2. Strategies for Scaling Up Evolutionary Instance Reduction Algorithms for Data Mining

Jose Ramon Cano[1], Francisco Herrera[2], and Manuel Lozano[2]

[1] Dept. of Electronic Engineering, Computer Systems and Automatics, Escuela Superior de La Rabida, University of Huelva, 21819, Huelva, Spain
jose.cano@diesia.uhu.es

[2] Dept. of Computer Science and Artificial Intelligence, University of Granada, 18071, Granada, Spain
herrera@decsai.ugr.es, lozano@decsai.ugr.es

Abstract: Evolutionary algorithms are adaptive methods based on natural evolution that may be used for search and optimization. As instance selection can be viewed as a search problem, it could be solved using evolutionary algorithms.

In this chapter, we have carried out an empirical study of the performance of CHC as representative evolutionary algorithm model. This study includes a comparison between this algorithm and other non-evolutionary instance selection algorithms applied in different size data sets to evaluate the scaling up problem. The results show that the stratified evolutionary instance selection algorithms consistently outperform the non-evolutionary ones. The main advantages are: better instance reduction rates, higher classification accuracy and reduction in resources consumption.

2.1 Introduction

Advances in digital and computer technology that have led to the huge expansion of the Internet means that massive amounts of information and collection of data have to be processed. Due to the enormous amounts of data, much of the current research is based on scaling up [2.5] Data Mining (DM) ([2.1, 2.20, 2.22]) algorithms. Other research has also tackled scaling down data. The main problem of scaling down data is how to select the relevant data. This task is carried out in the data preprocessing phase in a Knowledge Discovery in Databases (KDD) process.

Our attention is focused on Data Reduction (DR) [2.16] as preprocessing task, which can be achieved in many ways: by selecting features [2.15], by making the feature-values discrete [2.8], and by selecting instances([2.13]). We led our study to Instance Selection (IS) as DR mechanism ([2.3, 2.17, 2.18]), where we reduce the number of rows in a data set (each row represents and instance). IS can follow different strategies (see Fig. 2.1): sampling, boosting, prototype selection (PS), and active learning. We are going to study the IS from the PS perspective.

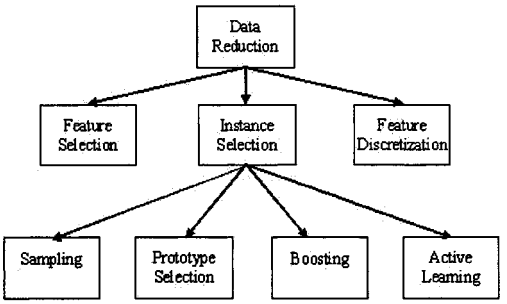

Fig. 2.1. Data reduction strategies

IS mechanisms have been proposed to choose the most suitable points in the data set to become instances for the training data set used by a learning algorithm. IS has been studied previously in the literature using different approaches, in particular by means of Genetic Algorithms (GA) ([2.11]) as PS approach. For example, in [2.14], a GA is used for carry out a k-nearest neighbor edition.

Evolutionary Algorithms (EAs) ([2.2]) are general-purpose search algorithms that use principles inspired by natural genetic populations to evolve solutions to problems. The basic idea is to maintain a population of chromosomes, which represent plausible solutions to the problem, which evolves over time through a process of competition and controlled variation. EAs in general and GAs in particular have been used to solve the IS problem, with promising results ([2.14]).

The issue of scalability and the effect of increasing the size of data sets are always present in the algorithm behavior. This scaling up drawback appears in EAs due to the increasing of the chromosome's size, which reduces the EAs convergence capabilities.

To avoid this drawback we offer a combination of EAs and a stratified strategy. In large size we can't evaluate the algorithms over the complete data set so the stratification is a way to carry out the executions. Combining the subsets selected from the strata we can obtain the subset selected for the whole initial data set. The stratification reduces the data set size, while EAs select the most representative prototype per stratus.

The aim of this chapter is to study the application of a representative and efficient EA model for data reduction, the CHC algorithm in IS ([2.6, 2.4]), and to compare it with non-evolutionary instance selection algorithms (hereafter referred to as classical ones) following a stratified strategy.

To address this, we have carried out a number of experiments increasing the complexity and the data set size.

