A Study of the Scaling up Capabilities of Stratified Prototype Generation.

I. Triguero, J. Derrac, F. Herrera
Department of Computer Science and Artificial Intelligence,
CITIC-UGR, University of Granada
18071 Granada, Spain
{*triguero,derrac,herrera*}*@decsai.ugr.es*

S. García
Department of Computer Science
University of Jaén
23071 Jaén, Spain
{*sglopez*}*@ujaen.es*

***Abstract*—Prototype generation is an appropriate data reduction process for improving the efficiency and the efficacy of the nearest neighbor rule. Specifically, evolutionary prototype generation techniques have been highlighted as the best performing methods. However, these methods can sometimes be inefficient when the data scale up. In other data reduction techniques, such as prototype selection, an stratification procedure has been successfully developed to deal with large data sets.**

In this study, we test the combination of stratification with prototype generation techniques, considering data sets with more than 10000 instances. We compare some of the most representative prototype reduction methods and perform a study of the effects of stratification in their behavior. The results, contrasted with nonparametric statistical tests, show that several prototype generation techniques present a better performance than previously analyzed methods.

***Keywords*-Stratification; Scaling up; Data Reduction; Prototype Generation; Nearest Neighbor**

I. Introduction

Nowadays, researchers in many fields such as marketing, medicine or biology have improved the way in which they acquire the data, allowing them to deal with greater and more difficult problems than before. It implies that data mining tools [2] must manage a higher number of data. Therefore, many classical techniques cannot work properly, or, at least, suffer several drawbacks in their application.

Data reduction processes are very useful in data mining to deal with large amounts of data. These techniques aim to obtain a representative training set with a lower size compared to the original one and a similar or even higher performance by any data mining application, e.g., supervised classification. In the literature, the most used data reduction techniques are feature selection, feature generation, instance generation and instance selection.

The Nearest Neighbor (NN) rule [5] is a simple and effective supervised classification technique. NN is a nonparametric classifier, which requires storing all training data instances. Unseen instances are classified by finding the class labels of the closest instances to them. Thus, the effectiveness of the classification process relies on the quality of the training data and the efficiency of this technique depends on the size of the problem tackled.

Prototype reduction techniques are useful data reduction techniques designed to improve the performance of the NN rule. These techniques can be divided into two different approaches, which are known as Prototype Selection (PS) [10] and Prototype Generation (PG) [16]. The former process consists of choosing a subset of the original training data, while the latter can also build new artificial prototypes to adjust the decision boundaries between classes in NN. The aim of both approaches is to avoid the excessive storage and time complexity of the NN classifier.

In the specialized literature, a great number of PS and PG techniques have been proposed. Since the first approaches for PS, such as CNN [11] and ENN [20], and the first attempts for PG [4], many other proposals have become well-known in these fields [12]. Given that PS and PG problems could be seen as combinatorial and optimization problems, evolutionary algorithms have been used to solve them [9], [17] and have been highlighted as the most promising prototype reduction techniques [18].

In both PS and PG, the increase in the size of the database is a staple problem. This problem is also known as the scaling up problem and produces a excessive storage requirements, time complexity and generalization capabilities of the prototype reduction techniques. These drawbacks tend to be more important in evolutionary approaches, where a excessive increment of the individual size can limit the practicality of these algorithm on many large-scale problems.

Several strategies have been proposed to tackle the scaling up problem in various domains [14]. Some of them try to scale up the algorithms by proposing faster and lower resource consumption approaches. Another interesting group of techniques works directly with the data set. They split the data set into various parts to make the application of a data mining algorithm easier, using a mechanism to join the solutions of each part in a global solution. The latter approach has been successfully applied for PS [3], [6].

In this work, we analyze the scaling up capabilities of previously proposed PG techniques when it is applied in conjunction with the stratification procedure [3]. An empirical study will be carried out to test the performance of the proposed combination. We select some of the main PG techniques, comparing them with a previous stratified

PS method [6]. The results will be compared in terms of accuracy, reduction rates and runtime, using a nonparametric statistical test to determine the most appropriate stratified prototype reduction technique.

In order to organize this paper, Section II describes the background of prototype reduction techniques, the scaling up problem and the stratification. Section III discusses the experimental framework and presents the analysis of results. Finally, in Section IV we summarize our conclusions.

