Proceedings of the 1995 European Symposium on Artificial Neural Networks
Brussels, Belgium, April 1995, pp. 297-302.

Learning the Appropriate Representation Paradigm by Circular Processing Units

Sandro Ridella, Stefano Rovetta, and Rodolfo Zunino

DIBE - Department of Biophysical and Electronic Engineering
University of Genoa - Italy

Abstract. This paper presents a neural model that exhibits, in a single architecture, the properties of other different, widely used schemes. Such a model can be regarded as the result of generalization of different feedforward architectures. The way its properties are achieved is very inexpensive in terms of computational load. Moreover, although the behavior of the single unit is conceptually different from that of the well-known "multi-layer perceptron", the network layout is almost identical, thus allowing one to port all the knowledge available about the classical model onto the new scheme at very low cost.

1 Introduction

The behavior of a neural network is usually analyzed by interpreting the set of parameters of each processing element in one of two possible ways. In the feedforward model, such parameters are generally viewed as weighting coefficients that describe a surface in the input space. The activation of an element is thus decided according to a rule, and the surface represented by the element weights is the decision boundary for that rule. The decision can be made in either a binary or a continuous way. Other neural models imply a different interpretation of the set of coefficients, which are viewed as coordinates of a reference point in the input space. The activation of an element is decided according to a best match (or minimum distance) criterion between the input pattern and the prototypal pattern learned by the processing element. This is the standard way of implementing a network of the vector quantization type, such as the family derived from Kohonen's self-organizing maps.
This duality between *rule-based* and *prototype-based* representations contributes to further increase the complexity of the task of inspecting the acquired knowledge embedded in a network's parameters.
We analyze a particular feedforward network model called Circular Back-Propagation (CBP), which is aimed at combining the advantages of both representation paradigms and at overcoming individual drawbacks. It differs from classical feedforward models – here denoted, for brevity, as Back-Propagation (BP) networks – in the inclusion of an additional quadratic term that enables hidden units to implement (hyper)spherical decision boundaries. A circular decision boundary allows a direct mapping of a weight vector into the input space, as a reference point or prototype. At the same time, the model's feedforward structure still permits the use of conventional and effective training algorithms – for example, back-propagation [5, 1].
In section 2, the CBP model is introduced and described. In section 3, experimental results on artificial problems are reported and the properties of a CBP network are assessed. In section 4, an experimental comparison between CBP and standard BP is

Proceedings of the 1995 European Symposium on Artificial Neural Networks
Brussels, Belgium, April 1995, pp. 297-302. © 1995 De Facto

made, concerning a real world classification task. In section 5, some concluding remarks are made.

2 CBP networks: structure and implementation

2.1 A general model for feedforward networks

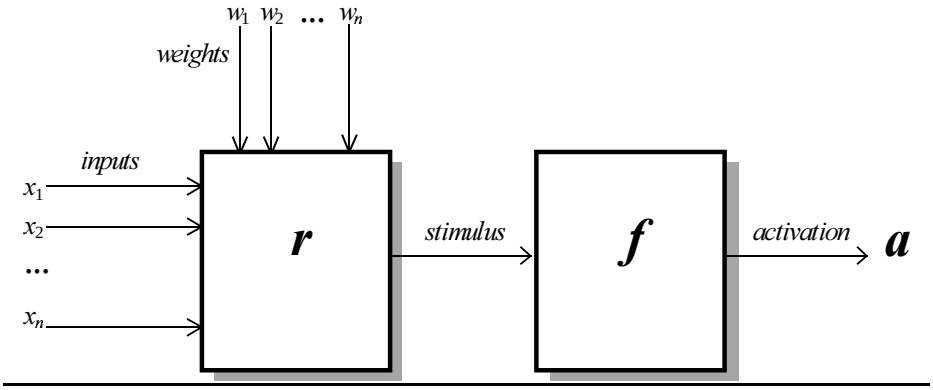


Fig. 1 The basic model of a general processing unit.

