Proceedings of the 2006 WSEAS International Conference on Mathematical Biology and Ecology, Miami, Florida, USA, January 18-20, 2006 (pp174-201)

NEW RECURRENT NEURAL ARCHITECTURES

Massimo Buscema, Marco Breda and Stefano Terzi (2005)
Semeion - Research Center of Science of Communication
Via Sersale 117
00128 Roma, Italy
tel. 0039 – 06 – 50652350
fax 0039 – 06 – 5060064

Proceedings of the 2006 WSEAS International Conference on Mathematical Biology and Ecology, Miami, Florida, USA, January 18-20, 2006 (pp174-201)

New Recurrent Neural Architectures

Abstract:

This paper presents two new neural networks, the TASM (Temporal Associative Subject Memory) and the SelfRecurrent network, described as a complex types of recurrent organisms. After a short general definition of recurrent neural networks we introduce the theoretical structure of the new architectures. The paper shows two relevant applications on an medical datasets which show the good classification performances of the networks proposed. We conclude with some reflection about the concept of meta-learning and capability of these architectures to act as meta-learning systems.

Keyword: Recurrent neural network, Dynamic system, Organism, Meta-learning, Artificial Intelligence.

1 Introduction

This paper introduces two new recurrent ANNs, based on the interaction of two ANNs, a supervised one and an unsupervised: the SelfRecurrent and TASM (Temporal Associative Subjective Memory) networks.

In the theoretical sections these new ANNs are presented as a special type of recurrence. In this case, in fact, the recurrence is not a feature of the process to be modeled, but it is a specific feature of the learning process itself. Consequently, these new ANNs can be used also for pattern recognition, where there is not a sequential implication among the input patterns.

Our analysis of recurrent networks will be based on a new concept of "timing". We named "exogenous" the timing of a network depending *only* upon external events. We will also refer to a kind of complex internal dynamics timing as "endogenous". For example, in Hopfield networks [8,9] we have more than a relaxation step for a single input variation. This exists, more specifically, with the interactions between more than one network. As we will see, these different timings make the networks we present effective systems that learn to learn: the *(n+1)-th* input model is not accompanied

with simple "traces" carried out on the previous *n-th* model, but rather with a *synthetic elaboration* (clustering) of the memory of the previous model.

In the experimental section we will show a synthesis of the results of two relevant studies obtained using this kind of networks and recently published: prediction of malignancy of lesion in contrast-enhanced mammography and our study on Alzheimer pathology. These results reinforce that the complex dynamics that characterizes these new types of networks, as system composed of two networks that cooperate and coordinate each other in a spontaneous way, could help to limit consequences of noisy data and of "overfitting" effect during the training phase and to allow for more accurate extrapolation during the recall phase.

In the conclusive section we discuss the main reason to design and to propose Tasm and SelfReflexive ANNs: the simulation of a complex and dynamic meta-learning process. At the most simple level human learning consists also to reflect on own learning process. Human learning is a complex way to learn to learn, a natural recurrent process.

2 Theory

Generally, a generic neural network is considered to be a discrete-time system on an input subspace $\Re^N$ and an output subspace $\Re^M$.

Let us consider this network during the recall phase, when all of its characteristic parameters are kept fixed. At each temporal step the network will associate to the input presented pattern, i.e. a X(t) vector belonging to the input subspace, an output pattern, i.e. a Y(t) vector in the output subspace.

If a Non Recurrent network works as a combinatory system, i.e. associating output patterns to input patterns according to the following relationship:

$$Y(t) = F(X(t))$$

a Recurrent Networks operate, on the contrary, as a sequential system, i.e. associating output patterns to input patterns depending upon an inner state which evolves, on its turn, with the presented inputs over time.

If we represent the internal state with a vector $Z(t)$, defined in a subspace $\Re^K$, we can determine network behavior according to two equivalent pairs of relationship:

$$\begin{cases} Z(t+1) = F(Z(t), \quad X(t)) \\ Y(t) = G(Z(t)) \end{cases} \tag{1}$$

called Mealy's Equations, or:

$$\begin{cases} Z(t+1) = F(Z(t), \quad X(t)) \\ Y(t) = G(Z(t), \quad X(t)) \end{cases} \tag{2}$$

called Moore's equations.

In general, a Recurrent Network does not establish an association between input patterns and out patterns, but an association between the set composed by the initial inner state and the sequence of input patterns and a sequence of output patterns.

Recurrent Networks basically are defined in terms of *equations* and in terms of *functions*. Equations of a Recurrent Network are characterized by the presence of state variables, which create a sequential system. The sequence of patterns constitutes a trajectory within the relative phase space. Basic functions are characterized by the creation of an association between input and output trajectories rather than by the creation of an association of points (mapping) between spaces themselves. The latter is typical in non-recurrent networks.

3 Internal Recurrence

A new concept is introduced in the analysis of the Recurrent Neural Networks: the "timing". When a simple FF network is trained or recalled, timing of the interaction between the network and the environment (essentially training and testing data) is fixed and is due only to the presentation of an input and a target pattern. Thus every time that there is a new input target pair, a new step manifests itself in the network. Given that the timing of a network depends *only* upon external events we named this timing "exogenous". More timings can operate in other kinds of networks,

which are defined by the internal dynamic evolution of the network. For example, in Hopfield networks we have more than a relaxation step for a single input variation. This exists, more specifically, with the interactions between more than one network. We will refer to this kind of complex internal dynamics timing as "endogenous".

This two different type of temporal dynamics allow to refine the concept itself of recurrence in ANNs literature.

Traditionally, the concept of recurrence is connected to an exogenous concept of time: there is recurrence when the learning process is modified by the sequential order of the input patterns. We can figure out a new type of ANNs able to generalize the concept of recurrence taking account the endogenous time of the learning process itself : there is recurrence when the learning process has its sequential timing, independent from the sequential order of the input patterns. The ANN , in this case, will be a combinatorial machine from a external viewpoint, but its inside learning process will work as a sequential machine.

