Self-Reflexive Networks

Article *in* Substance Use & Misuse · February 1998

DOI: 10.3109/10826089809115874 · Source: PubMed

Massimo Buscema
University of Colorado
298 PUBLICATIONS 4,194 CITATIONS

Substance Use & Misuse, 33(2), 409–438, 1998

Self-Reflexive Networks

Massimo Buscema, Dr.

Semeion Research Center of Sciences of Communication, Viale di Val Fiorita 88, Rome, Italy

INTRODUCTION

We present a new type of Artificial Neural Networks created by M. Buscema in 1989: the Self-Reflexive Networks (SR). The theoretical presuppositions are that their dynamics are analogous to those ascribed to autopoietic systems: self-referentiality, unsupervised learning and unintentionally cooperative and contractual activities of their own units. We also hypothesize a new concept of perception. We present the basic equations of Self-Reflexive Networks, new concepts such as that of dynamic target, of Re-entry with dedicated and fixed connections, and of MetaUnits. From the applied work presented, a few specifities and novelties of this type of Neural Networks emerge:

- the capability of spontaneously transforming its own learning inaccuracy in analogic capability and original self-organization capability.
- the capability of spontaneously integrating the models that it experienced in different moments in an acronical hyper-model.
- the capability of behaving as if it had explored a decisions graph of large dimensions, both in depth and in extension. The consequence of this is behaving as an Addressing Memory for *self-dynamic* Contents.
- the capability of answering simultaneously from different points of view, behaving, in this case, as a network that builds more similarity models for each vector-stimulus that it receives.
- the capability, through the MetaUnits, to integrate nodes with different saturation speeds in a unitary typology, and, therefore, with different memory; in fact, while the SR units are short memory nodes since each new stimulus zeros the previous stimulus, the MetaUnits memorize the different SR stimulus during time, functioning as an average length

memory. This fact should confirm that the average memory length is of a different level from the immediate memory and that it is based only upon *relations* among perceptive stimuli which are distributed in parallel and in sequence. In this context the weights matrix constitute the SR long-term memory. The MetaUnits, instead, work as a short memory. So, they can influence the same weights matrix during time. In any case, in the SR there are *service nodes* or *filter nodes* and *learning nodes* as if they were weights (the MetaUnits).

THE STRUCTURE OF PERCEPTION

One of the essential characteristics of a living organism is its capability of *adjusting to the environment by making the environment adjust to it* (see Grassé, 1973; Riedle, 1978; Waddington, 1970, 1975).

We refer to that particular interactive process "organism-environment" in which the organism "lives" the environment, trying to "adapt it" according to its life's needs, and simultaneously the environment responds to the organism's actions, forcing it to modify its structure. This complex negotiation changes the dynamic morphology step-by-step of both the environment and the organism defining it (on this level a particular relevance is assumed by Nicolis and Prigogine (Nicolis, 1987)).

In the Supervised Networks, the Network's learning is essentially a passive process: the target models to which the Network must adapt itself are either the Input Models' mirror, or external values decided outside of the network; for example:

Supervised Networks

Input Specular Targets

I		**T**
101	→	101
100	→	100
001	→	001
etc.		

Input Independent Targets

I		**T**
101	→	1
100	→	0
001	→	1
etc.		

On the contrary, the simulation of any organism assumes the capability of *restructuring* the Inputs in that organism, according to the way that organism is structured. For example, it includes generating actions towards the environment, which are the product of the restructuring that the organism is operating.

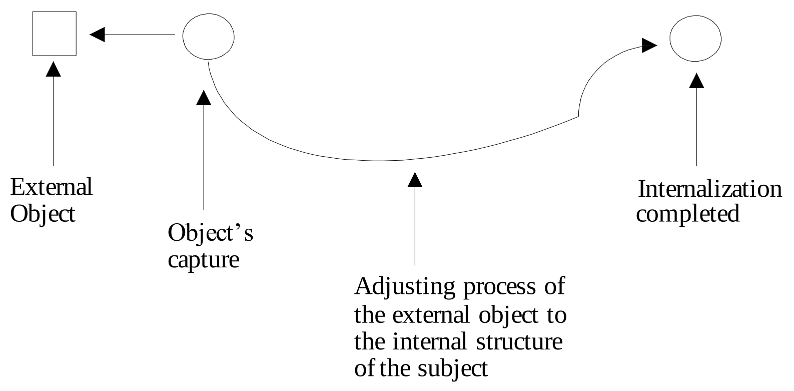
An essential condition is that the Network itself finds a behavioral target which is a subjective interpretation that the Network is making on the Input models it is testing.

By that we mean a neural Network simulating any organism must act as an *autopoietic* system (Maturana, 1980; Varela, 1979).

The *first characteristic* of an autopoietic system is to *perceive* the environment and not to *submit* to it.

At the moment, perception is considered as the restructuring strategy that a subject works out on a portion of the world, in order *to internalize* it into its own organizational structure (Fraisse, 1963; Linsday, 1977).

PERCEPTION


External Object
Object's capture
Adjusting process of the external object to the internal structure of the subject
Internalization completed

It is assumed that this process might be *iterative* (continuously repeated) and *unique* (different each time according to the state of the subject and to the functional nature of the perceived object).

But from our point of view, the perception activity is part of the existence itself of the organism. We mean by perception, that it is a collateral and obligated effect of a living organism's simple existence.

Also from this point of view, perception is a process, but it is not an activity that begins and finishes, nor is it an activity that produces itself in certain moments of the organism's life.

It is, instead, a *continuous* activity. The organism exists *as long as it perceives*. Of course, this continuity will be subjected to a certain *scanning* and to a different complexity, which is linked to the *type* of objects in the world that are perceived and under the *condition* of the perceiving subject who continuously defines the perceptive continuum.

For each living organism therefore, perception is a *socialization of its existence*.

A *second characteristic* of the perceptive activity is its functioning as a process, continuously reminding the organism of its *space-temporal cohesion* (in the same direction as Maturana and Varela (Maturana, 1980)).

That means that the perceptive activity is made up of a multiplicity of parallel and asynchronic micro-perception, although the difference between the external world and the organism is not important for either of them. The significant global effect is in the reformulation and/or reaffirmation of the autonomous topology of the perceiving organism (also agree to this point of view, Minsky 1986, and Rumelhart et al. 1986).

The *third characteristic* of perception should be the activity of trying to compensate for the living organism's *fundamental lack of balance*. This means that each organism is living because it is *structurally unstable*. Such a lack of balance "bounces" from one point of the organism to another, forcing it to continuously "attack" the external world, in order to locally re-balance itself (there are partial traces of these ideas in Von Foerster 1981 and Thom 1972, 1980).

Such dynamics are possible only if we imagine living structures as structures created in a world with at least 4 dimensions. That would imply a continuous instability, due to the major complexity of the organism in relation to the dimension of space within which the organism has to live. This situation would produce the manifestation of rejected dimensions in the form of time (on this point Von Foerster 1981 is the only one that hints about the relevance of the spatial dimension for the organizations of living structure).

Therefore, the organism continuously perceives parts of itself and of the external world as compensation activity for its structural unbalance.

The *fourth characteristic* of the perception process should be that this activity *creates* the units it perceives, on the basis of the perturbations that the organism generates towards its external world. In other words, the perceptive activity does not consist in "taking" units from the environment but rather in *reformatting* the units of the environment in units which are such only within its internal organization (it is a matter of activities that Maturana and Varela define of Self-Referentiality of Autopoietic Systems (Maturana, 1980)).

The Input, therefore, is only a mass of perturbations whose location depends on the location of the sensors of the organism. Such *articulation* depends on the resolution of conflicts between what the sensors of the organism perceive as location and intensity, and the organization of internal lack of balances. At that specific moment they are trying to compensate for this in order to continue to exist space-temporally.

