each strata.

4. Experimental methodology

We have carried out our study of the PS problem using three size problems: medium, large and huge. We try to evaluate the behavior of the algorithms when the size of the problem increases.

Section 4.1 is dedicated to describe the algorithms which appear in the experiments. In Section 4.2 we introduce the data sets evaluated. Section 4.3 shows the stratification and partition of the data sets that were considered, and finally, in Section 4.4 we describe the table contents that report the results.

4.1. Prototype selection algorithms for experiments

The algorithms studied can be divided in two groups, depending of their evolutionary nature. The algorithms selected are the most efficient ones shown in (Cano et al., in press).

4.1.1. Non-Evolutionary algorithms

In this section we present a summary of the non-evolutionary PS algorithms included in this study. The algorithms used are:

- Cnn (Hart, 1968)—It tries to find a consistent subset, which correctly classifies all of the remaining points in the sample set. However, this algorithm will not find a minimal consistent subset.
- Drop1 (Wilson and Martinez, 1997)—Essentially, this rule tests to see if removing an instance would degrade leave-one-out cross-validation generalization accuracy, which is an estimate of the true generalization ability of the resulting classifier.
- Drop2 (Wilson and Martinez, 1997)—Drop2 changes the order of removal of instances. It initially sorts the instances in TR by the distance to their nearest enemy (nearest instance belonging to another class). Instances are then checked for removal beginning at the instance furthest from its nearest enemy. This tends to remove instances furthest from the decision boundary first, which in turn increases the chance of retaining border points.

- Drop3 (Wilson and Martinez, 1997)—Drop3 uses a noise filtering pass before sorting the instances in TR. This is done using the rule: Any instance not classified by its k-nearest neighbours is removed.
- Ib2 (Kibbler and Aha, 1987)—It is similar to Cnn but using a different selection strategy.
- Ib3 (Kibbler and Aha, 1987)—It outperforms Ib2 introducing the acceptable instance concept to carry out the selection. The parameters associated to Ib3 appear in Table 1.

4.1.2. Evolutionary algorithms

We have evaluated the CHC algorithm as representative and efficient EA model.

During each generation the CHC (Eshelman, 1991) develops the following steps:

(1) It uses a parent population of size n to generate an intermediate population of n individuals, which are randomly paired and used to generate n potential offspring.
(2) Then, a survival competition is held where the best n chromosomes from the parent and offspring populations are selected to form the next generation.

CHC also implements a form of heterogeneous recombination using HUX, a special recombination operator. HUX exchanges half of the bits that differ between parents, where the bit position to be exchanged is randomly determined. CHC also employs a method of incest prevention. Before applying HUX to two parents, the Hamming distance between them is measured. Only those parents who differ from each other by some number of bits (mating threshold) are mated. The initial threshold is set at L/4, where L is the length of the chromosomes. If no offspring are inserted into the new population then the threshold is reduced by 1.

No mutation is applied during the recombination phase. Instead, when the population converges or the search stops making progress (i.e., the difference threshold has dropped to zero and no new offspring are being generated which are better than any members of the parent population) the population is reinitialized to introduce new diversity to the search. The chromosome representing the best solution found over the course of the search is used as a template to re-seed the population. Re-seeding of the population is accomplished by randomly changing 35% of the bits in the template chromosome to form each of the other $n - 1$ new chromosomes in the population. The search is then resumed.

The following table Table 1 introduces the parameters associated with the algorithms:

4.2. Data sets for experiments

To evaluate the behavior of the algorithms applied in different size data sets, we have carried out a number of experiments increasing complexity and size of data sets. We have selected medium, large and huge size data sets as we can see in Tables 2–4 (these data sets can be found in the UCI Repository in Merz and Murphy, 1996).

