Learning Long Sequences in Binary Neural Networks

Xiaoran Jiang, Vincent Gripon, and Claude Berrou
Telecom Bretagne, Electronics department
UMR CNRS Lab-STICC
Brest, France
xiaoran.jiang@telecom-bretagne.eu, vincent.gripon@telecom-bretagne.eu, claude.berrou@telecom-bretagne.eu

***Abstract*—An original architecture of oriented sparse neural networks that enables the introduction of sequentiality in associative memories is proposed in this paper. This architecture can be regarded as a generalization of a non oriented binary network based on cliques recently proposed. Using a limited neuron resource, the network is able to learn very long sequences and to retrieve them from only the knowledge of any sequence of consecutive symbols.**

***Index Terms*—oriented neural network; learning machine; associative memory; sparse coding; directed graph; sequential learning; efficiency.**

I. INTRODUCTION

Sequence learning in neural networks has been an important research topic in a large number of publications, since the forward linear progression of time is a fundamental property of human cognitive behavior. Different approaches have been carried on. Among them, the most important and commonly studied are the simple recurrent networks (SRN) [1] [2] and the short term memory (STM) [3] [4], which uses the dynamics of neural networks. Other structures have been proposed, especially those based on the Hopfield network principle [5]. However, many Hopfield-like connectionist networks do not have good performance when learning sequences, as the learning of new information completely disrupts or even eliminates that previously learnt by the network. This problem is identified as "catastrophic interference" [6] or "catastrophic forgetting" (CF) [7] in some literature. There are indeed strong interferences as the learning process relies on changing the connection weight (what is called plasticity by neurobiologists). Therefore, there is no guarantee that the ability of the network to recall messages will remain still when learning new ones.

A recently proposed non-oriented kind of network based on cliques and sparse representations [8] [9] follows a different approach by comparison with Hopfield-like networks. The neurons and the connections are all binary. The connection weight is equal to zero if the connection does not exist, otherwise this weight is equal to one. Subsequent learning will never impact on the weights of the existing connections. Therefore, we explain in this paper how the architecture of these networks can be efficiently modified to allow learning sequences with less degree of interference with the previously learned ones. However, the clique-based networks only enable the learning of fixed-length messages, and the learning and retrieving are rather synchronous than following time progression. In order to learn information arriving in separate episodes over time, one may replace the non oriented graph by an oriented one, and consider a more flexible structure than cliques.

The rest of paper is organized as follows: Section II recalls the principles of learning fixed length messages by non oriented clique-based networks, which is at the root of the works presented in this paper. In Section III, the oriented sparse neural networks based on original oriented graphs, called "chains of tournaments" are demonstrated to be good material to learn sequential information. Generalization is proposed in Section IV. Finally, a conclusion is proposed in Section V.

II. LEARNING FIXED LENGTH MESSAGES ON CLIQUES

Let $\mathfrak{M}_B$ be a set of binary messages of fixed length B bits. For each message $m \in \mathfrak{M}_B$, we split it into c sub-messages of length $\frac{B}{c}$: $m = m^1 m^2 ... m^c$. Each sub-message is then associated with a unique extremely sparse codeword (each sub-message is encoded by a single neuron), within a unique cluster of neurons in the network. For $1 \leq i \leq c$, m^i of length $\frac{B}{c}$ can take $2^{\frac{B}{c}}$ values, that leads to an extremely sparse code of length $2^{\frac{B}{c}}$, and the corresponding cluster of size $l = 2^{\frac{B}{c}}$. An example is represented in Figure 1, in which there are $c = 4$ clusters (filled circles, filled rectangles, rectangles and circles) of $l = 16$ neurons, that we call fanals, according to the vocabulary in [8]. In this figure, one message of 16 bits: 1110100111011010 is split into 4 sub-messages, $m^1 = 1110, m^2 = 1001, m^3 = 1101, m^4 = 1010$. Each sub-message is then mapped to a unique fanal in the corresponding cluster. The fundamental idea is to transform the learning of such a message into embedding a clique into the network (thick lines in Figure 1 for the message mentioned above). In graph theory, a clique in an undirected graph is a subset of its vertices such that every two vertices in the subset are connected by an edge. Any binary message in $\mathfrak{M}_{16}$ can be learnt by this network in embedding corresponding cliques. Let $(m_1, m_2, ..., m_N)$ be any N-tuple of binary messages in $\mathfrak{M}_B$. If we denote $W(m_n)$ the connection set of the corresponding clique after learning message m_n, the connection set of the associated graph after learning $(m_1, m_2, ..., m_N)$ can therefore