II. Prototype reduction techniques, scaling up and stratification

This section presents the main topics in the field in which this contribution is based. Section II-A defines the PS and PG problems in depth. Section II-B describes the main issues of the scaling up problem. Finally, Section II-C details the characteristics of the stratification procedure.

A. Prototype reduction techniques

This section presents the definition and notation for both PS and PG problems.

A formal specification of the PS problem is the following: Let $\mathbf{x}_p$ be an example where $\mathbf{x}_p = (\mathbf{x}_{p1}, \mathbf{x}_{p2}, ..., \mathbf{x}_{pN}, \omega)$, with $\mathbf{x}_p$ belonging to a class ω given by $\mathbf{x}_{p\omega}$ and a N-dimensional space in which $\mathbf{x}_{pi}$ is the value of the i-th feature of the p-th sample. Then, let us assume that there is a training set TR which consists of **n** instances $\mathbf{x}_p$ and a test set TS composed of **t** instances $\mathbf{x}_q$, with ω unknown. Let $RS \subseteq TR$ be the subset of selected samples resulting from the execution of a PS algorithm, then we classify a new pattern $\mathbf{x}_q$ from TS by the NN rule acting over RS.

The purpose of PG is to generate a reduced set RS, which consists of $\mathbf{r}$, $\mathbf{r} < \mathbf{n}$, prototypes, which are either selected or generated from the examples of TR. The prototypes of the generated set are determined to represent efficiently the distributions of the classes and to discriminate well when used to classify the training objects. Their cardinality should be sufficiently small to reduce both the storage and evaluation time spent by an NN classifier.

These techniques have been widely used in the literature. As far as we know, the most recent and complete studies performed for PS and PG can be found in [10] and [16], respectively. In [10] a memetic evolutionary approach, called SSMA [9], has been highlighted as the most promising PS technique. Furthermore, this technique has been applied in conjunction with the stratification procedure in [6]. Regarding PG methods, ENPC [7], RSP3 [15] and PSO [13] were stressed as the best performing methods in [16]. Recently, two different PG approaches have been proposed: IPADECS [17], [19] and SSMASFLSDE [18]. The former proposes an iterative optimization procedure which determine the most appropriate number of prototypes per class and their positioning. The latter is presented as a combination of a PS stage and a positioning adjustment of the prototypes.

In this work, we will focus in all these techniques to check their scaling up capabilities in conjunction with stratification.

B. The Scaling up problem

The scaling up problem appears when the number of training samples increases beyond the capacity of the traditional data mining algorithms, harming their effectiveness and efficiency. Due to large size data sets, it produces excessive storage requirements, increases time complexity and affects generalization accuracy.

This problem is also present when applying PS or PG methods to large data sets. The NN rule presents several shortcomings [10], [16], with the most well known being: the necessity of storing all the training instances in order to carry out the classification task, and its high time consumption in classification due to it having to search through all available instances to classify a new input vector.

These drawbacks are even more problematic when applying prototype reduction methods. The main issues are:

- **Time complexity**: The efficiency of the majority of the prototype reduction algorithms is at least, $O(N^2)$, with N being the number of instances in the data set; moreover, it is not difficult to find classical approaches with an efficiency order of $O(N^3)$, or more. The implication of this fact is that, although the methods could be very suitable when applied to small data sets, their application would become impossible as the size of the problem increases.
- **Memory consumption**: The majority of the PS and PG methods need to have the complete data set stored in their memory in order to be carried out. If the size of the data set is too big, the computer will be unable to execute the method or, at least, would need to use the hard disk to swap memory, which would have an adverse effect on efficiency due to the increased amount of access to the hard disk.
- **Generalization**: Prototype reduction techniques are affected in their generalization capabilities due to the noise and over fitting effect introduced by larger size data sets. Most of the classical PS and PG algorithms have been developed to handle small and medium sized data sets, and may not be suitable to find and delete every noisy or irrelevant instance in a greater data set.

These drawbacks produce a considerable degradation in the behavior of PS and PG algorithms. However, an important consideration must be taken into account: Assuming that a prototype reduction method is able to handle a problem, the time elapsed during its execution is not as important as the quality of the reduced set RS and the reduction rate achieved. In a similar way to which a decision tree or a neural net are built, the PG and PS processes need only be performed once; by contrast, the classification process with the RS must be performed every time a new example is required to be classified. Thus, a good prototype reduction technique should offer a high quality reduced set of instances, simultaneously achieving the best possible reduction rate. Thereby, a highly accurate and quick classification process could be carried out by the NN classifier.