Such an Input *reformatting* rule is not only valid for explaining the relation between environment and organism, but it also explains the interpretative dynamics through which each unit of the organism behaves with respect to any other unit.

Such a concept of perception obligates us to redefine the perception activity itself as a process marked by the following steps:

a) In order to exist, the organism is internally unbalanced; therefore, moved by an internal need, it perturbates the environment, and the energy spent in perturbing the environment balances itself again locally.

b) The perturbed environment reorganizes itself producing, in its turn, "clouds" of perturbations over the organism, which will "feel" these perturbations according to the typology, the location and the sensitivity of its sensor and of its more internal units.

c) The perturbative wave produced by the environment will travel from the outside towards the inside of the organism, according to an external point of view. Actually, each layer of the organism's micro-units will reinterpret the interpretations of the perturbations that have been carried out in the previous layer, with the possibility that the new internal micro-units of the organism may be created by the repetition of these processes.
   Therefore, what the organism feels is a fragmented, distributed and asynchronic set of reinterpretations at a level at which its units work starting from a cloud of "self-produced" perturbations. At this level, a perceiving consciousness does not yet exist.

d) Each reinterpretation that each micro-unit of the organism worked out on the arriving perturbation is the result of what that unit can perceive. It is the result of that unit's specific conditions, at the moment that it received the perturbations and of all the perturbations that the unit exchanges with other micro-units.
   Therefore, the reinterpretation that each unit makes on the variation of its course will simply be the result of the variation of its perturbation's exchanges with other units to which it is connected and/or to which it connects itself. Therefore, a fragmented process of unbalancing begins throughout the entire organism. This process is analogous to a "feedback loop", which could be identified as "training and perception work".

e) This process of unbalancing and readjustments tends to minimize the energy of the original perturbation, in order to reaffirm the space cohesiveness of the organism. Such a minimization begins to produce itself when the organism creates a logic of virtual perturbation towards the environment that emerges, which is more or less isomorphous to the fragmentated logic through which its process of distributed perturbation occurs.

At this point, the perception process begins to stabilize itself as a form of perceptive consciousness. The perceptive consciousness is the compensation that the organism self-generates, as a plan of virtual manipulation of the environment. This plan is isomorphous with the unbalancing that it undergoes and, therefore, it is a project that when it creates itself, it re-affirms the identity of the organism in the environment. *I feel, I change, I comprehend,* therefore *I exist.*

We have described in a linear way the process in which an organism was producing itself as the organism was continuing to produce its micro-improvements even on the environment, which was responding with new stimulations that were rendering the process more complex and in some ways continuous.

The organism, therefore, continuously generates *pieces of perceptive consciousness* every time it can minimize the energy which is self-produced by its continuous internal and external activity.

This activity of perpetual replanning is the virtual manipulation of the environment, which is isomorphic to the internal dynamics which the organism is living in order to exist in the environment. When this insists on repeating the same schemes, it is possible, then, that the virtual manipulation begins to transform itself into reality. This may slowly produce a modification of the topology of the organism itself and produce what we see as a biological change. This change can become genotypical.

In short, the biology describing an organism is a set of perceptions that the organism has come up with in its negotiation process with the environment.

Therefore, a gap between thoughts and biological shapes does not exist. The biological shapes are perceptions which the plan of manipulating the environment has slowly come up with.

From this perspective, the shape of the feline's claws is the biologically "frozen perception" of the reproduction cycle of a predator.

SELF-REFLEXIVE NETWORKS

M. Buscema in 1989, based on this thesis, started experimenting with the possibility of creating *unsupervised* Artificial Neural Networks (ANN), which would be able to use *cooperative* learning algorithms instead of competitive ones (Buscema, 1994b, 1995a; Pandin, 1996).

The importance of the unsupervised ANN is self-evident. Many complex systems are self-organizing and they do not need an outside given target. The importance of cooperative algorithms is less obvious but equally deep-rooted: a competitive process is the radicalization of a cooperative process. The last one, therefore, implies the first one showing all its possible shades.

The Self-Reflexive Artificial Networks were created, and their characteristics are quite simple (Buscema, 1993a).

These ANNs are provided with three layers. The Input layer is made up of a series of “k” nodes which are arranged in order to catch the signals from the environment. The Hidden units “j” layer and the Output units “i” layer actually involve the same units caught at two different moments. The Input layer of the ANN units work as Hidden units when receiving the signals. They appear as Output units when communicating their behavior to the environment.

This means that the number of Hidden and Output units of these ANNs is always the same: every Hidden node has a copy in an Output node.

The connections between the Input and the Hidden layers can be regulated in different ways. We have studied those with *the maximum connection*, *dedicated connections* and *mono-connections*.

The connections between the Hidden and the Output layers must be at the maximum gradient, as a rule, but other possibilities are being experimented.

The propagation algorithm of the signal, in these kinds of ANNs, is very simple and traditional: it can easily be summarized in the following equation:

$$(1) \qquad u_i = f\left(\sum_{j}^{N} u_j \cdot w_{ij} + \theta_i\right)$$

where *f* is a semilinear function of the already known functions (sigmoid, hyperbolic tangent, arc tangent, etc.).

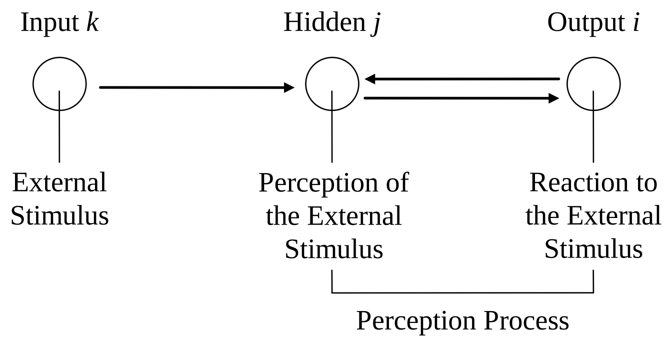
The learning algorithm is based on the Back Propagation (Rumelhart, 1986) and, therefore, it follows the descending gradient method. The Back Propagation error occurs through the comparison between the desired and the real Output at each Output node. For the Self-Reflexive ANN the Delta Output of each node is generated through the comparison between the real Output and the Hidden node value which is a copy of that Output node.

(2) $$\Delta out_i = \left(u_j - u_i\right) \cdot f'\left(u_i^O\right) \cdot \left(u_i^H - u_i^O\right)$$

where: $f'(.)$ = first derivative; u_i^O = i-th Input unit; u_i^H = i-th Hidden unit.

This small difference in the learning equation has great effects on the ANN behavior. The most obvious one is that the ANN will change its target at each cycle even if it is facing an identical Input.

The Self-Reflexive ANNs, therefore, have a dynamic target that they self-program during their learning. This fact makes them similar to *autopoietic systems*. Autopoietic systems determine each time the meaning of each external perturbation, according to their internal dynamics. In this sense, the learning process of a Self-Reflexive ANN is analogous to a perception process. The ANN has perceived some external stimulus when it has succeeded in structuring its own external reaction (Output) with *analogous dynamics* to the one through which it perceived the external stimulus (Hidden). In the figure,


Input *k*
Hidden *j*
Output *i*
External Stimulus
Perception of the External Stimulus
Reaction to the External Stimulus
Perception Process

it is clear that under in these conditions the Self-Reflexive ANN learns when it minimizes the Energy internally. This minimization occurs on the weights matrix. Therefore, the Error function of a Self-Reflexive ANN will be represented by:

(3) $$RMSE = \sqrt{\frac{\sum_{p}^{M} \cdot \frac{1}{2} \cdot \sum_{i}^{N} \left(u_i^H - u_i^O\right)^2}{M}}$$

where: M = number of the Input models; N = number of the Output and Hidden nodes; u^O = Output units; u^H = Hidden units.