 ISBN: 978-1-61208-218-9

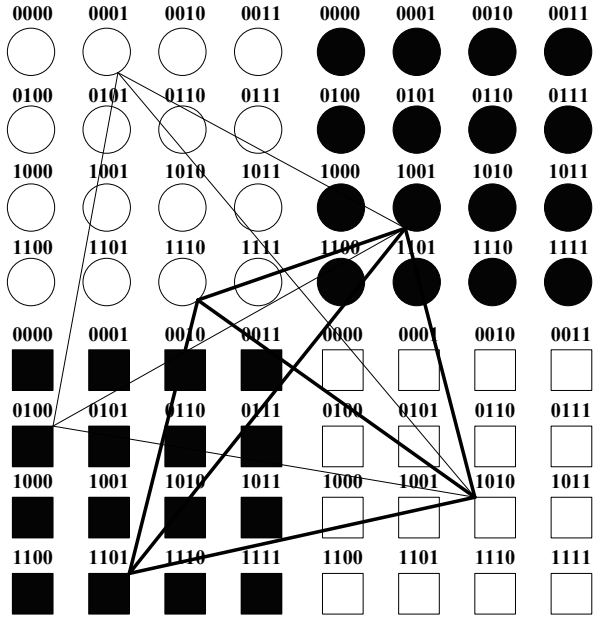

Fig. 1. Learning process illustration for non-oriented clique-based networks. The pattern to learn (with thick edges) connects fanals from four clusters composed of 16 fanals each (filled circles, filled rectangles, rectangles and circles).

be defined by the union:

$$W(m_1, m_2, ..., m_N) = \bigcup_{n=1}^{N} W(m_n) \tag{1}$$

The clique offers a large degree of redundancy that one can take advantage of during the retrieval process. For instance, there are 6 edges in a clique of 4 vertices, but only two of them are sufficient to identify such a clique. If some of the sub-messages are erased, it is likely that the whole message can still be retrieved thanks to this high redundant representation.

Let us denote by v_{ij} the actual value of n_{ij}, which is the j^{th} fanal in the i^{th} cluster. $\omega_{(ij)(i'j')}$ is the connection weight between n_{ij} and $n_{i'j'}$. $\omega_{(ij)(i'j')} = 1$ if this connection exists, 0 otherwise. The message retrieving can then be expressed by an iterative process as following:

$$\forall i, j, v_{ij} \leftarrow \sum_{i'=1}^{c} \min \left(\sum_{j'=1}^{l} \omega_{(ij)(i'j')} v_{i'j'}, 1 \right) + \gamma v_{ij} \tag{2}$$

$$v_i^{max} \leftarrow \max_j \ (v_{ij}) \tag{3}$$

$$\forall i, \forall j, v_{ij} \leftarrow \begin{cases} 1 & \text{if } v_{ij} = v_i^{max} \text{ and } v_i^{max} \geq \sigma \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

Equation (2) counts for each candidate in their corresponding cluster the number of connections to active fanals in other clusters. This equation offers an improvement with respect to that in [8] via the min function, which guarantees that the maximum contribution of a cluster can not exceed one. γ is the memory effect, which we generally set to 1. Equation (3) picks up the maximum fanal value in each cluster. If the input pattern contains the set of fanals corresponding to a learnt message, as these correct fanals connect to at least one active fanal per cluster, they will always have the maximum score after (2), and thus will be selected by (4), which expresses the "winner-take-all" rule. σ is a threshold, which deserves to be well chosen according to different applications. At a particular step of the process, there can be several fanals having the maximum score, which we call ambiguities, in a given cluster. Further iterations are helpful to continuously minimize the number of ambiguities, and hopefully to converge to a stable solution.

The number of messages that these clique-based sparse neural networks are able to learn and recall outperforms the previously state-of-the-art neural networks. For instance, for the same amount of used memory of 1.8×10^6 bits, the clique-based network model with $c = 8$ and $l = 256$ is 250 times superior to Hopfield Neural Networks (HNN) in terms of diversity (the number of messages that the network is able to learn and to retrieve) [8]. The diversity follows a quadratic law of the number of neurons per cluster, while that of HNN follows a sublinear law of the total number of neurons.

