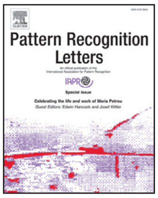
Pattern Recognition Letters 62 (2015) 61–67

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec



Parallel selective sampling method for imbalanced and large data classification[☆]

Annarita D'Addabbo, Rosalia Maglietta[*]

Institute of Intelligent Systems for Automation - National Research Council, via Amendola 122/D-O, Bari 70126, Italy

ARTICLE INFO

Article history:
Received 10 December 2014
Available online 5 June 2015

Keywords:
Imbalanced learning
Classification
Support vector machine
Selective sampling methods

ABSTRACT

Several applications aim to identify rare events from very large data sets. Classification algorithms may present great limitations on large data sets and show a performance degradation due to class imbalance. Many solutions have been presented in literature to deal with the problem of huge amount of data or imbalancing separately. In this paper we assessed the performances of a novel method, Parallel Selective Sampling (PSS), able to select data from the majority class to reduce imbalance in large data sets. PSS was combined with the Support Vector Machine (SVM) classification. PSS-SVM showed excellent performances on synthetic data sets, much better than SVM. Moreover, we showed that on real data sets PSS-SVM classifiers had performances slightly better than those of SVM and RUSBoost classifiers with reduced processing times. In fact, the proposed strategy was conceived and designed for parallel and distributed computing. In conclusion, PSS-SVM is a valuable alternative to SVM and RUSBoost for the problem of classification by huge and imbalanced data, due to its accurate statistical predictions and low computational complexity.

1. Introduction

Many real-word applications of machine learning classifiers have to identify rare events from very large data sets. For example, in the studies on the automated segmentation from magnetic resonance images [19–21], the number of training examples is very huge (up to millions), the classes are strongly imbalanced, and generating accurate statistical solution is not trivial. In addition, data imbalance in huge data sets is also reported in other applicative domains, such as marketing data [22], oil spill detection or land cover changes from remote sensing images [16,27], text classification [18] and scene classification [35]. In these areas, very large data sets have to be handled and the minority class is the one of interest, consequently two problematic issues add on: the computational complexity dependent on the size of the data set and the need to pursue a fairly high rate of correct detections in the minority class.

Many classification algorithms present great limitations on large data sets and show a performance degradation due to class imbalance [14]. For example, Support Vector Machines (SVM) [33], that are employed in many applicative domains [3,4,13,24], really become intractable and computationally too expensive when huge data sets are handled [32]. In fact, the training complexity of SVM is highly dependent on the size of the data set. Moreover, SVM classification performance can be hindered by class imbalance [1,30]. Compared with other standard classifiers, it is more accurate on moderately imbalanced data. The reason is that only SVs are used for classification and many majority samples far from the decision boundary can be removed without affecting the classification. However, an SVM classifier can be sensitive to high class imbalance, resulting in a drop of the classification performance on the minority class. In fact, it is prone to generate classifier that has a strong estimation bias toward the majority class: since the number of majority class patterns exceeds that of the minority class, the class boundary becomes vulnerable to be distorted [15]. Nevertheless, these limitations are common to many other classification schemes such as Multi-Layer Perceptron (MLP) [7] and Logistic Regression (LR) [23].

To overcome these problems, a selection of examples has to be performed sampling a small number of patterns from the majority class to reduce both the number of data and the imbalance. Such a procedure is well known in literature as "undersampling" method [12]. It generally improves the classification performance and reduces the computational complexity, however it presents a potential disadvantage of distorting the distribution of the majority class. If the sampled patterns from the majority class do not represent the original distribution, it may degrade the classification performance. This potential drawback comes dramatically true when the number of minority class patterns is very small [15]. However, other techniques are

[☆] This paper has been recommended for acceptance by Y. Liu.

[*] Corresponding author. Tel.: +39 80 5929454; fax: +39 80 5929460.
E-mail address: maglietta@ba.issia.cnr.it (R. Maglietta).

http://dx.doi.org/10.1016/j.patrec.2015.05.008

not feasible with very large data set because they work: (1) by modifying cost for misclassified patterns belonging to the minority class, without changing the number of original data [7], (2) by increasing the total number of examples by copying patterns from the minority class to balance the ratio of classes (”oversampling” method) [9], (3) by combining oversampling and undersampling techniques [34].

Several methods to select examples in a classification problem are presented in literature, using two different approaches: the example-selection method can be embedded within the learning algorithm or the examples can be filtered before passing them to the classification scheme [2,26]. It is worth noting that the first type of selection methods generally work by preserving the original ratio between classes [6,11]: if there is a great skew in the data, it continues to be. To overcome this problem, filtered methods are more suited for preprocessing data before the classification step. Numerous algorithms can be used taking into account the class-membership of samples to solve the imbalance in the data [2]. In this framework, a very interesting method has been developed by Evgeniou and Pontil in [10]. They present a preprocessing algorithm that computes clusters of points in each class, based on Euclidean distance, and substitute each cluster with the mass center of the points in the cluster. The algorithm tends to produce large (small) clusters of data points which are far (near) from the boundary between the two classes. These strategies did not focus on both large and imbalanced data learning. More recently, a method focused on both big and class imbalanced data classification was proposed [29]. It is a cost-sensitive support vector machine using randomized dual coordinate descent method (CSVM-RDCD) and it belongs to the class of embedded methods, i.e. the examples selection is integrated in the learning algorithm and classifier dependent. This method was tested on three data sets with relative class imbalance and three data sets with severe class imbalance, of which only one of them with a large number of examples. In all the experiments the recognition rates of the minority class, computed by CSVM-RDCD and SVM, are roughly comparable, with an improvement of about 1%. New studies are required in order to examine in more depth the case study of imbalanced and big data.

