The Multi-Class Imbalance Problem: Cost Functions with Modular and Non-Modular Neural Networks

R. Alejo[1,2], J.M. Sotoca[2], and G. A. Casañ[2]

[1] Centro Universitario Atlacomulco, Universidad Autónoma del Estado de México
Carretera Toluca-Atlacomulco Km. 60 (México)
[2] Dept. Llenguatges i Sistemes Informátics, Universitat Jaume I
Av. Sos Baynat s/n, 12071 Castelló de la Plana (Spain)

Abstract. In this paper, the behavior of Modular and Non-Modular Neural Networks trained with the classical backpropagation algorithm in batch mode and applied to classification problems with Multi-Class imbalance is studied. Three different cost functions are introduced in the training algorithm in order to solve the problem in four different databases. The proposed strategies show an improvement in the classification accuracy with three different types of Neural Networks.

Keywords: Multi-Class, imbalance, backpropagation, cost function.

1 Introduction

Typically supervised learning methods are designed to work with reasonably balanced Training Sets (TS) [1], but many real world applications have to face imbalanced data sets [2]. A TS is said to be imbalanced when several classes are under-represented (minority classes) in comparison with others (majority classes).

A feed-forward Neural Network (NN) trained on an imbalanced dataset is not able to learn with sufficient discrimination among classes [3]. Particularly, in the back-propagation algorithm with batch-mode, the majority class dominates the training process, therefore the minority classes converge very slowly [4].

In the machine learning field, most of the work in imbalanced problems are addressed to solve the problem for two classes [5], and only a few studies discuss the multi-class imbalance problem [4, 6].

This paper is focused mainly in the evaluation of different cost functions designed to improve the NN performance. Thus, the backpropagation algorithm is modified to deal with the multi-class imbalance problem. These cost functions will be calculated in relation to the proportion of samples used in to train the NN. The main contributions of this paper are the comparison between different approaches that apply cost functions in the multi-class imbalance problem learning directly and the effect of decoupling multi-class problems and solving two-class imbalance problem using a modular strategy.

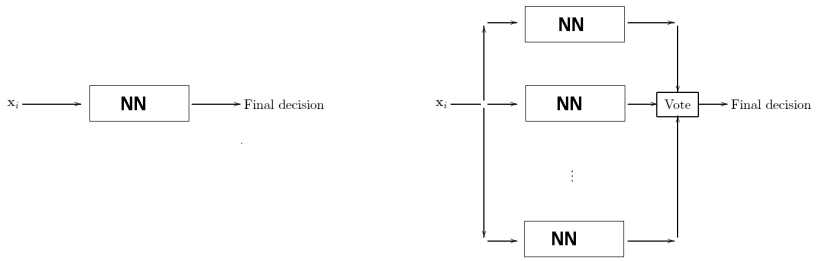
2 Modular and Non Modular Neural Networks

The Modular Neural Networks (Mod-NNs) present a new trend in NN architectural designs [7]. It has been motivated by the highly-modular nature in biological networks and based on the "divide and conquer" approach [8].

The use of Mod-NNs implies a significant improvement in the learning process in comparison with a Non-Modular NN (Non-Mod-NN) [8]. The Non-Modular classifiers tend to introduce high internal interferences because a strong coupling among their hidden-layer weights can appear [9].

The Mod-NNs show the following computational advantages [4]: a) the numbers of iterations needed to train the individual modules is less than the number of iterations needed to train a Non-Mod-NN for the same task; b) the modules in a Mod-NN are smaller than Non-Mod-NN; and c) the modules can be trained independently and in parallel.

Here, we use the Mod-NN architecture to face the multi-class imbalance problem. In this Mod-NN architecture, each module is a single-output NN (see Fig.1) which determines if a pattern belongs to a particular class. Thereby, a K-class problem is reduced to a set of K two-class problems. A module for class c_k is trained to distinguish between patterns belonging to c_k respect to the patterns of the rest of classes. So, in our Mod-NN, given a instance test x_i, the two class network with the highest rating is taken as the class label for that instance.



