A Robust and Effective Learning Algorithm for Feedforward Neural Networks Based on the Influence Function[*]

Héctor Allende[1,3], Rodrigo Salas[1], and Claudio Moraga[2]

[1] Universidad Técnica Federico Santa María
Dept. de Informática;
Casilla 110-V; Valparaíso-Chile;
{hallende, rsalas}@inf.utfsm.cl

[2] University of Dortmund; Department of Computer Science;
D-44221 Dortmund; Germany; moraga@cs.uni-dortmund.de

[3] Universidad Adolfo Ibañez; Facultad de Ciencia y Tecnología

Abstract. The learning process of the Feedforward Artificial Neural Networks relies on the data, though a robustness analysis of the parameter estimates of the model must be done due to the presence of outlying observations in the data. In this paper we seek the robust properties in the parameter estimates in the sense that the influence of aberrant observations or outliers in the estimate is bounded so the neural network is able to model the bulk of data. We also seek a trade off between robustness and efficiency under a Gaussian model. An adaptive learning procedure that seeks both aspects is developed. Finally we show some simulations results applied to the RESEX time series.

KEYWORDS: Feedforward Artificial Neural Networks, Robust Learning, Effective parameter estimate.

1 Introduction

In the last decades there have been a widespread interest in the use of artificial neural networks (ANN) in many different problems ranging from pattern classification to control engineering. A very important and widely applicable class of ANN models are the feedforward artificial neural networks (FANN) because they have been remarked as universal approximators of continous, bounded, nonlinear functions that can be trained from examples of input-output data.

The ANN are seen by researches as either highly parameterized models or nonparametric structures. ANN models are flexible, and have a demonstrated success in a variety of applications in which linear models fail to perform well. A statistical analysis has been made by considering ANN as nonlinear regression models and by casting network learning as a statistical estimation problem [6], [10].

[*] This work was supported in part by Research Grant Fondecyt 1010101 and 7010101, in part by Research Grant DGIP-UTFSM, and in part by Research Grant CHL-99/023 from the German Ministry of Education and Research (BMBF).

The learning algorithm for the parameters estimation of the neural model relies on the data. When the data are contaminated with outliers, for example, observations that are substantially different to the bulk of data due to gross errors, they can influence badly bringing degradation in the estimates [1], [2], [5]. Full effectiveness can be achieved only when the data agree with the assumptions underlying the data generating process but not when deviations from them occurs (See [8]), aspect widely investigated in statistics, but very poorly in the ANN literature [2], [3], [4], [5]. In this work we seek a compromise in terms of statistical efficiency and robustness of learning procedures by applying M estimators introduced by Huber [8].

This paper is organized as follows. In section 2, we introduce the notation and architecture of the feedforward neural networks (FANN). In section 3, we develop a robust analysis of the Learning Algorithm for the FANN, and then we propose a robust and effective estimator for the FANN parameters (weights). We will give simulation results in section 4 where the procedure is applied to Time Series modeling, and a comparative analysis based on the performance of the different learning algorithms is made. Concluding remarks and future extensions are presented in section 5.

2 Feedforward Artificial Neural Networks

A FANN consists of elementary processing elements called neurons, organized in three type of layers, the input, the output and the hidden layers, where the latter is located between the input and the output layers. The number of input and output units are determined by the application. The links of the neurons are from one layer to the successive without any type of bridge, lateral or feedback connections. For simplicity, a single-hidden-layer architecture is considered in this paper, consisting in only one hidden layer and one output neuron, this class of neural models can be specified by the number of hidden neurons by $S_\lambda = \{g_\lambda(\underline{x}, \underline{w}) \in \mathbb{R},\ \underline{x} \in \mathbb{R}^m,\ \underline{w} \in \mathcal{W}\}$, where $\mathcal{W} \subseteq \mathbb{R}^d$, $g_\lambda(\underline{x}, \underline{w})$ is a non-linear function of $\underline{x}$ with $\underline{w} = (w_1, w_2, ..., w_d)^T$ being its parameter vector, λ is the number of the hidden neurons and $d = (m+2)\lambda + 1$ is the number of free parameters. The results presented in this paper can be easily extended to FANN with a higher number of layers and output neurons.