C. Stratification for prototype reduction techniques

As we have stated, the *scaling up* problem has several drawbacks which can produce a considerable degradation in the behavior of prototype reduction algorithms. To avoid this, we use the stratification strategy proposed in [3].

This stratification strategy splits the training data into disjoint strata with equal class distribution. The initial data set D is divided into two sets, TR and TS, as usual (e.g. a tenth of the data for TS, and the rest for TR in a 10-fold cross validation). Then, TR is divided into t disjoint sets TR_j , strata of equal size, TR_1, $TR_2 \cdots TR_t$, maintaining class distribution within each subset. In this manner, the subsets TR and TS can be represented as follows:

$$TR = \bigcup_{j=1}^{t} TR_j \qquad (1)$$

$$TS = D \backslash TR \qquad (2)$$

Then, a prototype reduction method should be applied to each TR_j, obtaining a reduced set RS_j for each partition. The final reduced set is obtained joining every RS_j obtained, and it is denoted as Stratified Reduced Set (SRS).

$$SRS = \bigcup_{j=1}^{t} RS_j \qquad (3)$$

When the SRS has been obtained, the NN classifier can be applied to TS, employing SRS as training data. Figure 1 shows the basic steps of the process.

The used of the stratification procedure does not have a great cost in time. Usually, the process of splitting the training data into strata, and joining them when the prototype reduction method has been applied, is not time-consuming, as it does not require any kind of additional processing. Thus, the time needed for the stratified execution is almost the same as that taken in the execution of the prototype reduction method in each strata, which is significantly lower than the time spent if no stratification is applied, due to the time complexity of the PS and PG methods, which most of the times is $O(N^2)$ or higher.

The prototypes present in TR are independent of each other, so the distribution of the data into strata will not degrade their representation capabilities if the class distribution is maintained. The number of strata, which should be fixed empirically, will determine the size of them. By using a proper number it is possible to greatly reduce the training set size. This situation allows us to avoid the drawbacks that appeared due to the *scaling up* problem.

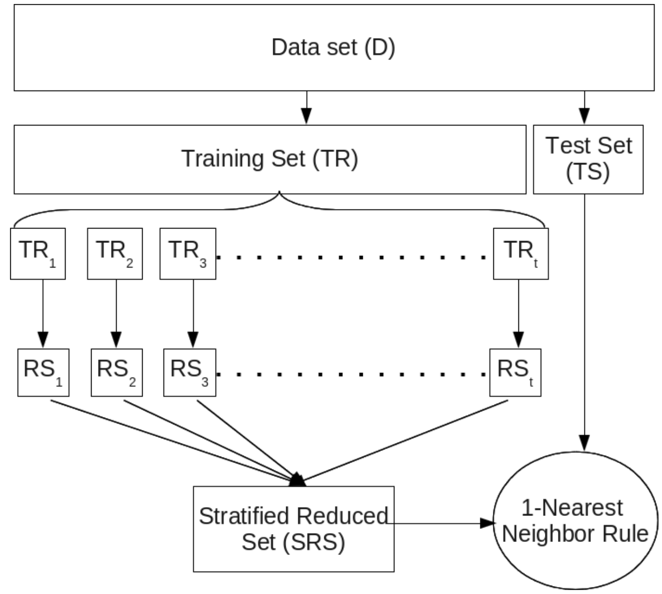

Figure 1. Scheme of the stratification strategy

III. Experimental Framework and Results

In this section we present the main characteristics related to the experimental study and the analysis of results. Section III-A describes the used techniques, provides the details of the problems chosen and the parameters of the algorithms. Finally, Section III-B shows a comparative study of the selected prototype reduction techniques.

A. Experimental Framework

We have selected some of the main proposed models in the literature of PS and PG. In addition, the NN rule with $k = 1$ (1NN) has been included as a baseline. These methods are described as follows:

- **Steady-state Memetic Algorithm (SSMA)**: This memetic algorithm makes use of a local search specifically developed for the PS problem. This interweaving of the global and local search phases allows the two to influence each other [9].
- **Evolutionary Nearest Prototype Classifier (ENPC)**: This follows a genetic scheme with 5 operators, which focus their attention on defining regions in the search space [7].
- **Particle Swarm Optimization (PSO)**: This adjusts the position of an initial set with the PSO rules, attempting to minimize the classification error [13]. We will use it as an optimizer in the proposed hybrid models.
- **Reduction by Space Partitioning-3 (RSP3)**: This technique tries to avoid drastic changes in the form of decision boundaries associated with TR by splitting it in different subsets according to the highest overlapping degree [15].