In order to do that, this chapter is set up as follows. In Section 2.2, we introduce the main ideas about IS, describing the processes which IS algorithms take part, and we also summarize the classical IS algorithms used in this study. In Section 2.3, we introduce the foundations of EAs and summarize the main features of them, giving details of how EAs can be applied to the IS problem in large size data sets. Section 2.4 is dedicated to the Scaling Up problem and we present our proposal solution. In Section 2.5, we explain the methodology used in the experiments. Section 2.6 deals with the results and the analysis of medium and large size data sets. Finally, in Section 2.7, we point out our conclusion.

2.2 Instance Selection on Data Reduction

In this section we describe the strategy which IS takes part in, as a DR mechanism, as well as a summary of classical IS algorithms.

2.2.1 Instance Selection

In IS we want to isolate the smallest set of instances which enable us to predict the class of a query instance with the same (or higher) accuracy as the original set [2.16]. By reducing the "useful" data set size, which can reduce both space and time complexities of subsequent processing phases. One can also hope to reduce the size of formulas obtained by a subsequent induction algorithm on the reduced and less noise data sets. This may facilitate interpretation tasks.

IS raises the problem of defining relevance for a prototype subset. From the statistical viewpoint, relevance can be partly understood as the contribution to the overall accuracy, that would be e.g. obtained by a subsequent induction. We emphasize that removing instances does not necessarily lead to a degradation of the results: we have observed experimentally that a little number of instances can have performances comparable to those of the whole sample, and sometimes higher. Two reasons come to mind to explain such an observation:

- First, some noises or repetitions in data could be deleted by removing instances.
- Second, each instance can be viewed as a supplementary degree of freedom. If we reduce the number of instances, we can sometimes avoid over-fitting situations.

2.2.2 Instance Selection for Prototype Selection

There may be situations in which there is too much data and this data in most cases is not equally useful in the training phase of a learning algorithm.

Instance selection mechanisms have been proposed to choose the most suitable points in the data set to become instances for the training data set used by a learning algorithm.

Fig. 2.2 shows a general framework for the application of an IS algorithm for PS. Starting from the data set, D, the PS algorithm finds a suitable set, Prototype Subset Selected (PSS), then a learning or DM algorithm is applied to evaluate each subset selected (1-nearest neighbor in our case) to test the quality of the subset selected. This model is assessed using the test data set, TS.

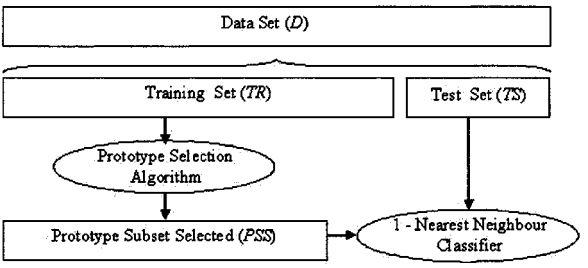

Fig. 2.2. Prototype selection strategy

2.2.3 Overview of Instance Selection Algorithms

Historically, IS has been mainly aimed at improving the efficiency of the Nearest Neighbor (NN) classifier. The NN algorithm is one of the most venerable algorithms in machine learning. This algorithm calculates the Euclidean distance (possibly weighted) between an instance to be classified and each training-neighboring instance. The new instance obtained is assigned to the class of the nearest neighboring one. More generally, the k-nearest neighbors (k-NN) are computed, and the new instance is assigned to the most frequent class among these k neighbors. The k-NN classifier was also widely used and encouraged by early theoretical results related to its Bayes error generalization.

However, from a practical point of view, the k-NN algorithm is not suitable for dealing with very large sets of data due to the storage requirements it demands and the computational costs involved. In fact, this approach requires the storage of all the instances in memory. Early research in instance selection firstly tried to reduce storage size. Taking as reference our study in [2.4] we select the most effective classic algorithms to evaluate them.

The algorithms used in this study will be:

Methods based on nearest neighbor rules.

- Cnn [2.12] - 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.
- Ib2 [2.13] - It is similar to Cnn but using a different selection strategy.
- Ib3 [2.13] - It outperforms Ib2 introducing the acceptable instance concept to carry out the selection.

Methods based on ordered removal.

- Drop1 [2.21] - 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 [2.21] - 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 [2.21] - 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 neighbors is removed.

