ESANN'2009 proceedings, European Symposium on Artificial Neural Networks - Advances in Computational Intelligence and Learning. Bruges (Belgium), 22-24 April 2009, d-side publi., ISBN 2-930307-09-9.

A brief introduction to Weightless Neural Systems

I. Aleksander,[1] M. De Gregorio,[2] F.M.G. França,[3] P.M.V. Lima,[3] H. Morton[4]

1- Imperial College London, UK
2 - Istituto di Cibernetica "E. Caianiello" – CNR, Pozzuoli, ITALY
3 – Universidade Federal do Rio de Janeiro, BRAZIL
4 – Brunel University Uxbridge, Middlesex, UK

Abstract. Mimicking biological neurons by focusing on the excitatory/inhibitory decoding performed by the dendritic trees is a different and attractive alternative to the integrate-and-fire McCullogh-Pitts neuron stylisation. In such alternative analogy, neurons can be seen as a set of RAM nodes addressed by Boolean inputs and producing Boolean outputs. The shortening of the semantic gap between the synaptic-centric model introduced by the McCullogh-Pitts neuron and the dominating, binary digital, computational environment, is among the interesting benefits of the *weightless* neural approach. This paper presents an overview of the most representative paradigms of weightless neural systems and corresponding applications, at abstraction levels ranging from pattern recognition to artificial consciousness.

1 Introduction

Mainstream artificial neural network (ANN) models are based on weighted-sum-and-threshold artificial neurons, as the pioneering *Threshold Logic Unit*, of McCullogh and Pitts [1]. The biological analogy behind this model lies on the mapping of the synaptic strength between the output produced and transmitted by the neuron's axon and the input of a post-synaptic neuron, into pseudo-continuous numerical weights. An important simplification happens in the way inputs to neuron are modeled: all synaptic connections terminate directly at the neuron's *soma*. Although such specific morphological arrangement is plausible in biological terms, the vast majority of synapses in the central nervous system terminate at the neuron's *dendritic tree* [2]. Nevertheless generalizations of artificial weighted-sum-and-threshold neurons, such as Sigma-Pi units [3], do exist, this means that the dendritic tree, the mostly noticeable morphological structure of the neuron cell, is not being taken into account in mainstream ANN paradigms.

Weightless neural networks (WNNs) are based on networks of Random Access Memory (RAM) nodes. A straightforward analogy between the address decoding in RAMs and the integration of excitatory and inhibitory signaling performed by the neuron's dendritic tree can be made: the closer to the neuron's soma an input synapse terminates, greater is the influence of such input in the definition of the neuron's output since it can *gate* the integration of other synaptic inputs coming from farer locations in the dendritic tree [4]. Mapping neuronal excitatory and inhibitory signaling into artificial binary input signals is a simple stylization, being Boolean representation the simplest and practical form, on which WNNs could have a basis for its biological plausibility. In other words, the "strength" of an input signal depends on which height in the dendritic tree the synaptic input connection is placed, quite similar to the way RAM address decoding is performed.

Independently of how wide this discussion could be, the biological plausibility of weightless neural networks had never an important role in the interesting explorations already made with such agile and practical neural models. As binary digital computing dominates, the semantic gap of the interpretation of WNNs in contemporary computers is quite short. In terms of functionality, it is easy to see that a weighted-sum-and-threshold artificial neuron having n inputs can be perfectly imitated by an n-tuple RAM node [5]. However, the way learning is performed in both models differ in the following: the functionality of a neuron can be modified by changes in the weight values in the former model, and by changes in the RAM contents in the later one.