- **Iterative Prototype Adjustment based on differential evolution (IPADECS)**: This algorithm follows an iterative prototype adjustment scheme with an incremental approach. At each step, an differential evolution algorithm is used to adjust the position of the prototypes, adding new ones if needed. The aim of this algorithm is to determine the most appropriate number of prototypes per class and adjust their positioning [19].
- **Hybridization of SSMA and differential evolution (SSMASFLSDE)**: This approach uses the SSMA algorithm to initialize a subset of prototypes, finding a promising selection of prototypes per class. Then, the resulting RS is inserted as one of the individuals of the population in a differential evolution algorithm, acting as a PG method, to adjust the positioning of the prototypes and increase the accuracy rate [18].

To compare the performance of the algorithms, we use four measures: *Accuracy* rate [2], the *Reduction rate* measured as

$$ReductionRate = 1 - size(RS)/size(TR) \tag{4}$$

Acc·Red measured as *Accuracy·Reduction rate* and *Runtime*.

Table I
SUMMARY DESCRIPTION FOR CLASSIFICATION DATA SETS

Data Set	#Ex.	#Atts.	#Cl.
Adult	48842	14	2
Census	299285	41	2
Connect-4	67557	42	3
Fars	100968	29	8
Magic	19020	10	2
Nursery	12690	8	5
Penbased	10992	16	10
Shuttle	58000	9	7

We have used 8 data sets from the KEEL-dataset repository[1] [1]. Table I summarizes the properties of the selected data sets. It shows, for each data set, the number of examples (#Ex.), the number of attributes (#Atts.), and the number of classes (#Cl.). These data sets are partitioned using the ten fold cross-validation procedure (10-fcv) and their values are normalized in the interval [0,1] to equalize the influence of attributes with different range domains when using the NN rule. In addition, instances with missing values have been discarded over the data sets.

It is also important at this point to select the specific stratification scheme to employ. As we related before, the use of the stratification procedure requires making the decision of how many strata to use to split the data. In this experimental study, we have fixed empirically a number of strata for each data set, trying to set the size of each strata as near as possible to 5000 instances, which corresponds to a medium sized problem which PS and PG methods should be able to handle adequately.

[1]http://sci2s.ugr.es/keel/datasets

Table II
PARAMETER SPECIFICATION FOR ALL THE METHODS EMPLOYED IN THE EXPERIMENTATION

Algorithm	*Parameters*
SSMA	Population=30, Evaluations=10 000, Crossover Probability=0.5, Mutation Probability=0.001
ENPC	Iterations = 500
PSO	SwarmSize = 40, Iterations = 500, ParticleSize (r) = 2%, C1 = 1, C2 = 3, Vmax = 0.25, Wstart = 1.5, Wend = 0.5
RSP3	Subset Choice = Diameter
IPADECS	PopulationSize = 10, iterations of Basic DE = 500, iterSFGSS = 8, iterSFHC = 20, Fl=0.1, Fu=0.9
SSMASFLSDE	PopulationSFLSDE= 40, IterationsSFLSDE = 500, iterSFGSS =8, iterSFHC=20, Fl=0.1, Fu=0.9

The parameters of the used algorithms, which are common to all problems, are presented in Table III-A. We focus this experimentation on the recommended parameters proposed by their respective authors, assuming that the choice of the values of the parameters was optimally made. In all of the techniques, euclidean distance is used as a similarity function and those which are stochastic methods have been run three times per partition.

All the methods have been run on an Intel(R) Core(TM) i7 CPU 920 (2.67GHz) with 12GB of RAM. Implementations of the original algorithms can be found in the KEEL software tool [1]. They have been adapted from the existing KEEL project implementations, performing the necessary changes to combine them with the stratification procedure.