2.3 Evolutionary Instance Selection Algorithms

EAs ([2.2]) are stochastic search methods that mimic the metaphor of natural biological evolution. All EAs rely on the concept of a population of individuals (representing search points in the space of potential solutions to a given problem), which undergo probabilistic operators such as mutation, selection, and (sometimes) recombination to evolve towards increasingly better fitness values of the individuals.

Most of the success of EAs is due to their ability to exploit the information accumulated about an initially unknown search space. This is their key feature, particularly in large, complex, and poorly understood search spaces, where classical search tools (enumerative, heuristic, etc.) are inappropriate. In such cases they offer a valid approach to problems requiring efficient and effective search techniques. Recently EAs have been widely applied to KDD and DM ([2.9, 2.10]).

In this section we firstly present the key-points of their application to our problem as well as the representation and the fitness function, and then describe the EA (CHC [2.6]) used in this study, according to the best results obtained by CHC in the study presented in [2.4].

In the Section 2.3.2 we describe the model of EA that will be used in this chapter as evolutionary instance selection algorithm. CHC is a classical model

that introduces different features to obtain a trade-off between exploration and exploitation.

2.3.1 Evolutionary Instance Selection: Key Points

EAs may be applied to the IS problem, because it can be considered as a search problem.

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

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 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.

Fitness function. Let PSS be a subset 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:

$$Fitness(\text{PSS}) = \alpha \cdot \texttt{clas\_rat} + (1 - \alpha) \cdot \texttt{perc\_red}. \tag{2.1}$$

The 1-NN classifier (Section 2.2.3) is used for measuring the classification rate, clas_rat, associated with PSS. It denotes the percentage of correctly classified objects from TR using only PSS to find the nearest neighbor. For each object y in TR, the nearest neighbor is searched for amongst those in the set PSS $\setminus$ {y}. Whereas, perc_red is defined as:

$$\texttt{perc\_red} = 100 \cdot (|\text{TR}| - |\text{PSS}|)/|\text{TR}|. \tag{2.2}$$

The objective 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 chapter, we have considered the value $\alpha = 0.5$ in the fitness function, as per a previous experiment in which we found the best trade-off between precision and reduction with this value.

2.3.2 The CHC Algorithm

During each generation the CHC 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 re-initialized 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.

2.4 The Scaling up Problem. The Stratified Approach

In this section we point our attention in the Scaling Up problem and finally we describe our proposal, the combination of the stratified strategy with the evolutionary instance selection.

2.4.1 Scaling up and Stratification

The algorithms we have studied, both classical and evolutionary, are affected when the size of the data set increases. The main difficulties they have to face are as follows:

- Efficiency. The efficiency of IS algorithms is at least $O(n^2)$, where n is the size of the data set. Most of them present an efficiency order greater than $O(n^2)$. When the size increases, 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 problem is 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.
- Representation. EAs are also affected by representation, due to the size of their chromosomes. When the size of these chromosomes is too big, then it increases the algorithm convergence difficulties.

To avoid these drawback we led our experiments towards a stratified strategy. This strategy divides the initial data set in strata. The strata are disjoints sets with equal class distribution. We evaluate the algorithm over each stratus to carry out the data selection and finally we reunite the partial subsets to conform the final one.

In the following section (Fig. 2.3) we describe the use of the stratified strategy combined with EA.

2.4.2 Evolutive Algorithms and Stratification Strategy

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

Prototype selection algorithms (classical or evolutionary) are applied to each D_j obtaining a subset selected DS_j, as we can see in Fig. 2.3.

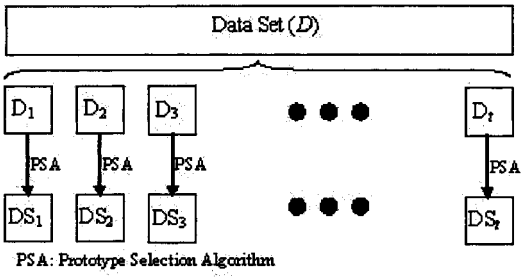

Fig. 2.3. Combination of prototype selection algorithms and stratified strategy

In Fig. 2.2, the PSS is obtained by the PS algorithm, applied on TR. In the stratified strategy, the PS algorithm is applied in each D_j to obtain its DS_j associated. PSS in stratified strategy is obtained using the DS_j (see Eq. (2.3)) and it is called Stratified Prototype Subset Selected (SPSS).

$$SPSS = \bigcup_{j \in J} DS_j, J \subset \{1, 2, ..., t\} \tag{2.3}$$

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

$$TR = \bigcup_{j \in J} D_j, J \subset \{1, 2, ..., t\} \tag{2.4}$$

$$TS = D \setminus TR \tag{2.5}$$

Our specific model will be described in Section 2.5.2.