Table 1
Algorithm's parameters

Algorithm	Parameters
Ib3	Acceptance level = 0.9, Drop level = 0.7
CHC	Population = 50, Evaluations = 10000

Table 2
Medium size data sets

Data set	Instances	Features	Classes
Pen-based recognition	10992	16	10
SatImage	6435	36	6
Thyroid	7200	21	3

Table 3
Large size data set

Data set	Instances	Features	Classes
Adult	30 132	14	2

Table 4
Huge size data set

Data set	Instances	Features	Classes
Kdd Cup'99	494 022	41	23

4.3. Partitions and stratification: An specific model

We have evaluated each algorithm in a ten fold cross validation process. In the validation process TR_i, i=1, …, 10 is a 90% of D and TS_i its complementary 10% of D.

In our experiments we have executed the PS algorithms following two perspectives for the ten fold cross validation process.

In the first one, we have executed the PS algorithms as we can see in Fig. 2. We call it classic Ten fold cross validation (`Tfcv classic`). This result will be used as reference versus the stratification ones.

In `Tfcv classic` the subsets TR_i and TS_i, i = 1,…,10 are obtained as the Eqs. (6) and (7) indicate:

$$TR_i = \bigcup_{j \in J} D_j,$$
$$J = \{j / 1 \leqslant j \leqslant b \cdot (i-1) \quad \text{and} \quad (i \cdot b) + 1 \leqslant j \leqslant t\} \tag{6}$$

$$TS_i = D \setminus TR_i \tag{7}$$

where t is the number of strata, and b is the number of strata grouped ($b = t/10$, to carry out the ten fold cross validation).

Each PSS_i is obtained by the PS algorithm applied to TR_i subset.

The second way is to execute the PS algorithms in a stratified process as the Fig. 3 shows. We call it stratified ten fold cross validation (`Tfcv strat`).

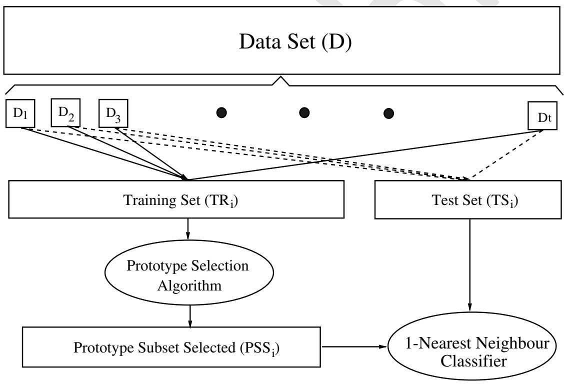

Fig. 2. Prototype Selection Strategy in Ten fold cross validation.

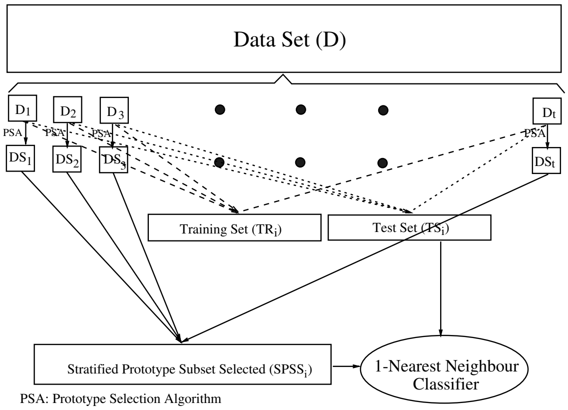

Fig. 3. Prototype Selection Strategy in stratified Ten fold cross validation.

In `Tfcv strat` each TR_i is defined as we can see in Eq. (6), by means of the union of D_j subsets (see Fig. 3).

In `Tfcv strat` (see Fig. 3) $SPSS_i$ is generated using the DS_j instead of D_j (see Eq. (8)).

$$SPSS_i = \bigcup_{j \in J} DS_j,$$
$$J = \{j / 1 \leqslant j \leqslant b \cdot (i-1) \text{and} (i \cdot b) + 1 \leqslant j \leqslant t\} \tag{8}$$

$SPSS_i$ contains the instances selected by PS algorithms in TR_i following the stratified strategy.

The subset TS_i is defined by means the Eq. (7). Both, TR_i and TS_i are generated in the same way in `Tfcv classic` and `Tfcv strat`.

As example, considering t=10, the subsets for each kind of validation process are presented in Table 5.

For each data set we have employed the partitions and number of strata that appear in Tables 6 and 7.

