On the Use of Surrounding Neighbors for Synthetic Over-Sampling of the Minority Class

V. GARCÍA J. S. SÁNCHEZ R. A. MOLLINEDA
Dept. Llenguatges i Sistemes Informàtics
Universitat Jaume I
Av. Sos Baynat s/n, 12071, Castelló de la Plana
SPAIN
vgarciaj@hotmail.com {sanchez,mollined}@uji.es

Abstract: - It has been observed that class imbalance may produce an important deterioration of the classification accuracy. One of the most popular methods to tackle this problem is the synthetic minority over-sampling technique (SMOTE). From the original SMOTE algorithm, we here propose the use of three surrounding neighborhood approaches with the aim of generating artificial minority examples, but taking both the proximity and the spatial distribution of the examples into account. Experiments with ten real data sets are conducted to compare the models introduced in this paper with SMOTE, demonstrating their effectiveness in a number of problems.

Key–Words: - Imbalance, Over-sampling, Surrounding neighborhood, SMOTE, Proximity graph, k-NCN

1 Introduction

Class imbalance constitutes one of the problems that has recently received most attention in areas such as Machine Learning and Pattern Recognition. A two-class data set is said to be imbalanced if one of the classes (the minority one) is represented by a very small number of instances in comparison to the other (the majority) class. It has been observed that class imbalance may cause a significant deterioration in the classifier performance attainable by standard supervised methods [1]. This issue is particular important in real-world applications where it is costly to misclassify examples of the minority class, such as diagnosis of an infrequent diseases [2], detection of fraudulent telephone calls [3], detection of oil spills in radar images [4], text categorization [5] and credit assessment [6]. Because of examples of the minority and majority classes usually represent the presence and absence of rare cases, respectively, they are also known as positive and negative examples.

Research on this topic can be categorized into three groups. One has primarily focused on several solutions for handling the imbalance both at the data and algorithmic levels. Another group has addressed the problem of measuring the classifier performance in imbalanced domains. The third consists of analyzing the relationship between class imbalance and other data complexity characteristics. From these three general topics in class imbalance, data level methods are the most investigated. These methods consist of balancing the original data set, either by over-sampling the minority class [7, 8, 9] and/or by under-sampling the majority class, until the classes are approximately equally represented [10, 11, 12, 13]. Several researchers have reported that over-sampling in general obtains more accurate results than the under-sampling methods [10, 13, 14].

A well-known over-sampling method is the Synthetic Minority Over-sampling Technique (SMOTE) proposed by Chawla et al. [11]. In this paper, we explore the convenience of using an alternative neighborhood concept, namely *surrounding neighborhood* [15], within the SMOTE algorithm. This kind of neighborhood takes both proximity and geometric information into account. Three different approaches to surrounding neighborhood are here used in the SMOTE and empirically compared with the plain or nearest neighborhood (that is, neighborhood just defined form the minimum distance).

2 Performance Evaluation Measures in Imbalanced Domains

Most of performance measures for two-class problems are built over a 2×2 confusion matrix as illustrated in Table 1. From this, four simple measures can

be directly obtained: *TP* and *TN* denote the number of positive and negative cases correctly classified, while *FP* and *FN* refer to the number of misclassified positive and negative examples, respectively.

Table 1: Confusion matrix for a two-class problem

	Predicted positive	*Predicted negative*
Positive class	True Positive (TP)	False Negative (FN)
Negative class	False Positive (FP)	True Negative (TN)

The most widely used metrics for measuring the performance of learning systems are the *error rate* and the *accuracy*, which can be computed as $(TP + TN)/(TP + FN + TN + FP)$. Nevertheless, researchers have demonstrated that, when the prior class probabilities are very different, these measures are not appropriate because they do not consider misclassification costs, are strongly biased to favor the majority class, and are sensitive to class skews [16, 17, 18]. Thus, in domains with imbalanced data, alternative metrics that measure the classification performance on positive and negative classes independently are required.