B. Analysis of results

In this subsection, we analyze the results obtained and discuss several issues about the use of the stratification procedure in association with PG techniques. Table III shows the accuracy test results for each method considered. For each data set, the mean accuracy and the standard deviation ($\pm$) are computed for the 10-fcv used. The best result for each column is highlighted in bold. The last row presents the average results considering all the data sets.

Considering only average results could lead us to erroneous conclusions. Due to this fact, we will accomplish statistical comparisons over multiple data sets based on nonparametric statistical tests [8]. Specifically, we use the Wilcoxon Signed-Ranks test. Table VI collects the results of applying Wilcoxon test comparing SSMASFLSDE, which obtains the best average result, with the rest of comparison techniques. This table shows the rankings R^+ and R^- values achieved and its associate p-value. Those p-values highlighted in bold point out methods outperformed by the SSMASFLSDE, at a $\alpha = 0.1$ level of significance.

Table IV shows the average reduction rates obtained (excluding 1NN). The best results are highlighted in bold.

Table III
ACCURACY TEST RESULTS

Datasets	1NN	SSMA	ENPC	PSO	RSP3	IPADECS	SSMASFLSDE
Adult	0.7960±0.0035	0.8268±0.0044	0.7394±0.0060	0.8135±0.0063	0.7846±0.0070	0.8263±0.0032	**0.8273**±0.0098
Census	0.9253±0.0010	0.9420±0.0011	0.8700±0.0029	0.9454±0.0012	0.9136±0.0024	0.9439±0.0005	**0.9460**±0.0009
Connect-4	0.6720±0.0036	0.6666±0.0142	0.6504±0.0049	0.6542±0.0031	0.6565±0.0041	0.6569±0.0009	**0.6794**±0.0061
Fars	0.7466±0.0034	0.7597±0.0048	0.7068±0.0041	0.7275±0.0064	0.7428±0.0028	0.7439±0.0218	**0.7625**±0.0036
Magic	0.8059±0.0090	0.8209±0.0005	0.7653±0.0077	**0.8300**±0.0093	0.8100±0.0057	0.8204±0.0088	0.8266±0.0040
Nursery	0.8267±0.0092	0.8083±0.0001	**0.8839**±0.0118	0.6335±0.0102	0.8279±0.0127	0.8301±0.0393	0.8289±0.0091
Penbased	0.9935±0.0023	0.9814±0.0002	0.9904±0.0031	0.9562±0.0085	0.9896±0.0037	0.9916±0.0091	**0.9940**±0.0036
Shuttle	**0.9993**±0.0004	0.9967±0.0083	0.9983±0.0006	0.9964±0.0005	0.9985±0.0006	0.9941±0.0021	0.9967±0.0021
Average	0.8457±0.0040	0.8503±0.0042	0.8256±0.0051	0.8196±0.0057	0.8404±0.0049	0.8509±0.0107	**0.8577**±0.0049

The reduction rate achieved has a strong influence on the efficiency of the training and test classifications phases, since it is directly related to the final size of the training set used with the 1NN rule ($O(N^2)$).

Figure 2 depicts the average $Acc{\cdot}Red$ obtained for each algorithm, showing the trade-off between accuracy and reduction rates.

Finally, Table V shows the average runtime (in seconds) taken by the prototype reduction methods to generate or select an appropriate SRS.

Observing these tables and Figure 2 we can state:

Table IV
REDUCTION RATE

Datasets	SSMA	ENPC	PSO	RSP3	IPADECS	SSMA SFLSDE
Adult	0.9882	0.7035	0.9801	0.7729	**0.9986**	0.9882
Census	0.9973	0.8926	0.9801	0.9203	**0.9994**	0.9973
Connect-4	0.9822	0.6414	0.9801	0.6047	**0.9990**	0.9822
Fars	0.9808	0.6494	0.9796	0.7762	**0.9968**	0.9808
Magic	0.9832	0.7578	0.9800	0.7553	**0.9991**	0.9832
Nursery	0.9452	0.7925	0.9800	0.5008	**0.9983**	0.9452
Penbased	0.9808	0.9679	0.9801	0.9068	**0.9973**	0.9808
Shuttle	0.9973	0.9881	0.9793	0.9870	**0.9974**	0.9973
Average	0.9819	0.7992	0.9799	0.7780	**0.9982**	0.9819

- In Table III, we can observe that SSMASFLSDE obtains the best average result and the best accuracy test in five of the eight data sets. As we can see in Table VI the Wilcoxon test supports this statement with a p-value lower than 0.1. Hence, the combination of PG techniques with the stratification allows it to tackle the problems considered and generate an appropriate SRS which increases the predictive power of the 1NN rule. Note that with this technique, IPADECS and SSMA algorithms are able to improve the baseline 1NN.
- IPADECS achieves the highest reduction rate in all the cases. Although, SSMASFLSDE significantly outperforms to the IPADECS algorithm in terms of accuracy test, it is important to note that IPADECS is the best performing method considering the trade-off between accuracy and reduction rates as we can observe in Figure 2.