Given the sample of observations, the task of neural learning is to construct an estimator $g_\lambda(\underline{x}, \underline{w})$ of the unknown function $\varphi(\underline{x})$ by

$$\hat{y} = g_\lambda(\underline{x}, \underline{w}) = \gamma_2 \left(\sum_{j=1}^{\lambda} w_j^{[2]} \gamma_1 \left(\sum_{i=1}^{m} w_{ij}^{[1]} x_i + w_{m+1,j}^{[1]} \right) + w_{\lambda+1}^{[2]} \right) \tag{1}$$

where $\underline{w}$ is a parameter vector to be estimated, γ_s' are linearity or non-linearity and λ is a control parameter (number of hidden units). An important factor in the specification of neural models is the choice of the 'activation' function γ_s', these can be any non-linearity as long as they are continuous, bounded and differentiable. The activation function of the hidden neurons γ_1 typically is a squashing or a radial basis function. A special type of squashing function is the logistic function $\gamma_1(z) = [1 + exp\{-z\}]^{-1}$ and is one of the most commonly used. For the output neuron the function γ_2 could be a linear function $f(z) = z$, or squashing function.

The estimated parameter $\underline{\hat{w}}_n^{LS}$ is obtained from the sample $\{\underline{x}^t, y_t\}_{t=1..n}$ of size n by minimizing iteratively a loss function $L_n(\underline{w})$, given for example by the ordinary least squares function (2). The loss function gives us a measure of accuracy with which the estimated model fits the observed data.

$$L_n(\underline{w}) = \frac{1}{2n} \sum_{t=1}^{n} \left(y_t - g_\lambda(\underline{x}^t, \underline{w})\right)^2 \tag{2}$$

3 Robust Analysis of FANN

In some earlier works is shown that FANN models are affected with the presence of outlying observations, in the way that the learning process and the prediction have a very poor performance (See [2], [5], [9]).

Let the data set $\chi = \{\underline{x}^t, y_t\}_{t=1..n}$ consists of an independent and identically distributed (i.i.d) sample of size n coming from the probability distribution $F(\underline{x}, y)$. A nonlinear function $y = \varphi(\underline{x})$ is approximated from the data by a feedforward artificial neural network, i.e., $y = g_\lambda(\underline{x}, \underline{w}^*) + r$, where $y \in \mathbb{R}$ is the desired output, $\underline{x} \in \mathbb{R}^m$ is the input vector, $\underline{w}^* \in \mathcal{W} \subset \mathbb{R}^d$ is the unknown parameters vector and $r \in \mathbb{R}$ is the residual error.

Assuming that $\mathcal{W}$ is an open convex set and r_t are independent to the $\underline{x}^t$, $t = 1..n$, with symmetric density $h(r/\sigma_r)$, where $\sigma_r > 0$ is the scale parameter and $k(\underline{x})$ is the density function of the $\underline{x}$, then the joint density function $f(\underline{x}, y)$ is given by $f(\underline{x}, y) = \frac{1}{\sigma_r} h\left(\frac{y - g_\lambda(\underline{x}, \underline{\hat{w}}_n)}{\sigma_r}\right) k(\underline{x})$.

An M-estimator $\underline{\hat{w}}_n^M$ is defined by $\underline{\hat{w}}_n^M = arg\ min\{RL_n(\underline{w}) : \underline{w} \in \mathcal{W}\}, \mathcal{W} \subseteq \mathbb{R}^d$, where $RL_n(\underline{w})$ is a robust functional cost given by the following equation,

$$RL_n(\underline{w}) = \frac{1}{n} \sum_{t=1}^{n} \rho\left(y_t - g_\lambda(\underline{x}^t, \underline{w})\right) \tag{3}$$

where ρ is the robust function that introduces a bound to the influence due to the presence of outliers in the data. Assuming that ρ is differentiable whose derivative is given by $\psi(r, \underline{w}) = \frac{\partial \rho(r, \underline{w})}{\partial r}$, an M-estimator $\underline{\hat{w}}_n^M$ can be defined implicitly by the solution of the following equation,

$$\sum_{t=1}^{n} \psi\left(\frac{y_t - g_\lambda(\underline{x}_t, \underline{w})}{\sigma_r}\right) Dg_\lambda(\underline{x}_t, \underline{w}) = \underline{0} \tag{4}$$

where $\psi : \mathbb{R} \times \mathcal{W} \to \mathbb{R}$, $r_t = y_t - g_\lambda(\underline{x}^t, \underline{\hat{w}}_n^M)$ is the residual error and

$$Dg_\lambda(\underline{x}, \underline{w}) = \left(\frac{\partial}{\partial w_1} g_\lambda(\underline{x}, \underline{w}), \ldots, \frac{\partial}{\partial w_d} g_\lambda(\underline{x}, \underline{w}) \right)^T \tag{5}$$

is the gradient of the FANN. We will denote $Dg_\lambda = Dg_\lambda(\underline{x}, \underline{w})$ for short.