4.4. Table of results

In the following section we will present the structure of tables where we present the results.

Our table shows the results obtained by the evolutionary and non-evolutionary prototype selec-

Table 5
Stratified ten fold cross validation subsets

	TR_i	TS_i	$SPSS_i$
$i = 1$	$D_2 \cup D_3 \cup \ldots \cup D_{10}$	D_1	$DS_2 \cup DS_3 \cup \ldots \cup DS_{10}$
$i = 2$	$D_1 \cup D_3 \cup \ldots \cup D_{10}$	D_2	$DS_1 \cup DS_3 \cup \ldots \cup DS_{10}$
...	...	...	...
$i = 10$	$D_1 \cup D_2 \cup \ldots \cup D_9$	D_{10}	$DS_1 \cup DS_2 \cup \ldots \cup DS_9$

Table 6
Stratification in medium size data sets

Pen-based recognition	SatImage	Thyroid
$t = 10$ Strata	$t = 10$ Strata	$t = 10$ Strata
$t = 30$ Strata	$t = 30$ Strata	$t = 30$ Strata

Table 7
Stratification in large and huge size data sets

Adult	Kdd Cup'99
$t = 10$ Strata	$t = 100$ Strata
$t = 50$ Strata	$t = 200$ Strata
$t = 100$ Strata	$t = 300$ Strata

tion algorithms, respectively. In order to observe the level of robustness achieved by all the algorithms, the table presents the average in the ten fold cross validation process of the results offered by each algorithm in the data sets evaluated. Each column shows:

- The first column shows the name of the algorithm. In this column the name is followed by the sort of validation process `Tfcv strat` and the number of strata, or `Tfcv classic` meaning classic ten fold cross process.
- The second column contains the average execution time (in seconds) associated to each algorithm. The algorithms have been run in a Pentium 4, 2.4 Ghz, 256 RAM, 40 Gb HD.
- The third column shows the average reduction percentage from the initial training sets.
- The fourth column contains the training accuracy associated to the prototype subset selected. The accuracy is calculated by means of 1-NN.
- The fifth column contains the test accuracy of the PS algorithms selection. This accuracy is calculated by means of 1-NN.

5. Experimental results and analysis

This section shows the results and the analysis.

5.1. Experimental results

Tables 8–10 contain the results obtained in the evaluation of Pen-based recognition, SatImage and Thyroid data sets, respectively. Due to their minor size we have developed the executions of the PS algorithms following both Ten fold cross validation procedures, classic and stratified one.

In Table 11, we present the results obtained in the evaluation of Adult data set. In this table we

Table 8
Results associated to Pen-based Recognition data set

Algorithm	Ex.Tim	Reduc. (%)	Ac. Trn (%)	Ac. Tst (%)
1-NN Tfcv classic	66		99.36	99.39
Cnn Tfcv classic	4	98.04	84.85	85.69
Cnn Tfcv strat 10	0.20	91.81	93.78	95.43
Cnn Tfcv strat 30	0.07	82.48	97.51	98.63
Drop1 Tfcv classic	374	98.45	86.23	86.02
Drop1 Tfcv strat 10	2	99.86	57.14	22.00
Drop1 Tfcv strat 30	0.23	99.70	68.96	38.90
Drop2 Tfcv classic	318	97.69	91.03	91.06
Drop2 Tfcv strat 10	1.9	98.50	52.98	62.92
Drop2 Tfcv strat 30	0.27	95.37	81.83	78.08
Drop3 Tfcv classic	391	98.07	90.33	90.05
Drop3 Tfcv strat 10	2.1	99.66	53.12	40.91
Drop3 Tfcv strat 30	0.23	98.60	90.51	57.53
Ib2 Tfcv classic	2	98.49	74.20	75.04
Ib2 Tfcv strat 10	0.1	94.31	93.73	91.41
Ib2 Tfcv strat 30	0.03	88.34	96.25	97.80
Ib3 Tfcv classic	9	96.42	96.73	98.00
Ib3 Tfcv strat 10	0.2	88.34	92.95	98.44
Ib3 Tfcv strat 30	0.1	83.05	97.07	98.63
CHC Tfcv classic	18845	98.99	96.29	98.94
CHC Tfcv strat 10	127	96.65	98.85	97.35
CHC Tfcv strat 30	31	93.78	99.69	97.53