Dr. Giulia Massini, of the Semeion Research Center, conducted some experiments on this type of ANN and showed their capability of mapping the Input patterns through the matrix of the Hidden-Output weights (Buscema, 1993a, chapter XIV, paragraph 14.2 and 14.3).

These experiments demonstrated how the Hidden-Output weights matrix tends to stabilize the fuzzy relationship of “*sociability*” and of “*antisociability*” each Input node has with all the others.

In this case, in fact, having assumed that each Input node is a “hypothesis” (Hinton, 1981), it is possible to maintain that a Self-Reflexive ANN generates the weights matrix that allows it to ask questions to this network as if it were a *Constraint Satisfaction* Network (see CS Netwoks in this volume).

Interesting experiments and discoveries have also been carried out on Self-Reflexive ANN with dedicated Input-Hidden connections. More particularly, the behavior of a Self-Reflexive ANN with dedicated connections in pairs towards the Hidden units has been studied: that is, each Hidden unit receives only 2 connections coming from 2 Input units. In this specific case the ANN maps into the Output the weighted and negotiated difference (DNP) between the 2 values of the 2 Input nodes (Buscema, 1993a, chapter VIII and chapter XIV, paragraph 14.4).

Other experiments have been carried out on the behavior of mono-dedicated Self-Reflexive ANN. In this case each input node is connected with only one hidden node.

The Self-Reflexive ANNs have shown 2 different capabilities with this architecture. They are capable of filtering, in order to carry on the analysis of similarities among forms, and of being a fuzzy database.

A monodedicated Self-Reflexive ANN to which different Input models are submitted — for example, the 21 graphic letters of the Italian alphabet — behaves as a Self-Reflexive Network with a quicker maximum gradient. It minimizes the error (RMSE), after a certain number of cycles.

The problem is that the Output mapping of the Input vectors, which the network is able to demonstrate after the learning has occurred, is rather imperfect. This is easy to explain. The ANN is structurally given to stabilize its own Hidden-Output weights by minimizing the differences among the various Input nodes. Therefore, it may occur that it varies at a certain Output node between 0.2 and 0.9, while the variation which it needs in order to distinguish another output node is possibly between 0.45 and 0.61. A Self-Reflexive ANN practically maps the input, *recoding* it in its internal system of differences. Nevertheless, this has the consequence of an imperfect mapping of the Input in Output.

In the monodedicated Self-Reflexive ANN, it is possible to avoid the problem through a normalization equation. In fact, we know that from the Hidden values of each node, it is possible to exactly stabilize the value of the correspondent Input node because in this type of ANN, there is only one weight connecting the Input to the Hidden.

Therefore, if:

$$u_j = \frac{1}{1 + e^{-(u_k \cdot w_{jk} + \theta_j)}} \tag{4}$$

where: $k = j$; u_j = Hidden node value; u_k = Input node value; θ_j = Hidden node bias; w_{jk} = connection between u_k and u_j; e = natural logarithm (e = 2.718281828459)

then:

$$u_k = \frac{\lg\left(u_j\right) - \lg\left(1 - u_j\right) - \theta_j}{w_{jk}} \tag{5}$$

Given that the learning is measured through the minimization of the distance of the Hidden and Output nodes, we can write equation (5) replacing u_j (Hidden) with its copy u_i (Output).

$$u'_k = \frac{\lg\left(u_i\right) - \lg\left(1 - u_i\right) - \theta_j}{w_{jk}} \tag{6}$$

where: $k = j = i$; u'_k = Normalized Output value.

I have called equation (6) a *normalization equation*. It is the one I needed in order to rewrite the Output values of the monodedicated Self-Reflexive ANN. It is clear that through this equation, once the learning has occurred, the mapping of the Input in the Output will appear very effective. But this is not the point. Equation (6) is, in fact, useful in order to ask questions of the ANN, once the learning phase has been concluded.

The interrogation of a monodedicated Self-Reflexive ANN, occurs through a very simple technique that we have elsewhere called *Re-entry* (Buscema, 1994a, chapter 3.2.9). The Re-entry consists in placing again the Output of an ANN in its Input vector until the difference between the Output and the Input will be minimized. In the Feed Forward ANN this technique is only possible with self-associative topologies. The result that we obtain is a *constraint* of the Output generated towards one of the targets that the ANN has previously learned (the most similar and the less far away from the initial Input).

We have used this technique in the mono-dedicated Self-Reflexive network putting the ANN *normalized Output* again in Input.

ADVANCED SELF-REFLEXIVE NETWORK: THE METAUNITS

The experiments carried out have favoured a methodological increase of the SR.

The SR that was submitted to training is, at the moment, the same that is questioned through the Re-entry. This *does not allow the SR* to generate answers on the patterns which it has learned (namely, on the records), but only regarding the properties by which it has learned them (namely, the field options).

Actually, it would be very important that during the Re-entry that the SR organized *resonances* among the records that it activates. This would allow one to enhance the *fuzzy similarity* among the various subjects.

This aim is easily reached in other ANN types. It is sufficient to add to these ANNs a pool of Hidden units, that are not handlable in Inputs, and equal to the number of the records of the DB.

In order to represent a specific record, each added Hidden unit will have some *positive connections* (i.e., +1), with all the properties which make up that record, and some *inhibitive connections* (i.e., -1) with all the other Hidden units that represent the other records.

Once this architecture is set, both the Input properties and the new Hidden units are treated as nodes of the same type by the updating equations of the units.

It is not possible to proceed with the same simplicity within the SR for at least 2 reasons:

- Since the SR are layer ANN, to which layer must the new units that represent the DB be added, with which connections, and where?

- Some ANNs receive the values of their connections directly from the DB architecture. Nevertheless, almost all of them have a specific updating algorithm of the units when they are used in a recall mode. Most of the time, this algorithm is not linked to the algorithm used to train the weights of the network. In the SR, instead, the algorithm for recall mode is part of the learning algorithm during the training phase. It is the same SR that learns and answers.

These problems induced us to proceed in the following way.

In the recall phase, one adds to the SR, *M* output units equal to the number of records that it learned during the training phase. Therefore, if during the training phase the SR will appear configured in the following way:

a. Input level:	$Input_k$	$1 \le k \le N$
b. Hidden Level:	$Hidden_j$	$1 \le j \le N$
c. Output Level:	$Output_i$	$1 \le i \le N$

{$i = j = k$; N = number of Nodes}

in the recall phase the SR will show its Output Level which will be modified in this way:

c. Output Level: $Output_i + NewOutput_p$

{$2 \le p \le M$; M = number of Learned Patterns}

We have called these new Output nodes of the SR "*MetaUnits*". Each MetaUnits is connected to all of the SR Hidden units with *specific weights*. More specifically, the value of each weight between a MetaUnit and all the Hidden units is a *function* of the Input nodes values that represent that MetaUnit. Therefore:

(7) $$w_{pj} = f^{m}(Input_j^p) \qquad \{p = \text{Record};\ j = \text{Hidden Node}\}$$

This is possible because each Hidden node is a *receptor* of the Input node with which it is connected.

The function, f^{m}, of manipulation of the Input is a classic function of MiniMax:

(7a) $$w_{pj} = Scale \cdot Input_j^p + Offset$$

where:

(7b) $$Scale = \frac{(high - low)}{(Max - Min)}$$

(7c) $$Offset = \frac{(Max \cdot low - Min \cdot high)}{(Max - Min)}$$

The *Max* and the *Min* values represent, respectively, the maximum and the minimum value which come into the function. Since the values are all taken from the Input vector, *Max* will be equal to 1.0, and *Min* will be equal to 0.0.

The *high* and *low* values, instead, represent the maximum and minimum values that the weight w_{pi} can take.