A valuable alternative is given by filter methods which are attractive because they adjust only the distribution of the original training set, independently of the given classifier. In this paper, we describe a novel approach, named Parallel Selective Sampling (PSS), that selects data from the majority class to reduce imbalance in big data sets. PSS is a filter method which can be combined with a variety of classification strategies. It is based on the idea (usually used in SVM) that only training data, near the separating boundary (for classification), are relevant. In this way the core information from the training data - i.e. the training data points near the separating boundary - is preserved while the size of the training set is effectively decreased. Relevant examples from the majority class are selected and used in the successive classification step using SVM. Due to the complex computational requirements, PSS is conceived and designed for parallel and distributed computing. Finally, PSS–SVM allows accurate statistical predictions keeping down the computational times.

The paper is organized as follows: in Section 2 we describe in details the proposed sampling method and we introduce the main properties of SVM for classification of large and imbalanced data sets. In Section 3 we discuss the experimental results obtained in the analysis of real and simulated data sets. Section 4 concludes the paper and summarizes the main results.

2. Methods

2.1. PSS

The PSS method can be used to preprocess very large training data with significant skew between classes. It is an undersampling method because it acts by reducing examples belonging only to the majority class. It is based on the computation of Tomek links [31], defined as a pair of nearest neighbors of opposite classes. Given $\{E_1, \ldots, E_n\} \in R^k$, a pair (E_i, E_j) is called a Tomek link if E_i and E_j have different labels, and there is not an E_l such that $d(E_i, E_l) < d(E_i, E_j)$ or $d(E_j, E_l) < d(E_i, E_j)$, where $d(\cdot, \cdot)$ is the Euclidean distance. Here Tomek links are used to remove samples of majority class staying in areas of input space dense of data belonging to the same class.

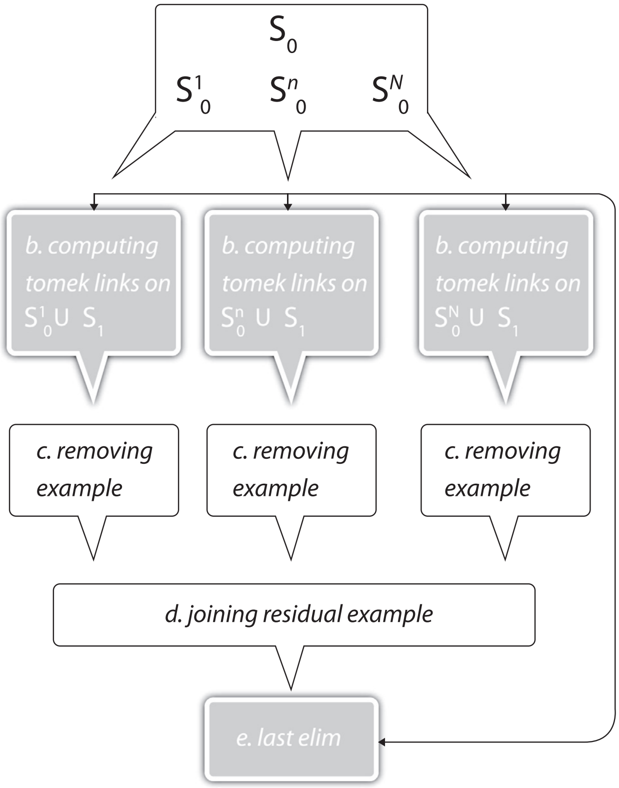

S_0
S^1_0
S^n_0
S^N_0
b. computing tomek links on S^1_0 U S_1
b. computing tomek links on S^n_0 U S_1
b. computing tomek links on S^N_0 U S_1
c. removing example
c. removing example
c. removing example
d. joining residual example
e. last elim

Fig. 1. Block diagram of PSS.

