Coupled Neural Associative Memories

Amin Karbasi, Amir Hesam Salavati, and Amin Shokrollahi
School of Computer and Communication Sciences
Ecole Polyetechnique Federale de Lausanne (EPFL)
Switzerland

Abstract—"THIS PAPER IS ELIGIBLE FOR THE STUDENT PAPER AWARD". **We propose a novel architecture to design a neural associative memory that is capable of learning a large number of patterns and recalling them later in presence of noise. It is based on dividing the neurons into local clusters and parallel plains, very similar to the architecture of the visual cortex of macaque brain. The common features of our proposed architecture with those of spatially-coupled codes enable us to show that the performance of such networks in eliminating noise is drastically better than the previous approaches while maintaining the ability of learning an exponentially large number of patterns. Previous work either failed in providing good performance during the recall phase or in offering large pattern retrieval (storage) capacities. We also present computational experiments that lend additional support to the theoretical analysis.**

I. INTRODUCTION

While relying on iterative operations of simple (and sometimes faulty) neurons, our brain is capable of retrieving the correct "memory" with high degrees of reliability even when the cues are limited or inaccurate. Not surprisingly, designing artificial neural networks capable of accomplishing this task, called *associative memory*, has been a major point of interest in the neuroscience community for the past three decades. This problem, in its core, is in fact very similar to the reliable information transmission faced in communication systems where the goal is to find mechanisms to efficiently encode and decode a set of transmitted patterns over a noisy channel. More interestingly, the novel techniques employed to design good codes are extremely similar to those used in designing and analyzing neural networks. In both cases, graphical models, iterative algorithms, and message passing play a central role.

Despite these similarities in the task and in the techniques, we witness a huge gap in terms of the efficiency achieved by them. More specifically, by using modern coding techniques, we are capable of reliably transmitting 2^{rn} binary vectors of length n over a noisy channel ($0 < r < 1$). This is achieved by intelligently introducing redundancy among the transmitted messages, that are later used to recover the correct pattern from the received noisy version. In contrast, until recently, artificial neural associative memories were only capable of memorizing $O(n)$ binary patterns of length n (see, [1], [2], [3], [4]).

Part of the reasons for this gap goes back to the assumption held in the mainstream work on artificial associative memories which requires the network to memorize *any* set of randomly chosen binary patterns. While it gives the network a certain degree of versatility, it severely hinders the efficiency.

To achieve an exponential scaling in the storage capacity

of neural networks Kumar et al. [5] suggested a different viewpoint in which the network is no longer required to memorize *any* set of random patterns but only those that have some common *structure*, namely, patterns all belong to a subspace with dimension $k < n$. By assuming that the connectivity matrix of the neural graph is given, the authors proposed a simple iterative algorithm for the recall phase. However, they did not propose any algorithm for the learning phase, i.e., learning the connectivity matrix. This task was then accomplished in [16]. Although the proposed approaches are capable of achieving exponential scaling for the pattern retrieval capacity, the performance of the algorithms employed in the recall phase are still far from being desirable.

In this work, we follow the same viewpoint as in [5] where we only focus on memorizing a set of patterns with some degree of redundancy. However, we propose a different neural architecture to capture local correlations among patterns, similar to the way that the receptive field in human visual cortex is arranged. We use the algorithm proposed in [16] for the learning phase. For the recall phase, we also employ the algorithm proposed in [11] as a building block. However, due to the novel structure, we will need a more powerful tool to investigate the performance of the recall algorithm. This is done by making use of the recent developments in the analysis spatially-coupled codes [14], [13].

II. RELATED WORK

The first artificial neural associative memory was introduced in the pioneering work of Hopfield [1]. A "Hopfield network" is a complete graph of n neurons, equipped with the Hebbian learning rule [6] to memorize a set of *randomly* chosen binary patterns of length n. It was shown by McEliece et al. that the pattern retrieval capacity of Hopfield networks is $\mathcal{C} = (n/2 \log(n))$ [7].