III. Learning long sequential messages on chain of tournaments

The clique-based networks offer good performance in learning fixed length atemporal messages. The way to map a sub-message to a particular fanal in the corresponding cluster via a very sparse code makes the length of the sub-message strictly equal to $\log_2(l)$, with l the number of fanals per cluster. All the clusters are synchronously involved in the learning and the retrieving process. An order of sub-messages is naturally predefined by the bijection between clusters and sub-messages. For instance, in Figure 1, this predefined ordering is : circle, filled circle, filled rectangle and rectangle. This ordering is not reflected in the decoding equations (2) - (4).

However, sequentiality and temporality is omnipresent in human cognitive behavior. Non oriented graphs and more particularly the cliques are not suitable to learn and retrieve sequential messages. An architecture with unidirectional links seems the right way to go, since for example it is much more difficult to sing a song in a reversed order. The information dependencies should also be limited in a certain neighborhood of time. For instance, in order to continue playing, a pianist only needs to remember a short sequence of several notes that he has just played, instead of what he played one hour ago. Inspired by clique-based networks, the main contribution of this paper is to propose an oriented graph regularly defined, which we call a "chain of tournaments" that is able to learn very long sequences using a limited number of neurons, and then to retrieve the next element of a sequence uniquely from the knowledge of part of the previous ones.

In graph theory, a tournament is a directed graph obtained by assigning a direction to each edge in a non oriented

 ISBN: 978-1-61208-218-9

complete sub-graph. A tournament offers less redundancy than a clique, since the number of connections is divided by two (one can consider an edge in non oriented graphs as two arrows in opposite direction). The progression of time is then reflected in the succession of tournaments. An example of "chain of tournaments" is illustrated in Figure 2. Clusters are represented by circles, and an arrow represents not a single connection between two fanals, but a set of possible connections between two clusters. One can consider such an arrow as a vectorial connection. The connections are authorized between the cluster i and j, only if $|j-i| \leq r$. r is the incident degree, which is the number of incoming vectorial connections of any cluster.

Let us take as an example the longest word in French "anticonstitutionnellement", which contains 25 letters. If one learns this word using the non-oriented clique-based network introduced in Section II, the network should be composed of 25 clusters of 43 fanals (cardinality of the French alphabet with accented letters). In fact, there are several ways to divide this word into sub-words, all of them leading to a network of an unreasonably large size. (If we divide it into c sub-words of length $\frac{25}{c}$, each cluster should contain $43^{\frac{25}{c}}$ neurons and the total number of neurons would be $c \times 43^{\frac{25}{c}}$. So, the best choice is $c = 25$.) But if one considers this word as a sequence of letters, it can be learnt by the "chain of tournaments" illustrated in Figure 2. The associated connectivity graph after learning this word is partially illustrated by Figure 3. The connections are successively established as the sequence is going on. Any sub-sequence of 4 letters is considered as an entity forming a tournament. For instance, the learning of the sub-sequence "anti" is equivalent to embedding 6 new arrows into the graph: a $\rightarrow$ n, a $\rightarrow$ t, a $\rightarrow$ i, n $\rightarrow$ t, n $\rightarrow$ i and t $\rightarrow$ i. Only three of them are sufficient to define this sub-sequence (a $\rightarrow$ n, n $\rightarrow$ t and t $\rightarrow$ i), and the rest of them serves as redundancy, which one can take advantage of during the decoding process. The learning of the next sub-sequence "ntic" adds another three arrows (n $\rightarrow$ c, t $\rightarrow$ c and i $\rightarrow$ c) to complete a new tournament. The loop structure of this graph enables the reuse of neuron resources. A cluster, and even a neuron, can be used at several times. In Figure 3, when the cluster on the top is solicited for the second time, connections to a new fanal corresponding to the letter "t" are established, without erasing any other existing connections.