3.1 The Influence Function of the M-estimator of the FANN

In order to study the robustness and effectiveness aspect of the M-estimator, we should analyze the *influence function* (IF) . The IF is a local measure introduced by Hampel [7] and describes the effect of an infinitesimal contamination at the point $(\underline{x}, y)$ on the estimate.

The IF of the M-estimator applied to the FANN model, $\underline{\hat{w}}_n^M$, and calculated at the distribution function $F(\underline{x}, y)$ is given by the following equation,

$$IF(\underline{x}, r; \underline{w}, F) = \psi(r, \underline{w}) M^{-1} Dg_\lambda(\underline{x}, \underline{w}) \tag{6}$$

where r is the residual, $Dg_\lambda(\underline{x}, \underline{w})$ is given by equation (5) and

$$M = \int_{\mathbb{R}} (\psi'(r, \underline{w}) Dg_\lambda Dg_\lambda^T - \psi(r, \underline{w}) D^2 g_\lambda) dF(\underline{x}, y) = \mathbb{E}_F[H(r, \underline{x}, \underline{w})] \tag{7}$$

where $H(r, \underline{x}, \underline{w}) = (\psi'(r, \underline{w}) Dg_\lambda Dg_\lambda^T - \psi(r, \underline{w}) D^2 g_\lambda)$ is the Hessian of $\rho(\cdot)$ with respect to the parameters $\underline{w}$ and $D^2 g_\lambda = [\frac{\partial^2 g_\lambda(\underline{x}, \underline{w})}{\partial \underline{w}_i \partial \underline{w}_j}]$ is the Hessian matrix of the FANN of side $d \times d$. In practice, M is not observable and must be estimated, White [10] demonstrated that a consistent estimator of M is $\hat{M}_n = \frac{1}{n} \sum_{t=1}^n H(r_t, \underline{x}_t, \underline{\hat{w}}_n^M)$, where $\underline{\hat{w}}_n^M$ are the parameters obtained from the data by the minimization of the risk function (3). With this result, we can estimate the influence at the point $(\underline{x}^*, y^*)$ by $\hat{IF}(\underline{x}^*, r^*; \underline{\hat{w}}_n^M) = \psi(r^*, \underline{\hat{w}}_n^M) \hat{M}_n^{-1} Dg_\lambda(\underline{x}^*, \underline{\hat{w}}_n^M)^T$.

3.2 Analyzing the Gaussian Case

As a special case we studied the case when the residual distribution is the standard normal, i.e., $h(r/\sigma_r) = \phi(r/\sigma_r)$, so the density function is given by $f(\underline{x}, y) = \phi(r/\sigma_r) k(\underline{x})$. If we assume a Gaussian model for the residuals and ψ is odd, then the second term in (7) can be neglected, so we get:

$$M = \mathbb{E}[\psi'] \mathbb{E}[Dg_\lambda Dg_\lambda^T] = \left(\int_{\mathbb{R}} \psi'(r) d\Phi(r) \right) \left(\int_{\mathbb{R}} Dg_\lambda Dg_\lambda^T dK(\underline{x}) \right) \tag{8}$$

From the equation (6) and (8) we can realize that the IF can be decomposed as the product of two factors, one dependent on the residual known as *residual influence* (IR) and the *influence due to the position* (IP), obtaining the *total influence*:

$$IF(\underline{x}, r; \underline{w}, F) = IT(\underline{x}, r; \underline{w}, F) = IR(r; \underline{w}, \Phi) IP(\underline{x}; \underline{w}, K) \tag{9}$$

where $IR(r;\underline{w},\Phi) = \psi(r)/\mathbb{E}[\psi'(r)]$ and $IP(\underline{x};\underline{w},K) = (\mathbb{E}[Dg_\lambda Dg_\lambda^T])^{-1}Dg_\lambda$.