Moreover, the pattern linear separability limitation suffered by the former neural model is easily overcame since it is possible to directly map exclusive-OR functions on a n-tuple RAM node [6]. In fact, the use of n-tuple RAM nodes in pattern recognition problems is dating 50 years by the work of Bledsoe and Browning [7]. Some years later, Aleksander introduced Stored Logic Adaptive Microcircuit (SLAM) and n-tuple RAM nodes as basic components for an adaptive learning network [8]. With the availability of integrated circuit memories in the late 70s, the WiSARD (Wilkes, Stonham and Aleksander Recognition Device) was the first artificial neural network machine to be patented and produced commercially [9][10]. Other WNN models followed, such as PLNs [11][12][13][14], GSNs [15][16][17] and GRAMs [18][19], and will be presented in more detail in the next sections.

2 RAM-discriminators and WiSARD

A RAM-discriminator consists of a set of X one-bit word RAMs with n inputs and a summing device (Σ). Any such RAM-discriminator can receive a binary pattern of $X{\cdot}n$ bits as input. The RAM input lines are connected to the input pattern by means of a biunivocal pseudo-random mapping. The summing device enables this network of RAMs to exhibit – just like other ANN models based on synaptic weights – generalization and noise tolerance.

In order to train the discriminator one has to set all RAM memory locations to 0 and choose a training set formed by binary patterns of $X{\cdot}n$ bits. For each training pattern, a 1 is stored in the memory location of each RAM addressed by this input pattern. Once the training of patterns is completed, RAM memory contents will be set to a certain number of 0's and 1's.

The information stored by the RAM during the training phase is used to deal with previous unseen patterns. When one of these is given as input, the RAM memory contents addressed by the input pattern are read and summed by Σ. The number r thus obtained, which is called the *discriminator response*, is equal to the number of RAMs that output 1. r reaches the maximum X if the input belongs to the training set. r is equal to 0 if no n-bit component of the input pattern appears in the training set (not a single RAM outputs 1). Intermediate values of r express a kind of "similarity measure" of the input pattern with respect to the patterns in the training set.

ESANN'2009 proceedings, European Symposium on Artificial Neural Networks - Advances in Computational Intelligence and Learning. Bruges (Belgium), 22-24 April 2009, d-side publi., ISBN 2-930307-09-9.

A system formed by various RAM-discriminators is called WiSARD (Wilkie, Stonham & Aleksander's Recognition Device) [10]. Each RAM-discriminator is trained on a particular class of patterns, and classification by the multi-discriminator system is performed in the following way. When a pattern is given as input, each RAM-discriminator gives a response to that input. The various responses are evaluated by an algorithm which compares them and computes the relative confidence c of the highest response (e.g., the difference d between the highest response and the second highest response, divided by the highest response). A schematic representation of a RAM-discriminator and a 10 RAM-discriminator WiSARD is shown in Figure 1.

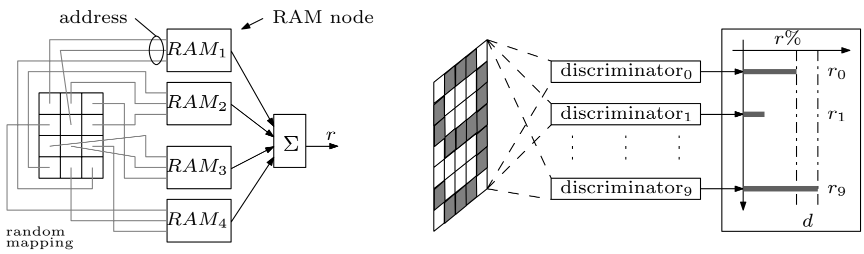

Fig. 1 – Example of a RAM-discriminator and of a WiSARD

The performance of the WiSARD depends on n. Specialized responses from WiSARD grows with n; on the other hand, generalization capabilities of WiSARD grows inversely with n [13].

3 PLNs, MPLNs and *p*RAMs

Most of the RAM-based neural systems are formed by a single layer of RAM-nodes. Aleksander introduced a multilayer architecture (pyramid) [12] formed by new nodes called PLN (Probabilistic Logic Node) [11]. A PLN node stores a 2-bit value at each of its memory locations: "0", "1", and "*u*". The latter represents the "unknown" or "undefined" state, which is stored in all PLN memory locations before the learning phase. The undefined state "*u*" allows a PLN node to, randomly, output either 0 or 1, with the same probability.