Table 9
Results associated to SatImage data set

Algorithm	Ex.Tim	Reduc. (%)	Ac. Trn (%)	Ac. Tst (%)
1-NN Tfcv classic	36		90.33	90.41
Cnn Tfcv classic	5	95.93	60.63	61.96
Cnn Tfcv strat 10	0.1	88.42	68.91	75.62
Cnn Tfcv strat 30	0.10	79.49	76.37	80.46
Drop1 Tfcv classic	206	93.66	84.29	81.68
Drop1 Tfcv strat 10	1.3	98.03	83.18	38.12
Drop1 Tfcv strat 30	0.13	97.89	86.20	30.69
Drop2 Tfcv classic	183	83.49	83.45	83.51
Drop2 Tfcv strat 10	1.2	83.55	58.21	79.53
Drop2 Tfcv strat 30	0.20	80.85	65.07	79.06
Drop3 Tfcv classic	301	93.25	87.93	81.03
Drop3 Tfcv strat 10	1.00	96.81	66.46	73.02
Drop3 Tfcv strat 30	0.13	96.65	71.14	57.65
Ib2 Tfcv classic	3	96.75	59.00	59.59
Ib2 Tfcv strat 10	0.20	91.87	72.15	66.87
Ib2 Tfcv strat 30	0.07	85.77	75.56	75.81
Ib3 Tfcv classic	22	84.66	84.51	86.45
Ib3 Tfcv strat 10	0.30	78.11	68.95	87.50
Ib3 Tfcv strat 30	0.10	73.71	77.40	87.90
CHC Tfcv classic	2479	99.06	89.45	89.67
CHC Tfcv strat 10	57	97.52	95.23	88.28
CHC Tfcv strat 30	30	94.32	97.19	89.76

Table 10
Results associated to Thyroid data set

Algorithm	Ex.Tim	Reduc. (%)	Ac. Trn (%)	Ac. Tst (%)
1-NN Tfcv classic	28		92.87	92.74
Cnn Tfcv classic	3	98.00	92.50	92.86
Cnn Tfcv strat 10	0.10	90.72	73.13	90.66
Cnn Tfcv strat 30	0.02	84.32	76.47	89.58
Drop1 Tfcv classic	182	98.06	63.47	62.86
Drop1 Tfcv strat 10	1.00	99.21	80.39	90.25
Drop1 Tfcv strat 30	0.13	99.36	82.22	92.5
Drop2 Tfcv classic	143	87.54	91.37	91.15
Drop2 Tfcv strat 10	0.70	87.67	53.40	81.19
Drop2 Tfcv strat 30	0.13	86.25	61.94	81.25
Drop3 Tfcv classic	322	97.44	88.82	85.24
Drop3 Tfcv strat 10	0.80	99.45	80.55	84.81
Drop3 Tfcv strat 30	0.10	99.71	91.17	91.66
Ib2 Tfcv classic	2	98.11	92.53	92.89
Ib2 Tfcv strat 10	0.10	92.92	76.50	90.80
Ib2 Tfcv strat 30	0.01	85.41	76.58	89.58
Ib3 Tfcv classic	94	33.93	93.22	93.38
Ib3 Tfcv strat 10	0.50	38.62	93.11	92.33
Ib3 Tfcv strat 30	0.03	33.17	93.70	94.16
CHC Tfcv classic	2891	99.83	94.20	91.98
CHC Tfcv strat 10	54	99.44	88.25	94.01
CHC Tfcv strat 30	33	99.16	96.49	93.33

have introduced, when the resources consumption permit us (in Cnn, Ib2 and Ib3 case), the evaluation of the algorithm following the Tfcv classic and the Tfcv strat. We have included the evaluation of 1-NN algorithm in the whole data set to note the benefits obtained by the application of our proposal.