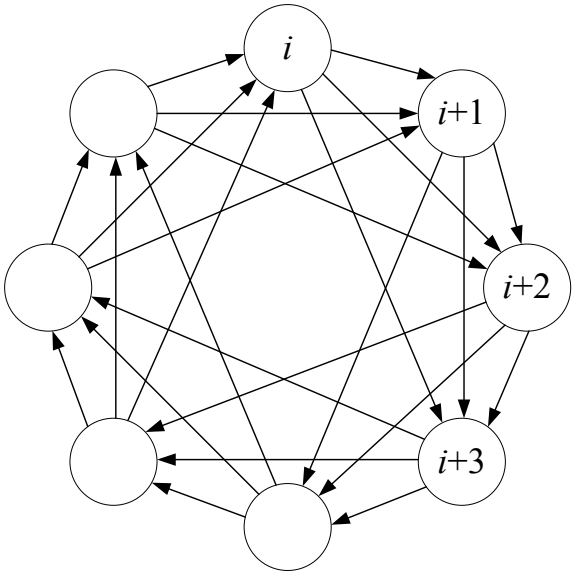

Fig. 2. Structure of the chain of tournaments with 8 clusters and incident degree $r = 3$.

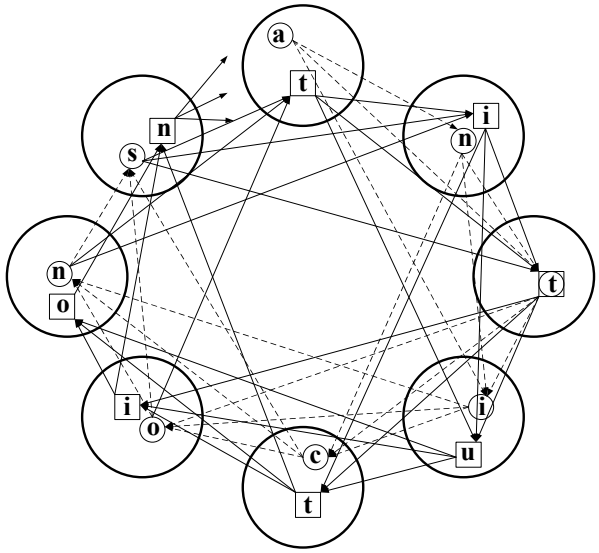

Fig. 3. The partial connectivity graph after learning the longest French word "anticonstitutionnellement" in the chain of tournaments of 8 clusters with incident degree r = 3. For the sake of clarity, only the beginning of the sequence and corresponding connections are represented. The fanals corresponding to the first passage are represented by small circles, while those corresponding to the second passage are represented by squares. All the fanals are exactly of the same nature despite the different representations.

The network is then able to retrieve the whole word from a very limited knowledge of the first three letters "a-n-t". The three fanals corresponding to the sub-sequence "a-n-t" are activated at the beginning. The decision of the fourth letter is made by selecting the fanal in the next cluster with the maximum number of connections to "a-n-t". The correct fanal "i" will be selected with a score of three. Then, the retrieval process continues decoding the next letter from the knowledge of three previous letters "n-t-i", and so on. Obviously, if this sequence contains a repetitive sub-sequence of length larger than 3, this illustrated network is potentially not able to make a correct decision. Fortunately, this is not the case for the word "anticonstitutionnellement". Anyway, it would be possible to add random signatures to complex sequences in order to solve this problem.

Formally, after learning S sequences of length L, the network (chain of tournaments composed of c clusters of l fanals each with parameter r) is defined by:

 ISBN: 978-1-61208-218-9

$$\forall (i,i') \in [|1;c|]^2, \forall (j,j') \in [|1;l|]^2,$$

$$\omega_{(i,j)(i',j')} = \begin{cases} \text{if } 1 \leq (i'-i) \bmod c \leq r \\ 1, \text{ and } \exists s \leq S, \exists k \leq \dfrac{L}{c}, \begin{cases} d^s_{i+(k-1)c} = j \\ d^s_{i'+(k-1)c} = j' \end{cases} \\ 0, \text{ otherwise} \end{cases} \tag{5}$$

$\mathbf{d}$ is the matrix of learnt sequences, where $d^s_{i+(k-1)c}$ refers to the fanal index in the cluster i corresponding to the k^{th} passage on this cluster by the s^{th} sequence.

