A hybrid method to face class overlap and class imbalance on neural networks and multi-class scenarios

R. Alejo, R.M. Valdovinos, V. García, J.H. Pacheco-Sanchez

1. Introduction

In supervised classification learning, the intrinsic difficulties in the data may significantly affect generalization performance of standard classifier algorithms. An important issue that has been identified into the 10 challenging problems is when the datasets suffer from skewed class distributions, that is, the number of samples of one class outnumbers the other classes (class imbalance) (Yang and Wu, 2006). Existing research indicates that the samples of one class outnumber the other classes (class imbalance) (Yang and Wu, 2006). The class imbalance problem causes seriously negative effects on the classification performance (Zhou and Liu, 2006), since the classifier suffers from skewed class distributions, that is, the number of samples is unbalanced. These studies suggest that the class imbalance is not a problem by itself, but the degradation of performance is also related to other factors, i.e., the degree of class overlapping (Batista et al., 2005; Denil and Trappenberg, 2010; García et al., 2008).

Class overlap and class imbalance have been widely studied in the literature and treated separately. Rarely are they treated at the same time. There are also very few approaches dealing with these complexities in multi-class scenarios. In this paper, we introduce a novel hybrid algorithm to face class imbalance and class overlapping simultaneously on multi-class problems.

Most of the research addressing the class imbalance problem can be grouped into three categories: (i) assigning distinct costs to the classification errors for positive and negative samples (Domingos, 1999; Zhou and Liu, 2006), (ii) resampling the original training dataset, either by over-sampling the minority class and/or under-sampling the majority class until the classes are approximately equally represented (Chawla et al., 2002; Estabrooks et al., 2004; García and Herrera, 2009), and (iii) internally biasing the discrimination-based process so as to compensate for the class imbalance (Anand et al., 1993; Bruzzone and Serpico, 1997; Oh, 2011).

It is generally accepted that the class imbalance is responsible for a significant degradation of the performance on individual classes. However, recent papers have pointed out that there is no direct correlation between class imbalance and the loss of performance. These studies suggest that the class imbalance is not a problem by itself, but the degradation of performance is also related to other factors, i.e., the degree of class overlapping (Batista et al., 2005; Denil and Trappenberg, 2010; García et al., 2008).

Class overlapping occurs in those zones where the decision boundary regions intersect. The overlapping samples have a high probability of being misclassified by any classifier. Hence, several instance selection (IS) methods have been developed to address this challenging task (Olivera-López et al., 2010). The IS approaches that seek to remove points that are noisy or do not agree with their neighbors are called Edition algorithms. The most popular editing methods are based on the nearest neighbor rule.

Class overlap and class imbalance have been widely studied in the literature and treated separately. Rarely are they treated at the same time. There are also very few approaches dealing with these complexities in multi-class scenarios. In this paper, we introduce a novel hybrid algorithm to face class imbalance and class overlapping simultaneously on multi-class problems.
square error) for dealing with the class imbalance problem, (b) we adapted the Gabriel graphs editing (GGE) to make it effective in reducing the class overlap in the neural network context and (c) to combine the points (a) and (b) in order to generate an effective strategy dealing with the class overlap and the class imbalance.

The rest of this paper is organized as follows. Related works are briefly reviewed in Section 2. In Section 3 we introduce the modified back-propagation algorithm for tackling the class imbalance problem. The editing algorithm is described in Section 4. In Section 5 we present a hybrid method dealing with the class overlap and the class imbalance. Sections 6 and 7 we show the experimental set up and results respectively. Finally, Section 8 is for conclusion.

2. Related works

Back-propagation is now the most widely used tool in the field of artificial neural networks (NN). However, despite the general success of back-propagation, several major deficiencies are still needed to be solved. The major disadvantage of back-propagation is the slow rate of convergence of net output error. This is especially difficult in “imbalanced” classification problems (Anand et al., 1993; Ramanan et al., 1998), i.e., where the training set contains many more samples of some “dominant” classes (majority classes) rather than the other “subordinated” classes (minority classes).

In the back-propagation algorithm, the class imbalance poses severe problems in the training stage as the learning process becomes biased towards the majority classes, ignoring the minority classes and leaving them poorly trained at the end of the training stage. The learning process also becomes slower and it takes longer to converge to the expected solution (Anand et al., 1993).