Let $S = \{(x_1, y_1), \ldots, (x_\ell, y_\ell)\}$ be the training set, where $x_i \in R^k$ and $y_i \in \{0, 1\}, \forall i = 1, \ldots, \ell$. We define S_0 the set of ℓ_0 training data belonging to class $y = 0$ and S_1 the set of ℓ_1 training data belonging to class $y = 1$, with $\ell_0 \gg \ell_1$. PSS achieves a reduced training set whose percentage M% of the minority class on the total number of examples is chosen by the user.

a: **data partitioning.** The S_0 set is divided into N subset S_0^n with $n = 1, 2, \ldots,$N, with N set by the user. In this way, N different undersampling procedures are performed in parallel computation (see Fig. 1).
For each S_0^n, with $n = 1, 2, \ldots,$N, the following steps are performed:

b: **computing Tomek links.** Let us define the set T^n of all examples in the majority class S_0^n that are first neighbors of one sample in S_1, that is $T^n = \{x \in S_0^n | (x, z)$ is Tomek link on $S_1 \cup S_0^n, z \in S_1\}$.

c: **removing examples.** Let us randomly select $\bar{x} \in D^n = S_0^n \backslash T^n$; the following steps are performed (see Fig. 2):

- the Tomek link $(\bar{x}, \bar{z})$ is computed over the data set $\bar{x} \cup S_1$, with $\bar{z} \in S_1$;
- the Euclidean distances $d(\bar{x}, x)$ are computed for each $x \in S_0^n$;
- let us define the subset $L = \{x \in S_0^n | d(\bar{x}, x) < d(\bar{x}, \bar{z})\}$, (see the red circumference in Fig. 2a). The Tomek link $(x^*, \bar{z})$ in $\bar{z} \cup L$ is computed, i.e. x^* is defined as the first neighbor in L of $\bar{z}$;
- let us define the set $R = \{x \in L | d(\bar{x}, x) < [d(\bar{x}, \bar{z}) - d(x^*, \bar{z})]\}$ (see the blue circumference in Fig. 2a). Let us delete all the points in R that are not Tomek links, i.e. each $x \in R'$ with

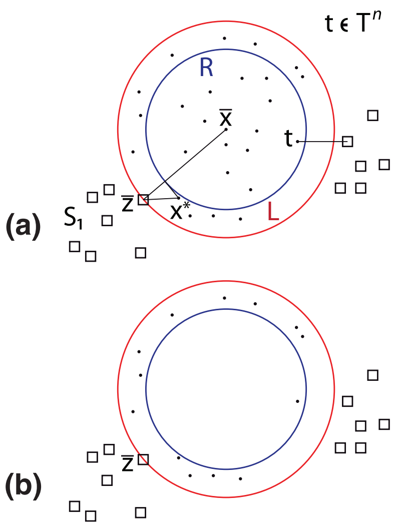


Fig. 2. Removing examples step (c) of PSS.

$R' = \{x \in R | x \notin T^n\}$. The remaining data points (shown in Fig. 2b) of the majority class are contained in $S_0^{n\prime} = S_0^n \setminus R'$;

- if the classes are balanced, the algorithm goes to the following step d; otherwise it randomly selects $\bar{x} \in D^{n\prime} = S_0^{n\prime} \setminus T^n$ and repeats the previous issues (step c).

d: **joining residual examples.** The majority class data, selected by each parallel computation, are then joined.

e: **last elimination.** The procedure previously described (step c) is repeated achieving a final reduced training set whose M% belongs to the minority class.

2.2. Support vector machines

Our methodology was combined with SVM, a powerful technique for data classification with many applications in literature. For details on the SVM algorithm we refer to [33], here we discuss its limitations on large and imbalanced training sets. In fact, despite its good theoretic foundations and high classification accuracy, it is not suitable for classification of huge data, because the training complexity of SVM is highly dependent on the size of data set. To overcome this bottleneck, different methods have been proposed in literature. A first approach consists of modifying SVM algorithm in order to make faster the training on large data sets; for example, Sequential Minimal Optimization (SMO) breaks the large QP problem into a series of smallest possible QP problems [25], allowing SMO to handle large training sets [25]. Nowadays SMO can be considered a standard procedure in the analysis of large data sets by using SVM. Another approach consists of matching selective sampling techniques with SVM. In [8] the authors proposed a novel classification approach for large data sets using Minimum Enclosing Ball (MEB) clustering: after partitioning the training data via MEB method, the centers of the clusters were used for the first time SVM classification; the algorithm used only the clusters whose centers are support vectors or those clusters which have different classes to perform the second time SVM classification. In this way many data points were recursively removed. However, the above mentioned methods are not helpful for classification of large data sets with imbalanced classes.

The effects of imbalanced data on SVM are related to the soft-margin maximization paradigm [1]: since SVM tries to minimize total error, it is biased toward the majority concept. In the linear case, a two-class space could be separated by a learned boundary that is very different from the ideal one. Moreover, if there was a lack of data representing the minority concept, there could be an imbalance of representative support vectors that could also degrade performance. These same characteristics are readily evident in non-linear separable spaces. In the worst case, SVM will classify all examples as pertaining to the majority class, a tactic that, if the imbalance is severe, can provide the minimal error rate across the data space [12]. There have been many works in literature that apply different techniques to the SVM framework in order to overcome problems due to imbalance. Most of them assign different error costs to different classes in order to shift the decision boundary and to guarantee that it is better defined [1,12]. Another major category of kernel-based learning research efforts focuses more concretely on the mechanics of the SVM itself; this group of methods is often called kernel modification methods [12]. However these methods could be useless for large data set, because they use all the data for training the classifier. The undersampling techniques are useful for large training set. In [30] the Granular Support Vector Machines - Repetitive Undersampling algorithm (GSVM-RU) was proposed to integrate SVM learning with undersampling methods. This method uses the SVM itself as a mechanism for undersampling in order to sequentially develop multiple subsets with different informative samples, which are later combined to develop a final SVM for classification. Also this method is not tailored for very huge data sets, because the SVM problem should be hard to solve due to the training set size. Instead, methods based on a smart undersampling of the majority class should be preferable for very large data sets analysis.