2.5 Experimental Methodology

We have carried out our study of IS problem using two size problems: medium and large. We intend to study the behavior of the algorithms when the size of the problem increases. Section 2.5.1 describes the data sets used and introduces the parameters associated with the algorithms, Section 2.5.2 introduces the stratification and partition of the data sets that were considered for applying the algorithms, and finally, in Section 2.5.3 we describe the table contents that show the results.

2.5.1 Data Sets and Parameters

The data sets used are shown in Table 2.1 and 2.2. They can be found in the UCI Repository (http://kdd.ics.uci.edu/).

Table 2.1. Medium size data sets

Data Set	Num. Instances	Num. Features	Num. Classes
Pen-Based Recognition	10992	16	10
Satimage	6435	36	6
Thyroid	7200	21	3

Table 2.2. Large size data set

Data Set	Num. Instances	Num. Features	Num. Classes
Adult	30132	14	2

The parameters used are shown in Table 2.3.

Table 2.3. Parameters

Algorithm	Parameters
Ib3	Acept. Level=0.9, Drop Level=0.7
CHC	Population=50, Evaluations=10000

2.5.2 Partitions and Stratification: A 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 evaluated 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. 2.4. We call it Ten fold cross validation classic (`Tfcv classic`). The idea is use this result as baseline versus the stratification ones.

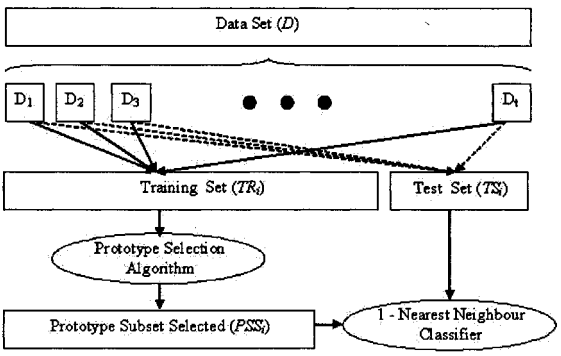

Fig. 2.4. Prototype selection strategy in Tfcv classic

In `Tfcv classic` the subsets TR_i and TS_i, i=1, ..., 10 are obtained as the Eqs. (2.6) and (2.7) indicate:

$$TR_i = \bigcup_{j \in J} D_j, J = \{j/1 \leq j \leq b \cdot (i-1) \; and \; (i \cdot b) + 1 \leq j \leq t\} \quad (2.6)$$

$$TS_i = D \setminus TR_i \quad (2.7)$$

where t is the number of strata, and b is the number of strata grouped ($b = t/10$).

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. 2.5 shows. We call it Ten fold cross validation strat (`Tfcv strat`).

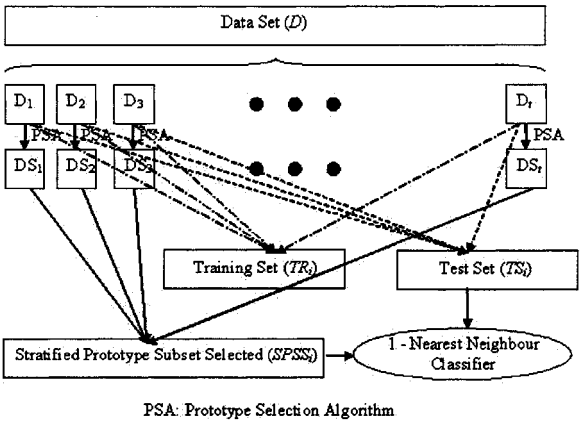

Fig. 2.5. Prototype selection strategy in Tfcv strat

In `Tfcv strat` each TR_i is defined as we can see in Eq. (2.6), by means of the union of D_j subsets (see Fig. 2.5).