After learning S sequences, the density of the network, which is defined as the ratio between the number of established connections and that of all potential ones, can be expressed as:

$$d = 1 - \left(1 - \frac{1}{l^2}\right)^{S\frac{L}{c}} \tag{6}$$

To start the retrieval process, the network should be provided with any r consecutive symbols, in particular the first r symbols if we want to retrieve the sequence from the beginning. It is important to note that if the provided part is in the middle of the sequence, one has to know the emplacement of the corresponding clusters to begin with. Formally, the decoding can be expressed as follows:

$$\text{for } r{+}1 \leq p \leq L : \begin{cases} i \leftarrow p \bmod c + 1 \\ \forall j, v_{ij} \leftarrow \\ \displaystyle\sum_{1 \leq \delta(i') \leq r} \min\left(\sum_{j'=1}^{l} \omega_{(i,j)(i',j')} v_{i'j'}, 1\right) \\ \text{where } \delta(i') = (i'-i) \bmod c \\ v_i^{max} \leftarrow \max_j (v_{ij}) \\ \forall j, v_{ij} \leftarrow \begin{cases} 1 \text{ if } v_{ij} = v_i^{max} \\ 0 \text{ otherwise} \end{cases} \end{cases} \tag{7}$$

The sequence retrieval error rate (SRER) is a measure of the network performance, which is here defined as the probability of getting at least one symbol error during the sequence retrieval process, given the first r consecutive symbols of a learnt sequence. After learning S sequences, SRER can be estimated by the following formula:

$$P_e = 1 - \left(1 - \left[1 - \left(1 - \frac{1}{l^2}\right)^{S\frac{L}{c}}\right]^r\right)^{(l-1)(L-r)} \tag{8}$$

By means of simulation, if one considers a chain of tournaments composed of 16 clusters of 512 fanals each with incident degree 9 learning 10000 random sequences of average length 90 (in symbols), that is to say 8.1 Mbits in total, in 98.4% of cases the network retrieves successfully the entire sequence only being provided with the 9 first symbols (10% of the whole sequence length).

For a fixed error probability, on can deduce the diversity of the network as:

$$S_{\max} = \frac{\log\left(1 - \left[1 - (1-P_e)^{\frac{1}{(l-1)(L-r)}}\right]^{\frac{1}{r}}\right)}{\frac{L}{c}\log\left(1 - \frac{1}{l^2}\right)} \tag{9}$$

The maximum number of bits stored by the network is expressed by:

$$C_{\max} = S_{\max} kc \log_2(l) \tag{10}$$

where $k = \frac{L}{c}$, the number of re-use of each cluster. The quantity of memory used by the network is:

$$Q = rcl^2 \tag{11}$$

This leads to the expression of the network efficiency, which is the ratio $\frac{C_{\max}}{Q}$:

$$\eta = \frac{S_{\max} k \log_2(l)}{rl^2} \tag{12}$$

Note that the efficiency is not directly dependant on the number of clusters c, but on k, the number of re-use of each cluster.

The previous equations lead to Table 1 that gives theoretical values for several different configurations of the network. With a sufficient incident degree r, the network efficiency reaches around 20%.

TABLE I
MAXIMUM NUMBER OF SEQUENCES (DIVERSITY) $S_{\max}$ THAT A CHAIN OF TOURNAMENTS IS ABLE TO LEARN AND RETRIEVE WITH AN ERROR PROBABILITY SMALLER THAN 0.01, FOR DIFFERENT VALUES OF c, l, r AND L. THE VALUES OF CORRESPONDING EFFICIENCY η ARE ALSO MENTIONED.

c	l	r	L	$S_{\max}$	η
8	512	2	16	155	0.5%
8	512	3	16	1513	3.4%
8	512	3	32	578	2.6%
8	512	2	64	225	2%
20	512	10	100	12741	21.9%
50	512	20	1000	1823	22.2%

The propagation of errors is especially harmful in successive decoding process. One can investigate the proportion of non propagative errors which do not cause a second error in following consecutive r decoding steps. In Figure 4, the chain of tournaments of 16 clusters of 512 fanals with $r = 9$ is able to learn and retrieve 15000 sequences of 90 symbols, that is to say 810 bits, while maintaining a satisfying proportion (more than 90%) of non propagative errors. Logically, a chain of tournaments with $r = c - 1$ offers the best performance possible.