The model of a neuron or processing unit can be developed in different ways. However, a large number of these ways can be unified into the scheme shown in Fig. 1. The model behavior is summarized by the basic input-output relationship:

$$a = f[r(\mathbf{x}, \mathbf{w})]$$

The M-dimensional vector $\mathbf{x}$ is the input to the neuron, and the M-dimensional vector $\mathbf{w}$, denotes a set of parameters of the unit (i.e., its weight vector). r is a scalar function of a vector variable, $r : \boldsymbol{R}^M \rightarrow \boldsymbol{R}$, representing what we call the neuron *stimulus*, and f is a scalar function of a scalar variable, called the activation function. Finally, a is the activation value that is obtained at the output. The properties of the model depend on the actual implementation of the functions r and f. For example, if we adopt a dot product as a stimulus function and a stepwise activation function, we have the standard perceptron. If the stimulus is an Euclidean norm and the activation is arranged so as to provide a winner-takes-all behavior, we obtain the class of vector-quantization and self-organizing networks.

In feedforward structures, one often assumes a layered architecture. Neurons are arranged into layers, each interconnected to only one other layer at its input; the same holds true at the output. Of course, a layer receives an input from the outside, and another layer sends its output to the outside.

The CBP schema is a particular instance of this general model. A Euclidean distance stimulus is applied to a sigmoidal activation:

$$r = g\left(\left\|\mathbf{x} - \mathbf{p}\right\|^2 + \theta\right) \qquad \text{and} \qquad f(r) = \frac{1}{1 + e^{-r}}$$

Proceedings of the 1995 European Symposium on Artificial Neural Networks
Brussels, Belgium, April 1995, pp. 297-302.

In this model, the parameter vector **p** holds the centers of the represented prototype, and θ implements a bias for the distance. Moreover, a gain factor g is assigned the ultimate task of shaping the sigmoid's slope.

2.2 Realization of a CBP network

In spite of the conceptual differences between a classical BP network and a CBP network, only a slight modification to the former is required to obtain the latter. Through a little algebra, its easy to show that

$$r = g\left[\mathbf{x}^T \cdot \mathbf{x} - 2\,\mathbf{p}^T \cdot \mathbf{x} + \mathbf{p}^T \cdot \mathbf{p} + \theta\right]$$

$$= w_{M+1}\sum_{k=1}^{M} x_k^2 + \sum_{i=1}^{M} w_i x_i + w_0$$

$$= \mathbf{w} \cdot \mathbf{X}$$

where we define a new parameter vector, **w**, and a new input vector, **X**, as follows:

$$w_0 = g \cdot \left(\sum_{j=1}^{M} p_j^2 + \vartheta\right) \qquad w_i = \left(-2gp_i\right) \quad \forall i = 1,\ldots,M \qquad w_{M+1} = g$$

$$X_0 = x_0 \qquad X_i = x_i \quad \forall i = 1,\ldots,M \qquad X_{M+1} = \sum_{j=1}^{M} x_j^2$$

The last component X_{M+1} does not require a weighted combination of the inputs, as it is a straightforward sum. Therefore, it is the equivalent of a supplementary input, with its usual weight (the procedure scheme is shown in Fig. 2). As a consequence, to change from the standard BP to the circular version, only one more input and the corresponding weight are necessary. This allows one to adopt all the well known techniques for back-propagation training, to achieve a more powerful neural architecture at the only expense of training one supplementary weight.
In general it is required that CBP units be placed only on the first computing layer, which is used as a trainable feature extractor, whereas a standard perceptron layer can be used for the actual classification process.