In `Tfcv strat` (see Fig. 2.5) $SPSS_i$ is generated using the DS_j (see Eq. (2.8)).

$$SPSS_i = \bigcup_{j \in J} DS_j, J = \{j/1 \leq j \leq b \cdot (i-1) \; and \; (i \cdot b)+1 \leq j \leq t\} \quad (2.8)$$

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

The subset TS_i is defined by means the Eq. (2.7). Both, TR_i and TS_i are generated in the same way in `Tfcv classic` and `Tfcv strat`.

For each data set we have employed the following partitions and number of strata:

Table 2.4. Stratification in medium size data sets

Pen-Based Recognition	Satimage	Thyroid
10 Strata	10 Strata	10 Strata
30 Strata	30 Strata	30 Strata

Table 2.5. Stratification in large size data set

Adult
10 Strata
50 Strata
100 Strata

2.5.3 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 classical and evolutionary instance selection algorithms, respectively. In order to observe the level of robustness achieved by all 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 ten fold cross-validation process classic).
- The second column contains the average execution time 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 fifth column contains the test accuracy of the PS algorithms selection.

2.6 Experimental Study

In this section we present the results obtained in the evaluation of medium and large data sets and their analysis.

2.6.1 Medium Size Data Sets

The following conclusions about the IS algorithms for PS can be made studying Table 2.6:

- In Table 2.6 we can see that the stratification strategy reduces significantly the execution time.
- Stratified strategy affects in different manner to the accuracy rates associated to the classic algorithms. Some of them, like `Ib2`, `Ib3` or `Cnn`, increase their accuracy, but other group (`Drop1`, `Drop2` and `Drop`) reduces it.

Table 2.6. Prototype selection for pen-based recognition data set

	Exec. Time(sec)	% Reduction	1-NN %Ac.Trn	1-NN %Ac.Test
1-NN	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%

- CHC and its stratified version have not been improved in their test accuracy by classic methods which offer small reduction rates. They offer the best balance between reduction and accuracy rates.
- The Stratified CHC is the one which presents the best behavior among time and resources consumption, and reduction and accuracy rates. The classic algorithm which can face to Stratified CHC is Ib3 following a `Tfcv classic`, which can be hard to use when the size of the problem is huge due to its resources necessities.

Table 2.7. Prototype selection for Satimage data set

	Exec. Time(sec)	% Reduction	1-NN %Ac.Trn	1-NN %Ac.Test
1-NN	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%

The following conclusions can be made studying Table 2.7:

- Execution time is decreased by the stratified strategy in the same way that we saw it in Table 2.6. We can see that the stratification strategy reduces significantly the execution time.
- Stratified strategy affects in different manner to the accuracy rates associated to the classic algorithms. We can see the group conformed by the Drop family algorithms and other group with the rest of classic algorithms. The first group reduces its accuracy associated when they are evaluated following a stratification strategy while the second group increase it.

- In Satimage, CHC and its stratified version offer the best balance between reduction and accuracy rates. They have not been improved in their test accuracy by classic methods which offer small reduction rates.
- Like in Pen-Based Recognition data set, the Stratified CHC presents the best behavior among time and resources consumption, and reduction and accuracy rates.

Table 2.8. Prototype selection for thyroid data set

	Exec. Time(sec)	% Reduction	1-NN %Ac.Trn	1-NN %Ac.Test
1-NN	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%

The following conclusions can be made studying Table 2.8:

- Execution time is reduced by the stratified strategy as in the Tables 2.6 and 2.7.
- In Thyroid data set, CHC and its stratified version have not been improved in their test accuracy by classic methods which offer small reduction rates. They offer the best balance between reduction and accuracy rates.
- The Stratified CHC is the one which present the best behavior among time and resources consumption, and reduction and accuracy rates.

The main conclusion that can be drawn when using medium size data sets is that Stratified CHC is the best algorithm for data reduction having

both high reduction rates and accuracy, and decreasing execution time and resources consumption.