Briefly, these new theoretical ANNs have to show a combinatorial dynamics in relation to the whole patterns, but they also have to show a sequential dynamics for each single pattern.

This independency of internal time from the external time happens with an automatic reset of the state vector when a new pattern is processed by the ANNs.

Consequently, we can figure out also an ANN able to manage this two independent dynamics simultaneously : in this case, the ANN does not reset its state vector when a new input is processed, mixing, in this way, an internal recurrence with an external one.

So, the new ANNs that we are going to define into details will show a meta-learning dynamics. This ability will consist in two features :

- their sequential learning process for each input pattern, not influenced by the other patterns;

- their capability to learn and cluster dynamically its own hidden layer, using a server ANN able to build in a recurrent way the state vector of the ANNs.

4 Composed networks

TASM and SelfRecurrent networks, the new ANNs we are going to define, represent a test for the hypothesis that meta-learning process can increase the learning capabilities of a ANN.

Such architectures are based on the consideration that networks belonging to the Feed Forward family as well as simple Recurrent Networks, such as Jordan's and Elman's [7,9,10,11], could present, at the same time, both a weakness and a strength in their makeup. Their layer of hidden units codes input vectors in a potentially *distributed* way, a very effective coding system from a computational point of view. But this kind of input vector coding is practically uncontrollable in terms of its power: we do not know which is the most effective one based upon the relationship which each input variable has with the others among many possible ways through which hidden units codify input vectors. An attempt have been made to answer this question, with TASM and SelfRecurrent architectures.

Two variants of these networks, obtained by suitably introducing meta-arches in the respective graphs will be described.

In order to better explain the two networks, we prefer, also in this case, to describe them, with the respective variants, autonomously one another, proposing in each exposition possible analogous aspects.

4.1 TASM networks

TASM networks were created by M.Buscema in the second half of the 1990s at the Semeion, Research Center of Sciences of Communication, Rome.

TASM networks are constituted, topologically, by a composition, in a feedback loop, of a normal Feed Forward network, a SelfReflexive (SR)

network with free hidden units, drift model [3] and an additional layer of PEs, called *extended input* (figure 1).

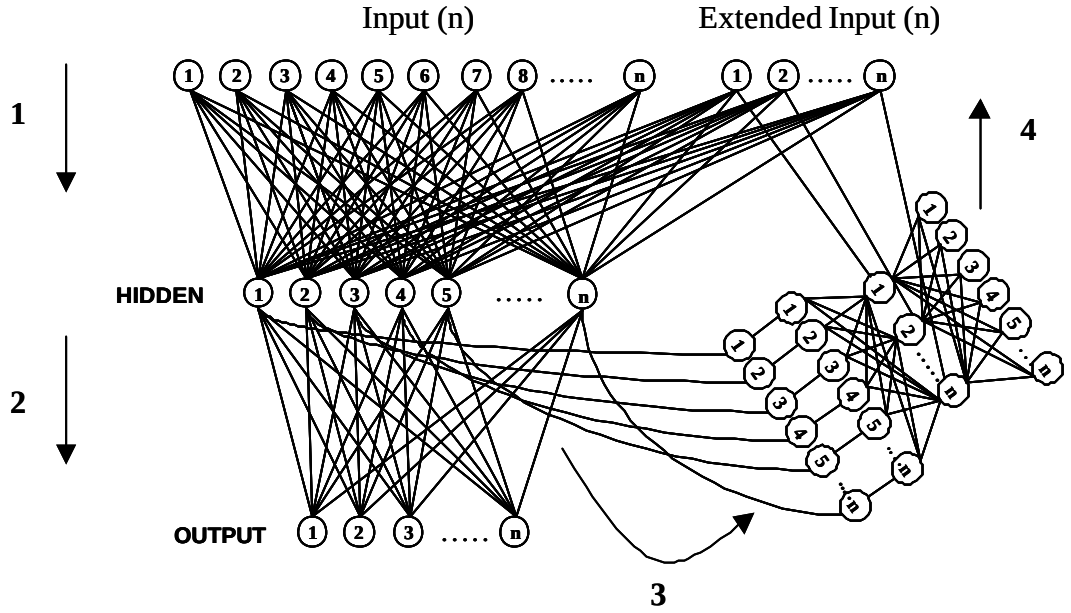

Figure 1 - Representation of a TASM network with signal flow

The basic structure of the SR consists of four layers: an input layer which passively receives the signal from environment and connected by a single weight to each of the units of the hidden layer which interprets the signal; a further layer of hidden units, free units not subject to any external constraint, whose number is completely independent of the number of units in the other layers. The connection between the hidden and free layers and free output was designed to be maximum gradient. SR networks are not supervised with the law of learning based on descending gradient. They define target by themselves and modify weights to match output nodes activation with the target, equal to the activation of hidden nodes.

The SR network receives in input the feedback signal from PEs of the hidden layer of the Feed Forward, through mono-dedicated connections. Outputs from PEs of the free hidden layer of the SelfReflexive are connected, through mono-dedicated connections, to correspondent PEs of the extended input layer. Outputs from PEs of the extended input layer are

distributed finally in input, to PEs of the Feed Forward hidden layer, through full grid connections.

Inner connections of Feed Forward and SelfReflexive networks, as well as those connecting the extended input layer to the Feed Forward hidden layer have an adaptive weight. On the other hand, connections linking the Feed Forward hidden layer to the SelfReflexive input layer, and those linking the SelfReflexive free hidden layer to the extended input layer have a fixed weight, equal to 1, since their only function is to copy or "clone" the activation value.

All activation functions work on the weighted sum of inputs. They are linear for input layers of Feed Forward, SelfReflexive and for the extended input layer, since such layers work only as a buffer. For all other layers they are tipically sigmoidal.