Since then, there have been many attempts to increase the pattern retrieval capacity of neural associative memories. In particular, Venkatesh et al. [2] sacrificed the online learning capability to increase the capacity by calculating the weight matrix offline [12]. The result is a pattern retrieval capacity of $n/2$ random patterns with the ability of one bit error correction. Jankowski et al. [3] investigated a different model with multi-state associative memory in which each neuron can be assigned a multivalued state from the set of complex numbers. It was shown by Muezzinoglu et al. [4] that the capacity of such networks can be increased to $\mathcal{C} = n$ at the cost of a prohibitive weight computation mechanism.

More recently, a new perspective has been proposed with the aim of memorizing only those patterns that posses some degree of redundancy. In this framework, a tradeoff is being made between versatility (i.e., the capability of the network to memorize any set of random patterns) and the pattern retrieval capacity. Pioneering this frontier, Berrou and Grippon considered memorizing patterns based on Walsh-Hadamard codes [9]. While the suggested approach increases the capacity beyond n, the complexity of operations in the recall phase was prohibitive. Gripon and Berrou [8] recently proposed another method based on neural clicks which increases the pattern retrieval capacity potentially to $O(n^2/\log(n))$ with a low complexity algorithm in the recall phase. The proposed approach is based on memorizing a set of patterns mapped from randomly chosen binary vectors of length $k = O(\log(n))$ to the n-dimensional space. This way, one could benefit from the extra redundancy within the patterns to increase the capacity as well as correcting errors, much in the same way error correcting codes do in communication systems.

By considering patterns that come from a subspace with dimension $k < n$, Kumar et al. were able to show an exponential scaling in the pattern retrieval capacity, i.e., $\mathcal{C} = O(a^n)$, with some $a > 1$. This model was later extended to modular patterns, i.e., those in which patterns are divided into non-overlapping sub-patterns where each sub-pattern come from a subspace [11], [16]. The authors have proposed a simple iterative learning algorithm as well as achieving an improved performance in the recall phase as compared to [5].

In this paper, we follow the same line of work by extending the model proposed in [11], [16] to neural networks with modular structure in three dimensions, similar to the way visual cortex of the macaque brain is organized [17]. The proposed model is based on *overlapping* local clusters with neighboring neurons, where clusters are arranged in parallel planes. At the same time, there are sparse connections between various clusters in different planes. The aim is to memorize only those patterns for which local sub-patterns in the domain of each cluster show a certain degree of redundancy.

Interestingly, this model is very similar to spatially-coupled codes on graphs [13]. This similarity helps us borrow analytical tools developed for analysing such codes [14] and investigate the performance of our proposed error correcting algorithm. Specifically, our suggested model is closely related to the spatially-coupled Generalized LDPC code (GLDPC) with Hard Decision Decoding (HDD) proposed in [15]. In our model, the clusters can be regarded as "component codes" in the GLDPC ensemble and the parallel planes as spatial-coupling of the individual decoders. However, in contrast to [15], our proposed error correction algorithm is purely based on simple operations performed by neurons, and message passing even within clusters (i.e., component codes). Furthermore, due to the structure of neural networks, a neuron sends out the same message to all of its neighbour as it cannot differentiate among them. This is different from the type of message passing algorithms (e.g., belief propagation) performed on LDPC codes.

III. PROBLEM SETTING AND NOTATIONS

Let $\mathcal{X}$ denote a dataset of $\mathcal{C}$ patterns of length n. In this paper, we assume that patterns are integer-valued with entries in $\{0, \ldots, S-1\}$. A natural way of interpreting this assumption is to consider the entries as the short-term firing rate of corresponding neurons. We divide each pattern into L sub-patterns of the same size and call them *planes*. Within each plane, we further divide the patterns into D *overlapping* clusters, i.e., an entry in a pattern can lie in the domain of multiple clusters (within the same or other planes). We also assume that each pattern neuron in plane ℓ is connected to at least one cluster in planes $\ell - \Omega, \ldots, \ell + \Omega$ (except at the boundaries). Therefore, each pattern neuron is connected to $2\Omega + 1$ planes, on average. Finally, to introduce redundancy, we assume that entries of each cluster belong to a subspace (or more generally, having negligible minor components).

