# ARTIFICIAL NEURAL NETWORKS WITH ADAPTIVE POLYNOMIAL ACTIVATION FUNCTION

*F. Piazza, A. Uncini, M. Zenobi*

Dip. di Elettronica ed Automatica, Universita` di Ancona
Via Brecce Bianche - 60131 Ancona, Italy

## ABSTRACT

The aim of this work is to study an extended multilayer perceptron made of neurons with an adaptive polynomial activation function. The adaptive polynomial neural network (APNN) gives a reduction in terms of dimensions and computational complexity both in learning and in forward phase compared with traditional MLPs with a sigmoidal activation function. Many experiments have been extensively carried out both on pattern recognition and data processing problems. The relationship of the APNNs with the Volterra expansion is also discussed.

## INTRODUCTION

In the last years, the artificial neural networks have been successfully applied to the solution of many real problems. Most of applications can be roughly grouped into two major categories: pattern recognition and data processing. The main scope of the first group of applications is to classify patterns by learning from examples, i.e. determine pattern statistics from a set of pattern samples. On the other hand, the main scope of the second group of applications is to process input data to obtain new modified output data, i.e. realise a proper rule (function) by which an input vector is transformed into an output vector. In both kinds of applications, the structural and computation complexity of the involved networks is one of the major hindrance to a broad spreading of the neural techniques.

A first way to reduce the complexity of the traditional approaches, as those based on the widely used Multilayer Perceptron (MLP) with the Back-Propagation (BP) learning algorithm [1], is to introduce different and powerful neural paradigms, as the well known polynomial networks [2], [3]. For classification purposes, the polynomial Adaline or Padaline [2] is a well established technique. As shown in [3], the polynomial Adaline can be trained to realize an approximate Taylor's series expansion of the Bayesian discriminant function, thereby implementing the optimum Bayes classifier. However, a suboptimal classifier can also be implemented training the Padaline to minimize the output mean square error with LMS algorithms, providing that enough sample points are available during learning. The particular structure of the polynomial Adaline has also a strict relationship with the Volterra series expansion [4] of a non-linear functional. For data processing purpose, in fact, it can be successfully employed to implement approximate realization of non-linear systems with memory. In this case, the Padaline is therefore characterised by a system order, i.e. the polynomial degree, and a memory, both tied to the number of the Volterra kernels associated with the truncated series. A typical application of such a network, the adaptive equalization of a communication channel, is reported in [5]. The polynomial networks, however, suffer of some limitations. The major drawback is the number of their coefficients which increases exponentially with the order and the memory of the truncated Volterra series. Moreover the correct order and memory of the network for a given set of data are unknown and must be inferred by experience, usually overestimating them with a huge increase of complexity and computational costs.

A second way to reduce the complexity of the traditional approaches is to design particular implementations of common neural networks which greatly decrease the computational burden. A good example is reported in [6], [7], where the complexity of a digital MLP implementation is reduced by constraining the synaptic weights to be simple powers of two. In this case, as in many other cases of digital realization of neural networks, the activation function of the neurons is implemented by the use of a look-up table. Thus it could be feasible to choose suitable (in general more complex) activation functions which allow to solve a given problem with a smaller network, hence reducing the complexity and the implementation costs. Extensive studies have been carried out on the capabilities and behaviours of the neuron activation functions in a MLP environment [see for example 8].

In this paper, a new neural network structure is presented that has a low complexity and is particularly suitable for digital hardware realization, especially in VLSI devices. The proposed structure is based on the "polynomial neuron", a classical additive neuron but with a polynomial activation function. Such a neuron is then used to build multilayer networks trainable with an algorithm very similar to the Back Propagation. These modified MLPs (with polynomial neurons) do not exhibit more complexity in digital implementations than the classical MLPs, since the activation functions are realized with look-up tables, nevertheless they are usually smaller in terms of layers and

neurons per layer than the classical MLPs on a large class of problems. It can be shown that the proposed networks are equivalent to Padalines in which some relationships exist among the polynomial coefficients; therefore they implement a truncated and constrained Volterra series expansion.

In the following section a detailed description of the proposed structure and the learning algorithm will be given, together with a demonstration of the relationship between this structure and the polynomial networks. In the last section, some experimental results will be presented in order to show the performance of the proposed network with respect to the traditional MLP.

## THE PROPOSED ARCHITECTURE

The scheme of the architecture proposed in this paper is presented in Fig. 1.