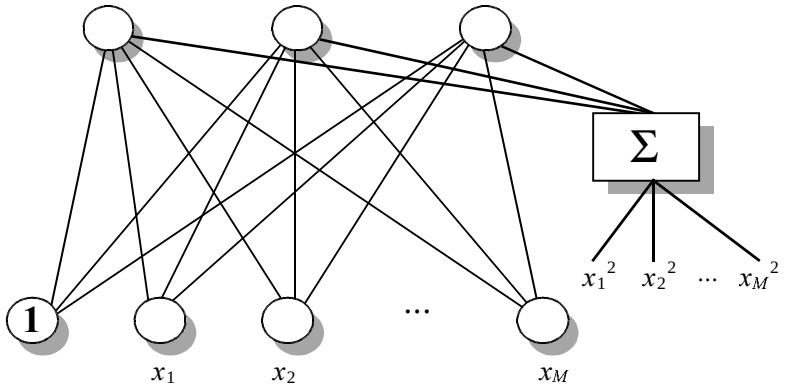


Fig. 2 A CBP network is an extended BP network.

Proceedings of the 1995 European Symposium on Artificial Neural Networks
Brussels, Belgium, April 1995, pp. 297-302.

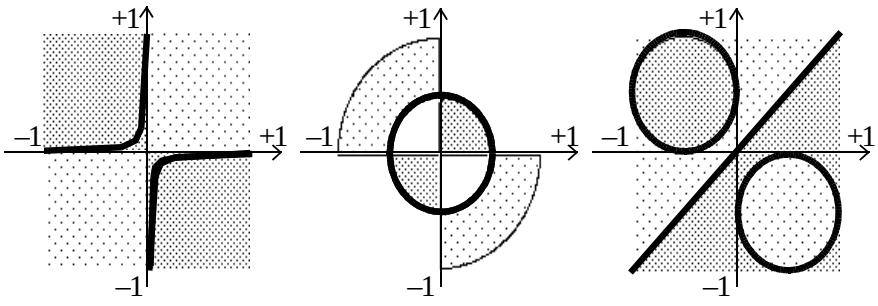
2.3 Data representation in CBP units

Once a CBP network is trained, it can be inspected to obtain information on the actual knowledge acquired. Given the particular stimulus selected, a single unit can easily be tailored to representing different data distributions by the same input-output relation; this requires only an (automatic) change in its trainable parameters.
The circular characteristic is defined by the presence of the "sum of squares" term. If the weight for this term is very small, the unit reduces to a standard BP perceptron. Therefore the knowledge representation can follow either of two possible paradigms: when the "squares" term is very small or null, the unit represents a *rule*; when it is comparable to the other terms, the unit represents a distance-based evaluation, that is, a *prototype*.
The prototypal data representation involves learning the position of the prototype in the pattern space. This also is automatically achieved by training the weights. The width of the decision region is given essentially by the bias term, which is related to the radius of the circular region.
The region smoothness, that is, the speed at which the output value (the activation) changes from the value 0 to 1 while entering the decision region, is learned independently of all the other parameters. Thus it is not linked to the region width, and can be tailored to implementing a decision at the level of smoothness required (from a very soft fuzzy decision to a hard step), even for very large regions. In other words, the function is more general than the Gaussian function used to approximate a probability density function according to Bayesian decision theory.

3 Experimental verifications

3.1 Some test problems

To investigate the properties of the proposed model, some test problems have been addressed, which are graphically represented in Fig. 3. The figure shows the training set, along with the decision regions (heavier lines) learned by a CBP layer. (Note that the dot density in the figures is not related to the sample density in the regions, which is a random density with uniform distribution.) Fig. 3a shows the standard two-dimensional generalized XOR, solved by a two-unit CBP layer. The decision borders



3a Generalized 2-Xor **3b** Non connected regions **3c** Mixed region shapes

Fig. 3 Test problems.

obtained are impossible to implement using a two-unit hidden BP layer. Fig. 3b presents another problem of the same nature, though an easier one. Whereas CBP can learn this task using a single unit, BP again requires a larger number of units to approximate a closed boundary, and of course needs at least two layers. A single BP unit (a sigmoidal perceptron) cannot perform better than 25% wrong classifications, as predicted by an elementary theoretical analysis.
Another problem is shown in Fig. 3c. Here the adaptive properties of the unit can be exploited to obtain both perceptron-like linear rules and prototyping capabilities in a single network. A CBP layer with three units allows the resulting weights to approximate very closely the two circular regions and the linear border.