Learning phase: This phase corresponds to learning the minor components or dual vectors for each cluster. In this paper, we use the learning algorithm proposed in [16]. The output of the learning algorithm is a matrix $W^{(\ell,d)}$ for cluster d in plane ℓ. The rows of this matrix correspond to the dual vectors and the columns correspond to the pattern nodes. Therefore, by letting $\mathbf{x}^{(\ell,d)}$ denote the sub-pattern corresponding to the domain of cluster d of plane ℓ, we have

$$W^{(\ell,d)} \cdot \mathbf{x}^{(\ell,d)} = \mathbf{0}. \tag{1}$$

By treating these matrices as connectivity matrices of the neural graph, one can consider each cluster as a bipartite graph composed of *pattern* and *constraint* neurons. The constraint neurons do not have any overlap (i.e. each one belongs only to one cluster) whereas the pattern neurons can have connections to multiple clusters. To ensure good error correction capabilities we aim to keep these connections sparse.

Putting all the local connectivity matrices together, we obtain the model shown in Figure 1. Interestingly, this model is very similar to the neural architecture to process visual signals in macaque brain [17]. As mentioned earlier, we have L planes with n/L pattern neurons. Each plane contains D clusters where cluster d in plane ℓ contains $m_{\ell,d}$ constraints neurons and is connected to $n_{\ell,d}$ pattern neurons. Note that due to overlaps among clusters, we have $\sum_d n_{\ell,d} > n/L$.

We also consider the overall connectivity graph of plane ℓ, denoted by $\tilde{W}^{(\ell)}$, in which the constraint nodes in each cluster are compressed into one *super node*. Any pattern node that is connected to a given cluster is connected with an (unwedighted) edge to the corresponding super node. Figure 2 illustrates this graph for plane 1 in Figure 1.

For graph $\tilde{W}^{(\ell)}$, let $\lambda_i^{(\ell)}$ and $\rho_j^{(\ell)}$ be the fraction of edges connected to pattern and constraint nodes with degree i and j, respectively. We define the degree distribution polynomials in plane ℓ from an *edge perspective* as $\lambda^{(\ell)}(x) = \sum_i \lambda_i^{(\ell)} x^{i-1}$ and $\rho^{(\ell)}(x) = \sum_j \rho_j^{(\ell)} x^{j-1}$.

Recall phase: This phase corresponds to retrieving correct memorized patterns in response to noisy queries. At this point, the neural graph has been learned (fixed) and we are looking
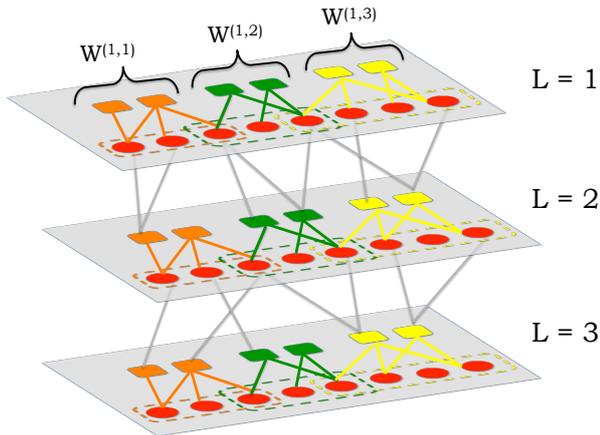
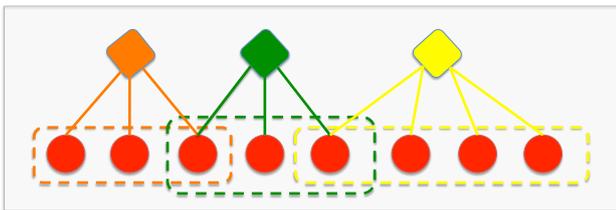Fig. 1: A coupled neural associative memory.