An important summary value based on the IF is the *gross error sensitivity* that measures the worst (approximate) influence which a small amount of contamination of fixed size can have on the value of the estimator. The gross outlier sensitivity is defined as $\gamma_u^*(\underline{\hat{w}}_n^M, F) := \sup_{\underline{x},r}\{\|IF(\underline{x}, r; \underline{\hat{w}}_n^M, F)\|\}$. It is a desirable feature that $\gamma_u^*(\underline{\hat{w}}_n^M, F)$ be finite obtaining a *B-robust* estimator.

In the Gaussian case, the gross error sensitivity is due to the residual and the position influence, i.e., $\gamma_u^*(\underline{\hat{w}}_n^M, F) = \sup_{\underline{x},r}\{|IR(r;\underline{\hat{w}}_n^M,\Phi)|\|IP(\underline{x};\underline{\hat{w}}_n^M,K)\|\}$. When the classical Least Square estimator is used, the gross error sensitivity is $\gamma_u^*(\underline{\hat{w}}_n^{LS}, F) = \sup_{\underline{x},r}\{|r|\|IP(\underline{x};\underline{\hat{w}}_n^{LS},F)\|\} = \infty$, i.e., this procedure lacks of robustness by being sensible to the contamination in the residual error that relies on the data.

To obtain a B-robust estimator we analyzed the influence due to the position (IP) and the influence due to the residual (IR). The influence due to the position is bounded ($sup_x\{\|IP(x;\underline{w},K)\|\} < \infty$), when γ_1 is a logistic function and γ_2 is a logistic or a linear function, because the gradient of the FANN model, $Dg_\lambda(\underline{x},\underline{w})$, has four types of derivatives:

$$\frac{\partial g_\lambda(\underline{x},\underline{\hat{w}}_n)}{\partial w_{ij}^{[1]}} = \gamma_2'(\cdot)w_j^{[2]}\gamma_{1,j}'(\cdot)x_i, i = 1..m, j = 1..\lambda;$$
$$\frac{\partial g_\lambda(\underline{x},\underline{\hat{w}}_n)}{\partial w_{m+1,j}^{[1]}} = \gamma_2'(\cdot)w_j^{[2]}\gamma_{1,j}'(\cdot), j = 1..\lambda;$$
$$\frac{\partial g_\lambda(\underline{x},\underline{\hat{w}}_n)}{\partial w_j^{[2]}} = \gamma_2'(\cdot)\gamma_{1,j}(\cdot), j = 1..\lambda; \text{ and,}$$
$$\frac{\partial g_\lambda(\underline{x},\underline{\hat{w}}_n)}{\partial w_{\lambda+1}^{[2]}} = \gamma_2'(\cdot),$$

because γ_2' is a constant if γ_2 is linear and $max_z\{\gamma_2'(z)\} = 0.25$ if γ_2 is a logistic function. Similarly for γ_1'. These factors decrease faster than x_i grows. The influence due to the residual must satisfy that $\sup_r\{|IR(r;\underline{\hat{w}}_n^M,\Phi)|\} = \sup_r\{|\psi(r)/\mathbb{E}[\psi'(r)]|\} < \infty$ to obtain a robust learning estimator that is insensible to the presence of outlying observations. For example the Huber and the Bisquare functions, given by equation (10) and (11) respectively, satisfy these requirements.

$$\psi_H(r,c) = sgn(r)min\{|r|,c\} \tag{10}$$

$$\psi_B(r,c) = \begin{cases} r(1-(r/c)^2)^2 & r \in [-c,c] \\ 0 & r < -c \;\; or \;\; r > c \end{cases} \tag{11}$$

By putting a bound on γ^* will often conflict with the aim of asymptotic effectiveness. In order to obtain an effective and robust estimator the value of the constant c of the $\psi - function$ should be estimated.