We introduce the following notation:

- $w_{ij}^{[hid]}$ and $w_{ij}^{[inE-hid]}$: input-hidden weights and extended input-hidden weights from *j-th* PE of extended input layer to *i-th* PE of hidden layer;
- $x_i^{[hid]}, x_i^{[in]}$ and $x_i^{[inE]}$: activation values of *j-th* PE of the hidden, input and extended input layers;
- $\vartheta_i^{[hid]}$: bias value of *i-th* PE of the hidden layer;
- *Nin, NinE:* number of PEs of the input layer and the extended input layer;
- $f(\cdot)$ a generic function.

The equation for the calculation of the activation values of the hidden layer is the following:

$$x_i^{[hid]} = f\left(\sum_{j=1}^{Nin} x_j^{[in]} \cdot w_{ij}^{[in-hid]} + \sum_{j=1}^{NinE} x_j^{[inE]} \cdot w_{ij}^{[inE-hid]} + \vartheta_i^{[hid]} \right) \qquad (3)$$

The SelfReflexive is used in this architecture to find out and code, with its free units, some characteristic properties of the pattern presented in input. In this case the presented pattern coincides with the hidden layer of the Feed Forward. Therefore, characteristics of its own input code are presented

again as an extended input to the Feed Forward. Moreover we want the network to have a certain number of cycles available to learn its input code. For this reason, presenting to the Feed Forward a pattern in input, calculation flow recycles more times through the SelfReflexive, producing at the same time the correspondent output patterns.

This network manifests both endogenous and exogenous timing:

- An input pattern is presented at each step of the external time,
- At each step of an internal time some cycles are carried out with the Self-reflexive network in order to learn the code of the hidden Feed Forward layer,
- Output patterns are presented.

So one step of the external time corresponds to more steps of the internal time. The activation value of the extended input (called "state") reached by the network at the last internal step is used again to calculate the output of the following pattern.

There are two kinds of behaviors of the network: the basic one, called *fixed feedback* and the advanced one, called *adaptive feedback*. They differ basically in the output condition from the internal cycles on the SelfReflexive. In the basic version of *fixed feedback*, we have a flow according to the steps described as follows, where number *k* of cycles, usually two, is a network parameter:

```
Fixed_Feedback_TASM_Flow
{
 extended_input=0;
 While(enough_patterns)
 {
  Get_input_pattern;
  num_cycles=k;
  do
  {
   Compute_hidden_activation;
   Compute_Selfreflexive_output_activation;
   Compute_extended_input_activation;
   Correct_Selfreflexive_weights;
   num_cycles= num_cycles -1;
  } while(num_cycles>0);
  Compute_hidden_activation;
  Compute_output_activation;
  Correct_hidden-output_weights(*);
  Correct_input-hidden_weights(*);
  Correct_extended_input-hidden_weights(*);
 }
}
```

The correction learning coefficient for all weights is equal to zero (i.e., all weights of the supervised network remain fixed in the particular case of recall only,) while the SelfReflexive weights are changed. The (*) steps are not performed during recall for the above described algorithm.

The *adaptive feedback* behavior makes the number of cycles adaptive through the SelfReflexive network. The adapting criterion can be chosen in different ways. One effective choice has been to compare the error of each cycle on the SelfReflexive to the error of the previous cycle until the error diminishes.

In this case, the necessary calculations follow these steps:

```
Error_Adaptive_Feedback_TASM_Flow
{
 extended_input=0;
 While(enough_patterns)
 {
  Get_input_pattern;
  selfreflexive_error=0;
  do
  {
   Compute_hidden_activation;
     Compute_Selfreflexive_output_activation;
     Compute_extended_input_activation;
     previous_selfreflexive_error = selfreflexive_error;
     selfreflexive_error =Compute_selfreflexive_error;
     Correct_Selfreflexive_weights;
  } while(selfreflexive_error > previous_selfreflexive_error);
  Compute hidden activation;
  Compute_output_activation;
  Correct_hidden-output_weights(*);
  Correct_input-hidden_weights(*);
  Correct_extended_input-hidden_weights(*);
 }
}
```

The (*) steps are not performed during recall.

TASM networks, from a functional point of view, are adaptive systems able to dynamically control the effectiveness of the hidden unit's coding of the input vectors through the synthesis operated by the free units of the SR. We can say that it is a system that learns to learn.

In a TASM network *(n+1)-th* input model is not accompanied with simple "traces" which the network had carried out on the previous *n-th* model, but rather with a *synthetic elaboration* (clustering) of the memory of the previous model. SelfReflexive free units supply the values for what we could define a *metamemory*.

We experimentally verified that SelfReflexive free units tend to code the principal components (with a eigenvalue >1) of their own input models. We know that the other values free units coding are not at all linearly correlated either between themselves or with those expressing the main components.

From a different point of view, this network architecture seems to simulate the transformation that happens in the brain between short-term memory and middle term memory. We believe, in fact, that SelfReflexive free units work as *active and adaptive selectors* of the traces of short-term memory. The free units, in fact, transform the latter into more stable components of a

syntactic level. They learn the model *x*, coded in *H* by the hidden units of the Feed Forward network, re-code in *G*, where *G* has a cardinality lower than *H* and represents the syntactic relationships that *G* has with the previous experiences, which have already been coded by the Supervised Network.

4.1.2 Dynamic Tasm Networks

In order to increase the architecture feedback dynamism, an interesting modification of the TASM networks consists in introducing in the activation functions of the Feed Forward hidden layer a dependence of the weights of the input-hidden connection on the activation values of the extended input. In the graph this dependence corresponds to the introduction of what we define a node-connection meta-arch. We add a connection to each input-hidden weight so that the *j-th* PE of the hidden layer will be the product of the activation values of all PEs of the extended input. The Dynamic Tasm Network (TASMDa) is illustrated in figure 2.