Table 1
Summary of training data sets,

Data set	Training set size	Percentage of minority class	♯ of attributes
S1	10^6	4.9%	2
S2	10^6	2.8%	2
S3	10^6	1.2%	2
D1	25,667	7.1%	54
D2	190,698	1%	54
D3	387,341	0.5%	54

3. Experimental results

3.1. Data set description and experimental design

In this study we used three synthetic and three real data sets (see Table 1). Each synthetic data set counts 10^6 training examples and each datum is composed of 2 attributes. The data distribution in the input space follows a checkerboard pattern, as shown in Fig. 3. The test sets are independently built with the same procedure and are made of 500, 000 examples. The percentage of the minority class of the training and test sets are reported in Table 1.

The three real data sets have been extracted from the Forest Cover Type data set of the UCI repository [5] having 7 classes and 581,012 samples. This is for the prediction of forest cover type based on 54 cartographic variables. Since our system works for binary classes, we extracted data for two classes from this data set as follows and divided the data in training set and test set.

- D1: Ponderosa Pine vs Cottonwood/Willow (training set 23,836 vs 1831 samples; test set 11,918 vs 916 samples);
- D2: Spruce-Fir vs Cottonwood/Willow (training set 188,867 vs 1831 samples; test set 94,434 vs 916 samples);
- D3: Cottonwood/Willow vs all (training set 1381 vs 385,510 samples; test set 192,755 vs 916 samples).

3.2. Evaluation of experimental performance

The accuracy as an objective function is inadequate for classification tasks with hard data imbalancing. For example, let us consider a

