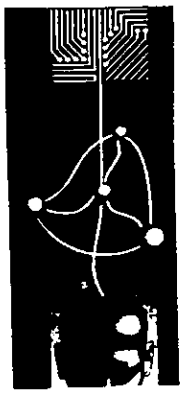

1° Congresso Brasileiro de Redes Neurais

Escola Federal de Engenharia de Itajubá
Itajubá. 24 a 27 de outubro de 1994

Equalization of the Training Set for Backpropagation Networks Applied to Classification Problems

Frederico dos Santos Liporace, Ricardo José Machado and Valmir C. Barbosa

Abstract— **One of the problems faced by multilayer perceptrons trained by the backpropagation algorithm when applied to classification problems is the low sensitivity of the resulting network to classes statistically less represented in the training set. This paper proposes that in such cases it is better to build a modular network, assigning to each independent module the task of recognizing one specific class and rejecting the others. The use of a modular architecture enables the construction of a modified training set for each module that tries to minimize the problem of the less represented classes. This modification or *equalization* assigns a *relevance degree* to each training sample. This gives each sample a different degree of importance, and has the same effect of the replication of some samples in the training set. This technique was applied in an application related to satellite imagery classification, and the obtained results show that the modules trained with the equalized training set exhibit far better results than others trained with the "plain" training set.**

Keywords— **Neural networks, backpropagation, classification.**

I. Introduction

The error backpropagation algorithm is by far the most used to build neural-network based applications nowadays. Contrasting to the ease that it is applied to construct neural networks able to solve "toy problems", there is a great amount of effort involved when designing a larger network required for some practical applications.

This paper describes one problem frequently found during the design of classifiers using multilayer perceptrons trained with the backpropagation algorithm, namely the low sensitivity of the classifier to classes less represented (prevalent) in the training set.

Frederico dos Santos Liporace and Ricardo José Machado are with IBM Rio Scientific Center, Caixa postal 4624, 20001-970, Rio de Janeiro - RJ, Brazil. e-mail:liporace@vnet.ibm.com

Valmir C. Barbosa is with Universidade Federal do Rio de Janeiro, Caixa postal 68511, 21945-970, Rio de Janeiro - RJ, Brazil.

The proposed solution is to build a modular network architecture as well as a different equalizations of the same training set. The resulting equalized training sets are then used to train each module separately.

This modular architecture consists of network modules with independent hidden and output layers, each one assigned to the task of recognizing one particular class and rejecting the others. Besides other advantages discussed later, this modular architecture permits the training set to be suitable equalized for each module, as these are trained separately. This equalization intends to promote the equilibrium between the number of examples favoring the module's class and the number of examples against the module's class. This is achieved by the virtual replication of some examples in the training set.

The training set equalization technique proposed in this paper is discussed in the context of an application related to the interpretation of satellite imagery, but we believe that it may be useful in the design of many others neural network based classifiers. Section II gives a short description of the application and the adopted neural network solution. Section III describes some important characteristics of the set used to train the neural network. Section IV describes the adaptive learning rate procedure employed to train the networks. Section V discusses the advantages of using a modular architecture, and why that was the selected option for use in our system. This Section describes also the procedure used to construct the equalized training set for each module. In Section VI the improvements obtained by the proposed modifications in the training set are presented, and the concluding remarks are made in Section VII.

II. The Classifier System

Our initial problem was to develop a neural-network based classifier to be applied in the automatic interpretation of Landsat-5 satellite images from the Amazon region. The main interest was to identify deforested areas such as road building, mining, agriculture and other kinds of human activity that could affect the environment. A detailed description of the problem, the adopted solution and the results can be found in [1], [2]. Here we present

