# CBP networks as a generalized neural model

Sandro Ridella, Stefano Rovetta, and Rodolfo Zunino
Department of Biophysical and Electronics Engineering – University of Genova, Italy
Via all'Opera Pia 11a 16145 Genova (Italy)
E-mail: {ridella,rovetta,zunino}@dibe.unige.it
Phone: +39–10–353 2268 – Fax: +39–10–353 2175

## Abstract

*This paper analyzes the* Circular backpropagation network*, a simple modification of the multilayer perceptron with interesting practical properties, especially well–suited to cope with pattern classification tasks. The proposed model unifies the two main representation paradigms found in the class of mapping networks for classification, namely, the* surface–based *and the* prototype–based *schemes, while retaining the advantage of being trainable by back–propagation. Multi–layer perceptrons, Radial–Basis–Function networks and Vector–Quantization networks are shown to be implementable with small modifications to the model under study*[1].

## 1. Introduction

Mapping neural networks are computing devices that implement, in a distributed way, a function $\psi$, from some input domain $\mathcal{D} \subset \mathbb{R}^d$ to some output domain $\mathcal{T}$, parametrized by a set of parameters and featuring only feedforward signal paths.

Mapping networks are widely used to approach classification problems when the task is to derive a rule from a set of examples. The present work focuses on the class of mapping networks in the context of classification problems (by default with two–class problems). We attempt to set up a framework to allow the study of a more general network model that may encompass different representation paradigms.

We refer to the multilayer mapping network model with a topological structure inherited from the multilayer perceptron (MLP). A single hidden layer will always be assumed in the following, without loss of generality. The network structure being fixed, we focus on the description and design of the (hidden) unit.

We adopt the following formalism. $\mathbf{x}$ indicates the input vector of dimension $d$. The parameters (weights, bias, etc.) are the components of the vector $\mathbf{w} \in \mathbb{R}^p$, which needs not (and usually does not) have the same dimension as $\mathbf{x}$. The unit is divided in two blocks, computing functions respectively denoted by $\overline{r}$ and $\overline{a}$. The first block outputs the value $r = \overline{r}(\mathbf{x}, \mathbf{w})$, which we call the *stimulus*. The second block outputs the *activation* $a = \overline{a}(r)$. This scheme, introduced in [7], on one side may help interpret a learned mapping from a representation standpoint, and, on the other hand, features interesting properties by itself.

This work deals with a modification to the activation function of the hidden neurons of a MLP, introducing a polynomial activation. The approach taken in many previous works is to consider polynomial activation functions as an alternative to the multilayer scheme. However, this introduces the need for additional constraints to keep the generality of the set of functions implementable by the model low enough for a good generalization [2][11]. Our approach aims instead to search for the *minimal* increment in the generality of the multilayer model that is capable of substantially improving the representation ability without affecting (and possibly enhancing) the generalization properties.

## 2. The CBP model

### 2.1. Generalized neural unit

Representing a neural unit with the two quantities $r$ and $a$ yields a quite straightforward interpretation in geometric terms. The stimulus results from the application of a "filter" sensitive to some geometric property of the input space. The activation is the response of

[1]The results reported in this paper are an extension of work previously presented in the *IEEE Transactions on Neural Networks* [8].
This work was supported by the Italian Ministry for the University and Research (MURST) 40%.

the unit to the geometric property pointed out by the stimulus.

Through the selection of appropriate functional forms for $\overline{r}$ and $\overline{a}$, the model can be used to represent all neural units usually adopted in practical applications. Some examples follow.

- The perceptron [9]: $r = \overline{r}(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{i=1}^{d} x_i w_i$; $a = \overline{a}(r) = \mathcal{H}(r)$ (where $\mathcal{H}$ is a Heaviside function).

- The sigmoidal multilayer perceptron unit [10]: $r = \overline{r}(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{i=1}^{d} x_i w_i$; $a = \overline{a}(r) = \sigma(r)$ (where $\sigma$ is a sigmoidal function).