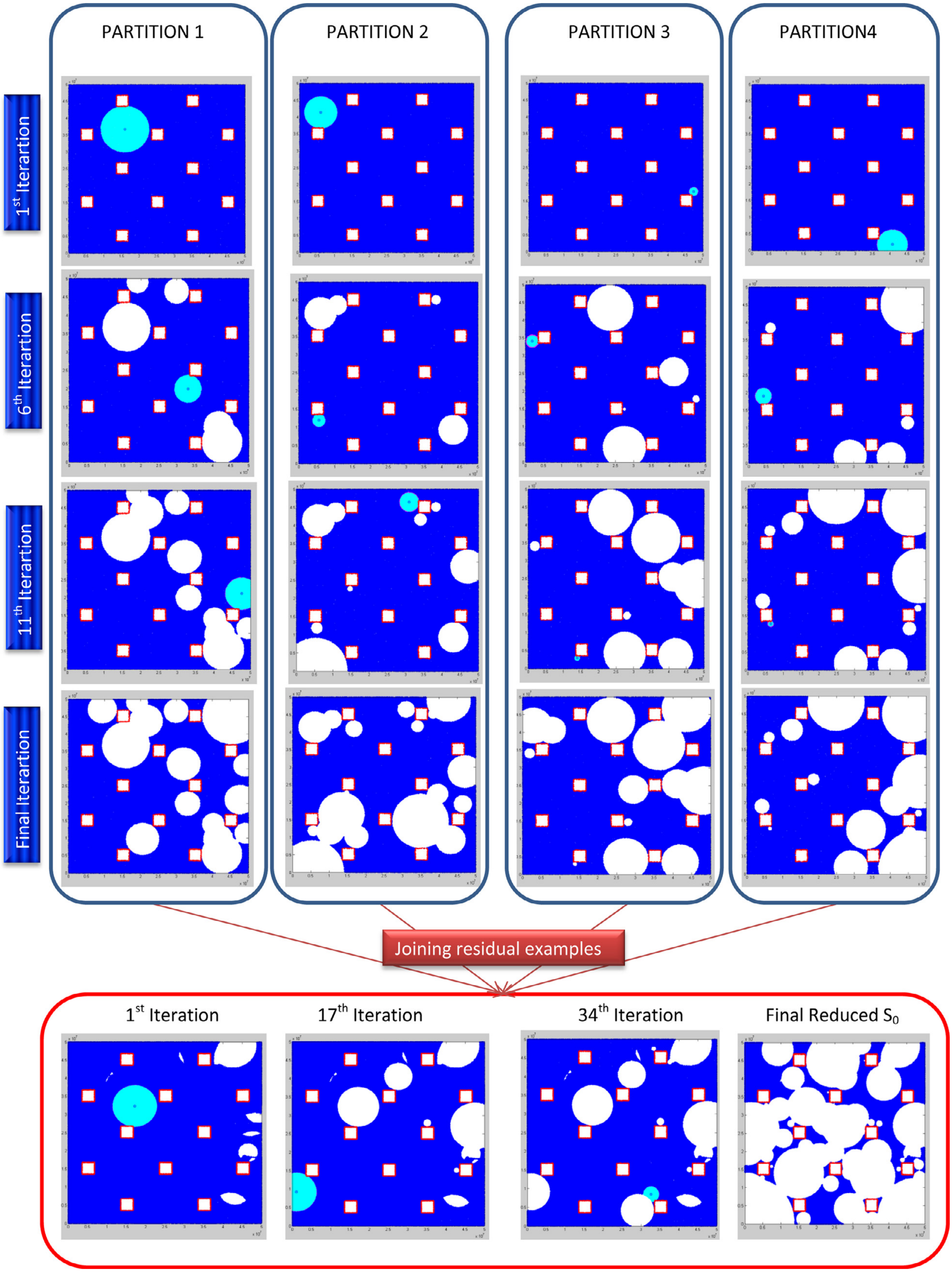


Fig. 3. Undersampling of the synthetic data set S1 by PSS. After the partition of S1 in four subsets, the 1st, 6th, 11th and final iterations are shown. The residual examples of the four subsets are joined in the first picture (1st iteration) of last row. Then, the 17th and 34th iterations of PSS to the joined residual examples and the final reduced majority class (S_0) are shown.

Table 2
Summary of the experimental results on the synthetic data sets: a) evaluation metrics computed on test set and computational time of PSS-SVM required for both preprocessing and training, b) mean values (standard deviation) of evaluation metrics computed on test set and computational time on 10 iterations of random undersampling SVM. Optimal parameters (SVM kernel, regularization parameter C and desired percentage M) are shown.

S1	PSS-SVM $M = 25\%$, RBF $\sigma = 0.7$, $C = 610$	Random undersampling SVM $M = 25\%$, RBF $\sigma = 0.7$, $C = 610$
Time (s)	2123	12740
Dice	0.87	0.79 (0.0031)
Precision	0.78	0.66 (0.0044)
Recall	0.97	1.00 (0.0002)
Relative Overlap	0.77	0.66 (0.0042)
S2	PSS–SVM $M = 25\%$, RBF $\sigma = 0.5$, $C = 620$	Random undersampling SVM $M = 25\%$, RBF $\sigma = 0.5$, $C = 620$
Time (s)	757	3635
Dice	0.86	0.72 (0.0171)
Precision	0.78	0.56 (0.0211)
Recall	0.97	1.00 (0.0002)
Relative Overlap	0.76	0.56 (0.0210)
S3	PSS-SVM $M = 10\%$, RBF $\sigma = 0.5$, $C = 650$	Random undersampling SVM $M = 10\%$, RBF $\sigma = 0.5$, $C = 650$
Time (s)	536	4020
Dice	0.80	0.56 (0.0071)
Precision	0.67	0.39 (0.0070)
Recall	0.99	1.00 (0.0000)
Relative Overlap	0.67	0.39(0.0070)

classification problem in which there are two classes, 1% of the patterns belonging to the minority class and 99% of the patterns belonging to the majority class. If a naive approach of classifying made a decision that all patterns should be classified into the majority class, it would achieve 99% of accuracy. This can be considered as a good performance in terms of simple accuracy, but this is of no use since the classifier does not catch any important information on the patterns of the minority class [12].

More appropriate performance measures may be derived from the confusion matrix, that compares predicted to true labels. We consider dice D, precision P, recall R and relative overlap R.O.: these metrics are effective to evaluate classification performance in imbalanced learning scenarios, and are defined as follows:

$$P = \frac{TP}{TP + FP}$$

$$R = \frac{TP}{TP + FN}$$

$$D = \frac{2 * P * R}{P + R} = \frac{2 * TP}{(TP + FP) + (TP + FN)}$$

$$R.O. = \frac{TP}{FP + TP + FN}$$

where TP is the number of True Positive, i.e. the actual positive data which are correctly classified as such, FP is the number of False Positive, i.e. negative data classified as positive, TN is the number of True Negative, i.e. the actual negative data which are correctly classified as such, and FN is the number of False Negative, i.e. positives incorrectly classified as negatives. Intuitively, precision is a measure of exactness (i.e., of the examples predicted as positive, how many are actually labelled correctly), whereas recall is a measure of completeness (i.e., how many examples of the positive class were labelled correctly) [12]. The D value is used to merge precision and recall into a single metric for convenience. The R.O. accounts for the fraction of TP on the total number of true and predicted positive examples. We considered the minority class as positive.

3.3. *Experimental results*

All of the experiments were carried out on a Workstation HP Z820 equipped with 2 CPU Intel Xeon and eight cores E5-2650, RAM 64Gb-2x1000Gb. All data were analyzed in MATLAB (MathWorks, Natick, MA). The parameters of the classifiers were tuned to obtain optimal values; models were built on the training set, and performances of the constructed classifier were tested on the test set. Also the percentage M% of the minority class obtained by PSS was considered as a parameter and was tuned in each experiment.

