Comparison Results Between Usual Backpropagation and Modified Backpropagation with Weighting: Application to Radar Detection

Diego Andina
Escuela Técnica Superior de Ingenieros de Telecomunicación
Universidad Politécnica de Madrid
Madrid, Spain

Aleksandar Jevtić
Escuela Técnica Superior de Ingenieros de Telecomunicación
Universidad Politécnica de Madrid
Madrid, Spain

***Abstract*—This paper presents some relevant results of a novel variant of the Backpropagation Algorithm to be applied during the Multilayer Perceptrons learning phase. The novelty consists in a weighting operation when the MLP learns the weights. The purpose is to modify the Mean Square Error objective giving more relevance to less frequent training patterns and resting relevance to the frequent ones. The inherent statistical distribution of training patterns is used to quantify how frequent a pattern is. The results, applied to a radar detector, show that Backpropagation with Weighting training requires much less training patterns maintaining the Artificial Neural Network performance.**

***KEYWORDS*: Neural Networks, Backpropagation Training Algorithm, Importance Sampling, Binary Detection, Detection Curves.**

I. Introduction

The whole idea of the experiment is to improve the basic BP algorithm [9][10] used to train an Artificial Neural Network (ANN) of the Multilayer Perceptron (MLP) type manipulating the Mean Square Error (MSE) objective function in order to give more relevance to less frequent training patterns and resting relevance to the frequent ones. If the MSE objective function is defined by the following expression:

$$E_{MS} = \varepsilon\left\{(Y - Y_d)^2\right\} \qquad (1)$$

where the random variable $Y = g(X)$ is the neural network output and X is a random variable of the training input vectors $\overline{x} = (x_1, x_2, ... x_n)$, ($\overline{x} \in R^n$), where R^n is the n-dimensional space. Y_d represents the desired output. From statistical inference theory applied to Eq. (1), an estimator of E_{MS} is given by [1]:

$$\hat{E}_{MS} = \frac{1}{N}\sum_{k=1}^{N}\frac{e(x_k^*)}{f_X^*(x_k^*)} \qquad (2)$$

where $x_k^*, k = 1, 2 ... N$, are independent sample vectors whose *pdf* is $f_X^*(x)$, and $e(\cdot)$ is the error as a function of the training inputs applied in MLP training to update the weights in each training iteration step. $f_X^*(x)$ is ideally given by [1]:

$$(f_X^*(x))_{opt} = \frac{1}{E_{MS}} e(x) \qquad (3)$$

and it is not possible to be known *a priori* because E_{MS} is not known and $e(\cdot)$ is changing in each iteration. Nevertheless, the suboptimal solutions can be tested, if $f_X^*(x) \neq 0$ wherever $e(x) \neq 0$, $\forall x \in R^n$.

II. Weighting Operation

The scheme in Fig. 1 represents the training cycle when applying the weighting function.

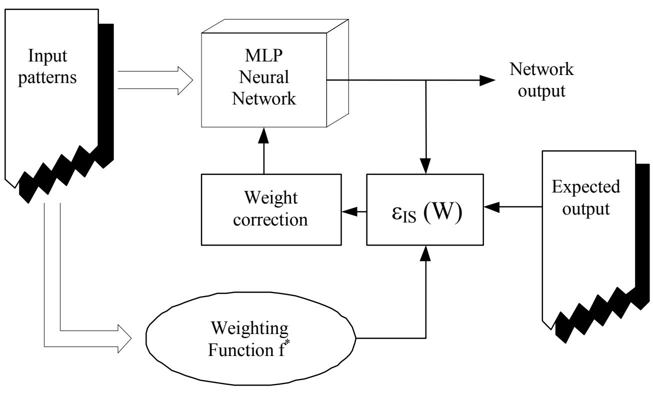

Fig. 1. Weighted Training Cycle

For weighting, we have tested two different functions:

$$f_{\overline{X}}^{*}(\overline{x}) = \frac{A}{\sqrt{(2\pi)^{N_1}} \cdot e^{\frac{B}{8}\sum_{i=1}^{8} x_i^2}}, \qquad (4)$$

and

$$f_{\overline{X}}^{*}(\overline{x}) = \frac{0.01}{\hat{y}}. \qquad (5)$$

Eq. (2) in fact shows that the MSE can be achieved if we divide the error function *e(·)* by a weighting function $f_X^*(x)$. In Eq. (4) a Gaussian function is proposed as weighting function supposing that the inputs also have Gaussian distribution. In Eq. (5) the advantage is taken from the inherent *a posteriori probabilities* estimation of the MPL output.