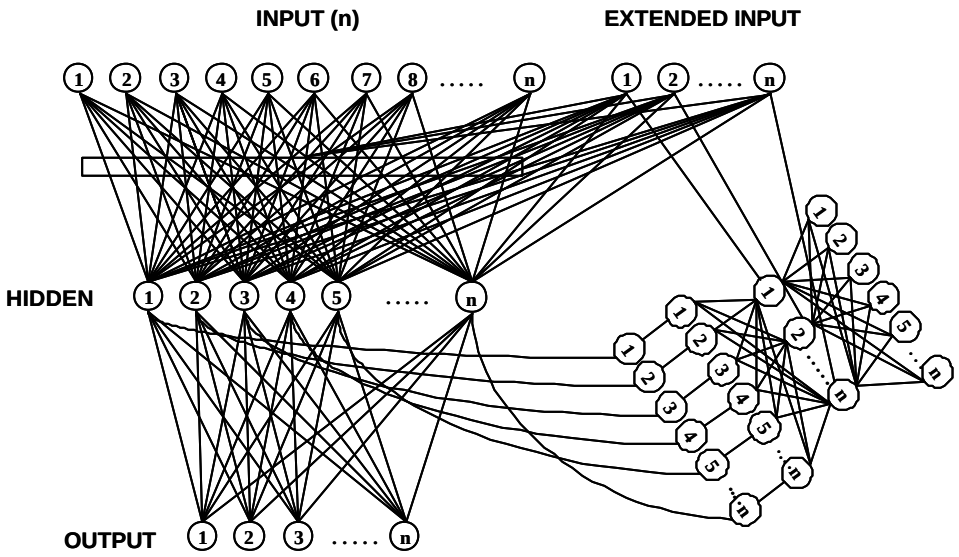

Fig. 2 - Representation of a dynamic TASM Network

Referring to the analogous expression for the activation of the hidden units of basic TASM, we have in this case:

$$x_i^{[hid]} = f\left(\sum_{j=1}^{Nin}\left(x_j^{[in]} \cdot w_{ij}^{[in-hid]} \cdot \prod_{k=1}^{NinE} x_k^{[inE]}\right) + \sum_{j=1}^{NinE} x_j^{[inE]} \cdot w_{ij}^{[inE-hid]} + J_i^{[hid]}\right) \quad (4)$$

4.2 SelfRecurrent networks

SelfRecurrent networks were ideated by to M.Buscema in 1996 [4].

From a topological point of view, SelfRecurrents Networks are composed in a feedback loop, by a normal Feed Forward network, an Auto-Associative Feed Forward network with two layers [1,14] and an additional layer of PEs, called *extended input* (fig. 10).

Auto-Associative network are non supervised, with two or more layers. The learning law is based on descending gradient. They present the same number of node in its layers since training target is to duplicate the input array on the output units. In the Self Recurrent architecture this network have only two layers and the weights connecting same index nodes are deleted to avoid convergence toward trivial solution.

More specifically, the Auto-Associative network receives the feedback signal in input from PEs of the Feed Forward of the hidden layer, through mono-dedicated connections. Outputs from PEs of the output layer of the Auto-Associative are connected, through mono-dedicated connections, to correspondent PEs of the extended input layer. Outputs from PEs of the extended input layer are distributed in input to PEs of the Feed Forward hidden layer, through complete net connections.

The patterns processed by the Auto-Associative coincide with the outputs of the hidden layer of the Feed Forward. As the target of the former is to reproduce patterns presented in input, through a filter enhancing the characteristic properties of the input space, the result of the feedback loop is representing in input to Feed Forward a filter coding of the input already processes by the same Feed Forward. So the signal will be elaborated several times. The Self Recurrent (SelfaSA) network is illustrated in figure 3.

As for the TASM network, inner connections of Feed Forward and Auto-Associative networks, as well as those connecting the extended input layer with the Feed Forward hidden layer have an adaptive weight. On the other hand connections linking the Feed Forward hidden layer to the Auto-Associative input layer, and those linking the Auto-associated output layer to the extended input layer have a fixed weight, equal to one, since their only function is to duplicate the activation value.

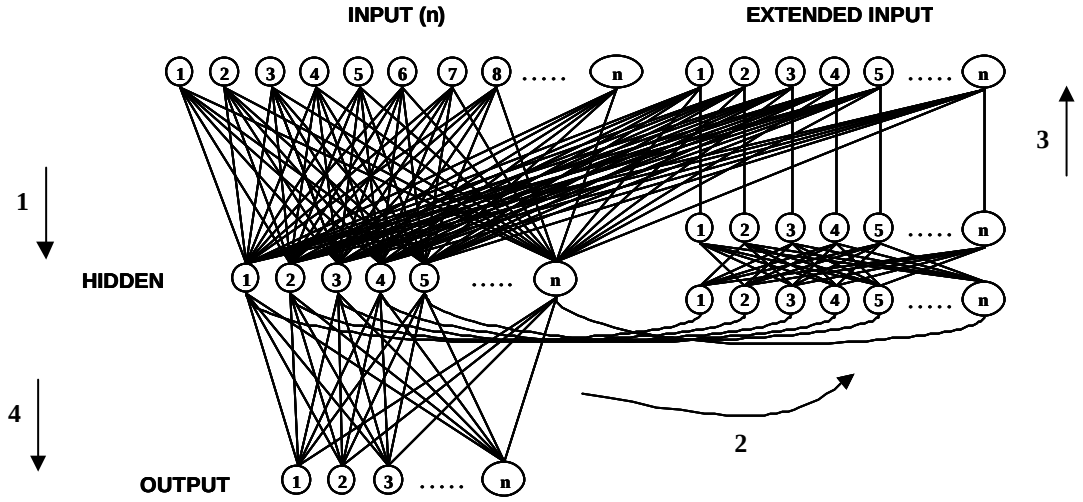

Figure 3 Representation of a SelfRecurrent Network with signal flow

All activation functions work on the weighted sum of inputs. They are linear for the input layers of Feed Forward and Auto-Associative networks and for the extended input layer, since such layers work only as a buffer. They are they are typically sigmoidal for all other layers.