(a) Non-Modular NN (b) Modular NN

Fig. 1. The NN architectures

Non-Mod-NN and Mod-NNs modules are studied in this work with Radial Basis Function NN (RBFNN), Random Vector Functional Link Net Networks (RVFLNN) and Multilayer Perceptron (MLP). MLP and RBFNN are two well-known NN in the pattern recognition field [10]. The main difference from the MLP is that the activations of the hidden neurons of RBFNN depend on the distance of an input vector to a prototype vector whereas MLP calculate the inner product of the input vector and the weight vector [11]. Both NN can be trained by supervised methods [10]. All parameters are adapted simultaneously by an optimization procedure.

The RVFLNN is a variant of the RBFNN. The RVFLN of Pao [12] is added to the RBFNN in order to obtain the last one, and it gives a extra connectivity of the FLN along with any functions put into the offset hidden neurons. The addition of connections between the hidden neurons adds extra learning power [10].

3 The backpropagation algorithm and the class imbalance problem

Empirical studies of the backpropagation algorithm [13], show that the class imbalance problem does not generate equal contributions to the mean square error (MSE) in the training phase. Obviously the main contribution to the MSE is produced by the majority class.

Let us consider a TS with two classes such that $N = \sum_i^m n_i$ and n_i is the number of samples from class i. Suppose that the MSE by class may be expressed as

$$E_i(U) = \frac{1}{N} \sum_{n=1}^{n_i} \sum_{p=1}^{L} (y_p^n - F_p^n)^2 \,, \tag{1}$$

so that the overall MSE can be expressed as

$$E(U) = \sum_{i=1}^{m} E_i = E_1(U) + E_2(U) \,. \tag{2}$$

If $n_1 << n_2$ then $E_1(U) << E_2(U)$ and $\|\nabla E_1(U)\| << \|\nabla E_2(U)\|$. Then $\nabla E(U) \approx \nabla E_2(U)$. So, $-\nabla E(U)$ it is not always the best direction to minimize the MSE in both classes.

Considering that the TS imbalanced problem affects negatively the backpropagation algorithm due to the disproportionate contributions to the MSE, it is possible to consider a cost function (γ) that balances the TS class imbalance as follows

$$\begin{aligned} E(U) &= \textstyle\sum_{i=1}^{m} \gamma(m) E_i = \gamma(1) E_1(U) + \gamma(2) E_2(U) \\ &= \tfrac{1}{N} \textstyle\sum_{m=1}^{M} \gamma(m) \sum_{n=1}^{n_m} \sum_{p=1}^{L} (y_p^n - F_p^n)^2 \,, \end{aligned} \tag{3}$$

where $\gamma(1)\|\nabla E_1(U)\| \approx \gamma(2)\|\nabla E_2(U)\|$ avoiding the fact that the minority class would be ignored in the learning process [14]. The previous process can be generalized to multi-class problems [6] to obtain the following cost functions:

- **Option 0**: $\gamma(m) = 1$, is the backpropagation algorithm without modifications.
- **Option 1**: $\gamma(m) = n_{max}/n_m$; where $m = 1, ..., M$ and n_{max} is the number of samples of the majority class.
- **Option 2**: $\gamma(m) = N/n_m$; where $m = 1, ..., M$ and N is the total number of samples.
- **Option 3**: $\gamma(m) = \|\nabla E_{max}(U)\|/\|\nabla E_m(U)\|$, where $\|\nabla E_{max}(U)\|$ is the majority class. This function is a simplification of [13].

4 Data sets

For the experimental phase, Cayo, Ecoli6, Feltwell and Satimage databases with multiple classes were used. Feltwell is related to an agriculture region near to Felt Ville, Feltwell (UK) and is divided in training data (5124 samples) and test data (5820 samples). Cayo, which represents a particular region in the gulf of Mexico, was partitioned using the holdout method (50% training and 50% test). Both are remote sensing images.