2.6.2 Large Size Data Set

Table 2.9. Prototype selection for adult data set

	Exec. Time(sec)	% Reduction	1-NN %Ac.Trn	1-NN %Ac.Test
1-NN	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	0.20	93.69%	66.51%	57.42%
Cnn Tfcv strat 100	0.02	90.09%	64.42%	58.27%
Drop1 Tfcv strat 10	44	95.09%	100.00%	25.64%
Drop1 Tfcv strat 50	1.2	94.59%	100.00%	24.96%
Drop1 Tfcv strat 100	0.15	94.49%	100.00%	24.83%
Drop2 Tfcv strat 10	48	70.33%	27.71%	61.30%
Drop2 Tfcv strat 50	0.7	68.03%	56.90%	70.27%
Drop2 Tfcv strat 100	0.13	66.96%	59.31%	71.85%
Drop3 Tfcv strat 10	41	95.57%	48.98%	63.46%
Drop3 Tfcv strat 50	0.8	95.34%	64.83%	71.19%
Drop3 Tfcv strat 100	0.11	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	0.1	98.66%	74.72%	45.68%
Ib2 Tfcv strat 100	0.03	94.33%	67.66%	54.30%
Ib3 Tfcv classic	210	79.42%	72.61%	74.09%
Ib3 Tfcv strat 10	3	76.69%	33.98%	70.96%
Ib3 Tfcv strat 50	0.4	73.48%	63.93%	74.36%
Ib3 Tfcv strat 100	0.05	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%

We point out the following conclusions:

- If we pay attention to Table 2.9 we can see that only `Cnn`, `Ib2` and `Ib3` have been evaluated in a `Tfcv classic` validation. This is due to the size of the data set makes too hard to evaluate the rest of algorithms. This is one of the reason to advice the use of a stratified strategy like the one proposed by us.
- There is an important reduction in execution time due to the stratified strategy. In Stratified `CHC` we have reduced its execution time associated from 40.391 seconds using 3 strata to 14 seconds using 100 strata.

- Stratified CHC offers the best behavior. It presents the best reduction rates and accuracy rates, combined with a lower execution time. The fifth column in Table 2.9 shows that stratified CHC is the best algorithm offering the highest accuracy and reduction rates.
- The classical algorithms which present higher accuracy rate, offer smaller reduction rates. Those which present higher reduction rates, show minimal accuracy rates.

Clearly, when we manage large size data sets, the stratified CHC algorithm improves the behavior of the classic ones, giving the best results for scaling down data.

If we take note of Table 2.9, the initial data set which needs 24 seconds to be evaluated using 1-NN, is reduced (in 14 seconds) in 97.03% by the stratified CHC, losing less than 1.5% in accuracy rate. This situation shows that our proposal is an effective data reduction alternative to be applied in large size data sets.

Taking Table 2.9 as reference, and more concretely the Stratified CHC as the best algorithm evaluated, we can study the effect of the number of strata over the algorithm's behavior.

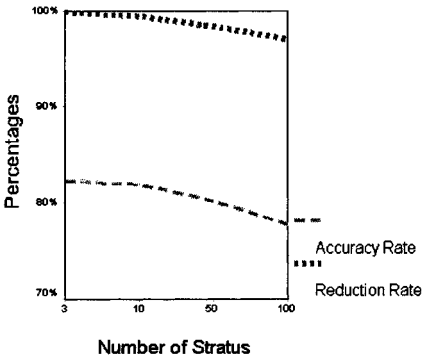

Fig. 2.6. Stratus effect on accuracy and reduction rates in Adult data set

As we can see in Fig. 2.6, when the number of strata increases, both the accuracy and reduction rate decrease. This situation is due to the selection carried out in each stratus. When the number of strata increases, the number of equivalent selected instances in different subsets also increases. This situation produces that the size of the final subset selected is bigger.

2.7 Concluding Remarks

This chapter addressed the analysis of the evolutionary instance selection by means of CHC and their use in data reduction for large data sets in KDD. We have studied the effect of the stratified strategy in the scaling up of the algorithms.

The main conclusions reached are the following:

- The Stratified Strategy reduces significantly the execution time and the resources consumed by classic and CHC algorithm. This situation offers two principal advantages: First, the evaluation of large data sets which needs too much resources is feasible, and second, the reduction in time associated to its execution.
- Stratified CHC outperform the classical algorithms, simultaneously offering two main advantages: better data reduction percentages and higher classification accuracy.
- In medium and large size data sets, classical algorithms do not present balanced behavior. If the algorithm reduces the size then its accuracy rate is poor. When accuracy increases there is no reduction.
- The increase in the number of strata can produce a small degradation in the algorithm behavior as we indicated in Fig. 2.6. The adequate number of them has to be chosen to produce a balance between time and resources consumption in one side, and reduction and accuracy rates by the other side.