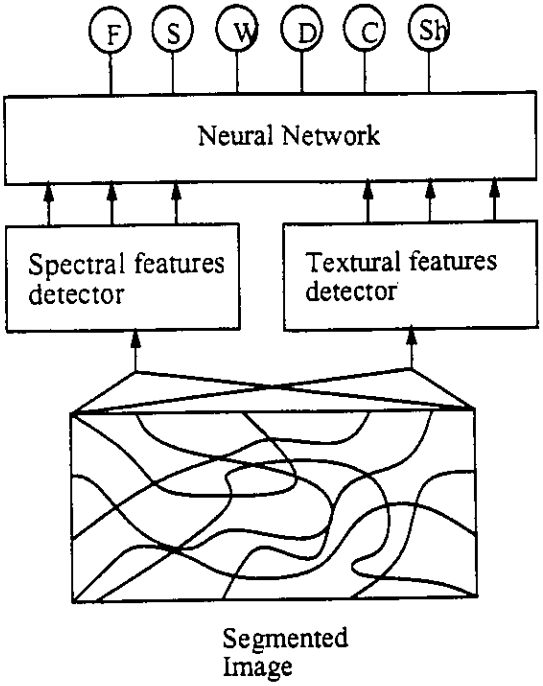

Fig. 1. Schematic of the proposed architecture

a short description of the system necessary to introduce the motivation of the equalization of the training set.

The classifier architecture is shown in Figure 1. In a first pass, the image generated by the satellite is partitioned in segments with homogeneous spectral characteristics. Later, a set of numeric attributes for each segment is calculated. These attributes consists of information related to spectral and textural features of a segment. We may cite as spectral features the gray level average of a segment in each spectral band monitored by the satellite, and as textural features the variance, entropy and correlation of each segment, as defined in [3]. These numerical attributes or *descriptors* of each segment are then presented to a feed-forward neural network trained by the backpropagation algorithm, which outputs the membership of the segment that is being analyzed in each of the defined categories. We employed a fuzzy classification approach, as it is allowed for a segment to have a partial degree of membership to more than one category. The conventional crisp classification is a particular case of the fuzzy one, where the segments are constrained to have a full membership to only one category.

There are four defined categories of interest in our classification problem, namely Forest (F), Savanna (S), Water (W) and Deforestation (D). These categories are called *basic categories*, as they embed all the relevant information to be monitored. There are also two *interfering categories*, namely Shadow (Sh) and Cloud (C), defined to account with the presence of interference caused by clouds and shadows in the images.

Situation	F	S	W	D	C	Sh
Deforested area	0	0	0	1	0	0
Deforested area with incipient reforestation	.25	0	0	.75	0	0
Cloud (opaque)	0	0	0	0	1	0
Tenuous shadow over forest	1	0	0	0	0	.5

TABLE I
EXAMPLES OF VALID CLASSIFICATIONS FOR A SEGMENT

The fuzzy classification approach allows to represent phenomena like the transition between two basic classes, such as the growth of forest in a area that was previously deforested but abandoned later, and the presence of interference like shadows and clouds that still permit the identification of the basic category of the segment, such as the presence of a transparent and thin cloud over a region of forest. Table I shows some examples of fuzzy classifications to clarify the idea.

The membership values of the segments in each category may vary in the interval $[0, 1]$, with 0 indicating empty and 1 indicating full membership of the segment in each category.

III. Characteristics of the Training Set

The database used to train and test the performance of the neural network is composed of five representative images of the Amazon region. These images were segmented by a region-growing technique [3], and the resulting segments were classified by an experienced photo-interpreter using the fuzzy-model approach described in Section II. To simplify the photo-interpreter's task, the allowed membership values were restricted to the set $\{0, 0.25, 0.5, 0.75, 1\}$.

This database is composed of approximately 17 thousand segments, from which two thirds were used to construct the training set of the network and the remaining third was used to measure the generalization ability of the trained network.

Table II shows the distribution of the segments that form the training set in respect to its classifications. Only the basic categories were considered, and a segment was considered to belong to the category that had the greatest membership among all the basic categories. Segments that were classified as having membership only in the cloud or shadow categories were not considered in this table, that's why the percentages doesn't sum up to 100%. As Table II shows, the distribution is highly unbalanced in favor of the F and D categories, leaving far less examples of the W and S ones.