Much research has been done in addressing the class imbalance problem (Zhou and Liu, 2006). In the NN field, the modified learning algorithm has been proposed for dealing with this problem. In (Anand et al., 1993) a modified back-propagation is proposed. This consists of calculating a direction in weight-space which decreases the error for each class (majority and minority class) in the same magnitude, in order to accelerate the learning rate for two-class imbalance problems. In (Bruzzone and Serpico, 1997; Lawrence et al., 1998; Ramanan et al., 1998; Oh, 2011), the error function was modified by introducing different costs associated with making errors in different classes. Basically, when the sum of square errors is calculated, each term is multiplied by a class dependent (regularization) factor. This compensates the class imbalance (Bruzzone and Serpico, 1997; Lawrence et al., 1998; Oh, 2011) and accelerates the convergence of the NN (Ramanan et al., 1998). However, the main drawback of these approaches is the use of free parameters, because these parameters control the updating amount of weights whether training samples are in the minority or majority classes.

The most popular strategies to deal with the class imbalance problem have been at the data level (under or over sampling techniques). These methods for balancing the classes are the most researched because they are independent of the underlying classifier and can be easily implemented for any problem. The simplest method to increase the size of 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 (He and Garcia, 2009). Nevertheless, since this method replicates existing examples in the minority class, overfitting is more likely to occur. Chawla et al. (2002) proposed an over-sampling technique that generates new synthetic minority samples by interpolating between several preexisting positive examples that lie close together. This method, called SMOTE (Synthetic Minority Over-sampling Technique), allows to the classifier to build larger decision regions that contain nearby samples from the minority class.

On the other hand, random under-sampling (Japkowicz and Stephen, 2002) aims at balancing the dataset through the random removal of negative examples. Despite its simplicity, it has empirically shown to be one of the most effective resampling methods. Unlike the random approach, many other proposals (García and Herrera, 2009; Batista et al., 2005; He and Garcia, 2009) are based on a more intelligent selection of the negative examples to be eliminated.

Several papers point out the class imbalance as an obstacle when applying machine learning algorithms to real world domains. However, in some cases, learning algorithms perform well on several imbalanced domains (Batista et al., 2005). Recent research shows that the class imbalance is not always a problem (Garcia et al., 2008; Denil et al., 2010). Japkowicz and Stephen (2002) suggest that some classifiers are not sensitive to the class imbalance problem in cases where the classes are separable. In the same way some researchers (Lawrence et al., 1998; Visa et al., 2005) affirm that the class imbalance is not an intrinsic problem if the distributions do not overlap.

The overlapping appears when the samples of the minority class share a region with the majority one, where all the samples are intertwined (this is an intrinsic problem of the data). García et al. (2008) have shown that overlap can play an even larger role in determining classifier performance than the class imbalance problem. Lawrence et al. (1998) suggest that when distribution is overlapped, it is desirable to pre-process or edit the data in a manner that results in reduced overlap. The similar idea was studied in (Batista et al., 2005). The latter work shows data cleaning strategies which usually lead to a performance improvement for highly overlapped datasets. Tang and Gao (2007) use the k-nearest neighbor (k-NN) and inverse k-NN algorithms to eliminate potential noisy patterns, and extraction of boundary pattern. The goal of the latter work is to deal with the classification problem, which involves class overlapping. Nevertheless, the main drawback of these approaches is that parameter setting in k-NN impacts directly on the classification performance. Kretzschmar et al. (2005) introduce variance-controlled NN (VCNNs), which were developed to handle class overlap. These VCNNs are feed forward models trained by minimizing an error function involving the class specific variance (CSV) computed at their outputs. This minimization suppresses abrupt changes in the responses of the trained classifiers in regions of the input space occupied by overlapping classes. The main restriction is that VCNNs require the selection of an additional free parameter (to adjust the influence of CSV) specified empirically by the user.

3. A modified back-propagation (MBP)

The multilayer perceptron (MLP) neural network usually comprises one input layer, one or more hidden layers, and one output layer. Input nodes correspond to features, hidden layers are used for computations, and output nodes are related with the number of classes. A neuron is the elemental unit of each layer. It computes the weighted sum of its inputs, adds a bias term and drives the result though a generally non-linear (commonly a sigmoid) activation function to produce a single output. The most popular training algorithm for MLP is back-propagation, which uses a set of training instances for the learning process. Given a feed-forward network, the weights are initialized to small random numbers. Each training instance sent through the network and the output from each unit is computed. The target output is compared with the estimated output of the network by calculating the error, which is fed back through the network. To adjust the weights, the back-propagation algorithm uses a gradient descent to minimize the squared error. At each unit in the network starting from the output unit and moving to the hidden units, its error value is
used to adjust the weights of its connections as well as to reduce the error. This process is repeated for a fixed number of times, or until the error is small.