Fig. 2: A connectivity graph with neural planes and super nodes. It corresponds to plan 1 of Fig. 1.

for a simple iterative algorithm to eliminate noise from the query. In this paper, we assume that the the noise is additive, i.e., the query is one of the memorized patterns plus some noise. The noise itself is integer-valued and for simplicity we assume that its entries are $\{-1, 0, +1\}$, where a -1 (resp. $+1$) corresponds to a neuron skipping a spike (resp. fire one more spike) than expected. More specifically, if the initial error probability is denoted by p_e, then each entry of the noise vector is $+1$ or -1 with probability $p_e/2$. In this paper, we propose an algorithm to eliminate this input noise.

Pattern Retrieval Capacity: Finally, an ideal neural associative memory should have a large pattern retrieval capacity. This is the maximum number of patterns that can be memorized by a network while still being able to return reliable responses in the recall phase.

IV. MAIN RESULTS

In this section, we briefly discuss our approach for addressing the aforementioned phases.

Learning phase: As we mentioned earlier, for the learning phase we use the learning algorithm proposed in [16]. In the remaining, we effectively assume that the connectivity matrices $W^{(\ell,d)}$ are known and satisfy Eq. 1.

Recall phase: The proposed recall algorithm in this paper is the extension of the one proposed in [11] to the coupled neural networks. For the sake of completeness, we have summarized this approach in Algorithm 1, the goal of which is to correct

---

Algorithm 1 Error Correction Within Cluster [11]

Input: Connectivity matrix $W^{(\ell,d)}$, threshold φ, iteration $t_{\max}$.
Output: Correct memorized sub-pattern $\mathbf{x}^{(\ell,d)}$.

1: **for** $t = 1 \rightarrow t_{\max}$ **do**
2:    *Forward iteration:* Calculate the weighted input sum $h_i = \sum_{j=1}^{n} W_{ij}^{(\ell,d)} x_j^{(\ell,d)}$, for each neuron $y_i^{(\ell,d)}$ and set:
$$y_i^{(\ell,d)} = \begin{cases} 1, & h_i < 0 \\ 0, & h_i = 0 \\ -1, & \text{otherwise} \end{cases}$$
3:    *Backward iteration:* Each neuron $x_j^{(\ell,d)}$ computes
$$g_j^{(\ell,d)} = \frac{\sum_{i=1}^{m_{\ell,d}} W_{ij}^{(\ell,d)} y_i^{(\ell,d)}}{\sum_{i=1}^{m_{\ell,d}} |W_{ij}^{(\ell,d)}|}.$$
4:    Update the state of each pattern neuron j according to $x_j^{(\ell,d)} = x_j^{(\ell,d)} + \text{sgn}(g_j^{(\ell,d)})$ only if $|g_j^{(\ell,d)}| > \varphi$.
5: **end for**

(at least) a single error in a cluster with high probability. Let us remind ourselves of the performance of Algorithm 1.

Theorem 1: [11] When $\varphi \rightarrow 1$, Algorithm 1 can correct a single error in each cluster with probability at least $1 - (\bar{d}/m)^{d_{\min}}$, where $\bar{d}$ and $d_{\min}$ are the average and minimum degree of the pattern nodes within the cluster domain.