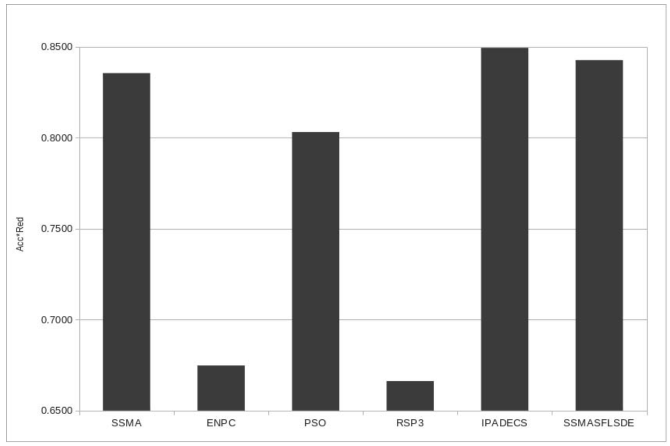

Figure 2. Average Acc*Red obtained

Table V
TIME ELAPSED (SECONDS)

Datasets	SSMA	ENPC	PSO	RSP3	IPADECS	SSMA SFLSDE
Adult	5714.00	7210.91	4581.29	429.78	961.92	7010.27
Census	20719.89	8300.12	29992.34	3083.68	1418.39	24745.75
Connect-4	9290.71	34511.09	14484.37	2226.89	1728.96	27436.94
Fars	15418.65	42565.59	16611.55	2406.72	11888.42	31912.23
Shuttle	4518.85	77.94	429.78	385.88	2270.88	5087.21
Magic	2649.11	2330.54	2492.85	187.63	236.03	7353.58
Nursery	3395.41	850.70	941.04	134.07	647.59	8082.74
Penbased	1612.68	84.44	1438.56	236.78	1854.48	6763.28
Average	7914.91	11991.42	8871.47	1136.43	2625.83	14799.00

- The lowest average runtime corresponds to the non-evolutionary technique RSP3. Observing the rest of evolutionary techniques, we can stress to IPADECS as the fastest evolutionary algorithm to generate an appropriate SRS. This behavior arises because IPADECS uses a smaller population size than the rest of evolutionary techniques.
- The stratification process allows to the PS and PG techniques to handle with large data sets in a reasonable time. Note that the runtime used in the selection or generation phase of SRS is only spent once. Therefore, the time consumption may not be an important drawback

Table VI
RESULTS OF THE WILCOXON TEST

Methods	R^+	R^-	p-value
SSMASFLSDE vs 1NN	33.0	3.0	**0.0299**
SSMASFLSDE vs SSMA	28.0	0.0	**0.0142**
SSMASFLSDE vs ENPC	31.0	5.0	**0.0587**
SSMASFLSDE vs PSO	33.0	3.0	**0.0299**
SSMASFLSDE vs RSP3	34.0	2.0	**0.0208**
SSMASFLSDE vs IPADECS	34.0	2.0	**0.0208**

if the reduction rate achieved is high enough to reduce the evaluation time spent by the NN classifier.

IV. CONCLUSIONS

In this contribution we have studied the combination of a stratification strategy with PG methods to tackle large data sets. Some of the most important prototype reduction techniques have been compared in order to determine the most appropriate combination to deal with large domains. An hybridization between PS and PG methods, SSMASFLSDE, has offered the best behavior over these problems in the study. However, the IPADECS approach has obtained the best trade-off between accuracy and reduction rates in most of the domains. Both algorithms allow to generate an adequate reduced training set, minimizing the time-consumption of the 1NN classifier and improving its performance. The results obtained have been analyzed from a statistical point of view, which has reinforced that SSMASFLSDE and IPADECS outperform previously proposed stratified prototype reduction techniques.