On the other hand, in the back-propagation algorithm the class imbalance problem generates unequal contributions to the mean square error (MSE) in the training phase (Anand et al., 1993). Clearly the major contribution to the MSE is produced by the majority class.

Let us consider a training dataset (TDS) with two classes \( j = 2 \) such that \( N = \sum n_j \) and \( n_j \) is the number of samples from class \( j \). Suppose that the MSE by class can be expressed as

\[
E_j(U) = \frac{1}{N} \sum_{i=1}^{N} (t_{pj} - z_{pj})^2,
\]

where \( t_{pj} \) is the desired output and \( z_{pj} \) is the actual output of the network for the sample \( i \). Then the overall MSE can be expressed as

\[
E(U) = \sum_{j=1}^{C} E_j(U) = E_1(U) + E_2(U).
\]

If \( n_1 \ll n_2 \) then \( E_1(U) \ll E_2(U) \) and \( \|\nabla E_1(U)\| \ll \|\nabla E_2(U)\| \), consequently \( \nabla E(U) \approx \nabla E_2(U) \). So, \( \nabla E(U) \) is not always the best direction to minimize the MSE in both classes (Anand et al., 1993).

Considering that the class imbalance problem affects negatively in the back-propagation algorithm due to the disproportionate contributions in the MSE, it is possible to consider a cost function \( (\gamma) \) that balances the MSE as follows:

\[
E(U) = \sum_{j=1}^{l} \gamma(j)E_j = \gamma(1)E_1(U) + \gamma(2)E_2(U)
\]

\[
= \frac{1}{N} \sum_{j=1}^{l} \gamma(j) \sum_{i=1}^{N} (t_{pj} - z_{pj})^2,
\]

where \( \gamma(1) \|\nabla E_1(U)\| \approx \gamma(2) \|\nabla E_2(U)\| \) avoiding that the minority class be ignored in the learning process. In this work, we propose a new cost function defined as:

\[
\gamma(j) = \frac{\|\nabla E_{\text{max}}(U)\|}{\|\nabla E_1(U)\|},
\]

where \( \|\nabla E_{\text{max}}(U)\| \) corresponds to the largest majority class.

On the other hand, when a cost function is included in the training process, the data probability distribution is altered (Lawrence et al., 1998). Nevertheless, the cost function \( \gamma(j) \) (Eq. (4)) reduces its impact in the data distribution probability because the cost function value is diminished gradually. In this way, the class imbalance problem is reduced in early iterations, and later \( \gamma(j) \) reduces its effect on the data distribution probability.

4. Editing technique for handling class overlap

The editing techniques have been proposed to remove noisy samples as well as close border classes (overlapping), leaving smoother decision boundaries (Wilson and Martinez, 2000). The aim is to improve the classifier accuracy. The most popular editing schemes are based on the well-known \( k \) Nearest Neighbor (k-NN) rule. However, this rule only takes into account the distances to a number of close neighbors. Alternative concepts of neighborhood have been proposed to consider geometrical relation between a sample and some of its neighbors (Dasarathy et al., 2000).

The Gabriel graph (GG) has recently been used for introducing a set of editing methods for the k-NN rule (Sánchez et al., 1997). The Gabriel graph editing (GGE) consists of applying the general idea of Wilson’s algorithm (Wilson and Martinez, 2000), but using the graph neighbors of each sample instead of the Euclidean or other norm-based distance neighborhood. Two samples \( x \) and \( y \) are graph neighbors in a GG = \( (V, E) \) if there exists an edge \( (x, y) \in E \) between them. Taking into account the definitions of GG, the graph neighborhood of a given point requires that no other point lies inside the union of the zones of influence (i.e. hypersphere of influence) corresponding to all its graph neighbors.