We apply Algorithm 1 *sequentially* to neural clusters and neural planes as follows. We start with the first cluster in the first plane and apply Algorithm 1. Once finished, we check the final state of the constraint nodes, expecting them to be all zero. If not, we revert the values of the pattern nodes to their initial values. We then proceed to the second cluster in the first plane and continue until we reach the last cluster in the last plane. We repeat the above sequential scheduling $t_{\max}$ times (see Algorithm 2). Repeating is particularly helpful since errors get corrected in each round which makes noisy clusters experience less amount of noise for the following rounds and thus higher success rate for the application of Algorithm 1 on the clusters. Note that by Theorem 1, Algorithm 1 can only guarantee the correction of a single error. Hence, applying Algorithm 1 to clusters with two or more errors might be unsuccessful. By sweeping $t_{\max}$ times over the clusters and planes, we try to correct noisy neurons one by one.

We consider two variants of the above error correction algorithm. In the first one, called *constrained* coupled neural error correction, we provide the network with some side information during the recall phase. This is equivalent to "freezing" a few of the pattern neurons to known correct states, similar to spatially-coupled codes [13], [14]. In the case of neural associative memory, the side information can come from the context. For instance, when trying to fill in the blank in the sentence "The _ at flies", we can use the side information (flying) to guess the correct answer among multiple choices. Without this side information, we cannot tell

Algorithm 2 Error Correction of the Coupled Network

Input: Connectivity matrix ($W^{(\ell,d)}, \forall \ell, \forall d$), iteration $t_{\max}$
Output: Correct memorized pattern $\mathbf{x} = [x_1, x_2, \ldots, x_n]$
1: **for** $t = 1 \rightarrow t_{\max}$ **do**
2:    **for** $\ell = 1 \rightarrow L$ **do**
3:      **for** $d = 1 \rightarrow D_\ell$ **do**
4:        Apply Algorithm 1 to cluster d of neural plane ℓ.
5:        Update the value of pattern nodes $\mathbf{x}^{(\ell,d)}$ only if all the constraints in the clustered are satisfied.
6:      **end for**
7:    **end for**
8: **end for**

---

if _ *at* corresponds to *bat* or *cat*.

In the other variant, called *unconstrained* coupled neural error correction, we perform the error correction without providing any side information. This is similar to many standard recall algorithms in neural networks and serves as a benchmark to compare our method with those of other work [5], [11].

Let $z^{(\ell)}(t)$ denote the *average* probability of error for pattern nodes across neural plane ℓ in iteration t. A cluster node in plane ℓ receives noisy messages from its neighbors with an average probability of $\bar{z}^{(\ell)}$, where

$$\bar{z}^{(\ell)} = \frac{1}{2\Omega + 1} \sum_{j=-\Omega}^{\Omega} z^{(\ell-j)} \text{ s.t. } z^{(l)} = 0, \ \forall l \notin \{1, \ldots, L\}.$$

In the following, we derive a recursive expression for $z^{(\ell)}(t)$ (all the proofs are provided in our technical report [19]).

Lemma 2: Let us define $g(z) = 1 - \rho(1-z) + z\rho'(1-z)$ and $f(z; p_e) = p_e \lambda(z)$. Then,

$$z^{(\ell)}(t+1) = f\left(\frac{1}{2\Omega + 1} \sum_{i=-\Omega}^{\Omega} g(\bar{z}^{(\ell-i)}(t); p_e)\right). \quad (2)$$

The decoding will be successful if $z^{(\ell)}(t+1) < z^{(\ell)}(t), \ \forall \ell$. As a result, we look for the maximum p_e such that

$$f\left(\frac{1}{2\Omega + 1} \sum_{i=-\Omega}^{\Omega} g(\bar{z}^{(\ell-i)}(t); p_e)\right) < z^{(\ell)} \text{ for } z^{(\ell)} \in [0, p_e].$$