First of all, we illustrate experimental results on synthetic data obtained by the proposed method PSS-SVM. Fig. 3 shows a snapshot of PSS procedure on data set S1. In each partition the progressive deletion of the points of the majority class is shown. After joining residual examples from parallel partitions, the output of PSS consisted of a reduced data set where the examples of majority class, nearest to the minority class, were clearly preserved.

Table 2 shows that the method achieved good performances, in particular high values of dice and recall for all the data sets. Correctly, evaluation metrics decreased when the imbalancing increased. Note that for severe imbalancing, high values of M were not optimal choice in term of classification performances. In fact, for data set S3 which had an imbalancing of 1.2%, the best performances were obtained with M = 10%. Moreover, in order to evaluate the repeatability of PSS–SVM, we run it on S3 data set 10 times. The average values of each metric correspond to those reported in Table 2 with standard deviations lower than 0.01.

In these experiments an undersampling method was mandatory: in fact due to the very huge amount of data points (1 million of examples), SVM did not achieve convergence. Hence the performances of PSS-SVM were compared with those of random undersampling SVM, in order to highlight the benefit deriving from using an ôintelligentö method for the undersampling. Then Gaussian kernel SVMs were trained on reduced data sets, obtained by combining minority class with random undersampling of the majority one, until the desired balance was achieved. This procedure implied a loss of information due to deleting examples from the training data. To generalize we considered a number of 10 iterations where

Table 3
Summary of the experimental results on real data sets. Evaluation metrics computed on test set and computational time for both preprocessing (if required) and training of the classifiers are shown.

D1	SVM
	RBF-$\sigma = 1$
	$C = 100$
		RUSBoost
		n.trees=1000
		n.leaf=5
			PSS-SVM
			$M = 15\%$, RBF-$\sigma = 0.8$
			$C = 10$

Time (s)	47	962	46
Dice	90.4	90.7	90.9
Precision	90.0	89.9	90.8
Recall	90.7	91.5	90.9
Relative overlap	82.4	82.9	83.3

D2	SVM
	RBF-$\sigma = 1$
	$C = 100$
		RUSBoost
		n.trees=100
		n.leaf=5
			PSS-SVM
			$M = 10\%$, RBF-$\sigma = 2$
			$C = 10$

Time (s)	761	38	82
Dice	99.3	99.4	99.8
Precision	99.9	99.2	99.9
Recall	98.7	99.6	99.8
Relative overlap	98.6	98.8	99.7

D3	SVM
	RBF-$\sigma = 1$
	$C = 100$
		RUSBoost
		n.trees=1500
		n.leaf=5
			PSS-SVM
			$M = 15\%$, RBF-$\sigma = 1$
			$C = 10$

Time (s)	833	1984	153
Dice	84.3	87.8	87.7
Precision	82.9	83.4	84.5
Recall	85.8	92.9	91.0
Relative overlap	72.9	78.3	78.0

a new random undersampling and SVM training was performed. The average values of evaluation metrics and standard deviation are shown in Table 2. The M values were chosen equal to the PSS-SVM setting.

Both methods achieved high recall values which referred to the ability of the classifiers to correctly identify positive examples. In clinical application, high recall values are important where the test is used to identify a serious but tractable disease [17]. In the analysis of the data set S1 which had an imbalancing of 4.9%, PSS-SVM had an higher precision values (P = 0.78) than random undersampling SVM (P = 0.66). In the analysis of data sets S2 and S3, the performances of PSS-SVMs in terms of precision (S2, P = 0.78; S3, P = 0.67) were considerably better than random unndersampling SVM (S2, P = 0.56; S3, P = 0.39). The precision of a test is very useful to clinicians since it answers the question: "How likely is it that this patient has the disease given that the test result is positive?"[17]. Consequently, Dice and R.O. computed by PSS-SVM had higher values than those obtained by random unndersampling SVM. These metrics are simple and useful summary measures of overlapping between actual and predicted labels, which are interestingly applied to studies of reproducibility and accuracy in medical image segmentation [36]. Moreover, PSS-SVM required shorter computational time than random undersampling SVM.

Now we discuss experimental results on real data sets shown in Table 3. We compared the performances of PSS-SVM with SVM on three data sets with significant variation of both data size and imbalancing. We also considered random undersampling SVM which raised poor performance and therefore it was excluded from the discussion. For an exhaustive analysis, we compared the performances of PSS-SVM with RUSBoost (Random Undersampling with Boosting) [28]. It is a boosting-based sampling algorithm that handles class imbalance randomly removing examples from the majority class until the desired balance is achieved.

In all the experiments, we trained SVMs with different combinations of parameters and we chose a gaussian kernel with optimal sigma and C as reported in Table 3. Similarly a tuning was performed for RUSBoost parameters and optimal choices are shown in the same table.

The data set D1 contained 25667 examples whose 7% belonged to minority class. Due to the small sample size and not extreme imbalancing, the performances of PSS-SVM, SVM and RUSBoost were excellent and roughly comparable, with best dice of 90.9% computed by PSS-SVM. The cost sensitive implemented in SVM worked well controlling this amount of imbalancing. Nevertheless the good results obtained with RUSBoost, its computational time was much higher than others. This was due to the number of trees, equal to 1000, useful to obtain the optimized evaluation metrics. In fact, using a number of 100 trees, the computational time decreased to a value comparable to that of PSS-SVM, but the dice decreased to 90.0.