The *true positive rate*, also referred to as *recall* or *sensitivity*, $TPR = TP/(TP + FN)$, is the percentage of correctly classified positive instances. Analogously, the *true negative rate* (or *specificity*), $TNR = TN/(TN + FP)$, is the percentage of correctly classified negative examples. The *false positive rate*, $FPR = FP/(FP+TN)$, refers to the percentage of misclassified positive examples. The *false negative rate*, $FNR = FN/(TP + FN)$ is the percentage of misclassified negative examples. Alternative criteria for evaluating classifier performance include ROC curves [17] and the geometric mean of accuracies [8]. These are good indicators of performance on imbalanced data because they are independent of the distribution of examples between classes.

The geometric mean of accuracies measured separately on each class is defined as g-mean$= \sqrt{TPR \times TNR}$. This measure closely relates with the distance to perfect classification in the ROC space and the idea is to maximize the accuracy on each of the two classes while keeping these accuracies balanced. In the present paper, the classifier performance measure used for the experiments will be the g-mean.

3 The SMOTE Algorithm

The simplest strategy to expand the minority class corresponds to random over-sampling, that is, a non-heuristic method that balances the class distribution through the random replication of positive examples [10, 19]. Although effective, this method may increase the likelihood of overfitting since it makes exact copies of the minority class instances [11].

In order to avoid overfitting, Chawla et al. [11] proposed a technique, called SMOTE, for over-sampling the minority class. Instead of merely replicating cases belonging to the minority class, this generates artificial examples of the minority class by interpolating existing instances that lie close together. It first finds the k nearest neighbors (k-NN) belonging to the minority class for each positive example and then, the synthetic examples are generated in the direction of some or all of the nearest neighbors.

SMOTE allows the classifier to build larger decision regions that contain nearby instances from the minority class. Depending upon the amount of over sampling required, neighbors from the k nearest neighbors are randomly chosen. In the experiments reported in the original paper, k is set to five. When, for example, the amount of over-sampling needed is 200%, only two neighbors from the five nearest neighbors are chosen and one prototype is generated in the direction of each of these two neighbors. Synthetic prototypes are generated in the following way: take the difference between the feature vector (instance) under consideration and its nearest neighbor. Multiply this difference by a random number between 0 and 1, and add it into the feature vector under consideration.

From the original SMOTE algorithm, a number of modifications have been proposed in the literature, most of them pursuing to determine the region in which the positive examples should be generated. Among them, Borderline–SMOTE [12] is one of the most known and consists of using only positive examples close to the decision boundary, since these are more likely to be misclassified.

4 Surrounding Neighborhood

Intuitively, the concept of neighborhood should be such that the neighbors are as close to an object as possible but also, the neighbors should lie as homogeneously around it as possible. The second condition is a consequence of the first in the asymptotic case but in some practical situations, the geometrical location can become much more important than the actual distances to appropriately characterize an object by its neighborhood. As the classical nearest neighborhood takes into account the first property only, the nearest neighbors may not be placed symmetrically around the object if the neighborhood in the data set is not spatially homogeneous. In fact, it has been shown that the use of local distance measures can significantly

improve the behavior of a classifier in the case of a finite sample size [20].

Alternative concepts of neighborhood have been proposed as a way to overcome the problem just pointed out for the k-NN rule. These consider proximity and symmetry so as to define the general concept of neighborhood. Thus they try to search for neighbors close enough (in the basic distance sense), but also in terms of their spatial distribution with respect to a given object. These methods have been generally referred to as *surrounding neighborhood* [15]. Among others, the *nearest centroid neighborhood* and the *graph neighborhood* have been demonstrated to behave better than the traditional nearest neighborhood for a number of pattern classification problems [15, 21, 22].