ACKNOWLEDGMENT

Supported by the Research Projects TIN2011-28488 and TIC-6858. I Triguero holds a scholarship from the University of Granada. J. Derrac holds an FPU scholarship from the Spanish Ministry of Education and Science.

REFERENCES

[1] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez and F. Herrera, *KEEL Data-Mining Software Tool: Data Set Repository, Integration of Algorithms and Experimental Analysis Framework*. Journal of Multiple-Valued Logic and Soft Computing, 17(2-3):255–287 (2011).

[2] E. Alpaydin, *Introduction to Machine Learning*. MIT Press, Cambridge, MA, 2nd edition (2010).

[3] J.R. Cano, F. Herrera, M. Lozano, *Stratification for scaling up evolutionary prototype selection*. Pattern Recognition Letters 26(7):953–963 (2005).

[4] C. Chang, *Finding Prototypes For Nearest Neighbor Classifiers*. IEEE Transactions on Computers 23(11):1179–1184 (1974).

[5] T. M. Cover, P. E. Hart, *Nearest neighbor pattern classification*. IEEE Transactions on Information Theory 13:21–27 (1967).

[6] J. Derrac, S. García, F. Herrera, *Stratified prototype selection based on a steady-state memetic algorithm: a study of scalability*. Memetic Computing 2(3):183–199 (2010)

[7] F. Fernández, P. Isasi, *Evolutionary Design of Nearest Prototype Classifiers* Journal of Heuristics 10(4):431–454 (2004)

[8] S. García, F. Herrera, *An Extension on "Statistical Comparisons of Classifiers over Multiple Data Sets" for all Pairwise Comparisons*. Journal of Machine Learning Research 9:2677–2694 (2008).

[9] S. García, J.R. Cano, F. Herrera, *A memetic algorithm for evolutionary prototype selection: A scaling up approach*. Pattern Recognition 41(8):2693–2709 (2008).

[10] S. García, J. Derrac, J.R. Cano, F. Herrera, *Prototype Selection for Nearest Neighbor Classification: Taxonomy and Empirical Study*. IEEE Transactions on Pattern Analysis and Machine Intelligence, in press, doi: 10.1109/TPAMI.2011.142 (2011).

[11] P. E. Hart, *The condensed nearest neighbor rule*. IEEE Transactions on Information Theory 18:515–516 (1968)

[12] W. Lam, C. Keung, D- Liu, *Discovering Useful Concept Prototypes for Classification Based on Filtering and Abstraction*. IEEE Transactions on Pattern Analysis and Machine Intelligence 24(8):1075–1090 (2002).

[13] L. Nanni, A. Lumini, *Particle swarm optimization for prototype reduction*. Neurocomputing 72(4-6):1092–1087 (2009).

[14] F. Provost, K. Venkateswarlu, *A Survey of Methods for Scaling Up Inductive Algorithms*. Data Min. Knowl. Discov. 3(2):131–169 (1999).

[15] J.S. Sánchez, *High training set size reduction by space partitioning and prototype abstraction* Pattern Recognition 37(7):1561–1564 (2004).

[16] I. Triguero, J. Derrac, S. García, F. Herrera, *A Taxonomy and Experimental Study on Prototype Generation for Nearest Neighbor Classification*. IEEE Transactions on Systems, Man, and Cybernetics–Part C: Applications and Reviews, in press, doi: 10.1109/TSMCC.2010.2103939 (2011).

[17] I. Triguero, S. García, F. Herrera, *IPADE: Iterative Prototype Adjustment for Nearest Neighbor Classification*. IEEE Transactions on Neural Networks 21(12):1984–1990 (2010).

[18] I. Triguero, S. García, F. Herrera, *Differential Evolution for Optimizing the Positioning of Prototypes in Nearest Neighbor Classification*. Pattern Recognition 44(4):901–916 (2011).

[19] I. Triguero, S. García, F. Herrera, *Enhancing IPADE Algorithm with a Different Individual Codification*. Proceeding of the 6th International Conference on Hybrid Artificial Intelligence Systems (HAIS). LNAI 6679, pp. 262–270 (2011).

[20] D. L. Wilson, *Asymptotic properties of nearest neighbor rules using edited data*. IEEE Transactions on System, Man and Cybernetics 2(3):408–421 (1972)