III. Computer Results

Experiments have been carried out in order to evaluate the Backpropagation with Weighting (BPW) [5][6][7] algorithm. The main objective of these experiments is the evaluation of the weighting function capabilities and limits. We present the results obtained from training of 100 Neural Networks (NNs) using a BPW algorithm consisting in Least Mean Square (LMS) criterion modified by the proposed weighting functions.

A. General Characteristics of the Experiments

The ANNs used are MLPs with structure 16/8/1 (that is 16 inputs, and one hidden layer of 8 units). The choice of the structure and the rest of parameters of the network was the optimal solution for the given example application [4]. The activation function is sigmoidal with scalar output in the range (0,1) and, it is the same for all the neurons.

For the training of the network we used balanced patterns of two classes, being class H_0 noise patterns and being class H_1 signal received with additive Gaussian noise. These patterns configure the problem of signal detection noise and the ANN acts as a binary detector. The application of the ANN is an elemental radar detection problem [4] when the basic parameter for the patterns is the Signal to Noise ratio, *SNR*, and the performance of the detectors is evaluated in terms of the Neyman-Pearson criterion. That is, maximizing probability of detection, P_d, (the probability of classifying correctly the patterns belonging to the class H_1) for a fixed false alarm probability, P_{fa} (the probability of classifying erroneously the patterns belonging to the class H_0). In the radar literature, performance is evaluated through the Detection curves (P_d vs. *SNR*), so we use these detection curves to present the results of our method.

In our previously conducted experiments the training of a network was limited to the error probability value in range of 0.1–0.2. Fig. 2 shows an example of NN training only using weighting function (4). As we can notice, classification error reached the value of 0.125, and this NN could not be considered completely trained. For this reason, the weighting function (1) was applied until the critical error probability value was reached, and from that point the weighting function was changed to (2). The function (5) is not valid until the output of the network is a sufficiently good approximation of the *a posteriori* probabilities of the inputs. In the first iterations, it can be $\hat{y} = 0$, and the NN stops learning. We conducted two experiments with the different critical error probability values: 0.2 and 0.15.

In each experiment 100 networks were trained in order to achieve mean results that does not depend on initial random

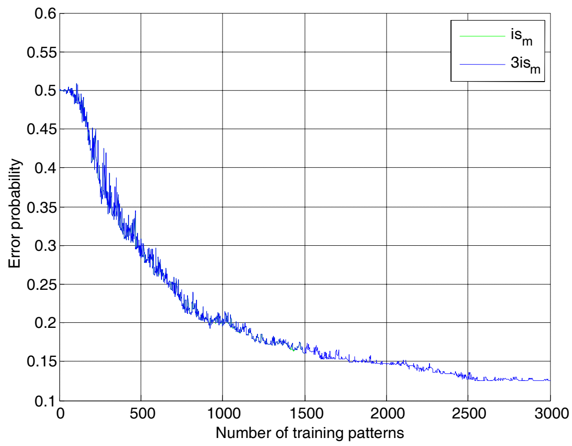

Fig. 2. Classification error in training phase with only one weighting function.

value of the weights of the ANN. Two different criterions were applied to stop the training: in one case it was stopped when the error reached zero (denoted as is_m) and in the other the training was conducted with a fixed number of 3000 patterns ($3is_m$).

As usual [2], three set of patterns have been used to design the network. A training set (composed of patterns of SNR=13.2 dB for class H_1), a test set to calculate the error during training and a validation set to obtain the detection curves.

B. Critical error probability 0.2

Fig. 3 shows the error evolution during the network training phase, calculated as the rate of misclassified patterns of the training set out of the total number of patterns. We can notice that the combination of the proposed weighting functions in this experiment made possible to override the threshold of error of 0.2 where the training was stopped when using only one function.