These two values are created by the analysis of the weights between the Hidden level and the Output level that the SR has developed during the learning phase. More specifically, one calculates the mean of the absolute value of the matrix weights w_{ij}:

(7d) $$\overline{X}w^o = \frac{\sum_i^N \sum_j^N \left| w_{ij} \right|}{N^2}$$

and, then, the *high* and *low* values are initialized:

(7e) $high = \overline{X}w^o$ and $low = -\overline{X}w^o$

It is also possible to initialize these values through other procedures. For example, instead of the mean of the weights $\overline{X}w$, it is possible to use the function $Max_w\left[\left|w_{ij}\right|\right]$, or $Min_w\left[\left|w_{ij}\right|\right]$.

In any case, the choice of the criteria for stabilizing the maximum and the minimum values of the new weights w_{pi} is a choice linked to the kind of *filter* you want to use. It is a matter of deciding which is the impact power that the Hidden units of the SR must have on the new Output units.

Once we have stabilized the new weights matrix w_{pi} and considered the added Output units, the SR is ready to be questioned through the Re-entry.

During the forward algorithm, in fact, the *NewOutputs* are calculated starting from the Hidden units through the weights w_{pi}, in the same way as for the old Output$_j$ units:

(8) $NewOutput_p = f^s(Net_p)$ where f^s = sigmoidal function

where:

(8a) $Net_p = \sum_j^N Hidden_j \cdot w_{pj}$

and then:

(8b) $NewOutput_p = \frac{1}{1+e^{-Net_p}}$

The *NewOutput*$_p$ values that are calculated at each cycle of the Re-entry process contribute to the global value of the *MetaUnits*$_p$.

The *MetaUnits*$_p$, then, are *saturation* units that increase and/or decrease their total value with respect to the values of the *NewOutput*$_p$ at each Re-entry and with respect to a certain parameter of *activation decay*.

The maximum *decay* parameter is stabilized by the same SR at each Re-entry. We have calculated it as the *mean value* of the values of the all *NewOutput*$_p$ at each Re-entry.

(9) $decay = \frac{\sum_p^M NewOutput_p}{M}$

The subsequent operation consists in calculating the *Delta*$_p$, by which each *MetaUnit*$_p$ will increase or decrease its value at each Re-entry.

We have stabilized this value with respect to the *Net Input* (*Net*$_p$) that each *MetaUnit*$_p$ receives from the others at each Re-entry.

(10) *if* $(Net_p > 0)$

then $Delta_p = Net_p \cdot \left(1 - MetaUnits_p\right)$

$$else \qquad Delta_p = Net_p \cdot MetaUnits_p$$

The Net Input, *Net*$_p$, that arrives at each *MetaUnit*$_p$ is stabilized by the other 2 parameters:

- a function, $f^{\,t}$, of the sum between a function, $f^{\,i}$, of the *NewOutput* value of the *MetaUnit*$_p$ that we are calculating, and the same function $f^{\,i}$ of the value of each of the left *MetaUnits*;
- a linear function of the *MetaUnit* value that we are calculating at that moment.

The equation regulating these parameters is, then, the following:

(11) $$Net_p = f^{\,t}(DpD_p)$$

where $f^{\,t}$ is the function of the *hyperbolic tangent* that binds the Net$_p$ value in a semilinear way among the excluded boundaries of -1 and +1.

(11a) $$Net_p = \frac{e^{DdD_p} - e^{-DdD_p}}{e^{DdD_p} + e^{-DdD_p}}$$

The values *DdD*$_p$ (Degree of Difference) is instead calculated in this way:

(11b)
$$DdD_p = \sum_j^M \left[\left(f^{\,i}(NewOutput_p) - f^{\,i}(NewOutput_j) \right) \cdot \left(1 - MetaUnit_p \right) \right]$$

The function $f^{\,i}$ that adjusts the *NewOutput* values of equation (11b) is a function that we could call a *virtual inversion* of these values. It is, in fact, expressed in this way:

(12) $$f^{\,i}(NewOutput_p) = \frac{\lg(NewOutput_p) - \lg(1 - NewOutput_p)}{\overline{X}_{w^H}}$$

where $\overline{X}_{w^H}$ is the mean of the values which connect the Input level and the Hidden level of the SR (we remind you, the SR is monodedicated); then:

(12a) $$\overline{X}_{w^H} = \frac{\sum_{j,k}^{N} w_{jk}}{N} \qquad j = k$$

At this point it is possible to note the equation which updates the *MetaUnits*$_p$ at each Re-entry.

(13) $$MetaUnits_{p(n+1)} = MetaUnits_{p(n)} + Delta_p - Decay \cdot MetaUnits_{p(n)}$$

The *MetaUnits* update their activation values at each Re-entry by this algorithm between the *included limits* 0 and 1.

All this occurs without the help of any external parameter of the SR. The SR always uses only local information which it has created for itself during the training phase.

The experiments which follow will document the capabilities and the limits of this architecture.

SYNTHESIS OF THE SELF-REFLEXIVE ANN EQUATIONS

We have already described the SR in the previous pages. We think it is useful, therefore, to synthesize the equations of the monodedicated SR that we have used in the following experiments.

a) Forward Algorithm

- $$u_i = f^s\left(\sum_j^N u_j \cdot w_{ij} + \theta_i\right) = f^s\left(Net_i\right) = \frac{1}{1+e^{-Net_i}}$$

b) Backward Algorithm (the exponents I, H, and O are given for the Input, the Hidden and the Output).

- $d_i = u_i^H - u_i^O$

- $\Delta out_{i(n)} = d_i \cdot f'(u_i^O)$

- $\Delta hidden_{j(n)} = \sum_{i}^{N} \Delta out_{i(n)} \cdot w_{ij}^{O} \cdot f'(u_j^H)$

- $\theta_{i(n+1)}^{O} = \theta_{i(n)}^{O} + \Delta out_{i(n)} \cdot Rate$

- $w_{ij(n+1)}^{O} = w_{ij(n)}^{O} + \Delta out_{i(n)} \cdot u_j^H \cdot Rate$

- $\theta_{j(n+1)}^{H} = \theta_{j(n)}^{H} + \Delta hidden_{j(n)} \cdot Rate$

- $w_{jk(n+1)}^{H} = w_{jk(n)}^{H} + \Delta hidden_{j(n)} \cdot u_k^I \cdot Rate$

c) Output Normalization

- $Opposite_i = \dfrac{\lg(u_i^O) - \lg(1 - u_i^O) - \theta_i^H}{w_{ii}^H}$

d) Re-entry

- $G_{(n)} = \sum_{i}^{N} Opposite_i$

- $While \quad G_{(n-1)} \neq G_{(n)}$

  $u_i^I = Opposite_i$

e) Weights of the MetaUnits (p = index of the MetaUnits, M = number of MetaUnits)

- $w_{pi}^{O} = f^{m}\left(u_i^{Ip}\right) = Scale \cdot u_i^{Ip} + Offset$

- $Scale = \frac{high - low}{Max - Min}$; $Offset = \frac{Max \cdot low - Min \cdot high}{Max - Min}$
- $Max = 1.0$; $Min = 0.0$
- $high = \overline{X}w^O$; $low = -\overline{X}w^O$ (w^O = Output weights matrix) ($\overline{X}$ = mean value)
- $\overline{X}w^O = \frac{\sum_i^N \sum_j^N \left| w_{ij}^O \right|}{N^2}$

f) MetaUnits: Forward Calculation

- $NewOutput_p = f^s\left(\sum_j^N u_j^H \cdot w_{pj}^O\right) = f^s(Net_p) = \frac{1}{1 + e^{-Net_p}}$

g) Calculation of MetaUnits

- $Decay = \frac{\sum_p^M NewOutput_p}{M}$
- *if* $(Net_p > 0)$

  then $\Delta_p = Net_p \cdot \left(1 - MetaUnits_p\right)$

  else $\Delta_p = Net_p \cdot MetaUnits_p$

- $Net_p = f^t\left(DdDp\right) = \frac{e^{DdDp} - e^{-DdDp}}{e^{DdDp} + e^{-DdDp}}$
- $DdD_p = \sum_j^M \left[\left(f^i(NewOutput_p) - f^i(NewOutput_j)\right) \cdot \left(1 - MetaUnits_p\right)\right]$
- $f^i\left(NewOutput_p\right) = \frac{\lg\left(NewOutput_p\right) - \lg\left(1 - NewOutput_p\right)}{\overline{X}w^H}$