4.1 Nearest Centroid Neighborhood

The first definition of surrounding neighborhood comes from the Nearest Centroid Neighborhood (NCN) concept [23]. Let p be a point whose k neighbors should be found from a set of points $X = \{x_1, \ldots, x_n\}$. These k neighbors are such that (a) they are as near p as possible and (b) their centroid is also as close to p as possible. Both conditions can be satisfied through an iterative procedure in the following way:

Algorithm 1 k-NCN

Input:
$X = \{x_1, \ldots, x_n\}$
k Number of neighbors
p A query point
Output:
$Q = \{q_1, \ldots, q_k\}$ Set of k neighbors
The first NCN of p is its nearest neighbor, say q_1
The i'th neighbor, q_i ($i \geq 2$), is such that the centroid of this and all previously selected neighbors, $q_1, \ldots, q_i$ is the closest to p

This definition gives rise to a kind of neighborhood in which both closeness and spatial distribution of neighbors are taken into account because of the centroid criterion. However, the iterative procedure clearly does not minimize the distance to the centroid because it gives precedence to the individual distances instead.

4.2 Graph Neighborhood

Let $X = \{x_1, \ldots, x_n\}$ be a finite set of n points in R^d, where d denotes the dimensionality of the feature space. From a general point of view, a proximity graph, say $G = (V, E)$, is defined as an undirected graph with the set of vertices $V = X$, and the set of edges, E, such that $(x_i, x_j) \epsilon E$ if and only if x_i and x_j satisfy some mutual neighborhood property. Thus the set of graph neighbors of a given point constitutes its *graph neighborhood* [21]. The graph neighborhood of a subset, $S \subseteq V$, consists of the union of all the graph neighbors of every node in S.

The Gabriel Graph (GG) and the Relative Neighborhood Graph (RNG) are two well-known examples of *proximity graphs* [24]. In this paper, we concentrate our examination on these two graph approaches.

4.2.1 Gabriel Graph

Let $d(\cdot, \cdot)$ be the Euclidean distance between two points in R^d. The GG is defined as follows:

$$(x_i, x_j) \epsilon E \quad (1)$$
$$\Leftrightarrow d^2(x_i, xj) \leq d^2(x_i, x_k) + d^2(x_j, x_k)$$
$$\forall x_k \epsilon X, k \neq i, j$$

Two points x_i and x_j are Gabriel Neighbors if and only if there is no other point from X lying in the *hypersphere of influence* Γ_{x_i,x_j} centered at their middle point and whose diameter is the distance between them.

4.2.2 Relative Neighborhood Graph

In a similar fashion, the set of edges in the RNG can be obtained in the following way:

$$(x_i, x_j) \epsilon E \quad (2)$$
$$\Leftrightarrow d(x_i, xj) \leq max[d(x_i, x_k), d(x_j, x_k)]$$
$$\forall x_k \epsilon X, k \neq i, j$$

In this case, its corresponding geometric interpretation is based on the concept of *lune* Λ_{x_i,x_j}, which is defined as the disjoint intersection between two hyperspheres centered at x_i and x_j and whose radii are equal to the distance between them. Two points x_i and x_j are said to be *Relative Neighbors* if and only if their lune does not contain other points of the set X.

5 Surrounding–SMOTE

As already mentioned, the surrounding neighborhood methods can be applied to a number of pattern classification problems. These approaches can effectively help in several situations (finite sample size case), in which training instances do not fully represent the

underlying statistics and/or the distance used (irrelevant in the asymptotic case) exhibits some undesirable properties. In fact, the ultimate goal of the surrounding neighborhood is to overcome some of the practical drawbacks outlined for the k-NN techniques and more specifically, to outperform the classification performance of these decision rules.

Hence, based upon the analysis stated above, we here propose to employ the three surrounding neighborhood approaches (NCN, GG, and RNG) for oversampling the minority class by means of the SMOTE algorithm. SMOTE finds the k nearest neighbors belonging to the minority class for each positive example in the training set and then, it generates artificial samples in the direction of some (or all) of the nearest neighbors.

Instead of nearest neighbors, now we propose to select surrounding positive neighbors for each instance of the minority class. The rationale behind this modification of the original SMOTE algorithm is that these surrounding neighbors will extend the region of new synthetic samples and therefore, it seems that the resulting over-sampled set can describe better the decision boundaries.