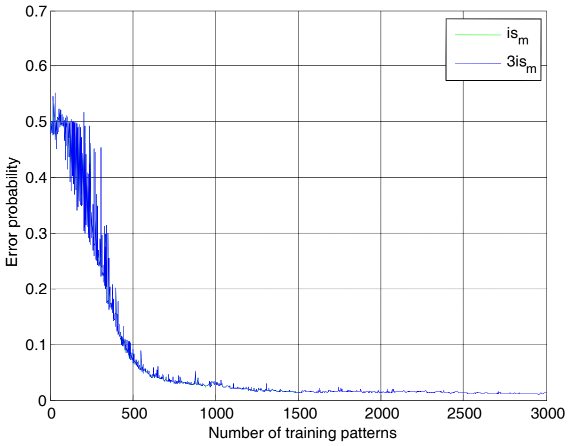

Fig. 3. Classification Error in Training Phase, Threshold 0.2.

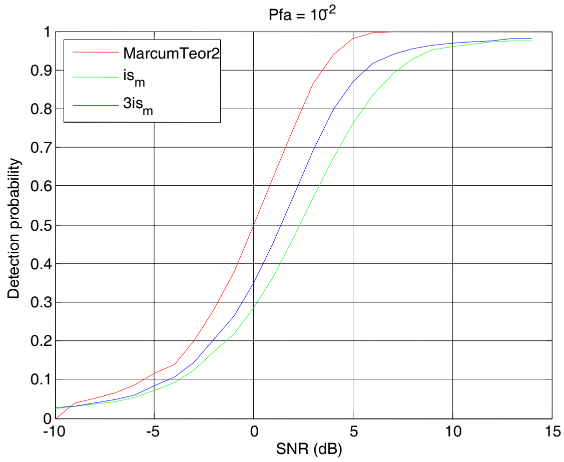

Fig. 4. Detection Probability, Pfa=10^{-2}, Threshold 0.2.

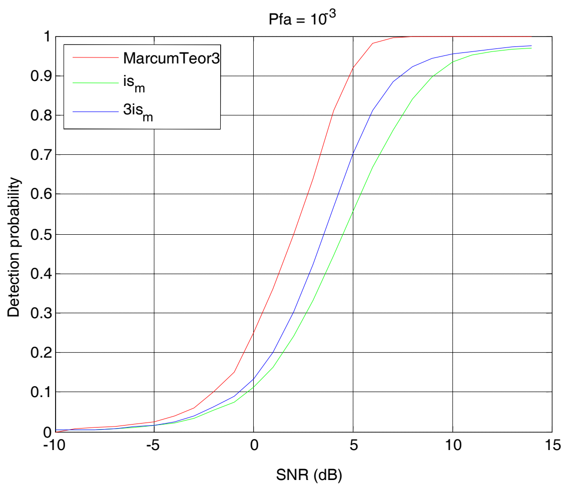

Fig. 5. Detection Probability, Pfa=10^{-3}, Threshold 0.2.

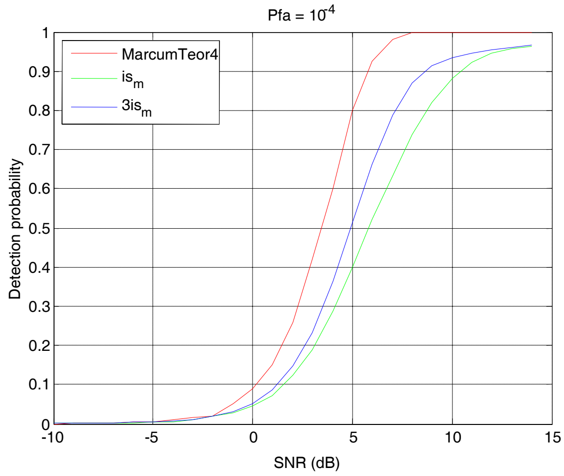

Fig. 6. Detection Probability, Pfa=10^{-4}, Threshold 0.2.

The detection probability for three different false alarm probability (probability of "decide H_0 when input corresponds to H_1") values related to the SNR are shown in Fig. 4, 5 and 6, respectively. The red line represents the theoretical maximum by Marcum theorem [8]. The green line represents average performance for the networks that were trained until the error probability reached zero and the blue line is used for the networks trained with the fixed number of patterns. False alarm probabilities, P_{fa}, of 10^{-2}, 10^{-3} and 10^{-4} have been considered. For the detection probability that corresponds to the false alarm probability of 0.01, we find that the results are noticeably better if the NNs were trained with the fixed number of patterns (3000) for all the values in relation to the SNR between 0 and 8 dB.