 ISBN: 978-1-61208-218-9

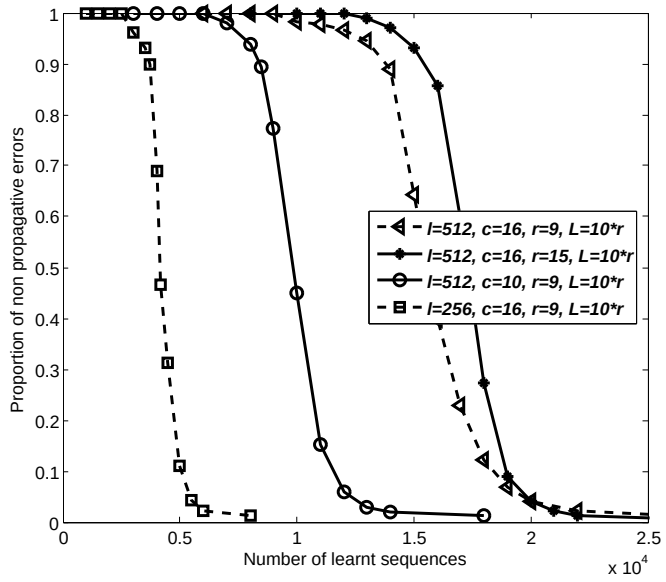

Fig. 4. Proportion of non propagative errors in function of the number of the sequences learnt by chains of tournaments. The first input symbols are of 10% length of the whole sequences.

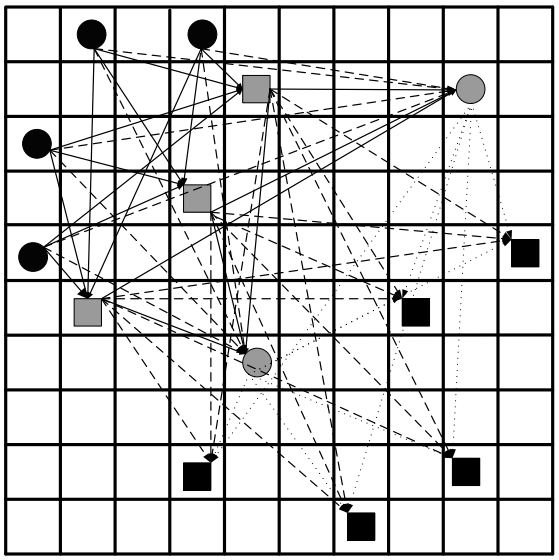

Fig. 5. Learning vectorial sequences in a network composed of 100 clusters by the generalized chain of tournaments. Clusters are represented by squares in the grid. Four patterns with different sizes are represented: filled circles, grey rectangles, grey circles and filled rectangles. The incident degree is $r = 2$.

IV. Learning vectorial sequences

As a matter of fact, the structure represented in Figure 2 is a generalization of clique-based networks. It becomes a clique by setting $r = c - 1$ (two oriented connections being equivalent to a non oriented one). It is still possible to generalize furthermore this topology:

1) A chain of tournaments is not necessarily a closed loop;
2) A given element of the sequence at time τ, s^τ, is not necessarily a single symbol, but a set of parallel symbols that corresponds to a set of fanals in different clusters. The sequence then becomes vectorial. We call these sets of parallel symbols as "vectors" or "patterns".

An illustration of this generalization is given in Figure 5. The network is composed of 100 clusters, represented by squares in the grid. Four patterns with different sizes are represented: filled circles (size 4), grey rectangles (size 3), grey circles (size 2) and filled rectangles (size 5). There are no connections within a pattern. Two patterns that are linked are associated through an oriented complete bipartite graph. The succession of patterns is then carried by a chain of tournaments with parameter r. In Figure 5, we have $r = 2$, since the first pattern (filled circles) is connected to the second (grey rectangles) and to the third (grey circles), but not to the fourth (filled rectangles). This concept is similar to that in [10], although the latter only considers connections between two consecutive patterns, and the way of the organization in clusters is different.

During the decoding process, the network is provided with r successive patterns. A priori, the locality of the next pattern is unknown. As a consequence, at each step of the decoding, one has to process a global "winner-take-all" rule instead of a local selection expressed in (3) or (7). In other words, one has to go through the whole network to select all the fanals with the maximum score, which is normally the product of the incident degree and the size of patterns, instead of doing this selection within selected clusters.