- The radial basis (Gaussian) unit [4]: $r = \overline{r}(\mathbf{x}, \mathbf{w}) = ||\mathbf{x} - \mathbf{c}||^2/\sigma^2$; $a = \overline{a}(r) = e^{-r}$

It is now possible to make a parallel analysis of many network models by comparing their stimuli and activations. The notion of *representation paradigm* can be a useful tool for performing such comparisons.

The representation paradigm is closely related to the geometrical properties of the stimulus. A distance-based stimulus (e.g., the Euclidean distance between the parameter vector and the input vector) can be associated with the *prototype-based* paradigm, according to which a network stores representative patterns (prototypes) and computes its output by measuring the match between a pattern and the stored prototypes. Nearest–neighbor classifiers [1] implement this paradigm.

By contrast, the *surface-based* paradigm is represented by those models that draw region borders (hypersurfaces) in the input space, usually composed of individual segments realized by different units, and compute their output according to the position of an input pattern with respect to the borders. The perceptron [9] is an example of this paradigm.

These two approaches can be regarded as being complementary.

## 2.2. The circular unit and the CBP network

The perceptron can be generalized by letting $\overline{r}(\mathbf{x}) = \sum_{i=1}^{p} w_i \xi_i = \mathbf{w} \cdot \boldsymbol{\xi}$, where the map $\mathbf{x} \mapsto \boldsymbol{\xi}$ ($\boldsymbol{\xi} \in \mathbb{R}^p$) is such that each component $\xi_i$ is given by a product of components of $\mathbf{x}$ (some power of a single component, or the product of powers of different components). Usually, one of the terms is a constant whose weight implements the bias. The parameter vector is of the same dimensions as $\boldsymbol{\xi}$, and the resulting stimulus is a polynomial with its components as coefficients.

The number of terms of a complete polynomial with $d$ variables of order $q$ is $p = \binom{d+q-1}{q}$, which is of order $d^q$. Therefore, we consider the selection of an appropriate number of polynomial terms. In the following, the *circular back-propagation* (CBP) model [8][7] will be studied from this standpoint. As previously remarked, the model features the standard multi layer topology with a single hidden layer. At the unit level, the CBP model is described by the following functions:

$$\overline{r}(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{i=1}^{d} x_i w_i + w_q x_q \tag{1}$$

where the last term is a compact form for $x_q = \sum_{i=1}^{d} x_i^2$, and

$$\overline{a}(r) = \sigma(r) \tag{2}$$

This is a special case of polynomial unit. There is one additional parameter, i.e., the coefficient $w_q$, which weights the sum of the squared inputs. By simple algebraic transformations, it is possible to obtain another form for the same stimulus:

$$r = g\left(||\mathbf{x} - \mathbf{c}||^2 - \theta\right) \tag{3}$$

in which the parameters $\mathbf{c}$ , $g$, and $\theta$ do not appear as weights but have the following geometrical interpretations.

The distance from the point $\mathbf{c}$ in the space of inputs is computed and compared with the value $\theta$. The result is scaled with the coefficient $g$ to obtain the actual stimulus $r$, and the activation $a$ is computed by the standard sigmoidal function. The output of the unit can be positive inside (for $g > 0$) or outside (for $g < 0$) a circular (in general, hyperspherical) region; anyway, a localized "bump" with a circular section is obtained around the point $\mathbf{c}$. Therefore, we describe the parameters as follows:

$\mathbf{c}$ = center or prototype
$\theta$ = radial threshold (hence $\rho = \sqrt{\theta}$ = radius)
$g$ = gain

We call these the "circular parameters."

The double form of each parameter reflects the double nature of the representation. The circular parameters implement a transfer function implementing the prototype-based paradigm. However, when the coefficient $w_q$ is very small, the circular parameters are not adequate anymore, and the stimulus collapses to the standard, linear perceptron stimulus. In this situation, the unit implements the surface-based paradigm.

The choice between the two representation forms depends only on the value of adaptable parameters, so it is left to the optimization process (*paradigm plasticity*).

This enables the network to adapt the representation form, without need for the user's supervision.

The only different feature of a CBP network, as compared with the MLP, is an additional input $x_q$. This means that a CBP network can be obtained by an *off-line* modification to the training set, i.e., by adding the quadratic term $x_q$ directly to the input patterns. The resulting network will be trained by plain backpropagation, at the only expense of an additional input (for a network with $h$ hidden units, this means $h$ additional weights).