In the case of false alarm probability of 0.001 and 0.0001 we get better results for training a network with the fixed number of patterns and the curve (blue) is much closer to the theoretical one (red). For the high SNR values the results could be improved, which could make a part of the future lines of investigation.

C. Critical error probability 0.15

Fig. 7 shows the results obtained for setting the threshold for changing the weighting functions at 0.15. Again, we considered two criterions for stopping the training of a network, when error reaches zero and with the fixed number of patterns.

We can see that the decision to change the weighting function when the threshold 0.15 was reached gave the satisfying results because the training continued lowering the error value. Fig. 8, 9 and 10 show characteristics of trained networks for false alarm probabilities, P_{fa}, of 10^{-2}, 10^{-3} and 10^{-4}. The results obtained are better in the case of training a network with the fixed number of patterns, as it was with the threshold of 0.2.

Finally, in both cases training continued over the limiting value detected using only one weighting function.

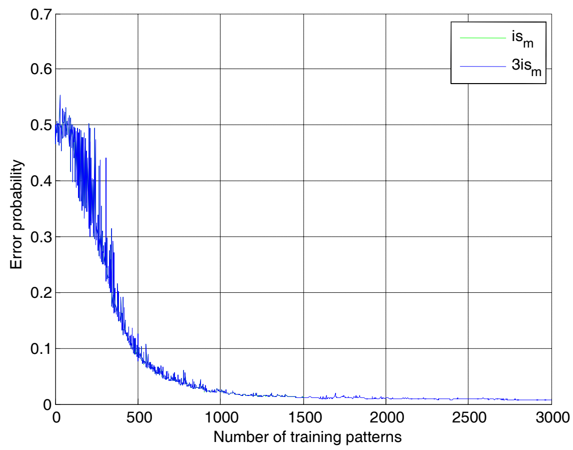

Fig. 7. Classification Error in Training Phase, Threshold 0.15.