Ecoli6 is obtained from Ecoli, a biological database created by the *Institute of Molecular and Cellular Biology* from Osaka University, Japan. It was originally distributed in eighth classes but in this work classes 7 and 8 have been eliminated since these only have two samples making it difficult to apply the method of cross validation. The Ecoli6 database was split using the five cross validation method in 10 parts (5 training and 5 testing), using the 80-20 proportion. Satimage were obtained from the *UCI Machine Learning Database Repository*, and used without changes. The data in Satimage is divided into: 4435 training and 200 testing samples. In Table 1, the most important characteristics of each database are summarized.

Table 1. A brief summary of the some basic characteristics of the databases

Dataset	Size	Attr.	Class	Class distribution
Cayo	6019	4	11	838/293/624/322/133/369/324/722/789/833/772
Ecoli6	332	7	6	5/143/77/52/35/20
Feltwell	10944	15	5	3531/2441/896/2295/1781
Satimage	6430	36	6	1508/1531/703/1356/625/707

5 Methodology

The NN was trained with the backpropagation algorithm in batch mode. This process was repeated five times and the results correspond with the average. The *K-Hold-Out Paired t* and *K-Fold Cross-Validation Paired t* statistical tests [15] were applied. We assume that the set of differences has an independently drawn sample from an approximately normal distribution. Then, under the null hypothesis (equal classification accuracies), the following statistic has a Test Student distribution with K - 1 degrees of freedom. The learning rate (η) was set to 0.0001 for RBFNN and RVFLNN, and a value of 0.9 was used for the MLP. Only one hidden layer was used in the last case. The number of neurons for the hidden layer (for all NN) was established to 16, 15, 6 and 12 for Cayo, Ecoli6, Feltwell and Satimage respectively.

In this study, *Accuracy*, *g-mean* and $Kappa$ coefficient are used as measure criteria for the classifiers performance. These measure criteria are often obtained from the confusion matrix, where real classes are in columns whereas predicted ones appears in rows (Table 2). The table built on this way is a general vision assignment, the right ones

(diagonal elements) like the wrong ones (elements out of the diagonal). From Table 2, the measure criteria $Accuracy = \frac{\sum_{i=1}^{k} n_{ii}}{n}$ is obtained, where n is the total number of samples and $\frac{n_{jj}}{n_{+j}}$ is the *Accuracy by class*. The proportions of the samples p_{ij} in the cell (i, j) correspond to the number of samples n_{ij}, i.e, $p_{ij} = n_{ij}/n$. So, define p_{i+} and p_{+j} as $p_{i+} = \sum_{j=1}^{k} p_{ij}$, and $p_{+j} = \sum_{i=1}^{k} p_{ij}$.

The $Kappa$ coefficient is used as a quality parameter and takes into consideration the marginal distributions for the confusion matrix. Its value provides gives us an idea of the right percentage obtained in the classification process, once the random part has been eliminated. It is defined as $\left(Kappa = \frac{p_o - p_c}{1 - p_c}\right)$ where $p_o = \sum_{i=1}^{k} p_{ii}$ is the well predicted percentage, and $p_c = \sum_{i=1}^{k} p_{i+}p_{+j}$ the random coefficient.

Other measure used to quantify the classifier performance in the class imbalance problem is the geometric mean (*g-mean*) [14]. The geometric mean is defined as *g-mean* $= (\prod_{i=1}^{k} p_{ii})^{\frac{1}{k}}$, where p_{ii} is the class accuracy i.

Table 2. Confusion matrix

	Real Classes			
Predicted Classes	1	2	k	total (n_{i+})
1	n_{11}	n_{12}	n_{1k}	n_{1+}
2	n_{21}	n_{22}	n_{2k}	n_{2+}
k	n_{k1}	n_{k2}	n_{kk}	n_{k+}
total (n_{+j})	n_{+1}	n_{+2}	n_{+k}	**n**

6 Experiments and Discussions

As can be seen in Table 1, all databases have different TS imbalance levels. From a moderate imbalance up to a severe imbalance in the same data set. As example, in Ecoli6 database, classes 1 and 2 present a severe imbalance between them (majority class has 143 samples and minority class has only 5), while classes 5 and 6 show a reasonable imbalanced problem. A similar case is observed in Cayo, while in Feltwell and Satimage databases only a moderate imbalance between classes can be noticed.