F	S	W	D
38.76 %	2.03 %	3.65 %	40.34 %

TABLE II
DISTRIBUTION OF THE SEGMENTS IN RESPECT TO ITS CLASSIFICATIONS

IV. The Adaptive Backpropagation

The backpropagation algorithm defines an error function between the desired and the actual output of the network and then searches for a set w of synaptic weight values that minimizes this function by a steepest-descent procedure. The error function usually employed is

$$E(\mathrm{w}) = \frac{1}{2}\sum_{\mu=1}^{m}\sum_{i=1}^{n}[D_i^\mu - S_i^\mu]^2 \qquad (1)$$

for a training set of m samples and a network with n neurons in its output layer. We called the actual output of the i^{th} neuron for the μ^{th} sample S_i^μ, and the desired output for the same sample D_i^μ.

The derivatives of Equation 1 in respect to each weight of the network are then calculated and each weight w_{ij} is updated following the equations

$$\Delta w_{ij} = -\alpha\frac{\partial E}{\partial w_{ij}} \qquad (2)$$

$$w_{ij}^{\text{new}} = w_{ij}^{\text{old}} + \Delta w_{ij} \qquad (3)$$

where α is a positive constant known as *learning rate*, selected experimentally.

The backpropagation algorithm calculates the Δw_{ij} terms for each synapse as each sample in the training set is presented to the neural network. Shortly, we have

$$\Delta w_{ij}^p = \alpha\frac{\partial E^p}{\partial w_{ij}} \qquad (4)$$

where p is a sample from the training set and E^p is the error for the p sample. The synapses can be updated after the presentation of each sample (*stochastic* learning), applying the result of Equation 4 in Equation 3. This procedure updates the synaptic weight values to minimize the error for the specific pattern that is being presented to the network at the moment.

Another way to update the synaptic weight values is to accumulate all the Δw_{ij}^p terms for all samples in the training set and use this result in Equation 3, in what is called *batch* learning. This procedure tries to minimize the error for *all* the samples in the training set.

The large size of our training set makes it difficult to find good values for the learning rate parameter by trial and error, due to the long time required to run each trial. It was also verified in our early experiments that values that were good in the beginning of the training process caused instability later [2], which makes this search even harder and requires a continuous monitoring or the error rate evolution. We chose then to use an adaptive learning rate procedure, as suggested in [4].

This procedure can be summarized as follows: the weights are updated after the presentation of all the samples in the training set (batch learning), and the previous synaptic weight values are stored. This way it is possible to "undo" the update. After each update, the behavior of the error evolution is checked. The learning rate parameter is then modified following the rule:

$$\alpha^{\text{new}} = \alpha^{\text{old}} + \Delta\alpha$$

with

$$\Delta\alpha = \begin{cases} +a & \text{if the error decreases consistently} \\ -b\alpha & \text{if the error increases} \\ 0 & \text{otherwise} \end{cases}$$

where a and b are appropriate positive constants.

If the new error value exceeds the previous one, the process overshot and the learning rate is reduced. In our implementation, the synaptic weight values were also restored to the previous ones conveniently saved and the iteration was retried with the new learning rate until it succeeds in reducing the error. In the other way, if the error decreases consistently, the learning rate is increased in hope to achieve a faster convergence. We considered consistent a decrease of the error value for 10 consecutive iterations of the algorithm.

V. The Proposed Solution

The proposed solution to the low sensitivity of the network to classes less represented in the training set is composed of two parts: the first is the construction of a modular network, which permits then the training set for each module to be equalized in a way that the effect of the uneven distribution is reduced.

A. Why a Modular Network?

The first approach that one might consider when designing a neural network for the application described in this paper is to build a structure like that shown in Figure 2. Within this structure, the hidden layer is shared between all the neurons of the output layer, and each neuron on the output layer is assigned to one category.

In the first tests we made with that structure we found that the resulting network had a very low sensitivity to examples belonging to the S and W categories. One possible explanation for that behavior is that the backpropagation algorithm tries to minimize the *global* error function shown in Equation 1.