The pyramid architecture formed by PLN nodes is characterized by having a fan-out of 1 and low fan-in. Having a low fan-in, the PLN node saturates very easily and it fails to learn new patterns before the training set has been completely presented.

The training algorithm consists of replacing *u*'s with 0's and 1's in the following way: a training pattern is propagated through the base of the pyramid up to the top, where the output is produced; if the pyramid output value matches the desired output, then a signal is propagated back to each node in order to make them store all randomly generated output values into the currently addressed location at each node. If the produced output is not the desired value, then the same input pattern is reapplied and the same procedure repeated.

In [13] Aleksander proposed a natural extension of the PLN node: the *m*-state PLN (MPLN). In this new model, a wider discrete range of values could be stored at each memory location. In addiction, the output function could be even a linear function or the sigmoid function. In this model, the learning phase allows for incremental changes of the stored values. New information is acquired after different steps, as incorrect information is threw away only after a certain number of errors.

In order to incorporate some properties of living neurons, an evolution of this model was proposed by Taylor [20]. It can be shown that this new model is equivalent to a network of noisy (*p*robabilistic) RAMs (*p*RAMs). In this model, values belonging to [0, 1] can be stored at the memory locations (continuous probability). Given a certain input, the contents of memory locations represent the probability that a value 1 is produced as output.

4 GSNs and GRAMs

The GSN (Goal Seeking Neuron) has been developed with the aim of preserving the corruption of the information previously stored in PLN nodes. Unlike a PLN, a GSN can input, store and output 0's, 1's, and *u*'s. Two different locations are addressed in a GNS if the input contains a *u*. In the same way, if a given memory content is *u*, the corresponding GSN will address two different nodes.

GSN has three different modes of operation: validating (can the pattern be learned?), learning and recall. The purpose of the validation phase is to check if a new pattern can be learnt without corrupting the previously stored knowledge. The value *u* on the output means that the net can learn any desired output. 0 or 1 as output suggests that the net can learn only the proposed pattern if the output of the net is the desired one. Otherwise, there would be a disruption of information previously stored. In case of success of the validation phase, one of the following conditions will occur (learning phase): if the desired output is 0 (1) and the addressed contents contain at least one 0 (1), then one of these locations is selected randomly and its input address becomes, effectively, the desired outputs for nodes in the previous layer; otherwise, an undefined location is selected at random (from the addressable set) and the desired output is stored in that location and, as above, the address of that location goes back to previous layer. The aim of the recall phase is that of producing the most occurring value (0 or 1) in the addressed contents. If there are equal numbers of 0's and 1's in the addressed contents, then the output is *u*.

GRAMs were introduced by Aleksander [18] in order to increase the generalization of WNNs at the node level by including a spreading phase in the learning algorithm, just after a training pattern is stored. The GRAM behaves like a bit-organized random access memory up to the point at which there is a clash between attempting to set an address to both 0 and 1, or an address cannot be set according to the Hamming distance rule. In such cases the address remains undefined and behaves as a random binary string generator. This form of generalization is described as a process of spreading, where information set in the RAM through the trained set spreads to other locations. In some applications, spreading limited to a radius is employed, in the sense that Hamming distances greater than a set limit are ignored.

5 GNU

A GNU [21] is a neural state machine formed by a single layer of GRAM nodes in which the binary output variables are the same as the state variables. This means that the set of mappings which define the binary output vector is exactly the same as the set of mappings which define the binary state vector.

The GNU can be configured as a feedforward and/or feed-back system with auto-associative memory properties, though hetero-associative mappings are also possible. The input field is composed of N bits and the external output field is composed of M bits. In the GNU architecture, each node of the network has n ($n \leq N$) inputs connected to the input field and m ($m \leq M$) nodes connected to the output field. Each node sampling, from both input and output fields, are randomly established.