The first column in the tables of classification performance shows the NN used and the second the applied evaluation criteria. The following columns contain the observed values due to the applied strategies. The data in parenthesis represents the standard deviation.

6.1 Results with Non-Mod-NN

The experience with the Cayo dataset shows the low performance of the classifier when imbalanced samples were used in the training phase without a cost function (Option 0).

In table 3, the zero value of *g-mean* is observed in the three network models without applying cost functions. As expected, the classes not correctly identified are minority classes (4 and 5) whose impact is small in the global performance. In the three network models using cost functions improve the results when the *g-mean* has a low value in Option 0 (for Cayo database it took value zero). The improvements of classifier accuracy are due to the increment in the minority classes accuracy when cost functions are applied in the training phase and consequently are identified correctly in the classifier phase. The options with better results are Option 2 for MLP and Option 3 for RBFNN and RVFLNN.

Table 3. Classification performance of **Cayo** dataset with Non Modular Networks

		Option 0	Option 1	Option 2	Option 3
MLP	Acc	76.7(1.8)	85.2(0.4)	85.9(0.3)	85.1(0.2)
	g-mean	0.0(0.0)	82.6(0.3)	84.1(0.7)	80.2(0.4)
	$Kappa$	0.7(0.0)	0.8(0.0)	0.8(0.0)	0.8(0.0)
RBFNN	Acc	75.0(4.5)	75.6(3.9)	78.4(4.8)	81.0(1.5)
	g-mean	0.0(0.0)	72.7(4.5)	73.7(4.8)	74.9(2.4)
	$Kappa$	0.7(0.1)	0.7(0.0)	0.8(0.1)	0.8(0.0)
RVFLNN	Acc	70.8(4.3)	75.4(2.1)	78.8(2.5)	82.0(1.4)
	g-mean	0.0(0.0)	73.1(2.4)	73.5(3.1)	74.1(2.4)
	$Kappa$	0.7(0.1)	0.7(0.0)	0.8(0.0)	0.8(0.0)

In data set Ecoli6 (Table 4) the *g-mean* values (see Option 0) are rather high specially for MLP, thus, the use of cost functions does not suppose an improvement in the results, and in same cases it obtains worse results. It should probably be related to the small size of the database.

Table 4. Classification performance of **Ecoli6** dataset with Non Modular Networks

		Option 0	Option 1	Option 2	Option 3
MLP	Acc	87.1(2.7)	82.8(6.8)	82.2(5.9)	85.8(5.2)
	g-mean	84.5(5.7)	82.1(6.2)	81.7(5.3)	84.4(6.7)
	$Kappa$	0.8(0.0)	0.8(0.1)	0.8(0.1)	0.8(0.1)
RBFNN	Acc	83.4(5.5)	83.1(5.4)	84.6(4.9)	84.6(5.5)
	g-mean	66.2(37.4)	84.1(3.6)	84.4(5.4)	84.3(7.3)
	$Kappa$	0.8(0.1)	0.8(0.1)	0.8(0.1)	0.8(0.1)
RVFLNN	Acc	84.9(4.3)	83.1(7.4)	84.9(4.4)	83.1(6.7)
	g-mean	62.0(35.4)	84.4(6.2)	84.1(6.2)	84.1(6.6)
	$Kappa$	0.8(0.1)	0.8(0.1)	0.8(0.1)	0.8(0.1)

Feltwell dataset presents a similar situation as Ecoli6 with high *g-mean* values, but when we use cost functions the accuracy results are similar or bit better. The difference

seems to come from the number of samples, that is significantly bigger in Feltwell. Option 2 shows the best results in MLP and RBFNN models, while Option 1 and 3 do the same for RVFLNN.