## 3. Equivalence to Gaussian radial basis function networks

In this section we shall show that the CBP model may be made equivalent to another widely used neural scheme, i.e., the network of locally tuned Gaussian units. In the next section, a similar proof will be given for vector–quantization based networks.

Equivalence between two network models requires two conditions to be satisfied. The first is that the sets of functions implementable by the two models coincide. The second is that the training procedures should allow them to learn the same mapping for the same training set.

The first condition is of architectural nature. It can be verified by comparing the structure and interconnections of the layers, and the activation functions of the units. The second condition is related to the algorithms used for training and not to the networks. It can be verified by comparing the iterative learning steps. However, if the performance criterion adopted in training is the same for both models (e.g., in classification, the percentage of correctly labeled patterns), we can concentrate on the architectural equivalence, since the goal of the optimization process coincides in the two cases.

The transfer function of a circular unit is radially symmetric. Hence a CBP net has by itself the structure of a radial basis function (RBF) network. However, in practice, the most commonly adopted basis functions are the isotropic Gaussians [5][6]:

$$\mathcal{G}(\mathbf{x}) = \exp\left(-\frac{||\mathbf{x}-\mathbf{c}||^2}{\sigma^2}\right) \tag{4}$$

The training of such networks requires the choice of appropriate values for the parameters $\mathbf{c}$ and $\sigma$, which is usually made independently. Here we show that a CBP network can implement a Gaussian RBF network; therefore, backpropagation training can be used to obtain the same results as those obtained by RBF training. More formally, we can state this fact as follows:

There is a 2-layer, sigmoidal-activation CBP network equivalent to a Gaussian RBF network with the same number of hidden units $h$.

To prove this statement, we can express the stimulus of the generic hidden unit of a $\sigma$-CBP network in terms of the circular parameters as per Equation 3. The activation function is:

$$\overline{a}(r) = \frac{1}{1+e^{-r}} \tag{5}$$

Therefore, if we let $r' = g||\mathbf{x}-\mathbf{c}||^2$, the overall transfer function of the unit can be expressed as:

$$a = \frac{1}{1+e^{-(r'-g\theta)}} \tag{6}$$

By some algebraic manipulations, this expression can be transformed as follows:

$$a = \frac{1}{1+e^{-r'}e^{-g\theta}} = \frac{e^{r'}e^{g\theta}}{e^{r'}e^{g\theta}+1}$$

A generic output unit will not receive this value directly as an input, but only after a multiplication by the weight $w$. Therefore, the output value of the hidden unit can be multiplied by an arbitrary constant, which will be compensated for by the subsequent weight:

$$ka = e^{r'}\frac{ke^{g\theta}}{e^{r'}e^{g\theta}+1}$$

Let the term $g\theta$ take on very large values. Let the constant $k$ take on correspondingly small values. The multiplying fraction can then take on values arbitrarily close to 1. Hence, including the weight in the expression for the output value, we can write:

$$|wa_{\mathrm{RBF}} - w'a_{\mathrm{CBP}}| < \epsilon$$

for any $\epsilon > 0$, where: $a_{\mathrm{RBF}}$ is the activation computed by using the Gaussian activation function and stimulus, as per Equation 4; $a_{\mathrm{CBP}}$ is the activation using the CBP activation function and stimulus; $w$ is the output weight; and $w'$ is the compensated output weight, $kw' = w$

After showing that a CBP network can encompass also the Gaussian RBF model, we may ask whether the converse is also true, which means that the two approaches are theoretically identical. However, this is not the case. This may be shown with the aid of the *alternate-labels problem*. Figure 1 shows an alternate-labels problem with 7 data points, and the one-dimensional activation profile of 2 CBP units. It is possible to see that the CBP activation profile can identify 7 zones, characterized by sign inversion, while

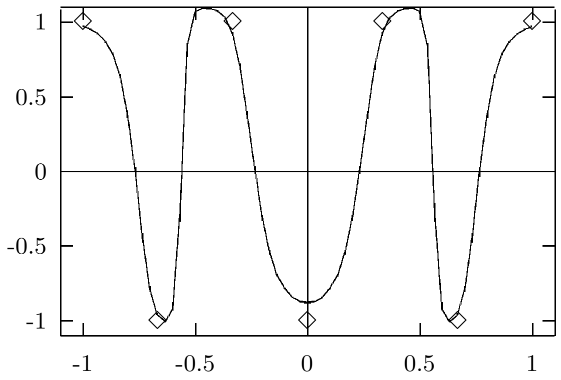


Figure 1. How CBP solves the alternate-labels problem.