The application of GGE has some additional properties as compared to the conventional methods. First, they consider the number of neighbors as a variable feature which depends upon every prototype. Secondly, since the graph neighborhood of a sample always tends to be widely distributed around it, the information extracted from samples close to decision boundaries may be richer in the sense of the prototypes distribution (Dasarathy et al., 2000).

The original GGE was proposed to improve the k-NN accuracy (Sánchez et al., 1997). However, in this work the original GGE was adapted to do it effective in the back-propagation context. The aim was to remove noisy and overlapping samples of the majority classes, but keeping all the positive samples. This task allows improving the back-propagation learning over the minority classes. The proposed GGE can be summarized as follows:

1. For each sample \( p \) in the TDS, constructs its corresponding GG.
2. To keep \( p \) in the TDS only if all its graph neighbors are of its same class.
3. Other issue, if \( p \) belongs to some majority class, then discard \( p \) from TDS.
4. If all majority class samples are eliminated then change the rule 2 by 5.
5. To keep \( p \) in the TDS only if the majority graph neighbors are of its same class.

5. Methodology for dealing with class imbalance and class overlapping on multi-class problems

This section provides an overview of the method here proposed to deal with the class imbalance and the class overlapping simultaneously, which consists of combining an editing technique and a cost function. This strategy can be summarized as follows:

1. MBP: To deal with the class imbalance problem.
   (a) To modify the back-propagation algorithm by applying a cost function (Eq. (4)), in order to avoid that the minority classes would be ignored in the training process, and to accelerate the convergence of the neural network.
2. GGE: To deal with class overlapping problem.
   (a) To edit the TDS with the GGE technique (Section 4), removing only majority samples in the overlap region and producing a local balance of the classes.
3. MBP + GGE (proposed strategy).
   (a) To train the MLP with the modified back-propagation algorithm over the edited TDS (see Fig. 1).

![Fig. 1. Block diagram of the proposed method.](image-url)
6. Experimental protocol

6.1. Database description

We used in our experiments five remote sensing datasets: Cayo, Feltwell, Sittingame, Segment and 92AV3C. Feltwell is related to an agricultural region near Feltville, Feltwell (UK) (Bruzzone and Serpico, 1997), Cayo presents a particular region in the gulf of Mexico, and Sittingame consists of the multi-spectral values of pixels in 3 × 3 neighborhoods in a satellital image (Asuncion, 2007). The Segment contains instances drawn randomly from a dataset of seven outdoor images (Asuncion, 2007). 92AV3C dataset corresponds to a hyperspectral image (145 × 145 pixels) taken over Northwestern Indiana's Indian Pines by the AVIRIS sensor.

In order to cover Cayo in a highly imbalanced dataset some of their classes were merged as follows: join classes 1, 3, 6, 7 and 10 to integrate class 1; join classes 8, 9 and 11 to integrate class 3, finally, the rest of classes (2, 4 and 5) were obtained from the original dataset. M92AV3C is a subset of 92AV3C, it contains six classes (2, 3, 4, 6, 7 and 8) and 38 attributes. The attributes were selected using a common features selection algorithm (Best-First Search (Kohavi and John, 1997)) implemented in WEKA.

Feltwell, Sittingame, Segment and 92AV3C were random undersampled with the goal of generating severe class imbalanced datasets. A brief summary of these multi-class imbalance datasets is shown in Table 1. Note that they are highly imbalanced datasets. For each database, a 10-fold cross-validation was applied. The datasets were divided into 10 equal parts, using nine folds as training set and the remaining block as test set.

6.2. Classifier performance and significance statistical test

The most traditional metric for measuring the performance of learning systems is the accuracy which can be defined as the degree of fit (matching) between the predictions and the true classes of data. However, the use of plain accuracy to evaluate the classifiers in imbalanced domains might produce misleading conclusions, since it is strongly biased to favor the majority classes (He and Garcia, 2009; Oh, 2011). Shortcomings of this evaluator have motivated the search for new measures. One of the most widely-used measures in imbalanced domains is the receiver operating characteristic curve (ROC), which is a tool for visualizing, organizing and selecting classifiers based on their trade-offs between true positive rates and false positive rates. Furthermore, a quantitative representation of a ROC curve is the area under it, which is known as AUC (Fawcett, 2006). The AUC measure for multi-class problems can be defined as:

\[ AUC = \frac{2}{|J|(|J| - 1)} \sum_{j,k \neq j} AUC_0(d_j, d_k), \]

where \( AUC_0(d_j, d_k) \) is the AUC for each pair of classes \( j_1 \) and \( j_2 \).