- $\overline{X}w^H = \frac{\sum_i^N |w_{ii}^H|}{N}$

- $MetaUnits_{p(n+1)} = MetaUnits_{p(n)} + \Delta_p - Decay \cdot MetaUnits_{p(n)}$

POSSIBLE USES OF SELF-REFLEXIVE ANN

When one considers the facility through which SR computes the Input-Output function, these ANN have been shown to be very effective in the *intelligent management* of Data Base and therefore in all CAM (Content Addressable Memory) type of problems.

In this applied field, the comparison between the SR and other ANN models has already been explored (Buscema, 1994b and 1995a). It seems that the SR are able to explore the conditions of a Data Base in a more flexible way than other ANN can. We mean by elasticity or flexibility of an ANN, the capacity of the SR to transfer the eventual contradictions of the external Inputs to each of its own internal nodes, without making them inflexible. This characteristic allows the SR to always suggest very fuzzy and not monotonic solutions in a recall phase (Buscema 1994b and 1995a).

The reason for these SR capacities to manage complex Inputs without simplifying them is due to two factors:

a. They are ANNs which dynamically reformulate their own Outputs during the learning phase. Therefore, they learn to constitute a self-internal or external representation of their own Input. Then, they are not ANNs whose target is already provided in a definitive way from the external in a static form.
b. Both the learning and answer algorithms of the SR are *cooperative* types of algorithm. The SR nodes don't compete among them in the presence of a new Input. They *cooperate by dividing* the codification quantity of the new Input among themselves. This helps the SR not to enlarge the incompatibilities among the different Inputs, but to "catch" the sharpest analogies.

A second use has been tried with the SR. Their ability to map different Input models in an interesting way, from a neurophysiological point of view, was tested (Buscema, 1994b and 1995a; Pandin, 1996).

Figure 1 shows the capability of the SR to create *receptive fields* spontaneously with its own weights at the sliding of *horizontal bars*. Furthermore, we had the possibility to research whether the SR, if appropriately questioned, is capable of discriminating between the sliding of the bars from up to down and from down to up.

Figure 2 shows the creation in an SR of specific receptive fields for its own weights after having codified the sliding of the vertical bars.

Figure 3 shows an analogous example carried out by subjecting the SR to a sliding of *oblique bars* each of different length.

The SR seems to be able to codify, in a two-dimensional map, vectors of N-dimensional Input. This makes them interesting from a neurophysiological point of view.

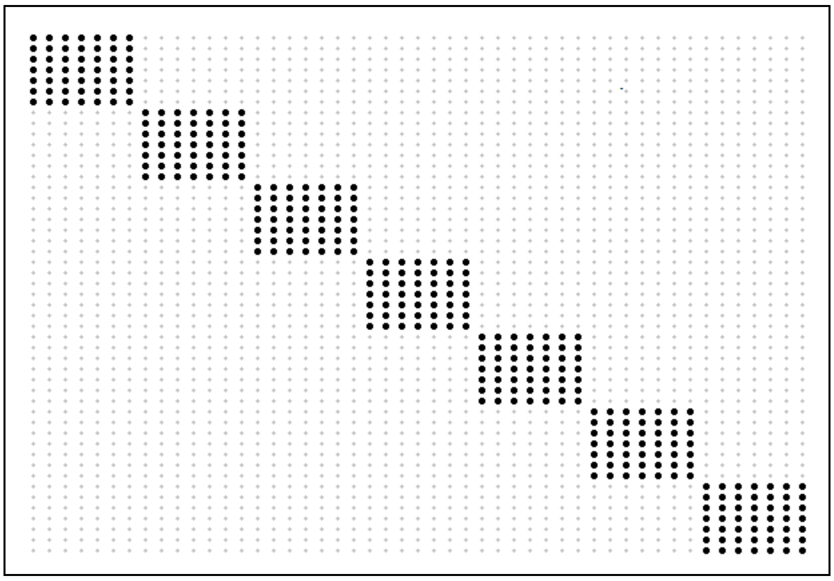


Figure 1. The Hidden-Output weights matrix once the learning has occurred. Receptive fields 7 × 7 took place spontaneously at the sliding of horizontal bars.

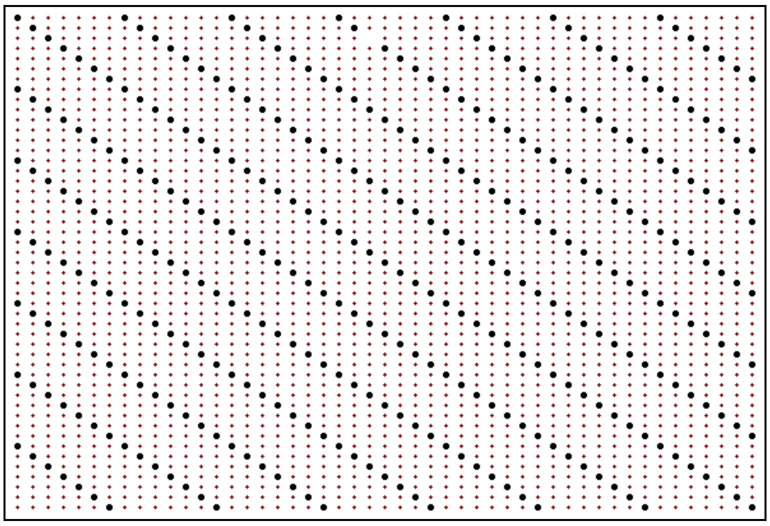


Figure 2. The Hidden-Output weights matrix once the learning has occurred. Specific receptive fields took place spontaneously at the sliding of vertical bars.

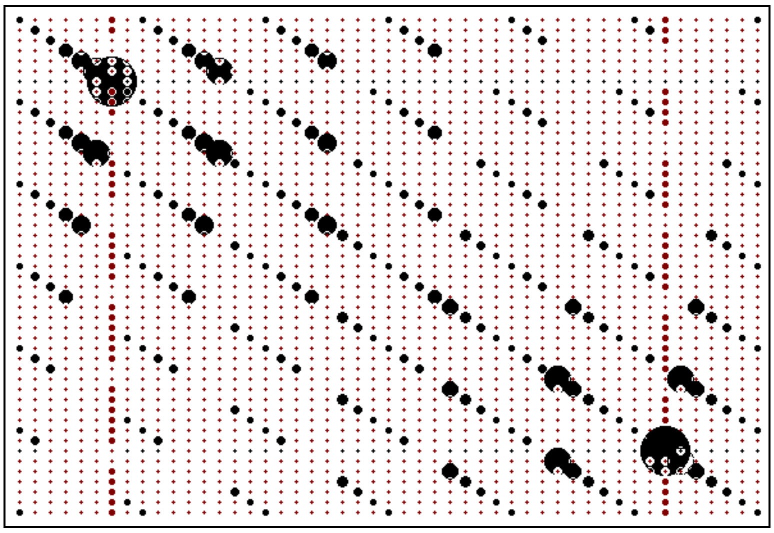


Figure 3. The Hidden-Output weights matrix once the learning has occurred. Specific receptive fields took place spontaneously at the sliding of oblique bars of different lengths.