The equation for the calculation of the activation values of the hidden layers is again

$$x_i^{[hid]} = f\left(\sum_{j=1}^{Nin} x_j^{[in]} \cdot w_{ij}^{[in-hid]} + \sum_{j=1}^{NinE} x_j^{[inE]} \cdot w_{ij}^{[inE-hid]} + J_i^{[hid]} \right) \qquad (2)$$

Moreover, unlike what we carried out in a classic Recurrent Network such as Elman's, we want the network output to consider its way to codify input at a hidden level, not in the following step for the following input, but in the same step for the same input. For this reason, presenting a pattern in input to

the Feed Forward, a calculation flow recycles more times through the Auto-Associative Network before the output pattern is produced.

As for the TASM network, Self recurrent network manifests two kinds of time: an external time, at each step of it an input and an output pattern are presented, and an internal time, at each step of it some cycles of the Auto-Associative are carried out to learn the coding on the Feed Forward hidden. One step of the external time corresponds to more steps of the internal time.

The state, i.e. the activation value of the extended input, reached by the network at the last internal step, is put to zero before executing internal cycles to calculate output for the next pattern. This allows the network to limit recurrence to calculate each single pattern.

In the *fixed feedback* network behavior, we have the flow according to the steps described as follows, where number *k* of cycles, usually two, is a network parameter:

```
Fixed_Feedback_SelfRecurrent_Flow
{
  While(enough_patterns)
  {
    Get_input_pattern;
    extended_input=0;
    num_cycles=k;
    do
    {
      Compute_hidden_activation;
        Compute_Autoassociate_output_activation;
      Correct_Autoassociate_weights;
      Compute_extended_input_activation;
        num_cycles = num_cycles -1;
      } while(num_cycles>0);
    Compute_hidden_activation;
    Compute_output_activation;  // As intermediate / final output
    Correct_hidden-output_weights(*);
    Correct_input-hidden_weights(*);
    Correct_extended_input-hidden_weights(*);
  }
}
```

In the particular case of recall only, the correction learning coefficient for all weights is equal to zero; i.e., all weights of the supervised network remain fixed, while the SelfReflexive weights are changed. The (*) steps are not performed during recall phase.

In the advanced version (adaptive feedback), the number of cycles through the Auto-associative network is made adaptive. Various options exist for choosing the adapting criterion. One which has demonstrated being appropriate is comparing the error on the Auto-Associative of each cycle with the one of the previous cycle, going out when error decreases.
In this case, we have the calculation flow according to the following steps:

```
Error_Adaptive_Feedback_SelfRecurrent_Flow
{
  While(enough_patterns)
    {
   Get_input_pattern;
   extended_input=0;
   autoassociation_error=0;
   do
   {
     Compute_hidden_activation;
        Compute_Autoassociate_output_activation;
        Compute_extended_input_activation;
        Correct_Autoassociate_weights
      previous_association_error=autoassociation_error;
      autoassociation_error=Compute_autoassociation_error;
   }while(autoassociation_error > previous_association_error);
   Compute_hidden_activation;
   Compute_output_activation;
   Correct_hidden-output_weights(*);
   Correct_input-hidden_weights(*);
   Correct_extended_input-hidden_weights(*);
    }
}
```

In the particular case of recall only, the correction learning coefficient is equal to zero for all weights, except for those of the Auto-associated, which is pushed toward convergence. Referring to the above algorithm description the (*) steps are not performed during recall.
Like TASM network, SelfRecurrent Network is an adaptive system able to *dynamically* control the effectiveness of coding of the input vectors performed by hidden units. It is a system that learns to learn. We created this complex evolution mechanism combining learning mechanisms with descendent gradients of the two networks by which SelfRecurrent is composed.

The *role* of the Auto-Associative Network and its *properties* define the specificity of this form of learning. Learning of the Auto-Associative works as a codification of the parameters $p_1, \ldots, p_r$ of the implicit function:

$$j\left(x_1^{[hid]}, \ ..., \ x_{Nhid}^{[hid]}, \ p_1, \ ..., \ p_r\right) = 0 \qquad (3)$$

which is not known and which expresses the dependency between the activation values of the hidden units of the Feed Forward which is connected with it.

The Auto-Associative Network works as an *adaptive filter* of the Feed Forward Network to which it is connected. It will reintroduce, as Feed Forward extended input, values that are never a copy of the hidden units that constituted its input.

In its working with the Feed Forward, during its learning phase, the Auto-Associative tends toward an attractor that is not necessary an attractor minimizing its error $E^{[Aut]}$. On the contrary, the Auto-Associative is dragged by the Feed Forward toward a value allowing it to filter in output values which represent hidden units more correlated with others and which better contribute to minimization of the error of supervised Feed Forward to which it is connected.

In this sense, we can state that the SelfRecurrent is a network conjugating *two gradients*, in order to establish the *optimal divergence* point of one of the two (that of the Auto-Associative), so that the *maximal convergence* of the other (that of the Feed Forward) can be obtained.

This network model's most important strengths are the following:

a. Precise generalization. This results through a learning phase during which the input coding itself is the object of learning. This reduces, at the same time, overtraining and coding errors problems.

b. A sufficiently good *tolerance and strength* to noisy inputs. The latter are contextualized together with clusters which they activate.

c. A good *biological plausibility*. External inputs are, in fact, not perceived as external inputs. They are *re-read* through the categories constituted by the system during its previous experiences. New inputs are, practically, re-categorized. This seems to have positive effects on the

stability and plasticity problems, which affects many artificial adaptive systems.

4.2.2 Dynamic SelfRecurrent Networks

An interesting modification of SelfRecurrent Networks consists in introducing a dependence of the activation functions of the Feed Forward hidden layer on the weights of the connections of the Auto-Associative. This dependence should increase feedback dynamism of this architecture . It corresponds, on the graph, to the introduction of what we defined connection-connection meta-arches.