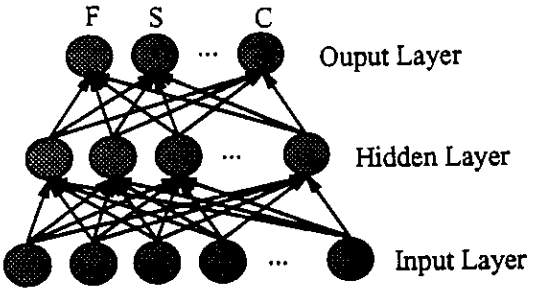

Fig. 2. A non-modular network

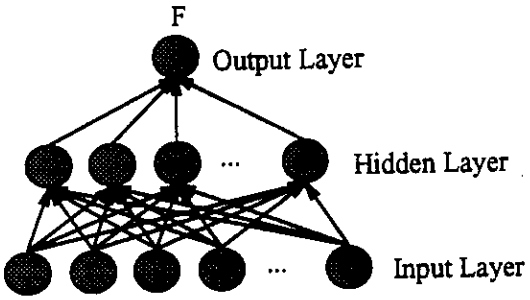

Fig. 3. A module associated to the F category

As the segments from the classes S and W have far less examples than that of the F and D categories, these two first are simply not considered as they have a very low influence on the global error.

Another way to build the network is to use a modular architecture. Figure 3 shows the structure of one of the modules. Each module is associated with a category, and the modules share only the input layer, maintaining independent hidden layers and an output layer with a single neuron. Each module is specialized in detecting one specific category and rejecting the others.

This second architecture has some very distinctive advantages over the first one. As there are no connections between the hidden layers, the modules can be trained separately, possibly in parallel using different machines. Moreover, one can select the modules that have a poor performance and concentrate the training on these modules, without affecting the others. This way, a more efficient use of the available computational power is allowed, specially because we observed that some modules trained significantly faster than others.

B. The Equalized Training Set

Another important advantage of the modular architecture is the possibility to adjust the training set of each module in a way that the effect of the unfair distribution between the classes is minimized. Each module of the network is specialized in detecting one category and rejecting the others. Hence, if we are concerned to one specific network module, we can classify the examples in the training set in two categories: the ones that are *in favor* of the module's class and the ones that are *against* the class, depending of the degree of membership in the module's class assigned to that examples by the photo-interpreter. One possible way to classify one example on the *in favor* or *against* categories is to check if the membership value for that example exceeds or not 0.5.

We can promote the equilibrium between these two antagonist categories by associating to each training set sample p a *relevance degree* $r(p)$. This relevance degree is then used to multiply the α term in Equation 4, yielding to the modified version:

$$\Delta w_{ij}^{p} = \alpha r(p) \frac{\partial E^{p}}{\partial w_{ij}} \qquad (5)$$

We may construct for each module c a set of relevance values r^c that achieve the equilibrium between the *in favor* and *against* samples by setting initially unitary relevance values to all samples and then raise the relevance of samples that belongs to the less represented group, being that the *in favor* or *against* one. We implemented that raising by simply increasing the relevance values for all the samples of the less represented group by one and repeating this pass until the equilibrium is achieved.

Note that when batch training is used the proposed modification is equivalent to $r(p)$ presentations of the p sample during the calculation of the Δw_{ij} terms, that is, it has the same effect of the simple replication of the p sample $r(p) - 1$ times in the training set.

VI. Experimental Results

This section compares the performance of the modules trained with the "plain" training set and with an equalized version of the same training set. We chose to show the results for the W module, since a similar behavior was encountered for all the remaining modules. We implemented the adaptive backpropagation method described in Section IV in an IBM Risc 6000 - Model 560 workstation, and each training run presented required approximately two days to complete.

Figure 4 shows the sensitivity and specificity of the module as the training process is executed. This module was trained without an equalization of the training set. The sensitivity and specificity are measured every 600 epochs, and the dots in the figures represent the values at each measurement. It may be observed from Figure 4 that the values of the sensitivity are very low for this network, starting at 0.04 in the beginning of the training and raising to 0.35, still a low value, after 8400 epochs. There is a clear tendency of the network trained with the "plain" training set to wrongly reject examples that belong to the water category.