The Surrounding–SMOTE algorithm (with NCN) can be written as follows:

Algorithm 2 Surrounding–SMOTE

Input:
$P = \{p_1, \ldots, p_{min}\}$ Minority class examples
min Number of minority examples
N Number of synthetic samples to generate for each minority class example
k Number of neighbors
Output:
$Synthetic$ Set of artificial samples examples
for $i = 1$ to min **do**
  Find k neighbors of p_i
  for $j = N$ to 1 **do**
    Choose randomly one of the k neighbors of p_i, say $n(p_i)$
    $diff = p_i - n(p_i)$
    $gap =$ Random number between 0 and 1
    $NewSample = p_i + gap * diff$
    $Synthetic \longleftarrow NewSample$
  end for
end for

The size of the set of synthetic samples will be $N \times min$. Note that in the case of GG and RNG we do not have to provide the number k of neighbors, since each training instance may have a different number of graph neighbors. Apart from this, the rest of the procedure for Surrounding–SMOTE with proximity graphs will be exactly the same as the algorithm above.

6 Experimental Data Sets and Results

The experiments were carried out on ten real data sets taken from the UCI Machine Learning Database Repository (`http://archive.ics.uci.edu/ml/`) (a summary is given in Table 2). All data sets were transformed into two-class problems. The fifth column in Table 2 indicates the original classes that have been joined to shape the majority class. For example, in Vehicle database the objects of classes 2, 3, and 4 were combined to form a unique majority class and the original class 1 was left as the minority class.

Table 2: Data sets used in the experiments

	P. Examples	N. Examples	Classes	Majority Class
Breast	81	196	2	1
Ecoli	35	301	8	1,2,3,5,6,7,8
German	300	700	2	1
Glass	17	197	9	1,2,4,5,6,7,8,9
Haberman	81	225	2	1
Phoneme	1586	3818	2	1
Pima	268	500	2	1
Satimage	626	5809	7	1,2,3,5,6,7
Vehicle	212	634	4	2,3,4
Yeast	429	1055	10	1,3,4,5,6,7,8,9,10

For each database, we have estimated the geometric mean by 5×10–fold cross–validation: each data set is divided into ten blocks of size $n/10$ (where n denotes the total number of objects), using nine folds as the training set and the remaining block as the test set. This is repeated 5 times. Values in Table 3 correspond to the average g-mean obtained with the 1-NN classifier, a support vector machine (SVM) and a multilayer perceptron (MLP). Each classifier has been applied to the original training set and also to sets that have been preprocessed by the original SMOTE (with k-NN) algorithm and by the three modifications here presented (with k-NCN, GG and RNG). In the case of NN and NCN, the value of k has been set to 5.

The first observation from results in Table 3 is that in all cases, the different resampling algorithms clearly improve the performance obtained on the original training sets (without any preprocessing). The most important gains are on the results of the SVM, in

Table 3: Experimental results (g-mean) for three different classifiers. Highlighted are the best values for each database

			1-NN		
	Original	k-NN	k-NCN	GG	RNG
Breast	58.81	59.89	60.10	60.39	59.74
Ecoli	69.59	83.30	82.68	82.90	78.58
German	58.57	60.69	58.42	60.64	60.07
Glass	54.48	70.54	67.50	67.66	63.45
Haberman	50.31	55.49	56.55	57.19	56.59
Phoneme	87.10	88.11	88.52	**88.76**	88.25
Pima	64.46	67.53	69.91	67.42	66.94
Satimage	82.28	**89.58**	89.24	89.53	88.74
Vehicle	62.20	66.49	67.04	67.60	67.13
Yeast	62.52	65.18	66.59	66.58	65.54
			SVM		
	Original	k-NN	k-NCN	GG	RNG
Breast	55.30	65.32	**66.73**	66.16	65.73
Ecoli	0.0	87.86	88.39	**88.49**	88.14
German	65.38	72.11	**72.41**	71.71	72.10
Glass	0.0	55.44	56.42	56.64	56.59
Haberman	0.0	56.38	55.52	55.58	55.79
Phoneme	70.06	74.72	74.96	74.79	74.80
Pima	69.29	74.17	**74.39**	73.87	74.17
Satimage	0.0	69.87	69.77	69.89	69.94
Vehicle	0.0	74.43	74.01	74.26	74.56
Yeast	42.11	70.70	70.68	**70.97**	70.78
			MLP		
	Original	k-NN	k-NCN	GG	RNG
Breast	55.75	56.78	57.89	58.49	55.63
Ecoli	82.73	88.20	86.02	86.88	85.72
German	63.91	65.91	66.13	66.50	66.42
Glass	36.97	74.29	70.28	**75.68**	73.63
Haberman	48.63	63.40	62.93	63.68	**64.36**
Phoneme	78.10	81.18	81.33	81.43	81.47
Pima	68.83	72.93	73.40	73.68	73.46
Satimage	78.47	86.33	87.30	87.08	85.88
Vehicle	74.87	77.61	**77.63**	77.26	76.52
Yeast	66.80	70.65	70.79	70.77	70.62