We add to each extended input-hidden weight a connection operating on *j-th* PE of the hidden layer the product of Auto-Associative weights linked to the *j-th* PE in output. Referring to the analogous expression (2) we have:

$$x_i^{[hid]} = f\left(\sum_{j=1}^{Nin} \left(x_j^{[in]} \cdot w_{ij}^{[in-hid]} \right) + \sum_{j=1}^{NinE} x_j^{[inE]} \cdot w_{ij}^{[inE-hid]} \cdot \prod_{k=1}^{NoutAA} w_{ik}^{[NoutAA]} + J_i^{[hid]} \right) \qquad (4)$$

where $w_{ji}^{[NautAA]}$ indicates the weight from *i-th* PE of the input layer of the Auto-Associative to its *j-th* PE of the output layer. The Dynamic Self Recurrent (SelfDA) network is illustrated in figure 4.

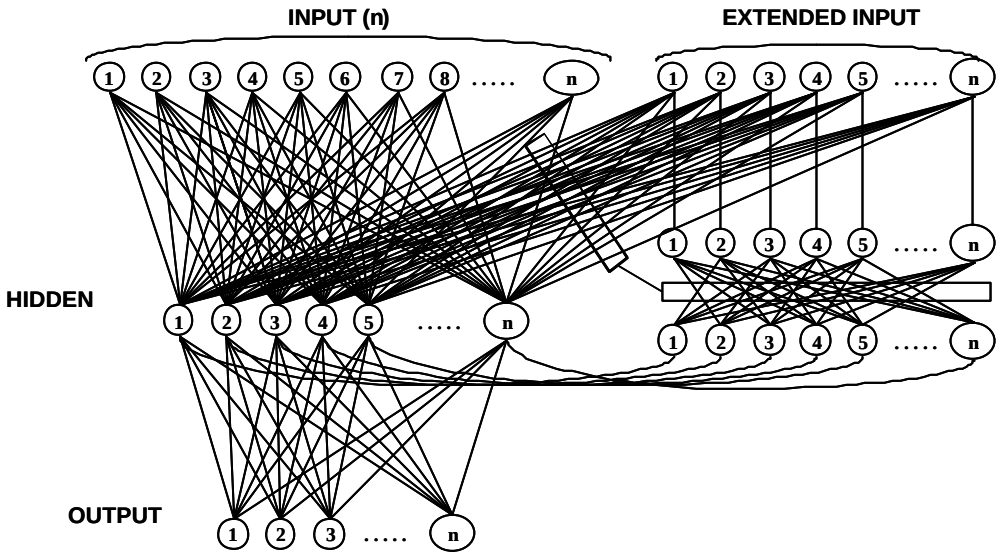

Figure 4 Representation of a dynamic SelfRecurrent network

5 Applications in medical field

5.1 Prediction of malignancy of lesions in contrast-enhanced MR-mammography

A dataset, collected at the University Hospital of Mainz, Germany, based on Contrast-Enhanced (CE) MRI of the female breast for the detection of breast cancer lesions, was used to train and test a Complex Recurrent Network against a BP network, and to evaluate this new architecture's performances accurately in terms of its generalization capabilities [15].

A 2 fold cross-validation analysis of this dataset, repeated 5 times (2CV5), was selected because these validation protocols seem to suffer less problems than the classical K fold cross-validations do. This is particularly so in terms of the first type error in algorithm's performance comparison [6].The dataset consisted of 604 histologically proven cases of contrast-enhancing lesions at MRI of the female breast analyzed between October 1996 and August 2001, 421 malignant and 183 benign tumors. Morphological parameters (e.g. size, shape or boundary of the lesion) as well as dynamic parameters (e.g. enhancement-style, central signal intensity at six different times during the investigation) were collected and stored in the database. This was enlarged by additional parameters (e.g. age of the patient, indication for the investigation, familiar and self-risk of breast cancer).

The Bp was configured with 4 hidden units, with sigmoidal activation and softmax [2] function for the output nodes. One node was for malignant and one was for benign breast lesion. All recurrent network used in this experiment have an adaptive definition of number of internal iterations. In following table TASM architecture is compared to statistic model and an expert radiologist. For details (optimization protocol and learning laws used) we refers the reader to the reference.

ANN	Sensitivity	Specificity
BP	91.7%	83.7%
TASMDa	93.6%	91.9%
Logistic Regression	90.5%	68.9%
Expert radiologist	92.1%	85.6%

Table 1 – Comparison between architecture.
The DA postfix means dynamic and represents the presence of second order connections.

5.2 Prediction of Alzheimer pathology only on the basis of cognitive and functional Status

Data from several studies have pointed out the existence of a strong correlation between Alzheimer's disease (AD) neuropathology and cognitive state. To predict the results of postmortem brain examinations, we applied ANNs to the Nun Study data set, a longitudinal epidemiological study, which includes annual cognitive and functional evaluation [5]. One hundred seventeen subjects from the study participated in this analysis. We determined how demographic data and the cognitive and functional variables of each subject during the last year of her life could predict the presence of brain pathology expressed as Braak stages, neurofibrillary tangles (NFTs) and neuritic plaques (NPs) count in the neocortex and hippocampus, and brain atrophy. The result of this analysis was then compared with traditional statistical models. ANNs proved to be better predictors than Linear Discriminant Analysis in all experimentations (+ ~10% in overall accuracy), especially when assembled in Artificial Organisms (+ ~20% in overall accuracy). Demographic, cognitive, and clinical variables were better predictors of tangles count in the neocortex and in the hippocampus when compared to NPs count. In the following table we show part of the results of prediction experiments with a comparison between two protocols (random vs optimization). For details (optimization protocol and learning laws used) we refers the reader to the reference.