A third possible use of the SR consists in their capacity to codify the *sequential processes in spatial paradigms* and therefore their consequent capacity to infer *abstract concepts* starting from *figurative details*.

Input models are shown in Figure 4a which document that an SR is able to compute the function after a few hundred cycles.

The Mono-Connections values between the Input nodes and the Hidden nodes of the already trained SR are visualized in Figure 4b. Note how the SR has translated, on this weights values, the succeeding of the perceived Input models. Therefore, it can be asserted that if each of the Input models perceived by the SR is imaginable as a "tree" in a different geographic position of a given space, the SR weights will demonstrate that they have caught the concept of "wood" as a synthesis of the single experiences.

Figure 4c strengthens this hypothesis. It is the visualization of the weight's values which connect the Hidden nodes to the Output nodes of the SR. The figure clearly documents how the SR has projected its *abstract representation* of the Input experiences on which it has been trained, which is also on a different scale from the original one.

The abstracting capacities of the SR are more evident and interesting when one leaves the possibility to an already trained SR to reelaborate its own experiences in complete absence of the Input. It is as if one allows an already trained SR to "sleep" (complete absence of external Inputs) and to reelaborate only the matrices that it has previously formed from its internal weights.

Figures 5a–5d present some of the SR Outputs during 100 cycles of Re-entry carried out in absence of external Inputs. Note how the SR demonstrates through a *chaotic* behaviour that it has understood not only the concept of "wood" (Figure 5a) starting from a "vision" in succession of many trees, but also the concept of "desert" (Figure 5b) and (empty Output) the concept of "complementary wood" (Figure 5c) to the one it has previously inferred (Output of value inverted) and many other synthetic and approximate concepts which are more complex than the simple Inputs through which the SR has been trained.

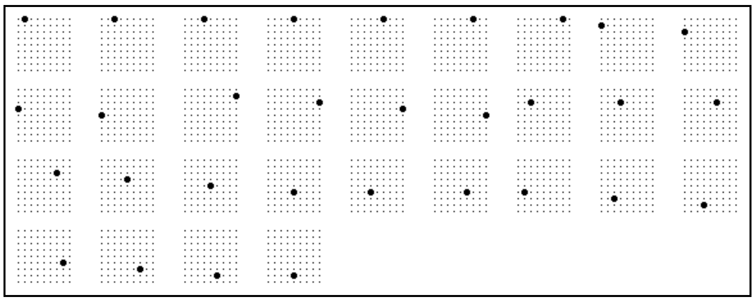


Figure 4a. The 31 Input Models, viewed by the Self-Reflexive Network for 25000 cycles, are grouped 9 × 9 for each of them.

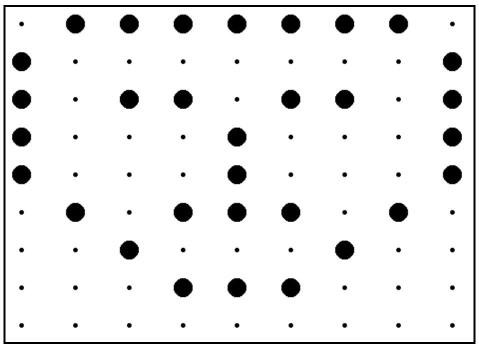


Figure 4b. The 81 Self-Reflexive Input-Hidden weights are shown once the learning has occurred.

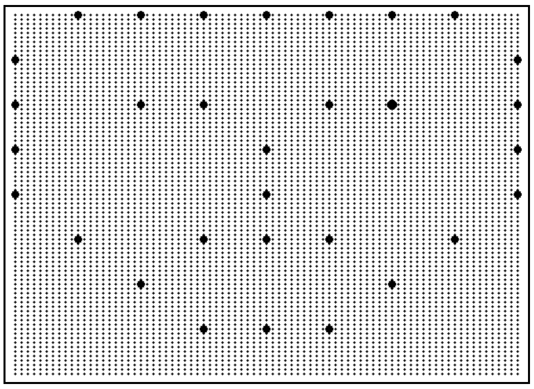


Figure 4c. The Self-Reflexive Hidden-Output weights once the learning has occurred.

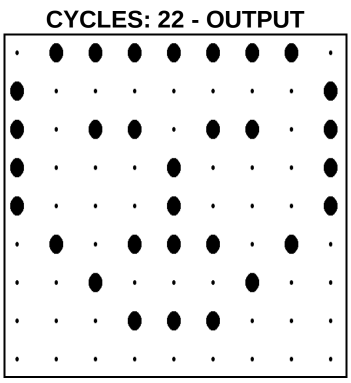


Figure 5a. The Self-Reflexive Outputs shown during 100 cycles of Re-entry in absence of external Inputs. Concept of "Wood".

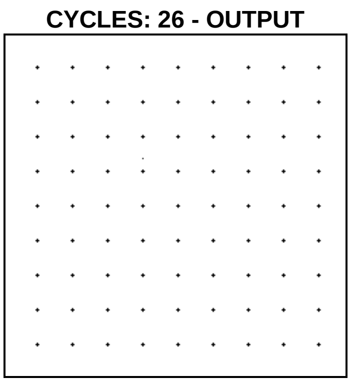


Figure 5b. Concept of "Desert".

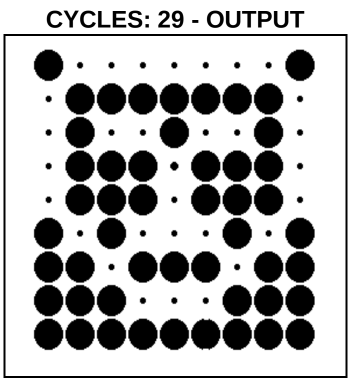


Figure 5c. Concept of "Complementary Wood".

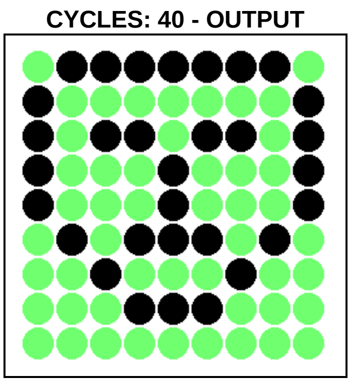


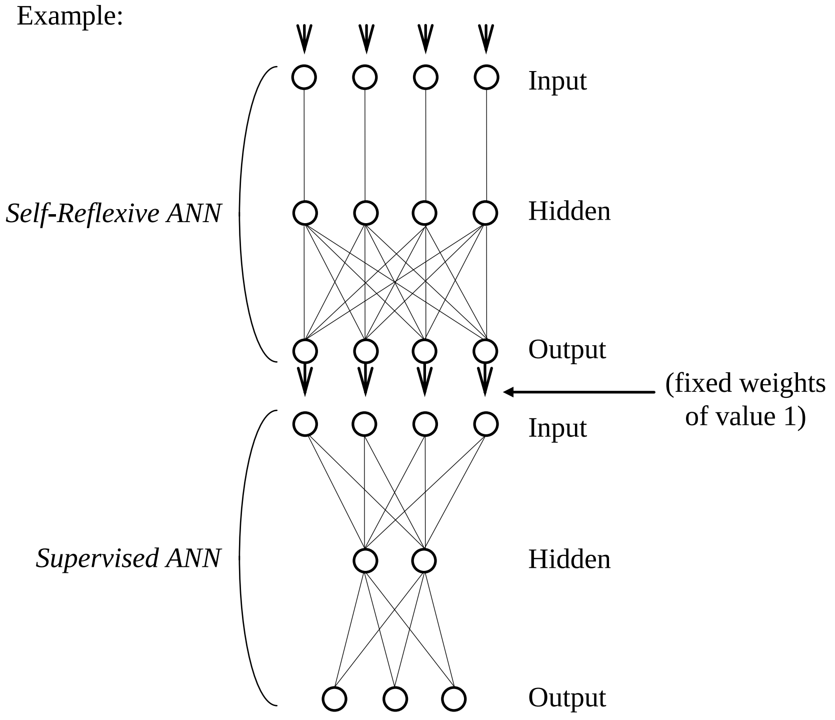
Figure 5d. Concept of "Mixed Wood".