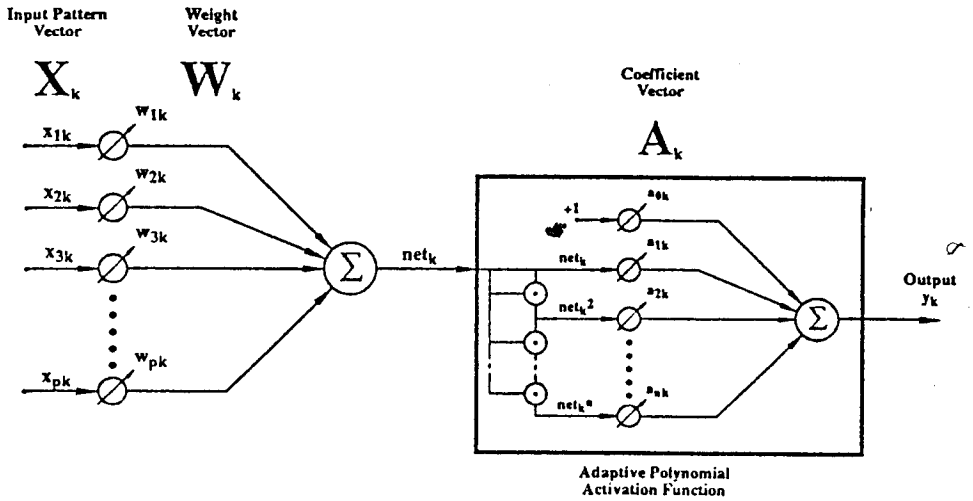


Fig. 1. A processing unit of APNN.

In order to analyze the behavior of the proposed structure, let's examinate a single neuron with two inputs and a 2nd degree polynomial nonlinearity. The two inputs, $x_1$ and $x_2$, multiplied by the respective weights, $w_1$ and $w_2$, and then summed together, become the argument of the nonlinearity, a quadratic polynomial with coefficients $a_0$, $a_1$ and $a_2$:

$$f[(w_1x_1 + w_2x_2)] = a_0 + a_1(w_1x_1 + w_2x_2) + a_2(w_1x_1 + w_2x_2)^2 =$$
$$= a_0 + a_1w_1x_1 + a_1w_2x_2 + a_2w_1w_2x_1x_2 + a_2w_1^2x_1^2 + a_2w_2^2x_2^2$$

after doing the following positions:

$$\alpha_0 = a_0$$
$$\alpha_1 = a_1w_1$$
$$\alpha_2 = a_1w_2$$
$$\alpha_3 = a_2w_1w_2$$
$$\alpha_4 = a_2w_1^2$$
$$\alpha_5 = a_2w_2^2$$

we obtain the expression:

$$\alpha_0 + \alpha_1x_1 + \alpha_2x_2 + \alpha_3x_1x_2 + \alpha_4x_1^2 + \alpha_5x_2^2$$

which contains all the terms of the Volterra expansion of order 2 in polynomial form.The nonlinear link between

the coefficients of the Volterra expansion and the parameters of the network does not allow to obtain analytically the optimal A and W vectors. Anyway, experimental tests point out that the BP algorithm, even if it cannot guarantee the optimal solution, converges to an adequate solution with an extremely reduced variance with respect to MLPs.

However, as stated in the introduction, APNNs exibit a remarkable save in terms of complexity, especially when digital implementations are involved, requiring a considerably smaller number of connections, as it can be seen from Fig. 2.

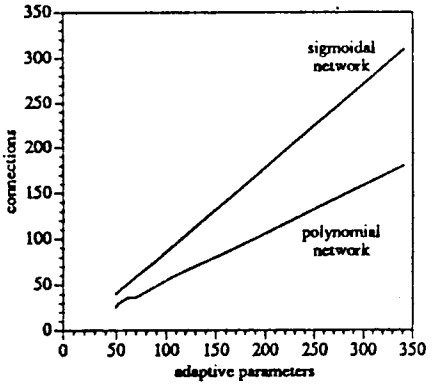


**Fig. 2.** Number of synapses (connections) vs number of adaptive parameters.

## LEARNING ALGORITHM

The proposed algorithm is an extension of the traditional Back-Propagation procedure. At each step, the updating of both the synaptic weights and the coefficients of the polynomial activation function is performed, according to the rule:

$$W_{k+1} = W_k + \Delta W_k$$

$$A_{k+1} = A_k + \Delta A_k$$

where $W_k$ and $A_k$ are respectively the weight vector and the coefficient vector. The objective of the procedure is to minimize the MSE (mean-square error),which in this case is a function not only of the weights, but also of the coefficients of the nonlinearity. At time index k we have:

$$\varepsilon = (d - y)$$

with d as the desired output and y as the output of the network:

$$y = f(a, net) = \sum_{i=0}^{n} a_i \, net^i$$

where net is the output of the summation unit:

$$net = X^T W$$

The minimization of MSE with the steepest-descent method, performed altering the weights and coefficients vectors in the direction corresponding to the negative of the measured gradient:

$$\nabla(\varepsilon^2) = \left[ \frac{\partial \varepsilon^2}{\partial W}, \frac{\partial \varepsilon^2}{\partial A} \right]^T = \left[ \frac{\partial \varepsilon^2}{\partial w_1}, \frac{\partial \varepsilon^2}{\partial w_2}, \cdots \frac{\partial \varepsilon^2}{\partial w_p}, \frac{\partial \varepsilon^2}{\partial a_0}, \frac{\partial \varepsilon^2}{\partial a_1}, \cdots \frac{\partial \varepsilon^2}{\partial a_n} \right]^T$$

where

$$\frac{\partial \varepsilon^2}{\partial w} = -2\varepsilon \left[ \frac{\partial f}{\partial net} \right] x \qquad \frac{\partial \varepsilon^2}{\partial a_j} = -2\varepsilon \left[ \frac{\partial f}{\partial a_j} \right]$$

with

$$\frac{\partial f}{\partial net} = \sum_{i=1}^{n} i\, a_i\, net^{(i-1)} \qquad \frac{\partial f}{\partial a_j} = net^j .$$

Finally we obtain:

$$\Delta W = 2\eta\varepsilon \left[ \frac{\partial f}{\partial net} \right] X$$

$$\Delta A = 2\eta\varepsilon \left[ \frac{\partial f}{\partial a_0}, \frac{\partial f}{\partial a_1}, \frac{\partial f}{\partial a_2}, \ldots \frac{\partial f}{\partial a_p} \right]^T$$

where $\eta$ is the learning rate. In some cases, to optimize the convergence time, two different learning rates have been used, one for weights and another one, smaller, for the polynomial coefficients.

## EXPERIMENTAL RESULTS

In order to estimate the performances of neural networks with an adaptive polynomial activation function with respect to the traditional MLPs, two problems available in literature are examinated. The first one is the example number 4 proposed by Narendra and Parthasarathy [9] and even used by Specht [10], and concerns the identification of a nonlinear system in the form:

$$y_p(k+1) = f\ [\, y_p(k), y_p(k-1), y_p(k-2), u(k), u(k-1)\,]$$

where $u$ and $y_p$ indicate the input and output of the plant respectively, and f assumes the form:

$$f\ [\, x_1, x_2, x_3, x_4, x_5\,] = \frac{x_1 x_2 x_3 x_5 (x_3 - 1) + x_4}{1 + x_2^2 + x_3^2} .$$

Using first the network proposed in [9] made of 20 neurons in the first hidden layer and 10 neurons in the second hidden layer, the system identification has been carried out for 100 times with an input pattern of 10,000 samples of noise uniformly distributed within the interval [-1.2, 1.2]. The procedure has been repeated for a network with adaptive polynomial activation function, made of 30 neurons with a 4th degree polynomial and 1 output neuron with a 2nd degree polynomial. This configuration has been chosen so that the two networks approximatively have the same amount of adaptive parameters. The typical course of the Mean Square Error (MSE) for the two networks is shown in Fig. 3.

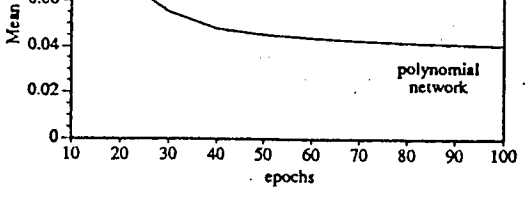


Fig. 3. Course of MSE during the identification procedure.

In a second time the procedure has been applied to other networks with several configurations and consequently with a different number of adaptive parameters. The result is that, at a parity of performances, neural networks with adaptive polynomial nonlinearity require a considerably smaller number of connections, as pointed out before.

The second problem considered concerns pattern recognition, and it was chosen to investigate of the generalization capabilities of networks built according to the proposed architecture. Ten 7x7 bits matrices, representing digits from 0 to 9 are considered as input patterns. The desired output has been coded with 4 bits using the Gray code. An APNN with a unique layer made of 4 neurons with a 3rd degree polynomial (p4g3) has been compared with two sigmoidal MLPs with one hidden layer, made respectively of 4 neurons (s4_4) and 8 neurons (s8_4), besides the 4 output neurons. The learning fase has been carried out for all the networks under examination introducing a 5% average flipping of the bits in the input patterns (about 2 bits per character), and stopping the training when the MSE has reached a previously fixed value. To perform the generalization phase, some input patterns with a fixed amount of flipped bits, from 0 to 7, have been used.

The experimental results indicate on average a certain equivalence among the networks with regard to the percentage of the correctly recognized characters (hit-rate), as shown in Fig. 4.