Table 12, contains the results associated to Kdd Cup'99 data set. This data set presents higher number of characteristics and instances than the previous data sets. This situation produces that some algorithms like the Drop family, which need more resources to be executed, cannot be evaluated.

5.2. Analysis

The analysis of Tables 8–12 allow us to make the following analysis according to different points of views.

5.2.1. Efficiency

As we can see in the second column of the tables, the stratified strategy reduces significatively execution time. Depending on the number of strata, this reduction allows us the execution of more demanding resources algorithms or decreases their evaluation time and resources needs. The reduction in execution time in the CHC case has to be highlighted. This reduction eliminates the efficiency problem that appears in the case of EAs applied to high size data sets.

In Table 11 dedicated to Adult data set, we can take note that the most resources consuming algorithms cannot be executed in Tfcv classic due to the resources necessities it involves.

The same situation appears in Table 12, where due to the dimension of this data set, some of the non-evolutionary algorithms cannot be evaluated in this case in anyone of the validation processes applied. We have mentioned this drawback in

PATREC 3554
27 October 2004 Disk Used
ARTICLE IN PRESS
No. of Pages 11, DTD = 5.0.1

Table 11
Results associated to Adult data set

Algorithm	Ex.Tim	Reduc. (%)	Ac. Trn (%)	Ac. Tst (%)
1-NN Tfcv classic	24		79.34	79.24
Cnn Tfcv classic	4	99.21	26.40	26.56
Cnn Tfcv strat 10	1	97.34	35.37	32.02
Cnn Tfcv strat 50	0	93.69	66.51	57.42
Cnn Tfcv strat 100	0	90.09	64.42	58.27
Drop1 Tfcv strat 10	44	95.09	100.00	25.64
Drop1 Tfcv strat 50	1	94.59	100.00	24.96
Drop1 Tfcv strat 100	0	94.49	100.00	24.83
Drop2 Tfcv strat 10	48	70.33	27.71	61.30
Drop2 Tfcv strat 50	0	68.03	56.90	70.27
Drop2 Tfcv strat 100	0	66.96	59.31	71.85
Drop3 Tfcv strat 10	41	95.57	48.98	63.46
Drop3 Tfcv strat 50	0	95.34	64.83	71.19
Drop3 Tfcv strat 100	0	93.71	65.82	70.19
Ib2 Tfcv classic	2	99.94	25.20	25.14
Ib2 Tfcv strat 10	1	99.57	52.33	26.89
Ib2 Tfcv strat 50	0	98.66	74.72	45.68
Ib2 Tfcv strat 100	0	94.33	67.66	54.30
Ib3 Tfcv classic	210	98.66	74.72	45.68
Ib3 Tfcv strat 10	3	76.69	33.98	70.96
Ib3 Tfcv strat 50	0	73.48	63.93	74.36
Ib3 Tfcv strat 100	0	71.21	68.12	71.52
CHC Tfcv strat 10	20172	99.38	97.02	81.92
CHC Tfcv strat 50	48	98.34	93.66	80.17
CHC Tfcv strat 100	14	97.03	94.28	77.81

Section 2. The second column in this table shows the significant cost associated to the execution of 1-NN algorithm over the whole data set. It is obvious that any kind of reduction is needed to carry out a successful use of this data set. A new reduction in execution time, due to stratified strategy, appears in this data set. 1-NN needs 18568 s, while the selection by means of stratified CHC is done, for example using 200 strata, in 418 s.

As summary, we can point the following:

- Stratification strategy reduces significantly execution time.
- The non-evolutionary algorithms evaluated improve the execution time of the evolutionary ones. We have to study if they are efficacy as well.