It is a well know fact that the LS estimator is the most effective estimator of the mean under a Gaussian model. By assuming that the residual has a distribution close to the Gaussian and outliers should appear in regions further than $3\sigma_r$. By assuming that $E[r] = 0$ and by considering a robust estimation of σ_r given by $\sigma_r = 1.483 median\{|r - median\{r\}|\}$, we should look for a constant c such that the distance between $\psi.(r^*, c)$ and $\psi_{LS}(r^*) = r^*$ is not bigger than

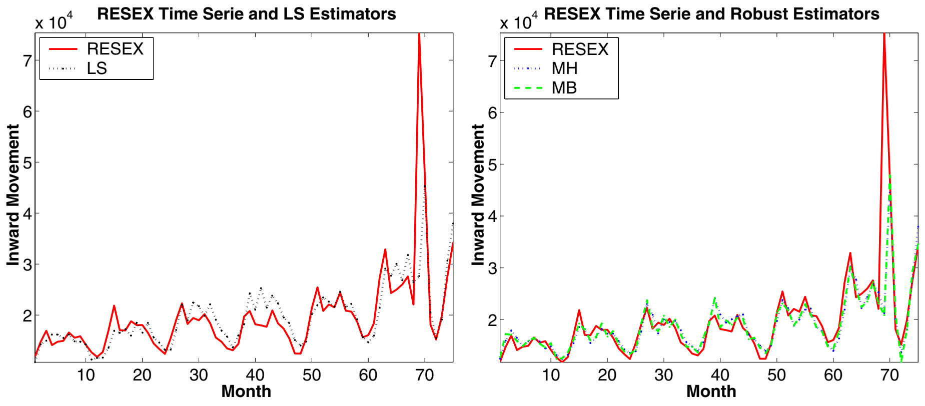

Fig. 1. a)(left) LS estimator. b)(right) Robust Learning Algorithm

a desired constant $k > 0$ given at some point r^* obtaining almost full efficacy inside the $[-r^*, r^*]$ region, and outside that point, the robust estimator start to have less efficacy than the LS estimator under Gaussianity.

The value of the constant c could be obtained analytically or by numeric methods, for example, in the Huber case, we choose $k = 0$, so the value of the constant is $c = r^*$ and for the Bisquare case, we take some small value for k and after some calculations and by using the absolute value as the distance metric, i.e., $|\psi_B(r, c) - r| = k$, we obtained $c = r^*/(\sqrt{1 - \sqrt{1 - k/r^*}})$.

Due to the fact that the estimation process is an adaptive learning algorithm σ_r varies while the model is approaching the training data, implying that $\sigma_r = \sigma_r(t)$ depends on the iteration t. The same holds for the constant $c = c(t)$.

4 Simulations Results Applied to the RESEX Data

In this section the procedure is applied to the Residence Telephone Extensions Inward Movement (Bell Canada) known as RESEX data. The chosen series is a monthly series of "inward movement" of residential telephone extensions of a fixed geographic area in Canada from January 1966 to May 1973, a total of 89 data points [1]. This serie has two extremely large values in November and December 1972 as it is shown in Figure 1. The two obvious outliers have a known cause, namely a bargain month (November) in which residence extensions could be requested free of charge. Most of the orders were filled during December, with the remainder being filled in January. This serie was identified as an ARIMA $(2, 0, 0) \times (0, 1, 0)_{12}$ model with the form $x_t = \phi_1 x_{t-1} + \phi_2 x_{t-2} + x_{t-12} + \phi_1 x_{t-13} + \phi_2 x_{t-14} + a_t$

After analyzing the performance of different architectures where the input and the hidden neurons where changed, good results were obtained for the FANN with lags X_{t-1}, X_{t-2}, X_{t-12}, X_{t-13}, X_{t-14} to predict X_t, i.e., five input neurons and one output neuron with linear activation function, and only one hidden neuron with logistic activation function. Due to the low number of data and in order to study the influence of the outlier in the learning algorithm, all the data were included in the training phase.

This architecture was trained using three different functional cost: the least square estimator (LS) described in (2), the M-estimators with the $\psi - function$ of Huber (MH) and the Tukey's bisquare (MB) given by the equation (10) and (11) respectively. To obtain the parameters that minimize the functional cost, the backpropagation with momentum algorithm was used [6].

The FANN were trained with all the data including the outliers (Serie 1) and with the data where the known outliers were edited (Serie 2). The training process was repeated 20 times.The results are shown in table 1 and 2, where the performance and the effectiveness of the prediction of the FANN trained with different estimators (first column) are shown. The performance was evaluated with the mean square error (MSE) and the effectiveness as the ratio between the MSE of the LS and the robust estimators.