The storage process in a GNU consists of creating an association between an external pattern and its representation, by producing a re-entrant, or stable, state in which the network stabilizes in the retrieval phase. The training phase consists of applying the same training procedure used at each individual GRAM. In the case of trained patterns, they are associated with themselves. This is called "iconic" training and the idea is to make a many-to-some mapping of the input in which each node samples the patterns that occur at the input. The word "iconic" is used to describe that the internal states mirror the patterns of input events. An example of GNU creating an association is illustrated in Figure 2.

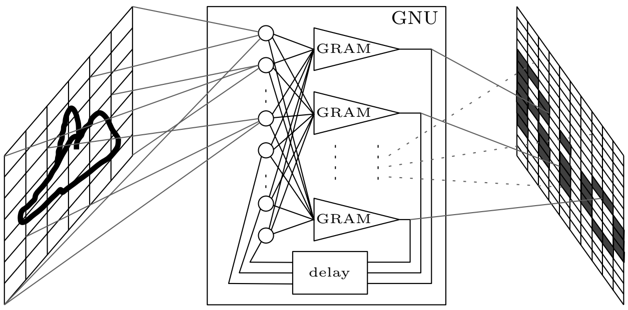

Fig. 2: GNU architecture

6 Some commercial and software developments.

The WISARD system was commercialized in 1986 and marketed by Computer Recognition systems on the UK. Systems were purchased by the UK police for face recognition and fingerprint processing. Further the GNU scheme above was a subject of a major grant in the UK that resulted in the MAGNUS system that allowed arbitrary structures of interconnected GNUs in order to model brain structures. Barry Dunmall of Imperial College also commercialized this system, which became known as the Neural Representation Modeller (NRM) which was developed to run under Microsoft Windows systems. This is the system that was most heavily used in order to arrive at the material described in [23] where references to details of the NRM system can be found.

7 Conclusions

A very brief review of the main weightless neural network paradigms has been presented. Although WNNs are not in the mainstream of ANN research, the agility and scalability of these models implemented in the existing computing environments is very attractive. Recent publications report that the role of WNNs were crucial in scientific investigations of very interesting subjects, such as artificial consciousness [22][23], as well as in recent development of commercial solutions to important and difficult problems, such as automated video surveillance [24], robotics [25], acceleration of 3D video animation [26] and automated text categorization and clustering [27].

References

[1] McCulloch, W. and Pitts, W., A logical calculus of the ideas immanent in nervous activity*, Bulletin of Mathematical Biophysics*, 7, pp. 115-133, 1943.

[2] D. Johnston, Foreword, in: *Dendrites*, G. Stuart, N. Spruston and M. Häusser, Eds, Oxford University Press, 1999.

[3] D. E. Rumelhart, G. E. Hinton, J. L. McClelland, A General Framework for Parallel Distributed Processing, in: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, Vol.1, MIT Press, 1988.

[4] C. Koch and T. Poggio, Biophysics of Computation: Neurons, Synapses and Membranes, in: *Synaptic Function*, G. M. Edelman, W. E. Gall and W.M. Cowan, John Wiley & Sons, 1987.

[5] Ferreira, V. M. G. and França, F. M. G. . Weightless implementations of weighted neural networks. In: Anais do IV Simpósio Brasileiro de Redes Neurais, pp. 53-54, 1997.

[6] I. Aleksander, Ideal neurons for neural computers, in: *Parallel Processing in Neural Systems and Computers*, North-Holland, Amsterdam, pp. 225-228, 1990.

[7] W.W. Bledsoe and I. Browning, Pattern Recognition and Reading by Machine, *Proceedings of the Eastern Joint Computer Conference,* Boston, pp. 225-232, 1959.