which differences are especially high for Ecoli, Glass, Haberman, Satimage, and Vehicle databases (here the g-mean corresponding to the original training sets is approximately 0).

It is also interesting to note the fact that in many cases, the Surrounding–SMOTE algorithms outperform the original SMOTE, although differences are not always significant. One can observe that the most important increases correspond to the results obtained with the SVM followed by those with the 1-NN classifier. For each preprocessing method there are a total of 30 results, since we have ten databases and three different classifiers. Having this in mind, the SMOTE with k-NCN has been 19 times better than the original SMOTE, that with GG 20 times, and the SMOTE with RNG 16 times.

When comparing the three Surrounding–SMOTE approaches, the k-NCN has been 9 time better than the GG and the RNG methods, whereas the GG has appeared 16 times better than k-NCN and RNG. Although the nature of the classifier used after the preprocessing plays an important role in the resulting performance, it seems that in general the GG-based approach can be deemed as the best over-sampling algorithm from those tested here.

7 Conclusions and Future Work

This paper has focused on the problem of expanding the minority class so as to balance the class distribution of the training set. Three alternatives to the original SMOTE algorithm have been proposed, all them based upon the concept of surrounding neighborhood. In particular, we have used the NCN, the GG and the RNG in the step of selecting neighbors for further generation of artificial positive examples. The aim of these modifications of SMOTE is to take both proximity and geometrical distribution of neighbors into account in order for extending the region of those synthetic created objects.

Experimental results on ten real databases and using three very different classifiers (1-NN, SVM, and MLP) have shown that the Surrounding–SMOTE algorithms achieve relative improvements in terms of the g-mean with respect to the original SMOTE. From the three alternatives, the GG seems to be the one with the highest values of performance, followed by the NCN-based approach.

Future work must include an exhaustive analysis of the approaches introduced in the present paper, by means of a larger number of experiments (more databases and more classifiers) that allow to draw conclusions about the merits of each method. Also, we are interesting in incorporating a filtering phase into the Surrounding–SMOTE algorithms in order to remove noisy examples of both classes.

Acknowledgment: This work has been partially supported by grants DPI2006–15542 from the Spanish CICYT, CSD2007–00018 from Consolider–Ingenio

2010, GV\2007\105 from Generalitat Valenciana and SEP–2003–C02–44225 from the Mexican CONACyT.

References:

[1] N. Japkowicz and S. Stephen, The Class Imbalance Problem: A Systematic Study, *Intelligent Data Analysis*, Vol.6, No.5, 2002, pp. 429–450.

[2] G. Cohen, M. Hilario, H. Sax, S. Hugonet and A. Geissbuhler, Learning from Imbalanced Data in Surveillance of Nosocomial Infection, *Artificial Intelligence in Medicine*, Vol.37, 2006, pp. 7–18.

[3] T. Fawcett and F. Provost, Adaptative Fraud Detection, *Data Mining and Knowledge Discovery*, Vol.1, 1996, pp. 291–316.

[4] M. Kubat, R. C. Holte and S. Matwin, Machine Learning for the Detection of Oil Spills in Satellite Radar Images, *Machine Learning*, Vol.30, 1998, pp. 195–215.

[5] S. Tan, Neighbor-weighted k-Nearest Neighbor for Unbalanced Text Corpus, *Expert Systems with Applications*, Vol.28, 2005, pp. 667–671.