The second and third column of both tables show the results of the FANN trained with Serie 1 and Serie 2 respectively. A fourth column was included to show the evaluation of FANN trained with the contaminated data (Serie 2) but evaluated by omitting the outliers. Finally in the table 1, two additional columns were added to show the peak value of the errors occurred in the location of the outliers data, the most importants contributors to the MSE.

As can be seen in table 2, the robust estimators shows almost full effectiveness under "uncontaminated" data (second column). Under the presence of outliers, the performance of the FANN with LS estimator was superior than the other two networks but as can be seen in the figure 1a), the model is separated to the bulk of data, so if we evaluate the networks without considering the outliers (fourth column), the FANNs with Robust learning over performed substantially the LS case with 212% of effectiveness (4th column of table 2). As a conclusion, first, the Robust networks approximated better the bulk of data meanwhile the FANN with LS learning tends to model the outliers, and, second, the MSE is a global measure of the prediction performance that introduce a distortion vision of the quality of the estimator because does not show the local behavior of the model.

Table 1. Performance of the learning process obtained for the different Learning algorithms ($\times 10^6$)

Est.	**Serie 1**	**Serie 2**	**Serie 2 without out.**	$error^2$ **Out. Nov.**	$error^2$ **Out. Dec.**
LS	2.3184 ± 1.4609	37.680 ± 0.0086	6.4264 ± 1.2678	2274.8	4.3
MH	2.3182 ± 0.0277	42.484 ± 0.0084	3.0203 ± 0.6122	2928.3	3.9
MB	2.3365 ± 0.0102	42.592 ± 0.0848	3.0304 ± 0.5806	2967.5	0.2

Table 2. Effectiveness of the Robust Learning compared to the LS case $(\frac{MSE_{LS}}{MSE_{M\cdot}} * 100\%)$ and square error of the outliers

Estimator	Serie 1	Serie 2	Serie 2 without out.
LS	—-	—-	—-
MH	100.01%	88.69%	212.77%
MB	99.22%	88.47%	212.06%

5 Concluding Remarks

The learning process of the FANN, based on the Least Square for the parameters estimate, were shown to be sensible to the presence of outliers, where they tend to model gross outliers due to their influence in the training. A Robust Learning Algorithm based on M-estimator was developed where the influence of the outlier in the estimation process was bounded. The robustness of the estimator will often conflict with the aim of asymptotic effectiveness, therefore the shape of the functional cost were adapted during the training.

Simulations results on real Time Serie were developed to show the improvement of the Robust Learning Algorithm over conventional least squares fitting for the RESEX Time Series.

Different types of M-estimators could be used in the Robust Learning Algorithm, so further studies can be made to choose the proper function for the data in study. The Robust technique used in this paper can be used in a different scope other than neural networks in time series. Future work in robust techniques and Neural Networks will center around making neural networks robust to changes in the variance of the noise.

References

[1] H. Allende, C. Moraga, and R. Salas, *Artificial neural networks in time series forecasting: A comparative analysis*, Kybernetika **38** (2002), no. 6, 685–707. 29, 33

[2] ———, *Robust estimator for the learning process in neural networks applied in time series*, ICANN 2002. LNCS **2415** (2002), 1080–1086. 29, 30

[3] E. Capobianco, *Neural networks and statistical inference. Seeking robust and efficient learning*, Comp. Statistics & Data Analysis (2000), no. 32, 443–454. 29

[4] D. Chen and R. Jain, *A robust back propagation learning algorithm for function approximation*, IEEE Trans. on Neural Networks **5** (1994), no. 3, 467–479. 29

[5] J. T. Connor and R. D. Martin, *Recurrent neural networks and robust time series prediction*, IEEE Transactions of Neural Networks **2** (1994), no. 5, 240–253. 29, 30

[6] Richard Golden, *Mathematical methods for neural networks analysis and design*, vol. 1, MIT Press, 1996. 28, 34

[7] F. R. Hampel, E. M. Ronchetti, P. J. Rousseeuw, and W. A. Stahel, *Robust statistics*, Wiley Series in Probability and Mathematical Statistics, 1986. 31

[8] Peter J. Huber, *Robust statistics*, Wiley Series in probability and mathematical statistics, 1981. 29

[9] R. Salas, *Robustez en redes neuronales feedforward*, Master's thesis, Universidad Técnica Federico Santa María, 2002. 30

[10] Halbert White, *Artificial neural networks: Approximation and learning theory*, Basil Blackwell, Oxford, 1992. 28, 31