The Self-Reflexive ANNs are also powerful preprocessing instruments.

Their capacity for reorganizing the Input data in Output data can be used as filter and grouping instruments of the relations among the data before these are analyzed by a Supervised ANN.

This is particularly useful in the case of noisy data.

Example:



There are two learning strategies in these cases:

a. The Self-Reflexive ANN is trained first, followed by the Supervised ANN. The latter will use as its own Input the Output values generated by the first.
b. The two ANN are trained together.

At the moment there is no reason for preferring the first or the second strategy. On the contrary, there are inconveniences for the whole pre-processing scheme carried out in this way.

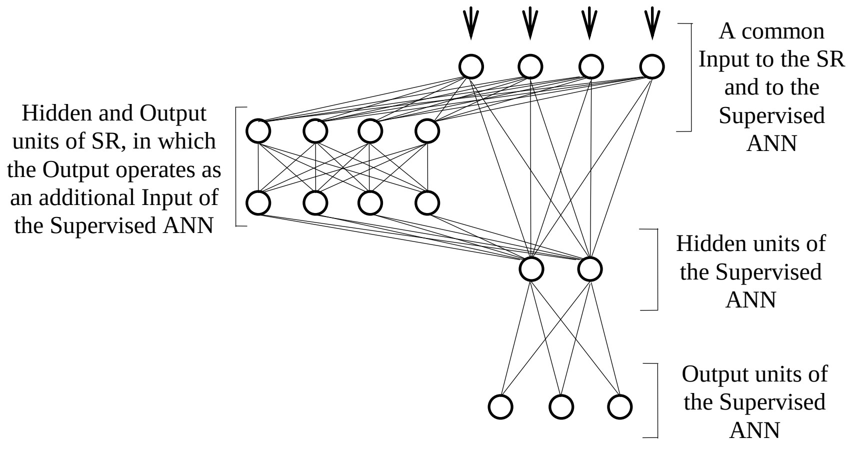
In fact, it can happen that the reorganization of the Input carried out by the SR would magnify the relations which are pertinent to the following function $y = f(x)$ that the Supervised ANN has to compute. In other cases the SR network can narcotize some features which could be useful to a Supervised network in the second step.

We can *double* the Input vector of the Supervised ANN in order to avoid this. It will be composed of the Output generated by the SR and by the original Input of the data.

The Supervised ANN has the advantage of two types of information in parallel: the primitive and *metainformative* produced by the SR.

This procedure has provided very good results in many real applications.

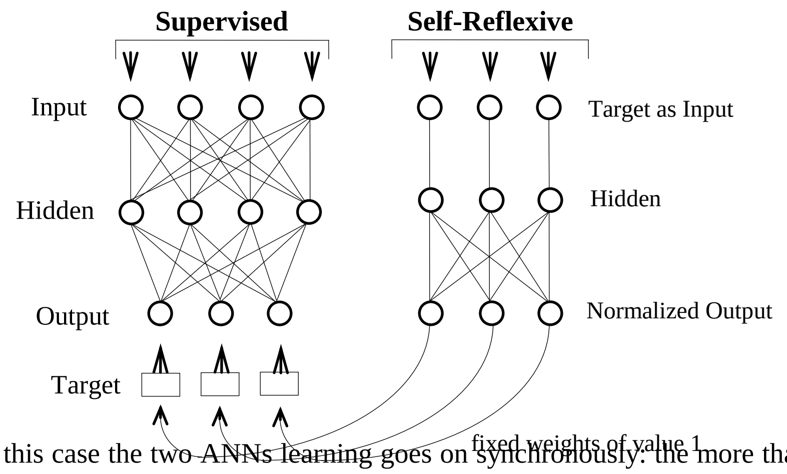
Example:


A common Input to the SR and to the Supervised ANN
Hidden and Output units of SR, in which the Output operates as an additional Input of the Supervised ANN
Hidden units of the Supervised ANN
Output units of the Supervised ANN

A further use of SR ANN has been carried out for the *gradual training* of Supervised ANN.

When the function $y = f(x)$ is demonstrated to be particularly complex, it has been effective to use an SR which provides the target to the Supervised ANN as it learns it more and more.

Example:



In this case the two ANNs learning goes on synchronously: the more that the SR improves its coding of the Target in Output, the more the Supervised ANN is driven towards the codification of a correct Output.

Many different uses and properties of SR are still being studied to currently determine their capacity to codify the temporal sequence of their Inputs.

REFERENCES

ARBIB, 1995: M. A. Arbib (ed.), *The Handbook of Brain Theory and Neural Networks*, The MIT Press, Cambridge, MA, London, England, 1995.

BISHOP: 1995: C. M. Bishop, *Neural Networks for Pattern Recognition*, Clarendon Press, Oxford University Press, New York, 1995.

BUSCEMA, 1993a: M. Buscema and G. Massini, *Il Modello MQ. Reti Neurali e Percezione Interpersonale*, Collana Semeion, Armando Editore, Rome [The MQ Model: Neural Networks and Interpersonal Perception, Semeion Collection by Armando Publisher].

BUSCEMA, 1993b: M. Buscema, *AF1: Shell to Program SelfReflexive Networks*, Semeion Software n. 7, Rome.

BUSCEMA, 1994a: M. Buscema, *Squashing Theory. Reti Neurali per la Previsione dei Sistemi Complessi*, Collana Semeion, Armando Editore, Rome [Squashing Theory: A Neural Network Model for Prediction of Complex Systems, Semeion Collection by Armando Publisher].

BUSCEMA, 1994b: M. Buscema et al., *Reti Neurali AutoRiflessive. Teoria, Metodi, Applicazioni e Confronti*, Quaderni Semeion, Armando Editore, Rome, n. 1 [Self-Reflexive Networks: Theory, Methods, Applications and Comparison, Semeion Research-book by Armando Publisher, n.1].

BUSCEMA, 1994c: M. Buscema, *Constraint Satisfaction Networks*, in BUSCEMA 1994b, pp. 93–126.

BUSCEMA, 1995a: M. Buscema, *Self-Reflexive Networks. Theory, Topology, Applications*, in Quality & Quantity, n. 29, Kluver Academic Publishers, Dordrecht, The Netherlands, pp. 339–403, November 1995.

BUSCEMA, 1995b: M. Buscema, L. Diappi, M. Ottanà, and S. Stabilini, *The Neural Vision of a Competitive City: An Exploration on the Urban Winning Assest in the European Context*, in “Sixth Colloquium of the European Association of Qualitative and Theoretical Geographic”, SPA, Belgium, September 1995.

CARPENTER, 1991: G. A. Carpenter and S. Grossberg, *Pattern Recognition by Self-Organizing Neural Network*, The MIT Press, Cambridge, MA, 1991.

CHAUVIN, 1995: Y. Chauvin and D. E. Rumelhart (eds.), *Backpropagation: Theory, Architectures, and Applications*, Lawrence Erlbaum, Hillsdale, N.J., 1995.

FOERSTER, 1981: H. von Foerster, *Observing Systems*, Intersystems Publications, Seaside, CA, 1991.

FRAISSE, 1963: P. Fraisse and J. Piaget (eds.), *Traité de Psychologie Experimentale: IV. La Perception*, Press Universitaires de France, Paris, 1963.

GRASSÉ, 1973: P. P. Grassé, *L'Evolution du Vivant*, Albin Michel, Paris, 1973.