RBF is limited to 5 zones. This has been experimentally demonstrated for CBP, as shown in the figure, with good convergence rate. The limitation for RBF can be proved as follows.

Consider a Gaussian RBF network with $d = 1$, $h = 2$, $b = 1$ to attempt representing the alternate labels problem with 7 data points. Symmetry considerations allow the stimulus of its output unit to be expressed as

$$r_{\text{out}} = w_0 + w_1 e^{-g_1 x^2} + w_2 e^{-g_2 x^2} \tag{7}$$

Derivation of this expression with respect to $x$ yields

$$\frac{\partial r_{\text{out}}}{\partial x} = -g_1 x w_1 e^{-g_1 x^2} - g_2 x w_2 e^{-g_2 x^2} \tag{8}$$

This expression vanishes for $x = 0$, for $x = \pm\infty$, and for $x = \pm\sqrt{\frac{ln(-w_1 g_1/w_2 g_2)}{(g_2 - g_1)}}$. (This pair of roots is defined only when the arguments of the logarithm and of the root are both positive. We assume this is the case, since we are interested in assessing the maximum number of roots.)

The roots of the derivative correspond to minimum, maximum and saddle points. Between pairs of these points, we can identify at most five regions corresponding to five different classification outputs. Therefore the 7-points problem cannot be solved.

We conclude with a note on the representation properties of the CBP activation function as compared with the Gaussian function. In the CBP network the parameters are expressed in the form of weights, rather than in the circular form. This means that degenerate radial functions are implementable in the CBP formalism, since an infinite radius is realizable when expressed as $w_q = 0$. In the RBF formalism, this would mean giving an infinite value to an actual parameter (the center's coordinates), which is unrealizable both in physical hardware and in software simulation. Therefore the equivalence between RBF and MLP could be theoretically assessed in the limit, but not physically attained, whereas the equivalence between CBP and MLP is feasible also in practice.

## 4. Equivalence to vector quantization networks

Competitive networks are often compared to models with localized activations because of their distance-based stimulus. In this subsection, we consider the competitive CBP model in the self-supervised form, in which the identical mapping is learnt by a network with as many outputs as inputs. For this model the target is defined as:

$$\mathbf{x}(l) = \mathbf{t}(l) \;\; \forall l. \tag{9}$$

The $j$-th hidden unit is defined as:

$$r_j = w_0 + \sum_{i=1}^{d} w_i x_i + w_q x_q \tag{10}$$

and

$$a_j = \frac{e^{G r_j}}{\sum_{j'=1}^{n_h} e^{G r_{j'}}} \tag{11}$$

This is the so-called "soft-max" function. Its behaviour is controlled by the non-adaptive parameter $G$, so that for $G$ very high it reduces to the normal "max" function. The output layer is a standard MLP layer with $n_o = d$ units.

The architectural equivalence between the above described auto-associative CBP network (AACBP) and a vector quantization (VQ) network is now apparent. In this situation the last layer is used as an output only in the training phase, for the error function computation. When using the network the actual classification is found on the hidden layer. Therefore, since the error computed on the last layer should correspond to the classification on the hidden layer, the weights should be constrained to have the same value on the two layers:

$$k = i \;\Rightarrow w_{ji} = w_{kj} \;\; \forall j \tag{12}$$

The usual VQ learning step is computed as a function of the distance between the input pattern $\mathbf{x}(l)$ and the winning unit.

$$\Delta w_{ji} = \eta'(w_{ji} - x_i) \tag{13}$$

The back propagation learning step is proportional to the gradient of the cost function.