Data set D2 contained 1% of the training examples belonging to the minority class and training set size 190698. Again the dice, equal to 99.8, computed by PSS-SVM was slightly better then others with short computational time.

Data set D3 contained large amount of data points (training set size = 387341), and the class imbalancing was very hard (0.5%). The highest dice of ~ 88% was obtained by both PSS-SVM and RUSBoost. The processing time (153 s) required by PSS-SVM was shorter than that of RUSBoost (1984 s). In order to obtain computational time for RUSBoost of about 150 s, 100 trees should be used which arose to a lower dice of 83.9%.

In the analysis of synthetic data we considered three data sets with very large training set size and decreasing percentage of minority class in order to evaluate the performances of the proposed method in a very hard condition. In this case, cost sensitive SVM does not work, while PSS-SVM showed very good performances. In the analysis of the real data sets, a less critical work condition was encountered. Indeed, PSS-SVM performs as well as cost-sensitive SVM on real data sets D1 and D2. Instead in the analysis of D3 real data set (the most critical among the real data sets), PSS-SVM outperforms cost sensitive SVM. The effectiveness and the advantages of using PSS-SVM are more evident in large scale - class imbalanced data set analysis.

4. Conclusions

In this paper, we introduced a new algorithm, called PSS, used as a preprocessing step to train SVM on very huge and imbalanced data sets. The comparison between PSS-SVM and SVM was carried on three synthetic data sets, having a very huge amount of data. SVM did not achieve convergence, then we considered random undersampling (RUS) techniques to handle the class imbalancing and compared RUS-SVM with PSS-SVM. The proposed algorithm performances were very good and considerably better. Moreover, PSS-SVM showed excellent performances and dice's indexes comparable to the ones of SVM and RUSBoost classifiers on three real data sets. Our analysis suggested that PSS-SVM is valuable alternative to SVM and RUSBoost classifiers for very imbalanced data. Importantly, PSS exhibited the great advantage to perform in parallel computation, drastically reducing the computational time. Moreover, it is a general selective sampling method that can be combined with different classification algorithms.

Acknowledgments

We would like to acknowledge A. Argentieri and R. Colella for technical assistance, P. Soria for graphical work and A. Lorusso for numerous and useful comments on the paper.

References

[1] R. Akbani, S. Kwek, N. Japkowicz, Applying support vector machines to imbalanced datasets, in: Machine Learning: ECML 2004, 2004, pp. 39–50.