Table 5. Classification performance of **Feltwell** dataset with Non Modular Networks

		Option 0	Option 1	Option 2	Option 3
MLP	Acc	88.6(1.3)	88.8(0.7)	89.3(1.2)	88.8(0.4)
	g-mean	84.7(3.2)	87.0(0.8)	87.4(1.3)	86.6(0.4)
	$Kappa$	0.9(0.0)	0.9(0.0)	0.9(0.0)	0.9(0.0)
RBFNN	Acc	87.0(0.8)	86.5(1.8)	87.8(1.8)	84.1(3.0)
	g-mean	81.1(2.5)	84.9(2.2)	86.3(1.7)	78.2(7.2)
	$Kappa$	0.8(0.0)	0.8(0.0)	0.8(0.0)	0.8(0.0)
RVFLNN	Acc	86.9(1.8)	88.0(0.8)	87.0(2.9)	88.1(1.1)
	g-mean	74.0(15.6)	85.8(0.7)	84.5(4.1)	84.7(2.4)
	$Kappa$	0.8(0.0)	0.8(0.0)	0.8(0.0)	0.8(0.0)

Table 6. Classification performance of **Satimage** dataset with Non Modular Networks

		Option 0	Option 1	Option 2	Option 3
MLP	Acc	82.2(0.2)	85.6(0.6)	87.4(0.5)	86.3(0.5)
	g-mean	50.8(2.8)	84.2(0.6)	86.3(0.4)	83.6(1.1)
	$Kappa$	0.78(0.0)	0.8(0.0)	0.8(0.0)	0.8(0.0)
RBFNN	Acc	83.2(0.7)	83.3(0.6)	85.6(0.5)	84.3(1.3)
	g-mean	76.8(1.8)	81.6(1.2)	84.6(0.5)	81.4(2.5)
	$Kappa$	0.8(0.0)	0.8(0.0)	0.8(0.0)	0.8(0.0)
RVFLNN	Acc	82.8(1.2)	83.2(1.3)	84.5(0.3)	85.3(0.3)
	g-mean	74.1(1.9)	81.1(1.2)	82.8(0.5)	82.2(0.5)
	$Kappa$	0.8(0.0)	0.8(0.0)	0.8(0.0)	0.8(0.0)

In Satimage dataset (Table 6) all implemented Options report improvements in the global classifier performance. Important accuracy, *g-mean* values improvements can be observed (specially in MLP) when options 1, 2 and 3 are applied. The best results are obtained with Option 2 in MLP and RBFNN and Option 3 in RVFLNN.

6.2 Results with Mod-NNs

For Cayo dataset with Modular NN, as happened Non-Modular NN, the use of cost functions improved the accuracy results, except with RVFLNN. It is important to note that the behavior of Modular and Non-Modular NN is different for the same database (compare the accuracy classification for the three different NN with Option 0 in tables

3 and 7). In Cayo dataset with Modular NN the best results where obtained for Option 1 for the MLP and RBFNN, and Option 3 for RVFLNN.

Ecoli6 database with Modular NN presents a similar behavior as the Non-Modular NN, using cost functions does not suppose an improvement in the results (table 8).

When Modular NN and cost functions are used with Feltwell dataset, several behaviours can be seen (table 9). We can be note that in MLP the results are worse when the Options are applied, but in RBFNN and RVFLNN we can talk of a light improvement in the global performance. Only Option 3 reports better classifier performance.

Table 7. Classification performance of **Cayo** with Modular Networks

		Option 0	Option 1	Option 2	Option 3
MLP	Acc	71.4(1.4)	83.2(0.3)	83.4(0.4)	83.0(0.7)
	g-mean	0.0(0.0)	79.1(1.0)	79.8(0.3)	77.4(0.5)
	$Kappa$	0.7(0.0)	0.8(0.0)	0.8(0.0)	0.8(0.0)
RBFNN	Acc	76.6(4.2)	82.3(0.5)	80.6(1.3)	81.7(3.0)
	g-mean	0.0(0.0)	78.8(1.9)	77.8(2.4)	76.5(2.6)
	$Kappa$	0.74(0.0)	0.8(0.0)	0.78(0.0)	0.8(0.0)
RVFLNN	Acc	83.18(1.3)	79.8(5.9)	81.5(1.8)	83.19(1.2)
	g-mean	74.0(3.5)	68.1(13.9)	77.3(2.9)	77.5(2.0)
	$Kappa$	0.8(0.0)	0.8(0.1)	0.8(0.0)	0.8(0.0)