[8] I. Aleksander, Self-adaptive universal logic circuits, *IEE Electronic Letters*, 2, p. 321, August 1966.

[9] I. Aleksander and T. J. Stonham, Guide to pattern recognition using random-access memories, *Computer and Digital Techniques*, 2, pp. 29-40, 1979.

[10] I. Aleksander, W. Thomas, and P. Bowden, WISARD, a radical new step forward in image recognition, *Sensor Rev*., 4(3), pp. 120-124, 1984.

[11] I. Aleksander and W.W. Kan, A Probabilistic Logic Neuron Network for Associative Learning, *IEEE Proceedings of the First Int. Conf. on Neural Networks*, pp. 541-548, 1987.

[12] I. Aleksander. Canonical neural nets based on logic nodes, in *Proc. of the IEE Int. Conf. on Artificial Neural Networks*, London, pp. 110-114, 1989.

[13] I. Aleksander, *An introduction to neural computing*, Chapman and Hall, London, 1990.

[14] I. Aleksander and C. Myers, Learning algorithms for probabilistic logic nodes, in: *Abstract of the First INNS88*, p 205, Boston, 1989.

[15] R. G. Bowmaker and G. G. Coghill, Improved recognition capabilities for goal seeking neuron, *IEE Electronics Letters*, 28, pp. 220-221, 1992.

[16] W. Martins and N. Allinson, Two improvements for GSN neural networks, *Proc. of the Weightless Neural Networks Workshop*, York, pp.58-63, 1993.

[17] A. de Carvalho, M. C. Fairhurst, D. L. Bisset, Progressive learning algorithm for GSN feedforward neural architectures, *IEE Electronics Letters*, 30, pp. 506-507, 1994.

[18] I. Aleksander, Ideal neurons for neural computers, in: *Parallel Processing in Neural Systems and Computers*, North-Holland, Amsterdam, pp. 225-228, 1990.

[19] J. Mrsic-Flogel, Approaching Cognitive System Design, *Proceedings of the International Conference on Artificial Neural Networks (ICANN 91)*, Helsinki, Vol. 1, pp. 879-883, 1991.

[20] J.G. Taylor, Spontaneous behaviour in neural networks, *Journal of Theoretical Biology*, 36, pp. 513-528, 1972.

[21] I. Aleksander and H. Morton, General neural unit: retrieval performance, *IEE Electronics Letters*, 27, pp. 1176-1778, 1991.

[22] I. Aleksander, Capturing consciousness in neural systems, in: *Artificial Neural Networks, 2. Proc. ICANN-92*, London: North-Holland, pp. 17-22, 1992.

[23] I. Aleksander, *The World in My Mind My Mind in the World: Key Mechanisms of Consciousness in Humans Animals and Machines*, Imprint Academic, Exeter, 2005.

[24] M. De Gregorio, An Intelligent Active Video Surveillance System Based on the Integration of Virtual Neural Sensors and BDI Agents, *IEICE Transaction on Information and Systems*, vol. E91-D, n. 7, pp. 1914-1921, 2008.

[25] P. Coraggio and M. De Gregorio, WiSARD and NSP for Robot Global Localization, in: *Nature Inspired Problem-Solving Methods in Knowledge Engineering* – Part II, J. Mira & J.R. Alvarez eds., LNCS 4528 Springer, pp.449-458, 2005.

[26] C. B. do Prado, F. M. G. França, E. Costa, L. Vasconcelos, A new intelligent systems approach to 3D animation in television, *Proc. of the 6th ACM international Conference on Image and Video Retrieval*, pp.117-119, 2007.

[27] C. B. do Prado, F. M. G. França, R. Diacovo, P. M. V. Lima, The Influence of Order on a Large Bag of Words, *Proc. of the 8th International Conference on Intelligent Systems Design and Applications*, IEEE CS Press, v. 1, pp. 432-436, 2008.