Other evaluation metrics are the geometric mean of accuracies measured separately on each class (He and Garcia, 2009). For multi-class problems it can be computed as:

\[ g\text{-mean} = \left( \prod_{i=1}^{k} \text{acc}_i \right)^{\frac{1}{k}}, \]

where \( \text{acc}_i \) is the accuracy on the class \( i \) and \( J \) the number of classes.

Statistical tests are used to evaluate whether the performance of a new method or learning algorithm on the same problem is significantly different. Into the framework of empirical analysis, the

Student’s paired t-test is the most widely used parametric statistical procedure. However, it is well-known that it is conceptually inappropriate and statistically unsafe to require certain assumptions like the data is normally distributed (Demšar, 2006). In this work, we adopt the non-parametric statistical Friedman test to perform a multiple comparison, which is equivalent to the repeated-measures ANOVA. This test used to check if all methods perform equal on the selected datasets can be rejected. The first step in calculating the test statistic is to rank the algorithms for each dataset separately; the best performing algorithm should have the rank of 1, the second best rank 2, etc. The Friedman test uses the average rankings to calculate the Friedman statistic, which can be computed as:

\[ X^2_F = \frac{12N}{K(K+1)} \left( \sum_{j} R_j^2 - \frac{K(K+1)^2}{4} \right). \]

where \( K \) denotes the number of methods, \( N \) the number of datasets, and \( R_j \) the average rank of method \( j \) on all datasets. Iman and Davenport (Iman and Davenport, 1980) showed that \( \chi^2_F \) presents a conservative behavior, so they proposed a better statistic distributed according to the \( F \)-distribution with \( K-1 \) and \( (K-1)(N-1) \) degrees of freedom,

\[ F_F = \frac{(N-1)\chi^2_F}{N(K-1) - \chi^2_F}. \]

When the null-hypothesis is rejected, we can use post hoc tests in order to find the particular pairwise comparisons that produce statistical significant differences. The Bonferroni–Dunn post hoc test is applied to report any significant difference between individual methods used. The test uses the average rank of each method and compare it to each other if these differ by at least the critical difference, which is given by

\[ CD = q_a \sqrt{\frac{K(K+1)}{6N}}, \]

where the value \( q_a \) is based on the studentized range statistic divided by \( \sqrt{2} \).

6.3. Resampling methods

SMOTE and random sampling (RUS) are used in the empirical study, because they are popular approaches to deal with the class imbalance problem. However, these methods have internal parameters that enable the user to set up the resulting class distribution obtained after the application of these methods. In this paper, we decided to add or remove examples until a balanced distribution was reached. This decision was motivated by two reasons: (a) simplicity (to avoid use many free parameters) and (b) effectiveness. Results obtained with the other classifiers (Weiss and Provost, 2003), have shown that when AUC is used as a performance measure, the best class distribution for learning tends to be near the balanced class distribution.
6.4. Neural network configuration

The MLP was trained with the standard back-propagation (SBP) and modified back-propagation (MBP) algorithm in batch mode. For each TDS, MLP was initialized ten times with different weights. The results here included correspond to the average of those achieved in the ten different initializations and of ten partitions. The learning rate \( \eta \) was set at 0.1 and only one hidden layer was used. The stop criterion was established at 25,000 epoch or an MSE below to 0.001. The number of neurons for the hidden layer was obtained from the trial and error strategy. So, the number of neurons was 7, 6, 12, 10 and 10, for MCayo, MFelt, MSat, MSeg and M92AV3C datasets respectively.

7. Results and discussion

In order to assure the performance of the proposed method, we have carried out an experimental comparison with respect to well-known resampling approaches. In total, seven strategies were examined:

(i) Standard back-propagation algorithm (SBP), (ii) modified back-propagation algorithm (MBP), (iii) standard back-propagation with Gabriel graph editing (SBP + GGE), (iv) modified back-propagation with Gabriel graph editing (MBP + GGE), (v) SMOTE, (vi) SMOTE with Gabriel graph editing (SMOTE + GGE) and (vii) random under sampling (RUS). The datasets that were preprocessed by the SMOTE, SMOTE + GGE and RUS strategies were applied to the SBP algorithm.
In this paper, we have omitted other neural networks approaches (to deal with the class imbalance problem) as the two-phase technique (Bruzzone and Serpico, 1997), threshold moving (Zhou and Liu, 2006), or modified error function (Oh, 2011), because these methods contain many prior free parameters, so it is difficult to make a fair comparison.