Table 12
Results associated to Kdd Cup'99 data set

Algorithm	Ex.Tim	Reduc. (%)	Ac. Trn (%)	Ac. Tst (%)
1-NN Tfcv classic	18568		99.91	99.91
Cnn Tfcv strat 100	8	81.61	99.30	99.27
Cnn Tfcv strat 200	3	65.57	99.90	99.15
Cnn Tfcv strat 300	1	63.38	99.89	98.73
Ib2 Tfcv strat 100	7	82.01	97.90	98.19
Ib2 Tfcv strat 200	3	65.66	99.93	98.71
Ib2 Tfcv strat 300	2	60.31	99.89	99.03
Ib3 Tfcv strat 100	2	78.82	93.83	98.82
Ib3 Tfcv strat 200	0	98.27	98.37	98.93
Ib3 Tfcv strat 300	0	97.97	97.92	99.27
CHC Tfcv strat 100	1960	99.68	99.21	99.43
CHC Tfcv strat 200	418	99.48	99.92	99.23
CHC Tfcv strat 300	208	99.28	99.93	99.19

5.2.2. Reduction rates

The final subset selected following the stratified strategy is slightly bigger than the one selected using the algorithm without stratification in the whole data set.

The best reduction rates are offered by the stratified CHC, overcoming to non-evolutionary ones in all size data sets.

5.2.3. Accuracy rates

The last column in the tables is dedicated to study the classification capabilities associated to the final subsets selected. As we can see, the non-evolutionary algorithms (with stratification or not) cannot improve the accuracy offered by the 1-NN (where 1-NN is evaluated in a Tfcv classic).

The best algorithm in test accuracy rate is the stratified CHC which presents rates similar than obtained by 1-NN.

Having accuracy rate as goal we can point the following:

- 1-NN applied to the whole data set offers the best result in most of the data sets.
- Stratified CHC is the algorithm which presents the accuracy rates with the best approximation to the 1-NN ones in all data sets.

5.2.4. Balance efficacy–efficiency

Stratified CHC offers the best balance between accuracy and reduction. It reduces the initial data set approximately at 98% in all data sets, maintaining and improving the accuracy rate provided by 1-NN. Stratified CHC presents the best results.

Non-evolutionary algorithms are faster than Stratified CHC, but they presents smaller reduction and accuracy rates. When the number of strata is increased, the execution time is reduced.

Stratified CHC, in a huge data set like Kdd Cup'99 (Table 12), presents the best balance between accuracy and reduction rates. It reduces the initial Kdd Cup'99 data set size (with 494 022 instances) around the 99.5% (2470 instances in the final subset selected), maintaining accuracy rates near to 99.2%, in 208 s.

Non-evolutionary algorithms following the stratified strategy can be executed more efficiently. The stratified execution reduces their resources needs, but they don't maintain their efficacy. They don't present a balanced behaviour between accuracy and reduction rates.

The CHC algorithm following a stratification strategy outperforms non-evolutionary PS algorithms, offering the best balance among resources necessities, reduction and accuracy rates. It decreases in all different data set the initial data set around the 99%, maintaining the accuracy rate similar than the one offered by 1-NN. The reduction in resources consumption induced by the stratified strategy presents a good solution to the Scaling Up problem, and improves the CHC efficiency maintaining its efficacy.

Briefly summarizing this section, we can point:

- Non-evolutionary algorithms are more efficient than evolutionary ones, but their result are worse.
- Stratified CHC presents the best balance among reduction rate, accuracy rate and execution time.

6. Concluding remarks

This paper addressed the Scaling Up problem involved when prototype selection algorithms are applied in large size data sets. The proposal is to combine a stratification strategy with the PS algorithm.

An experimental study has been carried out to compare the results of an EA model with the non-evolutionary Prototype Selection ones, in medium, large and huge size data sets, evaluating the drawbacks introduced by the Scaling Up problem.

The main conclusions reached are as follows:

- The proper election in the number of strata decreases significantly execution time and resources consumption, maintaining the algorithm's behaviour in accuracy and reduction rates.
- Stratification in non-evolutionary algorithms reduces their resources needs, improving their efficiency, but the EAs offer better results.
- Stratified CHC algorithm obtains best reduction rates in the data sets evaluated. It significantly reduces the size of the subset selected (>95% in reduction rate).
- Stratified CHC maintains classification capabilities similar than the offered by 1-NN applied over the whole data set.
- Stratified CHC offers the best results in all data sets, maintaining its behaviour when we increase the size of the data set (from 7200 instances in Thyroid to 494022 instances in Kdd Cup'99).
- Our proposal offers the best balance among accuracy, reduction rates, execution time and resources needs in all data sets evaluated, outperforming the non-evolutionary algorithms.