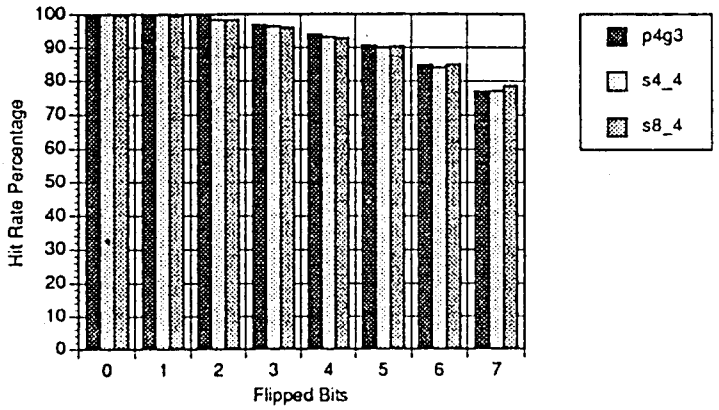


**Fig. 4.** Average values of the hit-rates for different classes of input sets.

Nevertheless, the variance of the hit-rates (see Fig. 5) show clearly that, for the case of APNNs, the values of the single tests are considerably more grouped around the average value.

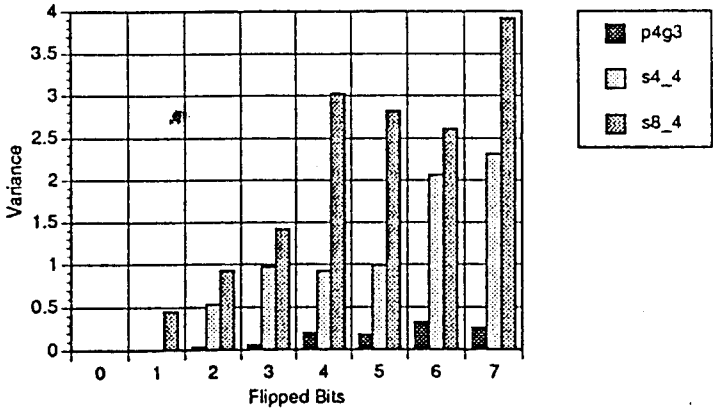


**Fig. 5.** Variance of the resulting hit-rates.

## CONCLUSIONS

APNNs have shown an excellent behavior on problems of pattern recognition and data processing.Their results are comparable with those obtained by classical MLPs, but a significantly smaller amount of adaptive parameters is needed. With respect to the known polynomial structures, for which a certain knowledge of the characteristics of the discriminant function is required, APNNs don't need any particular a priori information. Further studies will interest other kinds of adaptive nonlinearities, some applications of the powers-of-two technique [6], [7], and an extension of the model using the complex back-propagation algorithm.

## REFERENCES


[1] D. E. Rumelhart, J. L. McClelland, and the PDP Research Group, Parallel Distributed Processing (PDP): Exploration in the Microstructure of Cognition (Vol. 1), MIT Press, Cambridge (MA), 1986.

[2] M. Caudill, "The polynomial Adaline algorithm," Comput. Lang., Dec 1988.

[3] D. F. Specht, "Probabilistic neural networks and the polynomial Adaline as complementary techniques for classification", IEEE Trans. on Neural Networks, Vol. 1 No. 2, March 1990.

[4] V. Volterra, "Theory of functionals and of integral and integro-differential equations", New York: Dover, 1959.

[5] S. Chen, G. J. Gibson, C. F. N. Cowan, "Adaptive channel equalisation using a polynomial-perceptron structure", IEE Proceedings, Vol. 137, Pt. 1, No. 5, October 1990.

[6] M. Marchesi, G. Orlandi, F. Piazza, A. Uncini: "Fast neural networks without multipliers", accepted for publication on IEEE Transanctions on Neural Networks.

[7] M. Marchesi, G. Orlandi, F. Piazza, L. Pollonara, A. Uncini, "Multi-layer Perceptrons with discrete weights", Proc. of IJCNN International Joint Conference on Neural Networks, San Diego (CA-USA) pp II-623- II-630 June 1990.

[8] K. Hornik, M. Stinchcombe, H. White, "Multilayer feedforward networks are universal approximators", Neural Networks, Vol. 2, pp. 359-366, 1989.

[9] K. S. Narendra, K. Parthasarathy, "Identification and control of dynamical systems using neural networks," IEEE Trans. on Neural Networks, Vol. 1, No. 1, March 1990.

[10] D. F. Specht, "A general regression neural network", IEEE Trans. on Neural Networks, Vol. 2, No. 6, November 1991.