With the aim of showing the effectiveness of combining the MBP and the GGE techniques, in Fig. 2, the performance by class of the SBP, the MBP, the SBP + GGE and the proposed strategy (MBP + GGE), are presented separately (the bold boxes belong to minority classes). The results show that the minority classes performance is severely affected by the class imbalance. Some minority classes are not enough learned as observed in Fig. 2a, e, i, and q, due to the class imbalance problem. So these minority classes show 0% of accuracy. The effects of the class imbalance problem is to slow down the convergence of the SBP due to disproportionate contribution in the MSE in the training phase (see Section 3). An immediate consequence of this, is the difficulty of achieving effective performance (in terms of classification) in a "reasonable" period of time. Especially in situations where there is an extreme class imbalance.

On the other hand, Fig. 2 shows that when the class imbalance is compensated (MBP), the minority classes performance improves (Fig. 2b, f, j, n, and r). However, in high overlapped TDS is not enough (Fig. 2j and r).

GGE technique is used to reduce the overlapping between classes. Fig. 2c, g, k, o, and s, present the results obtained for applying the GGE technique. Note that the latter improves the minority
classes performance, especially in overlapped TDS (see class 5 in Fig. 2k and o). Nevertheless, the class imbalance problem continues affecting the dataset. For example, observe MFelt, and M92AV3C datasets (Fig. 2g and s respectively). A negative consequence of GGE technique is that when the minority classes accuracy increases, the majority classes performing is affected.

The four column of Fig. 2 presents the combining of both MBP and GGE (MBP + GGE). These results show a remarkable improvement in minority classes performance and exhibit a better performance by applying the MBP and GGE techniques individually.

The modification of the training algorithm including a cost function (MBP) increases the recognition rate of less represented classes, accelerating the convergence of the network, and applying the GGE technique reduces the confusion of the minority classes in the overlap region. So the results presented in Fig. 2d, h, l and t, demonstrate the effectiveness of combining the MBP and GGE techniques.

Fig. 3 shows experimental results when comparing the proposed method with respect to others well-known resampling approaches. The experimental results are presented in graphics where boxes represent the accuracy by class, and the bold boxes belong to minority classes. Fig. 3 exhibits the worst accuracy for the minority classes as shown by the RUS strategy (mainly over MFelt and MSat, see Fig. 3h and l). This means that when TDS is severely imbalanced, it removes samples to balance the class distribution, and it is not effective on back-propagation, because the RUS involves a loss of useful information that could be important for the training process. In the M92AV3C and MSat datasets, the RUS shows a good minority classes performance, however, it is not a tendency.

SMOTE was very successful in MCayo and MFelt, but in MSat and M92AV3C datasets as part of the minority classes performance are worse than the proposed method (see class 5 in Fig. 3j and n, and class 1 in Fig. 3r). We believe that the explanation is that these datasets present high level of overlapping. For example MSat dataset shows high overlapping level between the C-01 and C-05 classes, in other words, it is not enough to balance the TDS for improving the classifier performance over minority classes when the TDS overlaps. This is the reason of the low accuracy in class C-05 for RUS, MBP and SMOTE.

On the other hand, MBP + GGE presents better results than the SMOTE for overlapping datasets (see MSat, MSat, and M92AV3C, Fig. 3, n and r), this is due to the data cleaning method (GGE) is more efficient in highly overlapped regions. The MBP + GGE method starts to be less effective as overlapping is reduced (for example see MCayo and MFelt in Fig. 3).

The accuracy showed by SMOTE + GGE was very similar at SMOTE, however, in spite of SMOTE + GGE include GGE technique, this method was ineffective on overlapping datasets (see MSat, Fig. 3k). The explanation is that, as SMOTE was firstly applied the overlap level was increased too, thus GGE was not able to remove enough overlap for improving the accuracy of minority class. To prove this, we repeat the experiment: we first applied GGE over MSat, and then MSat was over-sampled using SMOTE. The results obtained were very successful and similar as achieved by MBP + GGE. The AUC = 0.756(0.050), g-mean = 0.713(0.071), C-05 accuracy = 0.91(0.02). This results show the effective of the GGE technique to reduce the class overlap and for improving the accuracy of the classifier over minority classes.