Therefore, as a final concluding remark, we consider the stratified strategy combined with CHC to be a good mechanism for data reduction, facing to the problem of Scaling Up. It has become a powerful tool to obtain small selected training sets and therefore scaling down data. CHC can select the most representative instances, satisfying both the objectives of high accuracy and reduction rates. Stratified strategy permits a reduction of the search space so we can carry out the evaluation of the algorithms with acceptable execution time, and decreasing the resources necessities.

Finally, we point out that future research could be directed towards the study of hybrid strategies between classical and evolutionary instance selection algorithms.

References

2.1 Adriaans, P., Zantinge, D. (1996): Data mining. Addison-Wesley
2.2 Back, T., Fogel, D., Michalewicz, Z. (1997): Handbook of evolutionary computation. Oxford University Press
2.3 Brighton, H., Mellish, C. (2002): Advances in instance selection for instance-based learning algorithms. Data Mining and Knowledge Discovery **6**, 153–172

2.4 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)
2.5 Domingo, C., Gavalda, R., Watanabe, O. (2002): Adaptative sampling methods for scaling up knowledge discovery algorithms. Data Mining and Knowledge Discovery **6**, 131–152
2.6 Eshelman, L. J. (1991): The CHC adaptive search algorithm: how to have safe search when engaging in nontraditional genetic recombination. (Foundations of Genetic Algorithms-1) , Rawlins, G.J.E. (Eds.), Morgan Kauffman, 265–283
2.7 Esposito, F., Malerba, D., Semeraro, G. (1997): A comparative analysis of methods for pruning decision trees. IEEE Transactions on Pattern Analysis and Machine Intelligence, **19**, 476–491
2.8 Frank, E., Witten, I. H. (1999): Making better use of global discretization. (Proc. Sixteenth International Conference on Machine Learning), Bratko, I., Dzeroski, S. (Eds.), Morgan Kaufmann, 115–123
2.9 Freitas, A. A. (2002): Data mining and knowledge discovery with evolutionary algorithms. Springer-Verlag
2.10 Freitas, A.A. (2002): A survey of evolutionary algorithms for data mining and knowledge discovery. (Advances in evolutionary computation), Ghosh, A., Tsutsui, S. (Eds.), Springer-Verlag, 819–845
2.11 Goldberg, D. E. (1989): Genetic algorithms in search, optimization, and machine learning. Addison-Wesley
2.12 Hart, P. E. (1968): The condensed nearest neighbour rule. IEEE Transaction on Information Theory, **18**, 431–433
2.13 Kibbler, D., Aha, D. W. (1987): Learning representative exemplars of concepts: An initial case of study. Proc. of the (Fourth International Workshop on Machine Learning) , Morgan Kaufmann, 24–30
2.14 Kuncheva, L. (1995): Editing for the k-nearest neighbors rule by a genetic algorithm. Pattern Recognition Letters, **16**, 809–814
2.15 Liu, H., Motoda, H. (1998): Feature selection for knowledge discovery and data mining. Kluwer Academic Publishers
2.16 Liu, H., Motoda, H. (2001): Data reduction via instance selection. (Instance Selection and Construction for Data Mining), Liu, H., Motoda, H. (Eds.) , Kluwer Academic Publishers, 3–20
2.17 Liu, H., Motoda, H. (2002): On issues of instance selection. Data Mining and Knowledge Discovery, **6**, 115–130
2.18 Reinartz, T. (2002): A unifying view on instance selection. Data mining and Knowledge Discovery, **6**, 191–210
2.19 Safavian, S. R., Landgrebe, D. (1991): A survey of decision tree classifier methodology. IEEE Transaction on Systems, Man. and Cybernetics, **21**, 660–674
2.20 Shanahan, J. G. (2000): Soft computing for knowledge discovery. Kluwer Academic Publishers
2.21 Wilson, D. R., Martinez, T. R. (1997): Instance pruning techniques. (Proceedings of the International Conference), Morgan Kaufmann, 403–411
2.22 Witten, I. H., Frank, E. (2000): Data mining: practical machine learning tools and techniques with Java implementations. Morgan Kaufmann