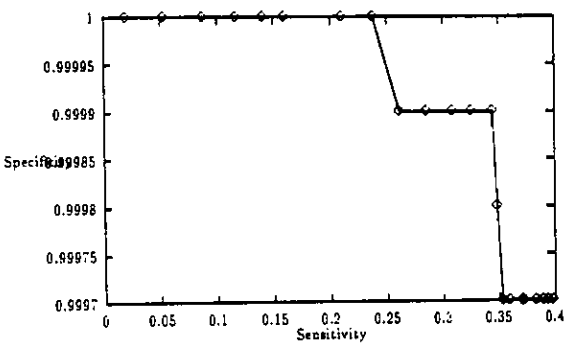

Fig. 4. Evolution of sensitivity and specificity to the W class for the "plain" training set

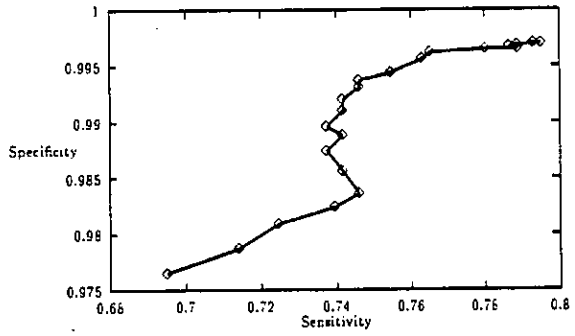

Fig. 5. Evolution of sensitivity and specificity to the W class for the equalized training set

The same module was trained again, now with an equalization of the training set as proposed in Section V-A. The use of the equalized training set clearly yielded to much larger values of sensitivity, as Figure 5 shows. After the first 600 epochs, the sensitivity value was already near 0.7, which is much better than the obtained after 8400 training epochs without the equalized training set. We may also note that the specificity value is still as high as it was before. As the training proceeds, both values still increase.

Another interesting behavior was observed during the training of the D module. Most of the errors made by this module were in S examples, since the spectral characteristics of this category are very similar to that of the D category. We also found that in that cases it is better to assign greater relevance values to S samples, as this makes the error function to be minimized by the backpropagation algorithm more sensitive to errors between S and D categories.

VII. Concluding remarks

One of the problems that the backpropagation algorithm exhibits when applied to classification problems is its low sensitivity to classes less represented in the training set. Our proposed solution to that problem consists in building a modular architecture, in which modules with different hidden layers are assigned the task to detect one class and reject the others. The use of this modular architecture enables the construction of an equalized training set for each module.

The proposed equalization has the same effect of simply replicating examples that belong to one less represented class in the training set, but without a significant increase in the training time.

The technique here described may be applied to fuzzy classification problems in general, and of course apply to the particular case of crisp classification. The learning should be preferably done in batch, as its allows the use of an adaptive learning rate procedure. This adaptive control of the learning rate and the possibility of training the network's modules in parallel on different machines are valuable advantages in complex problems domains that require long training times.

The results we obtained demonstrate that with the proposed modular architecture and the equalization of the training set for each module it is possible to construct networks that are more sensitive to classes less represented in the training set.

References

[1] Valmir Carneiro Barbosa, Ricardo José Machado, and Frederico dos Santos Liporace. A neural system for deforestation monitoring on Landsat images of the Amazon region. *International Journal of Abstract Reasoning*. To appear.

[2] Frederico dos Santos Liporace. Um sistema neural para monitoração do desflorestamento na região Amazônica utilizando imagens do Landsat. Master's thesis, Universidade Federal do Rio de Janeiro, 1994. In Portuguese.

[3] D. H. Ballard and C. M. Brown. *Computer Vision*. Prentice-Hall, 1982.

[4] John Hertz, Anders Krogh, and Richard G. Palmer. *Introduction to the Theory of Neural Computation*. Addison-Wesley, 1991.