SMOTE and SMOTE + GGE strategies have made great improvement on the minority classes. However, they add information to the TDS by introducing new (non-replicate) minority classes samples, which involves a larger TDS and longer training times for the same number of training epochs. In addition, when the dataset presents high overlapping, the SMOTE can be not good choice, because it can increase the class overlapping, and the order SMOTE + GGE is not recommended. Experimental results show that is better to apply first GGE and after the SMOTE, i.e., GGE + SMOTE.

Fig. 3 shows that the results obtained by MBP + GGE are very competitive with the results obtained by SMOTE and SMOTE + GGE. As well as MBP + GGE which does not have internal parameters that the user needs to set up before applying it and use of a TDS sight is more reduced (much less training time). These are main advantages of MBP + GGE over SMOTE and SMOTE + GGE.

Table 2 summarizes the experimental results in terms of AUC and g-mean on the five datasets when using six different strategies previously enumerated. For each method, the average ranking is shown. As can be seen in the table, the original (imbalanced) training set has the highest Friedman score, which means that this strategy performs worse than other methods, whereas MBP + GGE is the best performing algorithm for AUC and g-mean. Note, that SMOTE performs equal to MBP + GGE when the results are evaluated with AUC.

The Iman and Davenport statistic computed using Eq. (8) yields $F_T = 4.43$ and $F_T = 4.06$, for AUC and g-mean respectively. The critical value of the $F$-distribution with 6 and 24 degrees of freedom for $x = 0.05$ is 2.51. Given that the Iman and Davenport statistics are clearly greater than their associated critical value, the null-hypothesis that all methods perform equally can be rejected with a level of significance $x = 0.05$. Then a post hoc statistical analysis was used to detect significant differences for the control algorithm (method with the lowest ranking) in each measure.

### Table 2

Performance on five datasets measured using AUC, g-mean and average rank (AR).

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Imbalanced</th>
<th>MBP</th>
<th>GGE</th>
<th>MBP + GGE</th>
<th>SMOTE</th>
<th>SMOTE + GGE</th>
<th>RUS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AUC</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MCayo</td>
<td>0.477 (0.020)</td>
<td>0.715 (0.034)</td>
<td>0.636 (0.064)</td>
<td>0.828 (0.040)</td>
<td>0.860 (0.040)</td>
<td>0.847 (0.024)</td>
<td>0.722 (0.035)</td>
</tr>
<tr>
<td>MFelt</td>
<td>0.658 (0.022)</td>
<td>0.839 (0.033)</td>
<td>0.700 (0.017)</td>
<td>0.880 (0.031)</td>
<td>0.895 (0.046)</td>
<td>0.884 (0.027)</td>
<td>0.749 (0.028)</td>
</tr>
<tr>
<td>MSat</td>
<td>0.663 (0.026)</td>
<td>0.752 (0.044)</td>
<td>0.774 (0.049)</td>
<td>0.757 (0.041)</td>
<td>0.826 (0.038)</td>
<td>0.705 (0.038)</td>
<td>0.726 (0.031)</td>
</tr>
<tr>
<td>M92AV3C</td>
<td>0.871 (0.032)</td>
<td>0.916 (0.058)</td>
<td>0.905 (0.030)</td>
<td>0.918 (0.095)</td>
<td>0.880 (0.053)</td>
<td>0.882 (0.031)</td>
<td>0.914 (0.027)</td>
</tr>
<tr>
<td><strong>AR</strong></td>
<td>7.0</td>
<td>4.2</td>
<td>4.6</td>
<td>2.4</td>
<td>2.4</td>
<td>3.8</td>
<td>3.6</td>
</tr>
<tr>
<td><strong>g-mean</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MCayo</td>
<td>0.00 (0.00)</td>
<td>69.18 (4.18)</td>
<td>48.38 (24.67)</td>
<td>81.99 (4.18)</td>
<td>82.24 (2.48)</td>
<td>80.63 (2.86)</td>
<td>70.22 (4.10)</td>
</tr>
<tr>
<td>MFelt</td>
<td>0.00 (0.00)</td>
<td>82.29 (4.10)</td>
<td>0.00 (0.00)</td>
<td>87.54 (3.42)</td>
<td>89.05 (5.30)</td>
<td>88.14 (2.88)</td>
<td>53.05 (27.77)</td>
</tr>
<tr>
<td>MSat</td>
<td>0.00 (0.00)</td>
<td>49.36 (28.5)</td>
<td>73.89 (6.71)</td>
<td>72.27 (5.38)</td>
<td>80.12 (5.28)</td>
<td>0.000 (0.00)</td>
<td>0.00 (0.00)</td>
</tr>
<tr>
<td>M92AV3C</td>
<td>66.60 (31.81)</td>
<td>90.09 (0.91)</td>
<td>89.21 (3.82)</td>
<td>91.29 (9.46)</td>
<td>78.83 (24.62)</td>
<td>85.57 (9.53)</td>
<td>9037 (3.46)</td>
</tr>
<tr>
<td><strong>AR</strong></td>
<td>6.7</td>
<td>4.0</td>
<td>4.9</td>
<td>2.2</td>
<td>2.4</td>
<td>4.2</td>
<td>3.6</td>
</tr>
</tbody>
</table>