Table 8. Classification performance of **Ecoli6** with Modular Networks

		Option 0	Option 1	Option 2	Option 3
MLP	Acc	87.1(2.9)	84.9(3.5)	85.2(3.9)	85.5(3.9)
	g-mean	83.7(5.9)	83.3(3.5)	83.8(3.6)	83.3(5.9)
	$Kappa$	0.8(0.0)	0.8(0.0)	0.8(0.0)	0.8(0.0)
RBFNN	Acc	86.4(3.0)	85.2(7.4)	84.9(5.0)	85.2(6.0)
	g-mean	84.6(4.9)	84.7(8.2)	84.9(5.4)	86.0(4.8)
	$Kappa$	0.8(0.0)	0.8(0.1)	0.8(0.0)	0.8(0.0)
RVFLNN	Acc	87.9(2.8)	85.9(5.8)	85.2(3.4)	87.0(5.0)
	g-mean	85.0(5.7)	84.9(4.6)	85.6(3.6)	85.6(6.0)
	$Kappa$	0.8(0.0)	0.8(0.1)	0.8(0.0)	0.8(0.1)

In Satimage, a better classifier performance is observed when the cost functions are used (in table 10). Option 1 reports better accuracy results in models MLP and RVFLNN, and a similar improvement is shown with Option 2 in RBFNN.

Table 9. Classification performance of **Feltwell** with Modular Networks

		Option 0	Option 1	Option 2	Option 3
MLP	Acc	89.8(0.1)	87.2(0.3)	86.8(0.4)	88.6(0.4)
	g-mean	87.1(0.2)	84.7(0.3)	84.3(0.4)	85.9(0.5)
	$Kappa$	0.9(0.0)	0.8(0.0)	0.8(0.0)	0.8(0.0)
RBFNN	Acc	89.5(0.6)	89.0(1.0)	89.0(1.0)	90.4(0.5)
	g-mean	87.4(1.0)	87.1(1.7)	87.0(1.1)	88.2(0.2)
	$Kappa$	0.9(0.0)	0.9(0.0)	0.8(0.0)	0.8(0.0)
RVFLNN	Acc	88.3(2.0)	88.7(1.0)	89.9(1.2)	90.7(0.6)
	g-mean	82.4(4.2)	86.1(1.0)	88.1(1.2)	88.2(1.0)
	$Kappa$	0.8(0.0)	0.9(0.0)	0.8(0.0)	0.8(0.0)

Table 10. Classification performance of **Satimage** with Modular Networks

		Option 0	Option 1	Option 2	Option 3
MLP	Acc	81.8(0.8)	85.4(0.6)	84.7(0.5)	84.8(0.3)
	g-mean	47.5(2.8)	82.8(1.0)	82.8(0.9)	81.4(0.8)
	$Kappa$	0.8(0.0)	0.8(0.0)	0.8(0.0)	0.8(0.0)
RBFNN	Acc	83.9(1.5)	85.4(1.4)	86.2(0.8)	85.0(1.3)
	g-mean	73.9(5.9)	84.0(1.6)	84.4(0.9)	81.5(1.5)
	$Kappa$	0.8(0.02)	0.8(0.0)	0.8(0.0)	0.8(0.0)
RVFLNN	Acc	84.9(1.0)	86.0(0.4)	86.0(0.1)	85.9(1.0)
	g-mean	77.3(1.3)	83.9(0.2)	83.9(0.4)	82.5(1.0)
	$Kappa$	0.8(0.0)	0.8(0.0)	0.8(0.0)	0.8(0.0)

7 Conclusions

In this work, the class imbalance problem for multiple classes is analyzed by means of Modular and Non Modular NN trained with the backpropagation algorithm in batch mode for four different databases. Three strategies have been studied in order to balance the MSE for each class contribution. Each strategy consists of using different kinds of cost functions in the training algorithm.