[2] A.L. Blum, P. Langley, Selection of relevant features and examples in machine learning, Artif. Intell. 97 (1997) 245–271.
[3] M. Ambriola, R. Bellotti, M. Circella, R. Maglietta, S. Stramaglia, Supervised algorithms for particle classification by a transition radiation detector, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment 510 (2003) 362–370.
[4] N. Ancona, R. Maglietta, E. Stella, Data representation in kernel based learning machines, Mach. Learn. Appl., Proc. 11 (2004) 129–136.
[5] C. Blake, C. Merz, UCI Repository of Machine Learning Databases, Department of Information and Computer Sciences, University of California, Irvine, 1998.
[6] L. Breiman, Random forests, Mach. Learn. 45 (2001) 5–32.
[7] L. Bruzzone, S. Serpico, Classification of imbalanced remote-sensing data by neural networks, Pattern Recognit. Lett. 18 (11) (1997) 1323–1328.
[8] J. Cervantes, L. Xiaoou, Y. Wen, L. Kang, Support vector machine classification for large data sets via minimum enclosing ball clustering, Neurocomput. 71 (2008) 611–619.
[9] N. Chawla, L. Hall, W. Kegelmeyer, Smote: synthetic minority oversampling techniques, J. Artif. Intell. Res. 16 (2002) 321–357.
[10] T. Evgeniou, M. Pontil, Support vector machines with clustering for training with very large datasets, in: I. Vlahavas, C. Spyropoulos (Eds.), Methods and Applications of Artificial Intelligence, Springer-Verlag, Berlin, 2002, pp. 346–354.
[11] Y. Freund, R. Schapire, A decision-theoretic generalization of on-line learning and an application to boosting, J. Comput. Syst. Sci. 55 (1997) 119–139.
[12] H. He, E. Garcia, Learning for imbalanced data, IEEE Trans. Knowl. Data Eng. 21 (9) (2009) 1263–1284.
[13] B. Heisele, T. Poggio, M. Pontil, Face Detection in Still Gray Images, Technical Report, Artificial Intelligence Laboratory,Massachusetts Institute of Technology, Cambridge, MA, 2000. A.I. Memo No. 1687.
[14] X. Jiang, R. El-Kareh, L. Ohno-Machado, Improving predictions in imbalanced datavusing pairwise expanded logistic regression, in: Annual Symposium on Biomedical and Health Informatics (AMIA'01), 2001.
[15] P. Kang, S. Cho, Eus svms: Ensemble of under-sampled svms for data imbalance problems, in: I. King, J. Wang, L. Chan, D. Wang (Eds.), Lecture Notes in Computer Science LNCS: ICONIP 2006, 4232, 2006, pp. 837–846.
[16] M. Kubat, R. Holte, S. Matwin, Machine learning for the detection of oil spills in satellite radar images, Mach. Learn. 30 (2) (1998) 195–215.
[17] A. Lalkhen, A. McCluskey, Clinical tests: sensitivity and specificity, Contin. Educ. Anaesth., Crit. Care Pain 8(6) (2008) 221–223.
[18] Y. Liua, H. Lohb, A. Sunc, Imbalanced text classification: a term weighting approach, Expert Syst. Appl. 36(1) (2009) 690–701.
[19] R. Maglietta, N. Amoroso, M. Boccardi, S. Bruno, A. Chincarini, G. Frisoni, P. Inglese, A. Redolfi, S. Tangaro, A. Tateo, R. Bellotti. Automated hippocampal segmentation in 3d mri using random undersampling with boosting algorithm. Accepted for publication in Pattern Analysis and Applications.
[20] R. Morey, C. Petty, Y. Xu, Y. Hayes, H. Wagner, D. Lewis, K. LaBar, M. Styner, G. McCarthy, A comparison of automated segmentation and manual tracing for quantifying hippocampal and amygdala volumes, Neuroimage 45(3) (2009) 66–855.
[21] J. Morra, Z. Tu, L. Apostolova, A. Green, A. Toga, P. Thompson, Comparison of adaboost and support vector machines for detecting alzheimer's disease through automated hippocampal segmentation, IEEE Trans. Med. Imaging 29 (2010) 30–43.
[22] V. Nikulin, G. McLachlan, Classification of imbalanced marketing data with balanced random sets, J. Mach. Learn. Res. 7 (2009) 89–100.
[23] T. Oommen, L. Baise, R. Vogel, Sampling bias and class imbalance in maximum-likelihood logistic regression, Math. Geosci. 43 (2011) 99–120.
[24] G. Pasquariello, N. Ancona, P. Blonda, C. Tarantino, G. Satalino, A. D'Addabbo, Neural network ensembles and support vector machine Neural network ensembles and support vector machine, Proceedings of IEEE International Geosc and Remote Sensing Symposium IGARSS (2002).
[25] J.C. Platt, Fast training of support vector machines using sequanzial minimal optimization, in: B. Scholkopf, C.J.C. Burges, A.J. Smola (Eds.), Advanced in Kernel Methods – Support Vector Learning, MIT press, Cambridge, MA, 1998, pp. 185–208.
[26] F. Provost, V. Kolluri, A survey of methods for scaling up inductive algorithms, Data Mining and Knowledge Discovery 3 (2) (1999) 131–169.
[27] A. Refice, D. Capolongo, G. Pasquariello, A. D'Addabbo, F. Bovenga, R. Nutricato, F. Lovergine, L. Pietranera, SAR and InSAR for flood monitoring: examples with COSMO/SKYMED data, IEEE J. Selected Top. Appl. Earth Observ. Rem. Sens. 7 (7) (2014) 2711–2722.
[28] C. Seiffert, T. Khoshgoftaar, J.V. Hulse, A. Napolitano, Rusboost: a hybrid approach alleviating class imbalance, IEEE Trans. Syst. Man Cybern. Part A 40 (1) (2010) 185–197.
[29] M. Tang, C. Yang, K. Zhang, Q. Xie, Cost-sensitive support vector machine using randomized dual coordinate descent method for big class-imbalanced data classification, Abstr. Appl. Anal. 2014 (2014), doi:10.1155/2014/416591. Article ID 416591, 9 pages.
[30] Y. Tang, Y. Zhang, N. Chawla, S. Krasser, Svms modeling for highly imbalanced classification, IEEE Trans. Syst., Man and Cybern.ùPART B: Cybern. 39 (1) (2009) 281–288.
[31] I. Tomek, Two modifications of cnn, IEEE Trans. Syst. Man Cybernet. 6 (11) (1976) 769–772.
[32] I. Tsang, J. Kwork, P. Cheung, Core vector machines: fast svm training on very large data sets, J. Mach. Learn. Res. 6 (2005) 363–392.
[33] V. Vapnik, An overview of statistical learning theory, IEEE Trans. Neural Netw. 10 (5) (1999) 988–999.
[34] Q. Wang, A hybrid sampling SVM approach to imbalanced data classification, Abstr. Appl. Anal. 2014 (2014), doi:10.1155/2014/972786. Article ID 972786, 7 pages.
[35] R. Yan, Y. Liu, R. Jin, A. Hauptman, On predicting rare classes with SVM ensembles in scene classification, in: IEEE International Conference on Acoustics, Speech and Signal Processing(ICASSP'03), 2003.
[36] K. Zou, S. Warfield, A. Bharatha, C. Tempany, M. Kaus, S. Haker, W. Wells, F. Jolesz, R. Kikinis, Statistical validation of image segmentation quality based on a spatial overlap index, Acad. Radiol. 11(2) (2004) 178–189.