GROSSBERG, 1982: S. Grossberg (ed.), *Studies of Mind and Brain*, vol. 70 of Boston Studies in The Philosophy of Science, D. Reidel Publishing Company, Dordrecht, The Netherlands, 1982.

GROSSBERG, 1987a: S. Grossberg (ed.), *The Adaptive Brain, Vol. I: Cognition, Learning, Reinforcement and Rhythm*, North Holland, Amsterdam, 1987.

GROSSBERG, 1987b: S. Grossberg (ed.), *The Adaptive Brain, Vol. II: Vision, Speech, Language and Motor Control*, North Holland, Amsterdam, 1987.

GROSSBERG, 1988: S. Grossberg (ed.), *Neural Networks and Natural Intelligence*, MIT Press, Cambridge, MA, 1988.

HINTON, 1981a: G. E. Hinton and J. Anderson, *Parallel Models of Associative Memory*, Erlbaum, Hillsdale, N. J., 1981.

HINTON, 1981b: G. E. Hinton, *Implementing Semantic Networks in Parallel Hardware*, in HINTON 1981a, pp. 161–188.

HINTON, 1987: G. E. Hinton and T. J. Sejnowski, *Neural Network Architectures for A.I.*, Tutorial n. MP2, AAAI87, Seattle, WA, 1987.

HINTON, 1988: G. E. Hinton and J. L. McClelland, *Learning Representation by Recirculation*, Proceedings of IEEE Conference on Neural Information Processing Systems, November 1988.

HOPFIELD, 1982: J. J. Hopfield, *Neural Networks and Physical Systems with Emergent Collective Computational Abilities*, Proceedings of the National Academic Sciences USA, Bioscience, n. 79, 1982, pp. 2554–2558.

HOPFIELD, 1984: J. J. Hopfield, *Neurons with Graded Response have Collective Computational Properties like those of Two-State Neurons*, Proceedings of the National Academic Sciences USA, Bioscience, n. 81, 1984, pp. 3088–3092.

KOHONEN, 1984: T. Kohonen, *Self-Organization and Associative Memory*, Springer Series in Information Sciences, vol. 8, Springer-Verlag, Berlin, 1984.

KOHONEN, 1988: T. Kohonen, *The Neural Phonetic Typewriter*, Computer, 21(3), 1988, pp. 11–22.

KOHONEN, 1995: T. Kohonen, *Self-Organizing Maps*, Springer Verlag, Berlin, Heidelberg, 1995.

KOSKO, 1992a: B. Kosko, *Neural Networks and Fuzzy Systems: A Dinamical Systems Approach to Machine Intelligence*, Prentice Hall, Englewood Cliffs, N.J., 1992.

KOSKO, 1992b: B. Kosko, *Neural Networks for Signal Processing*, Prentice Hall, Englewood Cliffs, N.J., 1992.

LINDSAY, 1997: P. Linsday and D. Norman, *Human Information Processing*, Academic Press, New York, 1977.

MATURANA, 1980: H. Maturana and F. Varela, *AutoPoiesis and Cognition. The Realization of Living*, by D. Reidel Publishing Company, Dordrecht, Holland, 1980.

McCLELLAND, 1988: J. L. McClelland and D. E. Rumelhart, *Explorations in Parallel Distributed Processing*, The MIT Press, Cambridge, MA, 1988.

MINSKY, 1969: M. Minsky and S. A. Papert, *Perceptrons*, The MIT Press, Cambridge, MA (Expanded Edition 1988).

MINSKY, 1986: M. Minsky, *The Society of Mind*, Simon & Schuster, N.J., 1986.

NICOLIS, 1987: G. Nicolis and I. Prigogine, *Exploring Complexity. An Introduction*, R. Piper GmbH and Co. KG, Munich, 1987.

PANDIN, 1996: M. Pandin and G. Didoné, *Chaos in Information Processing: Simulation of a biological learning process by time evolution in an Unsupervised Neural Network*, in International Journal of Bifurcation and Chaos, World Scientific Publishing Co., vol. 6, n. 1, 1996, pp. 203–210.

RIEDL, 1978: R. Ridle, *Order in Living Organism*, Wiley Interscience, Chichester, New York, 1978.

RIPLEY, 1996: B. D. Ripley, *Pattern Recognition and Neural Networks*, University of Oxford, Cambridge University Press, Cambridge, MA, 1996.

RUMELHART, 1986: D. E. Rumelhart and J. L. McClelland (eds.), *Parallel Distributed Processing, Vol. 1: Foundations, Explorations in the Microstructure of Cognition. Vol. 2: Psychological and Biological Models*, The MIT Press, Cambridge, MA, 1986.

SCHMUCHER, 1984: K. J. Schmucher, *Fuzzy Sets, Natural Language Computation, and Risk Analysis*, Computer Science Press, Rockville, Maryland, 1984.

THOM, 1972: R. Thom, *Stabilité Structurelle et Morphogenese. Essay d'une Thèorie Générale des Modèls*, Inter Editions, Paris, 1972.

THOM, 1980: R. Thom, *Modèles Mathematiques de la Morphogénèse*, Christian Bourgois, Paris, 1980.

VARELA, 1979: F. Varela, *Principles of Biological Autonomy*, Elsevier, North Holland, New York, 1979.

WADDINGTON, 1970: C. Waddington (ed.), *Toward a Theoretical Biology*, Edimburgh University Press, 3 Volumes, Edinburgh, 1970.

WADDINGTON, 1975: C. Waddington, *The Evolution of an Evolutionist*, Edinburgh University Press, Edinburgh, 1975.

ZADESH, 1987: L. A. Zadesh, *Fuzzy Sets and Applications*, John Wiley & Sons, New York, 1987.

THE AUTHOR

Massimo Buscema, Dr., computer scientist, expert in artificial neural networks and adaptive systems. He is the founder and Director of the Semeion Research Center of Sciences of Communication, Rome, Italy. He is formerly Professor of Science of Communication, University of Charleston, Charleston, West Virginia, USA, and Professor of Computer Science and Linguistics at the State University of Perugia, Perugia, Italy. He is a member of the Editorial Board of *Substance Use & Misuse*, a faculty member of the *Middle Eastern Summer Institute on Drug Use*, co-editor of the Monograph Series *Uncertainty* and co-creator and co-director of *The Mediterranean Institute*. He is consultant of *Scuola Tributaria Vanoni* (Ministry of Finance), *Ufficio Italiano Cambi* (Bank of Italy), *ENEA* (Public Oil Company), *Sopin Group* (Computer Science Corporation) and many Italian Regions. He has published books and articles; among them: *Prevention and Dissuasion*, EGA, Turin, 1986; *Expert Systems and Complex Systems*, Semeion, Rome, 1987; *The Brain within the Brain*, Semeion, Rome, 1989; *The Sonda Project: Prevention from self and heterodestructive behaviors*, Semeion, Rome, 1992; *Gesturing Test: A Model of Qualitative Ergonomics* ATA, Bologna, 1992; *The MQ Model: Neural Networks and Interpersonal Perception*, Armando, Rome, 1993; *Squashing Theory: A Neural Networks Model for Prediction of Complex Systems*, Armando, Rome, 1994; *Self-Reflexive Networks: Theory, Topology, Application*, Quality & Quantity, 29, Kluver Academic Publishers, Dordrecht, Holland; *Idee da Buttare*, Edizioni Sonda, Turin, 1994; *Artificial Neural Networks and Finance*, Armando, Rome, 1997; *A General Presentation of Artificial Neural Networks*, in *Substance Use & Misuse*, 32(1), Marcel Dekker, New York, 1997; *The Sonda Project: Prevention, Prediction and Psychological Disorder*, in *Substance Use & Misuse*, 32(9), Marcel Dekker, New York, 1997.