* Classification using SBP.
Fig. 4 display a graphical representation of the results of Bonferroni–Dunn’s post hoc test, where for each method on the y-axis (ordered in ascending rank), the AR is plotted on the x-axis. For each AR we sum the critical difference obtained by the Bonferroni method, \( CD = 3.60 \) for \( \alpha = 0.05 \) in the two measures considered. The vertical dashed line segment represents the end of the best performing algorithm and the start of the next significant method. MBP + GGE is the best algorithm, although according to Bonferroni–Dunn’s test, only the difference to the Imbalanced approach is different.3

The effects of the MBP + GGE can be better analyzed by considering the number of samples that remain in the TDS after its application. Results in Fig. 5 suggest a higher decrease in the size of the dataset when it is processed with the GGE, whereas using SMOTE increase twice of the original size. RUS reduce the TDS size even more, however, it does not always present good classifier performance. Reducing the dataset involve to reach a better neural network learning time and reduce storage requirements.

8. Conclusions

In this paper, we propose a hybrid method (MBP + GGE) for dealing with class imbalance and class overlapping on multi-class problems. The MBP + GGE is based on combination of modified back-propagation (MBP) with a Gabriel graph editing technique (GGE). For a modified back-propagation algorithm we proposed to include a new cost function (based on MSE) in the algorithm, and to make the Gabriel graph editing effective we adapted it in the back-propagation context. MBP + GGE generates two effects: (a) MBP: to compensate the class imbalance during the training process and (b) GGE: to reduce the confusion of the minority classes in the overlap region. With the edition of the majority classes it is possible to reduce the confusion between the minority and majority classes.

The MBP + GGE strategy was compared with the conventional class imbalance techniques: RUS, SMOTE, MBP, GGE and SMOTE + GGE. Results show that SMOTE and SMOTE + GGE are very effective even with highly imbalanced datasets, but inadequate on overlapped datasets. MBP + GGE show a better performance on class overlap problems. The data cleaning step used in the MBP + GGE seems to be specially suitable in situations having a high degree of overlapping, moreover, GGE produces a small training dataset.

The SMOTE needs to find the most appropriate resampling rate, i.e., to determine the number of samples when we introduce them in the minority classes before applying it. So the main advantages of MBP + GGE over SMOTE and SMOTE + GGE are: (a) does not have internal parameters that the user needs to set up before applying it and (b) use of a TDS sight more reduced (much less training time). As we see from the results, MBP + GGE is a very competitive strategy for dealing with the class imbalance and the class overlapping on multi-class problems.

Further research is required to investigate the potential of the strategy proposed in this paper in “severe” multi-class imbalance and highly class overlapping problems. So, the exploration of the other editing strategies is necessary when approaching the graph based on editing scheme. Also, the study of new cost functions which help to speed up the neural network convergence in order to avoid altering the data probability distribution.

Acknowledgments

This work has partially been supported by the UAEM (Project 3072/2011), the Mexican SEP (PROMEP/103.5/11/3796), the TESJO (SDMAIA-010), the Spanish Ministry of Education and Science (CSD2007-00018, TIN2009-14205) and the Generalitat Valenciana (PROMETEO/2010/028).

References


---

3 Other powerful tests, such as Holm and Hochberg’s ones would be necessary, for comparing the control algorithm with the rest of algorithms.