3.2 A real-domain application

Once the basic properties of the model have been tested, the CBP network's efficiency has been verified in an application using a complex, real-world data base, the Lyme disease data, which constitutes a standard for our research group. The data refer to the description of a number of persons with suspected Lyme borreliosis, and were collected by the Italian Working Group for the Lyme Disease. The database features 684 patterns (54 inputs plus a binary classification for the diagnosis), of which 500 were used for training and 184 for validation. The classification results were compared with these yielded by a standard BP network [3, 4]. The validation error for BP was 10.7%, while for CBP was 9.6%. The percent improvement obtained by the circular stimulus is about 11%. The BP network features 605 weights, while the CBP network has 616 parameters; hence the additional computational load is only due to a 1.8% increment in the number of weights to be trained. The sum of squares, which has to be computed for each activation pattern, can be put off line in a pre processing phase, as it is required only for the first layer, hence only for each input pattern.

4 Discussion and conclusion

The model exhibits properties that are partly shared by other approaches. However, none of these approaches includes all the properties in a single architecture. The straightforward model realization, similar to that of back-propagation networks, allows one to exploit all the techniques already used for BP training, such as adaptive training parameters, second-order algorithms, conjugate-gradient minimization, and so on. In the standard architecture (a single CBP layer), the modification to the BP model can be performed at the pre-processing step. The activation of each unit can be local, thus can be similar to that of RBF networks. However, the transition slope and the region size can be adapted to the actual needs of the representation. The distance-based activation may also suggest the application of competitive training laws to emulate vector-quantization networks in either a supervised or an unsupervised way.
The model belongs to the class of polynomial networks. However, it uses only one parameter more than the linear threshold unit; as a consequence, the Vapnik-Chervonenkis dimension (i.e. the standard measure of the representation capability of a learning machine) is kept small, thus enabling the user to tune the overall VCdim of the whole network by selecting an appropriate architecture (as is usually done for standard BP networks) [2, 6]. This allows the network to reach the desired tradeoff

between good learning during training and good generalization after training. Instead, the general polynomial model is often used in tasks where a large number of small classes have to be stored, without requiring a great amount of generalization, as for recognition of complex but densely sampled patterns (e.g., vocalization of syllables).
In conclusion, the proposed neural model generalizes threshold feedforward networks, by adding a distance-based feature that is tuned during a standard training phase at a very low additional cost. This allows a notable improvement in the representation capabilities of the system. Moreover, the procedure of analyzing the trained network is easier, thanks to the adaptive nature of the representation paradigm itself. However, the model benefits from the advances in research on feedforward networks and back-propagation training, and can exploit them successfully.

References

[1] Anguita D, Parodi G C, and Zunino R "An efficient implementation of BP on RISC-based workstations," *Neurocomputing*, 1994, 6, pp. 57-65.
[2] Baum E B, and Haussler D "What size net gives valid generalization?" *Neural Computation*, No. 1, 1989, pp. 151-160.
[3] Bianchi G, Buffrini L, Monteforte P, Rovetta G, Rovetta S, and Zunino R "Neural approaches to the diagnosis and characterization of the Lyme disease," *7th IEEE Symp. Computer-Based Medical Systems,* Winston-Salem, June 1994, pp. 194-199.
[4] Moneta C, Parodi G C, Rovetta S, and Zunino R "Automated diagnosis and disease characterization using neural network analysis," *IEEE Int. Conf. on Systems, Man and Cybern.*, Chicago, IL, Oct. 1992, IEEE Press, pp. 123-128.
[5] Rumelhart D E, Hinton G H, and Williams R J "Learning internal representation by error propagation," in Rumelhart D E, McClelland J L (Eds.) *Parallel Distributed Processing – Explorations in the Microstructure of Cognition* vols. 1-2, MIT Press, 1986.
[6] Vapnik V N, *Estimation of dependences based on Empirical Data*, Berlin, Springer-Verlag, 1982.