By means of simulation, our network learns a set of long vectorial sequences composed of randomly generated patterns, and it shows outstanding performance in retrieving them. For example, with the incident degree $r = 1$, the network composed of only 6400 neurons (100 clusters $\times$ 64 fanals/cluster) is able to learn a sequence of 40000 random patterns of size 20 each, that is to say about 10 Mbits, with a SRER = 10% despite a relatively high network density $d = 0.32$. Nevertheless, this network is more at ease to learn sequences of big patterns (for instance, size 20) rather than those of small patterns (for instance, size 3), which suffers more from the problem of error propagation and diaphony. Since the patterns are randomly generated, one has few chance to get too many similar patterns. Anyway, in a similar way as mentioned in Section III, our model is also able to learn sequences of correlated patterns as well as those of non correlated ones, although the provided input patterns should not be strongly correlated with the rest of the sequence. The cost is to add random signatures in order to decorrelate the source, and to correspondingly double the number of clusters and neurons.

V. Conclusion and opening

While the clique-based non oriented networks enable the learning of fixed length atemporal messages, the model proposed in this paper is able to learn very long sequences, the length of which is not limited by the size of the network, but

 ISBN: 978-1-61208-218-9

only by its binary resource. As described in Section IV, the network made of 6400 neurons is able to learn a vectorial sequence composed of 40000 patterns of 20 parallel symbols, which corresponds to about 10 Mbits of information. This property could give them the ability to encode the flows with voluminous information, such as multimedia streams.

This model remains simple to be implemented, since all the connections and the neurons are binary. Oriented graphs are biologically plausible, as synapses (neuronal inputs) and axons (neuronal outputs) are not interchangeable.

Generally, time is embodied in a temporal message in two ways: temporal order and time duration. By now, the time involved in our model is discrete. It would be thus interesting to introduce the notion of duration in associative memories, which will have utilities to applications like natural language processing considering phoneme sequences. The learning of abstract structure [11] [12], which might lead to a hierarchical architecture, is another interesting possibility that remains open to further investigation.

REFERENCES

[1] J. L. Elman, “Finding structure in time”, *Cognitive Science*, vol. 14, pp. 179-211, 1990.

[2] A. Cleeremans, D. Servan-Schreiber, and J. L. McClelland, “Finite state automata and simple recurrent networks”, *Neural Computation*, vol. 1, pp. 372-381, 1989.

[3] D. Wang and M. A. Arbib, “Complex temporal sequence learning based on short-term memory”, *Proceedings of the IEEE*, vol. 78, no. 9, September 1990.

[4] D. Wang and B. Yuwono, “Incremental learning of complex temporal patterns”, *IEEE Transactions on Neural Networks*, vol. 7, pp. 1465-1481, 1996.

[5] J. J. Hopfield, “Neural networks and physical systems with emergent collective computational properties”, *Proceedings of the Natural Academy of Sciences, Biophysics*, vol. 79, pp. 2554-2558, USA, 1982.

[6] M. McCloskey and N. J. Cohen, “Catastrophic interference in connectionist networks: The sequential learning problem”, *The Psychology of Learning and Motivation*, vol. 23, Academic Press, pp. 109-164, New York, 1989.

[7] R. M. French, “Catastrophic forgetting in connectionist networks: causes, consequences and solutions”, *Trends in Cognitive Science*, vol. 3(4), pp. 128-135, 1999.

[8] V. Gripon and C. Berrou, “Sparse neural networks with large learning diversity”, *IEEE Transactions on Neural Networks*, vol. 22, no. 7, July 2011.

[9] V. Gripon and C. Berrou, “A Simple and efficient way to store many messages using neural cliques”, *IEEE Symposium Series on Computational Intelligence*, Paris, April 2011.

[10] G. J. Rinkus, “TEMECOR: An associative, spatio-temporal pattern memory for complex state sequences”, *Proceedings of the World Congress on Neural Networks*, Washington, D.C., pp. I.442-I.448, 1995.

[11] G. F. Marcus, S. Vijayan, S. Bandi Rao, and P. M. Vishton, “Rule learning by seven-month-old infants”, *Science*, vol. 283, no. 5398, pp. 77-80, January 1999.

[12] T. Lelekov and P. F. Dominey, “Human brain potentials reveal similar processing of non-linguitic abstract structure and linguistic syntactic structure”, *Neurophysiologie Clinique*, vol. 32, pp. 72-84, 2002.

 ISBN: 978-1-61208-218-9