Let p_e^* and $p_e^\dagger$ denote the maximum success probability for the constrained and unconstrained coupled system, respectively. We will use the analytical approaches recently proposed in [14] to find these thresholds. To this end, a potential function is defined to track the evolution of the error probability in Eq. 2. Let $\mathbf{f}(z; p_e) : \mathbb{R}^n \rightarrow \mathbb{R}^n$ and $\mathbf{g}(z) : \mathbb{R}^n \rightarrow \mathbb{R}^n$ be two component-wise vector functions operating on a vector $\mathbf{z}$ such that $[\mathbf{f}(\mathbf{z}; p_e)]_i = f(z_i; p_e)$ and $[\mathbf{g}(\mathbf{z})]_i = g(z_i)$, where $f(z_i; p_e)$ and $g(z_i)$ are defined in Lemma 2 . Using these definitions, we can rewrite Eq. 2 in the vector form as [14]:

$$\mathbf{z}(t+1) = A^\top \mathbf{f}(A\mathbf{g}(\mathbf{z}(t)); p_e) \quad (3)$$

where A is the *coupling matrix* defined as[1]:

---

[1]Matrix A corresponds to the unconstrained system. A similar matrix can be defined for the constrained case.

$$A = \frac{1}{2\Omega+1} \begin{bmatrix} \overbrace{1 \quad 1 \quad \cdots \quad 1}^{\Omega} & 0 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 1 & \cdots & 1 & 1 & 0 & 0 & \cdots & 0 & 0 \\ \vdots \\ 0 & 0 & \cdots & 0 & 0 & 1 & 1 & \cdots & 1 & 1 \\ 0 & 0 & \cdots & 0 & 0 & 0 & 1 & \cdots & 1 & 1 \end{bmatrix}$$

At this point, the potential function of the unconstrained coupled system could be defined as [14]:

$$\begin{aligned} U(\mathbf{z}; p_e) &= \int_C \mathbf{g}'(\mathbf{u})(\mathbf{u} - A^\top \mathbf{f}(A\mathbf{g}(\mathbf{u}))).d\mathbf{u} \\ &= \mathbf{g}(\mathbf{z})^\top \mathbf{z} - G(\mathbf{z}) - F(A\mathbf{g}(\mathbf{z}); p_e) \quad (4) \end{aligned}$$

where $\mathbf{g}'(\mathbf{z}) = \text{diag}([g'(u_i)])$, $G(\mathbf{z}) = \int_C \mathbf{g}(\mathbf{u}) \cdot d\mathbf{u}$ and $F(\mathbf{z}) = \int_C \mathbf{f}(\mathbf{u}) \cdot d\mathbf{u}$.

The potential function is defined in the way that $\min\{U(\mathbf{z}; p_e)\} > 0$ for $p_e < p_e^*$. In contrast, for the unconstrained coupled system, we have $U'(\mathbf{z}; p_e) > 0$ for $p_e \leq p_e^\dagger$. In other words, in order to find p_e^*, it is sufficient to find the maximum p_e such that $\min\{U(\mathbf{z}; p_e)\} > 0$ [14]. We will use this fact and compare the two thresholds later in the simulations section. Intuitively, we expect to have $p_e^\dagger \leq p_e^*$ (side information only helps), and as a result a better error correction performance for the constrained system.

The next theorem shows that for $p_e \leq p_e^\dagger$, the error probability of the unconstrained coupled system goes to zero.

Theorem 3: For $p_e \leq p_e^\dagger$, the fixed point of Eq. 2 is 0 and is achieved by iterative updates of Algorithm 2.

Similarly, to address the performance of the constrained coupled system, we use the results of [14] and [13].

Theorem 4: For the constrained coupled neural associative memory, when $p_e < p_e^*$ the potential function decreases in each iteration. Furthermore, if $L > \|U''(\mathbf{z}; p_e)\|_\infty / \Delta E(p_e)$, the only fixed point of Eq. 3 is $\mathbf{0}$.

Pattern retrieval capacity: In this part, we prove that the number of patterns that can be memorize by the proposed scheme is exponential in n, the pattern size. First, note that this number is only a function of the size of the subspace that sub-patterns come from. In other words, the number of patterns $\mathcal{C}$ which the network memorizes does not depend on the learning or recall algorithms (except for its obvious effect on the running time). Therefore, in order to prove that $\mathcal{C}$ could exponentially scale with n, we show that there exists a subspace with exponentially large number of members (in terms of n). The following theorem is the extension of the one we proved in [11], [16] to coupled neural associative networks.