SUMMARY RESULTS						
Experiments	**Models**	**Num**	**Target Classification**		**A. Mean**	**W. Mean**
			Tangles < 1.0	**Tangles > 1.0**		
Tangles NeoCortex	LDA - Random	10	74.12%	73.48%	73.80%	73.85%
	ANN FFSn* - Random	20	86.51%	81.81%	84.16%	84.57%
	Best ANN FFBp Optimized	10	89.18%	86.49%	87.83%	87.08%
	SelfDABp	40	93.36%	87.37%	90.36%	89.95%
			Tangles > 10	**Tangles < 10**		
Tangles Hippocampus	LDA - Random	10	76.83%	74.21%	75.52%	75.62%
	ANN FFSn*- Random	20	84.12%	83.77%	83.95%	83.92%
	Best ANN-FFBm Optimized	10	84.65%	88.70%	86.68%	86.33%
	TasmSaBm	40	88.00%	87.68%	87.83%	87.87%
			Plaques > 1.5	**Plaques < 1.5**		
Plaques NeoCortex	LDA - Random	10	64.03%	75.50%	69.77%	66.37%
	ANN FFSn* - Random	20	87.11%	80.25%	83.68%	85.16%
	Best ANN- FFBp Optimized	10	89,74%	88.13%	88.93%	88.11%
	SelfDABp	40	88.59%	91.25%	89.92%	88.62%
			Plaques > 1.5	**Plaques < 1.5**		
Plaques Hippocampus	LDA - Random	10	61.68%	68.53%	65.10%	65.36%
	ANNs - FFSn Random	20	79.24%	74.16%	76.70%	76.26%
	Best ANN- FFBp Optimized	10	77.01%	79.93%	78.47%	78.14%
	SelfSaBp	40	79.40%	85.02%	82.21%	81.94%

Table 2 Prediction by Artificial Neural Networks and Linear Discriminant Analysis of the Neuropathological Outcomes by Using Cognitive and Functional Variables in 117 Subjects
*FFSn: Feed Forward Sine Net: feed forward with Sine learning law (Semeion)
TasmSaBm: Tasm Static with Bimodal learning law (Semeion)
Num: number of elaborations; A. Mean: arithmetic mean; W. Mean: weighted mean.

The performance of these netowrks has been confirmed by other various other applications analyzing real data [8,12,13]. This reinforces that the complex dynamics that characterize these new types of networks could help, during the training phase, to limit consequences of noisy data and of "overfitting" effect and, during the recall phase, to allow for more accurate extrapolation.

It must be emphasized that these results were obtained using a system composed of two networks that cooperate and coordinate each other in a spontaneous way. The analysis of these networks obviously needs to be extended given the encouraging experimental results. This could be a way to build complex organisms which are able to enhance the capabilities of classical networks.

6 Conclusion: recurrence and meta-learning

The main reason to design and to propose Tasm and SelfReflexive ANNs is to simulate a complex and dynamic meta-learning process.

Learning is not enough: human learning is not a simple way to adapt brain connections to a new situation. At the most simple level it consists also to reflect on own learning process. Human learning is a complex way to learn to learn.

This process of Meta-learning is a natural recurrent process. And the number of recurrences has to be adaptive, according to the interaction between the prior knowledge and the new ones.

In a first approximation we can define meta-learning as the learning of a new input with the code through which the same new input was learnt one step before. This dynamic recoding is the background of meta-learning process :

- an ANN codes in its hidden layer a new input;
- the same ANN codes again in its hidden layer the new input plus the way the ANN understood it in the precedent step;
- and so on until the ANN finds no new information in the vector composed by the new input and the its precedent interpretation codified in its hidden layer.

We can try to formalize this dynamic:

$$y^t = F^t(G^t(x^t)) \qquad (5)$$

where:

y^t Output at time t

F^t Output function at time t

G^t Hidden function at time t

x^t Input at time t

The equation (5) is the general representation of a supervised feed forward multilayer ANN. The time, *t*, points out the external time of the process: when the Input-Output pair changes the time increases.

The equation of one of the more classical and simple recurrent ANN, instead, should have this representation :

$$y^t = F^t(G^t(x^t, G^{t-1}(x^{t-1}, G^{t-2}(...,...))))) \qquad (6)$$

Also in this case, the time, *t*, is an external time: the time of the hidden layer, G, is always the same time of the input, *x*. The ANN is not reflecting on its way to code the input. It is simply forcing the precedent input coding on the actual input coding. In this case we cannot say that the ANN has a memory. We have to say that the ANN architecture forces the ANN to link the actual learning to the precedent one. We can talk about "memory" if the ANN reflects and weights on this link. We can talk about memory if a meta-learning process is implied.

Meta-learning is possible if the time of the function and the time of its argument are not the same. It implies the existence of two different times: the external time, connected to the Input-Output pairs changes, and an internal time, dependent from the iterative processes between the hidden layer function and the input layer.

Memory happens when the ANN tries to understand now the way it understood the actual and the precedent input. Memory is much more complex than meta-learning, and meta-learning is much more complex than passive learning.

In the case of the Self Recurrent network, this more complex condition makes the Auto-Associative ANN weights matrix updating temporally independent from the external time of the global process. Internal time is completely independent from the external time: the Auto-Associative ANN updates its weights at any recurrence, not making difference between two different input vectors or the iteration of the same input vector.

The criterion trough which the meta-learning recurrence process stops depends from the dynamics of the Auto-Associative ANN : when the Auto-Associative ANN error at the internal time decreases respect to the same

error at the time *(t-1)*, the feed back loop terminates and the signal can flow up to the output, *y*, of the global ANN. The error reduction, in fact, means that the global ANN is beginning to understand the connection between the new input and its understanding of the same new input.

This specific time articulation works also during the recall phase of the trained ANN. For any new input the ANN works updating the Auto-Associative weights matrix after every loop, but maintaining fixed the other weights matrices.

This dynamic answering is interesting because it means that the ANN continues to iterate the meta-learning process also during the recall phase.

Tasm ANN works in the same way of the SelfReflexive ANN.

The main difference is that in Tasm ANN, the server ANN for the meta-learning process is an unsupervised SelfReflexive ANN. The free units of SelfReflexive ANN represent the main features of what the Tasm understood of the input vector. So, this condensed understanding modules the weights strength from the same input vector to the ANN hidden nodes. The meta-learning process, by the way, will happen on a double background:

- the indirect effect of input vector added with the condensed understanding that the ANN did of the same input in the precedent step;
- the direct modulation of the input vector itself on the hidden units, because of the interpretation of ANN precedent coding.

This process should be able to optimize the Tasm training further.