Therefore, as a final concluding remark, we consider stratified strategy combined with CHC to be the best mechanism in Prototype Selection in large size data sets. It has become a powerful tool to face to the Scaling Up problem. CHC selects the most representative instances, satisfying both objectives: high accuracy and reduction rates. Stratified strategy reduces the search space so we can carry out the evaluation of the algorithms in acceptable running time decreasing the resources that it needs.

Acknowledgement

This work was supported by Project TIC2002-04036-C05-01.

References

Aha, D.W., Kibbler, D., Albert, M.K., 1991. Instance based learning algorithms. Machine Learning 6, 37–66.

Angluin, D., Laird, P., 1987. Learning from noisy examples. Machine Learning 2 (4), 343–370.

Back, T., Fogel, D., Michalewicz, Z., 1997. Handbook of evolutionary computation. Oxford University Press.

Cano, J.R., Herrera, F., Lozano, M., 2003. Using evolutionary algorithms as instance selection for data reduction in KDD: An experimental study, IEEE Transaction on Evolutionary Computation, in press.

Cover, T., Hart, P., 1967. Nearest neighbour classification. IEEE Trans. on Inf. Theory IT-13 (1), 21–27.

Eshelman, L.J., 1991. The CHC adaptive search algorithm: How to have safe search when engaging in nontraditional genetic recombination. In: Rawlins, G.J.E. (Ed.), Foundations of Genetic Algorithms 1. Morgan Kauffman, pp. 265–283.

Forrest, S., Mitchell, M., 1993. What makes a problem hard for a genetic algorithm? Some anomalous results and their explanation. Machine Learning 13, 285–319.

Gates, G.W., 1972. The reduced nearest neighbour rule. IEEE Trans. on Inf. Theory 18 (5), 431–433.

Goldberg, D.E., 1989. Genetic algorithms in search, optimization, and machine learning. Addison-Wesley.

Hart, P.E., 1968. The condensed nearest neighbour rule. IEEE Trans. on Inf. Theory 18 (3), 431–433.

Kibbler, D., Aha, D.W., 1987. Learning representative exemplars of concepts: An initial case of study. In: Proceedings of the Fourth International Workshop on Machine Learning. Morgan Kaufmann, pp. 24–30.

Kuncheva, L., 1995. Editing for the *k*-nearest neighbors rule by a genetic algorithm. Pattern Recognition Lett. 16, 809–814.

Merz, C.J., Murphy, P.M., 1996. UCI repository of machine learning databases, University of California Irvine, Department of Information and Computer Science, Available from: <http://kdd.ics.uci.edu>.

Nakashima, T., Ishibuchi, H., 1998. GA-based approaches for finding the minimum reference set for nearest neighbour classification. Proceedings of the IEEE International Conference on Evolutionary Computation, 709–714.

Ravindra, T., Narasimha, M., 2001. Comparison of genetic algorithm based prototype selection schemes. Pattern Recognition 34, 523–525.

Ritter, G.L., Woodruff, H.B., Lowry, S.R., Isenhour, T.L., 1975. An algorithm for a selective nearest neighbour decision rule. IEEE Trans. on Inf. Theory 21 (6), 665–669.

Shinn-Ying, H., Chia-Cheng, L., Soundy, L., 2002. Design of an optimal nearest neighbour classifier using an intelligent genetic algorithm. Pattern Recognition Lett. 23 (13), 1495–1503.

Wilson, D.L., 1972. Asymptotic properties of nearest neighbour rules using edited data. IEEE Transactions on Systems Man. and Cybernetics 2, 408–421.

Wilson, D.R., Martinez, T.R., 1997. Instance pruning techniques. In: Proceedings of the 14th International Conference. Morgan Kaufmann, pp. 403–411.

Wilson, D.R., Martinez, T.R., 2000. An integrated instance-based learning algorithm. Computational Intelligence 16 (1), 1–28.

Wilson, D.R., Martinez, T.R., 2000. Reduction techniques for instance-based learning algorithms. Machine Learning 38, 257–268.