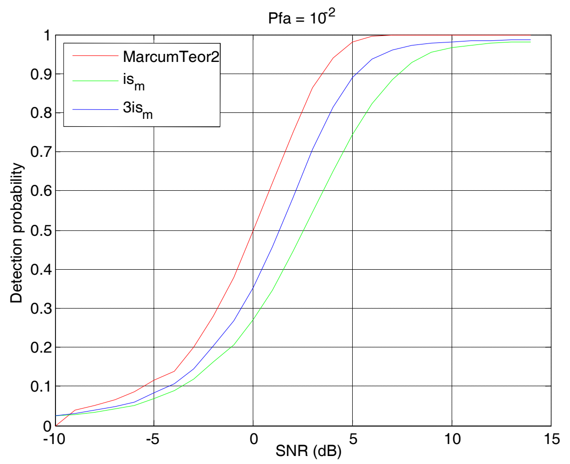

Fig. 8. Detection Probability, Pfa=10^{-2}, Threshold 0.15

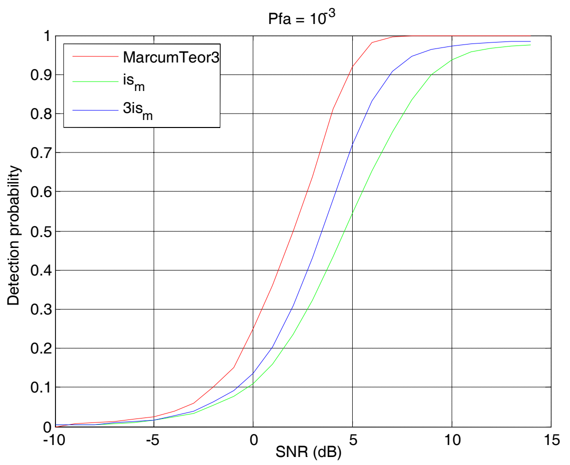

Fig. 9. Detection Probability, Pfa=10^{-3}, Threshold 0.15

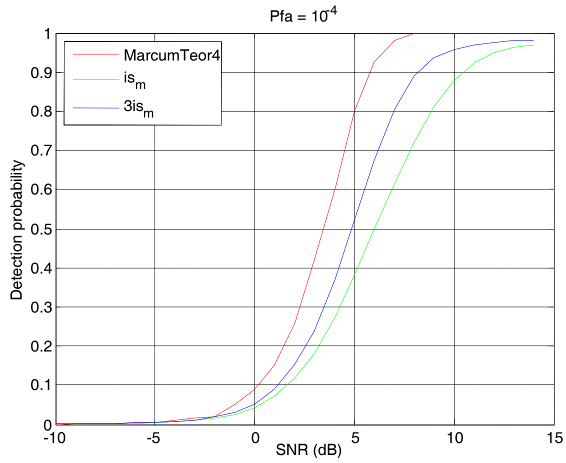

Fig. 10. Detection Probability, Pfa=10^{-4}, Threshold 0.15.

D. *The Best Obtained Network*

The error probability evolution of the best network obtained is shown in Fig. 11. Only 355 iterations were needed to reach the zero classification error. We can see that the network has a rapid error evolution to the zero value, with a low number of iterations. This allows us to save time and resources. The threshold for changing the weighting function was set to 0.2.

Fig. 12, 13 and 14 show the characteristics of trained network for false alarm probabilities, P_{fa}, of 10^{-2}, 10^{-3} and 10^{-4}. We can see that the distance between two curves is less than 1 dB. Even though the number of iterations used was small, we can conduct the training with fixed number of patterns and get values even closer to the theoretical maximum. These results demonstrate, one more time, the performance of NNs achieved by training with the small number of iterations using BPW criterion with two weighting functions. We generated NNs with similar or better characteristics than those obtained using BPW with only one weighting function or the classical BP. From this last experiment we just may extract some conclusions about the performance a neural detector trained by BPW might reach in the most favorable conditions.