$$\Delta w_{ji} = \eta'' \frac{\partial c}{\partial w_{ji}} \tag{14}$$

In the AACBP case, the derivative of the hidden units activation function is:

$$\frac{\partial a_j}{\partial r_j} = G \frac{e^{Gr_j} \sum_{j'=1}^{n_h} e^{Gr_{j'}} - e^{Gr_j} e^{Gr_j}}{\sum_{j'=1}^{n_h} e^{Gr_{j'}^2}} =$$

$$G a_j (1 - a_j) \tag{15}$$

The components of the gradient of $c$ in the weight space are:

$$\frac{\partial c}{\partial w_{ji}} = -\frac{\partial c}{\partial a_j} \frac{\partial a_j}{\partial r_j} x_i \tag{16}$$

for the first (hidden) layer, and

$$\frac{\partial c}{\partial w_{kj}} = -\sum_{k=1}^{n_o} 2(t_k - a_k) w_{kj} \tag{17}$$

for the last layer. By imposing conditions (9) and (12), we have:

$$\frac{\partial c}{\partial w_{kj}} = -2(x_k - a_k) a_j$$

$$+ \sum_{k'=1}^{n_o} 2(x_{k'} - a_{k'}) w_{k'j} a_j (1 - a_j) G x_k \tag{18}$$

which corresponds to

$$\frac{\partial c}{\partial w_{kj}} = -2(x_k - a_k) a_j + \delta_j x_k \tag{19}$$

where the coefficient of the term $x_k$ in equation (18) has been given the standard symbol $\delta_j$.

Usually, the "max" activation function is adopted in VQ training. Hence we will assume $G \to \infty$, and the above value is:

$$\frac{\partial c}{\partial w_{kj^*}} = 1 \quad \text{and} \quad \frac{\partial c}{\partial w_{kj}} = 0 \quad \forall j \neq j^* \tag{20}$$

where $j^*$ denotes the index corresponding to the maximum activation ($j^* = \text{argmax}_j\{a_j\}$).

Therefore an appropriate modification to the activation function of a CBP network makes it possible to map a surface-based feedforward network onto a prototype-based representation, trainable by backpropagation; moreover, the backpropagation updating rule is equivalent to the Kohonen-type updating step [3] used in vector quantization networks.

## 5. Concluding remarks

In this paper, the extensions of the circular backpropagation multilayer network have been investigated. Theoretical analysis and experimental evidence suggest that this model is especially well-suited to implement classification tasks. The paradigm plasticity featured by the model allows the implementation of classification principles which have different interpretations. This allows apparently different networks to be encompassed by the same framework, extending the applicability of backpropagation training to prototype-based models.

## References


[1] T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Trans. Info. Theory*, IT-13:21–27, January 1967.

[2] I. Guyon, B. E. Boser, and V. Vapnik. Automatic capacity tuning of very large VC-dimension classifiers. In S. J. Hanson, J. D. Cowan, and C. L. Giles, editors, *Advances in Neural Information Processing Systems V*, volume 5, 1992.

[3] T. Kohonen. *Self Organization and Associative Memories*. Springer, 3rd edition, 1989.

[4] R. P. Lippmann. An introduction to computing with neural nets. *IEEE Acoustic, Speech and Signal Processing Magazine*, 4:4–22, 1987.

[5] J. Moody and C. Darken. Fast learning in networks of locally-tuned processing units. *Neural Computation*, 1:281–294, 1989.

[6] T. Poggio and F. Girosi. Networks for approximation and learning. *Proceedings of the IEEE*, 78:1481–1497, September 1990.

[7] S. Ridella, S. Rovetta, and R. Zunino. Adaptive internal representation in circular back-propagation networks. *Neural Computing and Applications*, 3:222–233, 1995.

[8] S. Ridella, S. Rovetta, and R. Zunino. Circular back–propagation networks for classification. *IEEE Transactions on Neural Networks*, 8(1):84–97, January 1997.

[9] F. Rosenblatt. *Principles of Neurodynamics*. Spartan, New York, 1962.

[10] D. E. Rumelhart and J. L. McClelland. *Parallel Distributed Processing*. MIT Press, Cambridge MA, 1986.

[11] V. N. Vapnik. *The Nature of Statistical Learning*. Springer–Verlag, New York, 1995.