The proposed strategies improve the Modular and Non Modular NN (implemented for MLP, RBFNN and RVFLNN models) performance over the less representative classes contained in the TS, reducing the class imbalance problem in the training process with the added objective of improving the general performance during the classification phase. But in some cases it does not improve the global results. The most clear cases is the Ecoli6 database, which has a very reduced sample set.

We can not decide which cost function (Option 1, 2, 3) is better: in Non Modular NN the Option 1 has significance statistical improvements in 8.3% of the experiments, Option 2 in about 41.6% and Option 3 in 33.3%. While in the Modular NN we have 33.3% for Option 1 and Option 3, and 16.7% for Option 2.

Future research must consider the relationship between TS imbalance and data complexity (overlapping, noise or decision frontiers), and include the techniques for to re-

duce the data complexity. Also severe TS imbalance (for example in remote perception images) must be further considered.

Acknowledgment

This work has been partially supported by grants DPI2006-15542-C04-03 from the Spanish CICYT , SEP-2003-C02-44225 from the Mexican CONACyT, and Generalitat Valenciana under the project GV/2007/105.

References

1. N. Japkowicz and S. Stephen. The class imbalance problem: a systematic study. *Intelligent Data Analysis*, 6:429–449, 2002.
2. S. Kotsiantis and P. Pintelas. Mixture of expert agents for handling imbalanced data sets. *Annals of Mathematics and Computing & TeleInformatics*, 1(1):46–55, 2003.
3. N.V. Chawla, K.W. Bowyer, L.O. Hall, and W.P. Kegelmeyer. Smote: Synthetic minority over-sampling technique. *J. Artif. Intell. Res. (JAIR)*, 16:321–357, 2002.
4. R. Anand, K. Mehrotra, C.K. Mohan, and S Ranka. Efficient classification for multiclass problems using modular neural networks. *Neural Networks, IEEE Transactions on*, 6(1):117–124, 1995.
5. Z.-H. Zhou and X.-Y. Liu. Training cost-sensitive neural networks with methods addressing the class imbalance problem. *IEEE Transactions on Knowledge and Data Engineering.*, 18:63–77, 2006.
6. L. Bruzzone and S.B. Serpico. Classification of imbalanced remote-sensing data by neural networks. *Pattern Recognition Letters*, 18:1323–1328, 1997.
7. G. Auda and M. Kamel. Modular neural network classifiers: A comparative study. *Journal of Intelligent and Robotic Systems*, 21(2):117–129, 1998.
8. R. Eric and P. Gawthrop. Modular neural networks: a state of the art. Technical Report CSC-95026, Centre for System and Control. Faculty of mechanical Engineering, University of Glasgow, Uk., 1995.
9. R.A. Jacobs, M.I. Jordan, S.J. Nowlan, and S.J. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3:79–87, 1991.
10. C. Looney. *Pattern Recognition Using Neuronal Networks - theory and algorithms for engineers and scientists.* Oxford University Press, New York, 1 edition, 1997.
11. C. Ding, S.Q.and Xiang. From multilayer perceptrons to radial basis function networks: a comparative study. In *EEE. Conference on Cybernetics and Intelligent Systems*, volume 1, pages 69–74, 2004.
12. Y-H. Pao, G.H. Park, and D.J. Sobajic. Learning and generalization characteristics of the random vector functional-link net. *Neurocomputing*, 6(2):163–180, 1994.
13. R. Anand, K.G. Mehrotra, C.K. Mohan, and S. Ranka. An improved algorithm for neural network classification of imbalanced training sets. *IEEE Transactions on Neural Networks*, 4:962–969, 1993.
14. R. Alejo, V. García, J.M. Sotoca, R.A. Mollineda, and J.S. Sánchez. Improving the performance of the rbf neural networks with imbalanced samples. In *9th International Work-Conference on Artificial Neural Networks*, pages 162–169. Springer, 2007.
15. L.I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms*. Wiley-Interscience, July 2004.