With the SelfRecurrent ANN we have followed a different line of thought, linked to the different mathematic nature of the Auto-Associative ANN, connected to the main ANN.

The Auto-Associative ANN has the same cardinality in input as in output. But its cardinality is also the same cardinality of the hidden units of the main ANN.

So, it is possible to assert that each output of the Auto-Associative ANN represents an internal interpretation of the correspondent hidden node of the main ANN.

In fact, the fan in of each output node of the Auto-Associative ANN is equal to the number of the hidden nodes of the main ANN.

When we desire to increase the order of the SelfRecurrent ANN, we need to connect directly the Auto-Associative weights of each Auto-Associative output node to each connection coming from the Auto-Associative Output node to the correspondent hidden node of the main ANN.

In this way, the meta-connections will module the sign and the strength of meta-learning on every specific hidden node, according to the internal understanding that the Auto-Associative ANN is developing of what the global ANN understood of the input vector [16].

Also in this process every meta-learning feed back loop is a double meta-learning : what the ANN understands of the new input with the content of what it understood of the same input the step before, considering what the ANN has meta-understood with the precedent input vector.

Finally, Tasm and SelfReflexive are “double time” ANNs, showing specific features that make them different from the usual recurrent ANNs.

For this reason we named them composite recurrent ANNs.

Composite ANNs :

- are composed from different ANNs (at least two) , working together in recurrent way;
- their mutual and local convergence is not globally supervised;
- work following two different times: a external time and an internal one;
- the number of inside feed back loops is adaptive;
- can present with easy modifications an high order of dynamics;
- can be structured also for classical temporal learning;
- are meta-learning oriented;
- are able to simulate more likely than classic recurrent ANNs human memory processes.

REFERENCES

[1] Anderson J.A., *Associative Networks*, in Arbib M.A.(ed), The Handbook of Brain Theory and Neural Networks, The MIT Press, Cambridge, Massachusetts, 1995.

[2] Bridle J.S. (1989), *Probabilistic Interpretation of Feedforward Classification Network Outputs, with Relationships to Statistical Pattern Recognition*, in F. Fogelman-Soulié, and J. Hérault (eds.), Neuro-computing: Algorithms, Architectures, Springer-Verlag, New York. 1989.

[3] M.Buscema, *SelfReflexive Networks. Theory, Topology, Applications*, in Quality & Quantity, Kluwer Academic Publisher, Dordtrecht, The Netherlands, November 1995, vol.29(4), 339-403.

[4] M. Buscema, *Self-Recurrent neural network*, in Special Issue of Substance Use & Misuse, The International Journal of the Addictions, Marcel Dekker, New York ,Vol. 33 n. 2, 1998, pp. 495-501.

[5] M. Buscema, E. Grossi, D. Snowdon, P. Antuono, M. Intraligi, G. Maurelli, R. Savarè, *Artificial Neural Network can predict Alzheimer pathology in individual patients only on the basis of cognitive and functional status*, in NeuroInformatics, Humana Press, Winter 2004 2(4), pp. 399-416.

[6] Dietterich T.G. (1997), *Approximate statistical tests for comparing SuperVised classification learning algorithms* in Neural Computation, by The MIT Press, Vol. 10, 1998, pp. 1895-1923,.

[7] J.L.Elman, *Finding Structure in Time*, in Cognitive Science, vol.14, pp.179-211.

[8] E.Grossi, M.Buscema, M.Intraligi, A. Seeberg *Use of Artificial Neural Networks in Predicting Adverse Reactions in Individual Patients Receiving Contrast Media*, in Proceedings of "Artificial Intelligence meets R&D" , 2002.

[9] J.J.Hopfield, *Neural Networks and physical systems with emergent collective computational abilities*, Proceedings of the National Academy of Sciences 79:2554-2558, in J.A.Anderson and E.Rosenfeld (Eds) Neurocomputing Foundations of Reasearch, The MIT Press, Cambridge, MA, 1988.

[10] J.J.Hopfield, *Neurons with graded response have collective computational properties like those of two state neurons*, Proceedings of the National Academy of Sciences USA, Bioscences 81, pp.3088,3092.

[11] M.I.Jordan, *Serial Order: A parallel distributed processing approach*, in J.L.Elman, D.E.Rumelhart, Advances in Connectionist Theory: Speech, Erlbaum, Hillsdale, NJ.

[12] P.Mecocci, E.Grossi, M.Buscema, M.Intraligi, R.Savaré, P.Rinaldi, A.Cherubini, U.Senin , *Use of artificial neural networks in Clinical Trials: a Pilot Study to predict Responsiveness to Donepezil in Alzheimer Disease Patients*, in JAGS, vol. 50, n. 11, November 2002, pp. 1857-1860.

[13] F.Pace, M.Buscema, M.Intraligi, E.Grossi, P.Dominici, R.Cerutti, G.Bianchi Porro, *Neural Networks in GERD to Discriminate between Patients with or without Pathological Reflux on the Basis of Clinical data alone*, in Proceedings of "Artificial Intelligence meets R&D", 2002.

[14] Rumelhart D.E., Hinton G.E., and Williams R.J., (1986), "*Learning internal representations by error propagation*", in Rumelhart D.E. and McClelland J. L., eds. (1986), Parallel Distributed Processing: Explorations in the Microstructure of Cognition, MA: The MIT Press. 1986, Volume 1, 318-362, Cambridge.

[15] T. W. Vomweg, M. Buscema, H. U. Kauczor, A. Teifke, M. Intraligi, S. Terzi, C.P. Heussel, T. Achenbach, O. Rieker, D. Mayer, M. Thelen, *Improved Artificial Neural Networks in dignity prediction of enhancing lesions in contrast-enhanced MR-Mammography*, in Medical Physics, W. R. Hendee Ed., vol. 30 (n. 9) September 2003, pp. 2350-2359.

[16] C.Yonghong, J.Yaolin, X.Janxue, *Dynamic properties and a new learning mechanism in higher order neural networks*, Neurocomputing, Elsevier Ed., Vol. 50, January 2003 pp. 17-30.