[6] Y. Huang, C. Hung and H.C. Jiau, Evaluation of Neural Networks and Data Mining Methods on a Credit Assessment Task for Class Imbalance Problem, *Nonlinear Analysis: Real World Applications*, Vol.7, 2006, pp. 720-747.

[7] R. Barandela, J. S. Sánchez, V. García and E. Rangel, Strategies for Learning in Class Imbalance Problems *Pattern Recognition*, Vol. 36, 2003, pp. 849–851.

[8] M. kubat and S. Matwin, Addressing the Curse of Imbalanced Training Sets: One-sided Selection, *Proceedings of the 14th International Conference on Machine Learning*, 1997, pp. 179–186.

[9] S. Yen, Y. Lee, C. lin and J. Ying, Investigating the Effect of Sampling Methods for Imbalanced Data Distributions, *2006 IEEE Inernational Conference of Systems, Man and Cybernetics*, 2006, pp. 4163-4168.

[10] G. E. A. P. A. Batista, R. C. Prati and M. C. Monard, A Study of the Behavior of Several Methods for Balancing Machine Learning Training Data, *SIGKDD Explorations Newsletter*, Vol.6, No.1, 2004, pp. 20–29.

[11] N. V. Chawla, K. W. Bowyer, L. O. Hall and W. P. Kegelmeyer, SMOTE: Synthetic Minority Over-Sampling Technique, *Journal Of Artificial Intelligence Research*, Vol.16, 2002, pp. 321–357.

[12] H. Han, W. Wang, and B. Mao, Borderline–SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning, *Proceedings of the International Conference on Intelligent Computing*, 2005, pp. 878–887.

[13] G. He, H. Han and W. Wang, An Over-Sampling Expert System for Learning from Imbalanced Data Sets, *Proceedings of the International Conference on Neural Networks and Brain*, Vol.1, 2005, pp. 537–541.

[14] J. V. Hulse, T. M. Khoshgoftaar and A. Napolitano, Experimental Perspectives on Learning from Imbalanced Data, *Proceedings of the 24th International Conference on Machine Learning*, 2007, pp. 935–942.

[15] J. S. Sánchez, F. Pla and F. J. Ferri, On the Use of Neighbourhood-Based Non-Parametric Classifiers, *Pattern Recognition Letters*, Vol.18, 1997, pp. 1179–1186.

[16] S. Daskalaki and N. Avouris, Evaluation of Classifiers for an Uneven Class Distribution Problem, *Applied Artificial Intelligence*, Vol.20, 2006, pp. 381–417.

[17] T. Fawcett, An Introduction to ROC Analysis, *Pattern Recognitiobn Letters*, Vol.27, 2006, pp. 861-874.

[18] J. Huang and C. X. Ling, Using AUC and Accuracy in Evaluating Learning Algorithms, *IEEE Trans. on Knowledge and Data Engineering*, Vol.17, 2005, pp. 299–310.

[19] C. X. Ling and C. Li, Data Mining for Direct Marketing: Problems and Solutions, *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining*, 1998, pp. 73–79.

[20] R. D. Short and K. Fukunaga, A New Nearest Neighbour Distance Measure, *Proceedings of the 5th Internat. Conf. on Pattern Recognition*, 1980, pp. 81–86.

[21] J. S. Sánchez, F. Pla and F. J. Ferri, Prototype Selection for the Nearest Neighbour Rule Through Proximity Graphs, *Pattern Recognition Letters*, Vol.18, 1997, pp. 507–513.

[22] J. Zhang, Y.-S. Yim and J. Yang, Intelligent Selection of Instances for Prediction Functions in Lazy Learning Algorithms, *Artificial Intelligence Review*, Vol.11, 1997, pp. 175–191.

[23] B. B. Chaudhuri, A New Definition of Neighborhood of a Point in Multidimensional Space, *Pattern Recognition Letters*, Vol.17, 1996, pp. 11–17.

[24] J. W. Jaromczyk and G. T. Toussaint, Relative Neighbourhood Graphs and Their Relatives, *Proc. IEEE*, Vol.80, pp. 1502-1517.