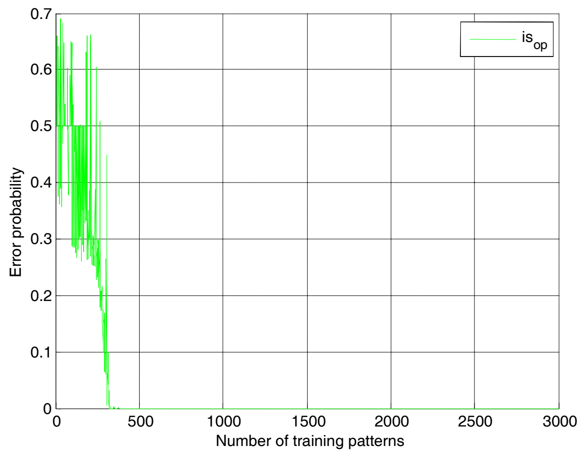

Fig. 11. Classification Error in Training Phase, Threshold 0.2, The Best Case

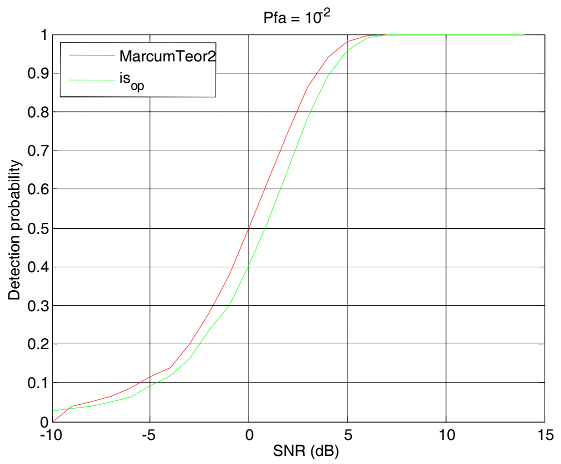

Fig. 12. Detection Probability, Pfa=10^{-2}, Threshold 0.2, The Best Case

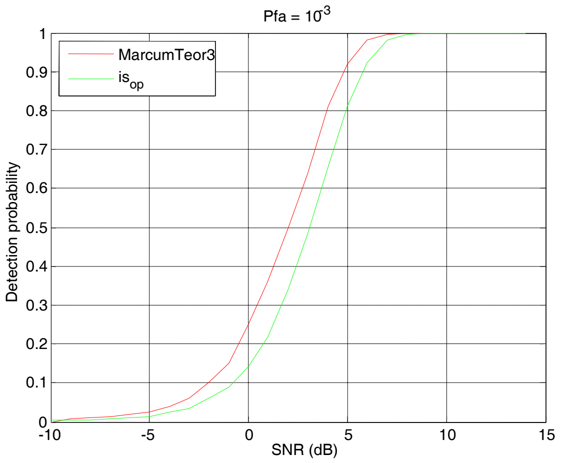

Fig. 13. Detection Probability, Pfa=10^{-3}, Threshold 0.2, The Best Case

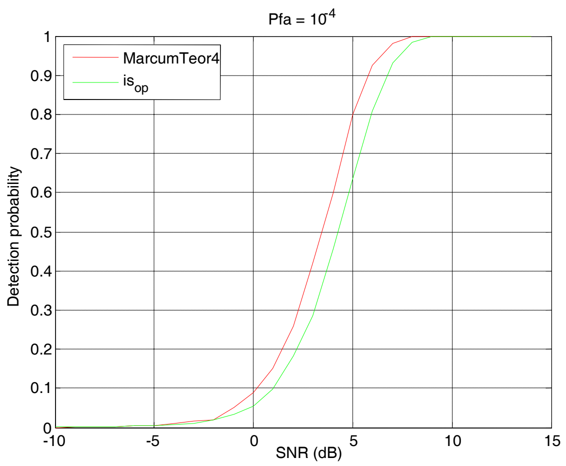

Fig. 14. Detection Probability, Pfa=10^{-4}, Threshold 0.2, The Best Case

IV. Conclusions

The results of the experiments presented in this paper show a drastic reduction in the number of training patterns (one order of magnitude) for Backpropagation algorithm optimized by using Weighting techniques. Combination of two weighting functions makes possible to override the critical value of error probability. In many practical applications, when few patterns are available for NN training, the proposed Weighting technique could be extremely useful.

Acknowledgment

The authors wish to thank Mr. Juan Fombellida, graduated student from Technical University of Madrid, Spain, for his work in the realization of the experiments.

References

[1] Andina D, Martínez-Antorrena J, Melgar I. “Importance Sampling in Neural Detector Training Phase”, *Computing with Industrial Apllications*, TSI Press Series on Intelligent Automation and Soft Computing Vol. 17, Albuquerque NM, USA September 2004, pp. 309 - 313.

[2] Hu, Y.H., Hwang, J.N. “Introduction to Neural Networks for Signal Processing” *Handbook of Neural Network Signal Processing*, CRC press, Boca Ratón, FL, USA, 2002, pp. 12-41.

[3] Andina D, Torres-Alegre S, Vega-Corona A, Álvarez-Vellisco A. “Advances in Neyman-Pearson Neural Detectors Design” in *Lecture Notes in Computer Science*, Núm. 2686, Editors: J. Mirá, J.R. Álvarez, Springer-Verlag, Berlin-Heidelberg, 2003, pp. II-249-256.

[4] Andina D, Ballesteros F (eds). *Recent Advances in Neural Networks*. Ed. International Institute of Informatics and Systemics, IIIS press, Illinois, 260 pp. USA, 2000. ISBN 980-07-7261-8.

[5] Smith P.J & Gao, H. “Quick simulation: A review of importance sampling techniques in communications systems” IEEE J. Select. Areas Commun., Vol. 15, pp. 597-613, May 1997.

[6] Gerlach, K. “New results in importance sampling” IEEE Trans. Aerosp. Electr. Systems, Vol. 35, pp. 917-925, July 1999.

[7] Maqbool Aliani Geoffrey C. Orsak, Behnaam Aazhang. “On the use on importance sampling in the optimization of classification systems”. George Mason University, www.cse.iitd.ernet.in/~sandeepj/avail_papers/introsmall.ps.

[8] Marcum J. “A statistical theory of target detection by pulsed radar”, IEEE Transactions on Information Theory, Vol. 6, Issue 2, pp: 59-267, April 1960.

[9] RumelHart, Hinton, G.E. & Williams, R.J. “Learning representations by back-propagating errors” Nature, 323, 533-536, 1986.

[10] Phansalkar, V.V. & Sastry, P.S. “Analysis of the back-propagation algorithm with momentum”. IEEE Trans. On Neural Networks, vol. 5, Nº 3, pp 505-506, May 1994.