Theorem 5: Let $\mathcal{X}$ be the $\mathcal{C} \times n$ dataset matrix, formed by $\mathcal{C}$ vectors of length n with entries from the set $\mathcal{S}$. Let also $k = rn$ for some $0 < r < 1$. Then, there exists a set of patterns for which $\mathcal{C} = a^{rn}$, with $a > 1$, and $\text{rank}(\mathcal{X}) = k < n$.

V. Simulations

In this paper, we are mainly interested in the performance of the recall phase and demonstrate a way, by the means of spatial coupling, to improve upon the previous art. To this end, we assume that the learning phase is done (by using our proposed algorithm in [16]) and we have the weighted connectivity graphs available. For the ease of presentation, we can simply produce these matrices by generating sparse random bipartite graphs and assign random weights to the connections. Given the weight matrices and the fact that they are orthogonal to the sub-patterns, we can assume w.l.o.g that in the recall phase we are interested in recalling the all-zero pattern from its noisy version.

We treat the patterns in the database as $2D$ images of size 64×64. More precisely, we have generated a random network with 29 planes and 29 clusters within each plane (i.e., $L = D = 29$). Each local cluster is composed of 8×8 neurons and each pattern neuron (pixel) is connected to 2 consecutive planes and 2 clusters within each plane (except at the boundaries). This is achieved by moving the 8×8 rectangular window over the $2D$ pattern horizontally and vertically.

We investigated the performance of the recall phase by randomly generating a $2D$ noise pattern in which each entry is set to ± 1 with probability $p_e/2$ and 0 with probability $1 - p_e$. We then apply Algorithm 2 to eliminate the noise. Once finished, we declare failure if the output of the algorithm, $\hat{\mathbf{x}}$, is not equal to the pattern $\mathbf{x}$ (assumed to be the all-zero vector).

Figure 3 illustrates the final error rate of the proposed algorithm, for the constrained and uncosntrained system. To obtain the final recall error probability, we have repeated Algorithm 2 for $t_{\max} = 10$ times. For the constrained system, we fixed the state of a patch of neuron on size 3×3 at the four corners of the $2D$ pattern. In the same figure, the results are also compared to the similar algorithms in [5] and [11]. In [5], there are no clustering while in [11] the network is divided into some *non-overlapping* clusters with a second nerual level to compensate for the lack of collaboration among non-overlapping clusters. As obvious from the figure, the performance of the proposed algorithms in this paper is significantly better than the ones in [5] and [11].

VI. Conclusions

In this paper, we proposed a novel architecture for neural associative memories. The proposed model comprises a set of neural planes with sparsely connected overlapping clusters. Given the similarity of the suggested framework to spatially-coupled codes, we employed recent developments in analyzing these codes to investigate the performance of our proposed neural algorithm. We also presented numerical simulations that lend additional support to the theoretical analysis.
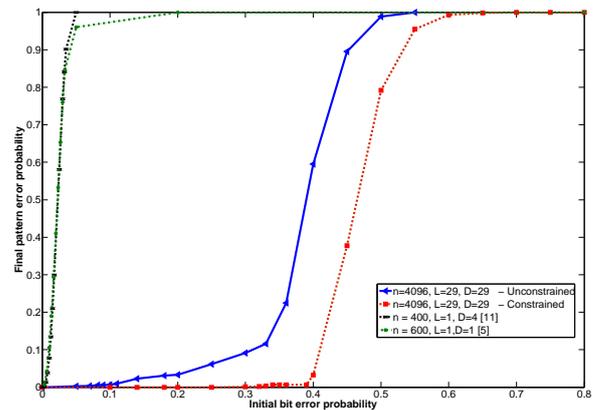
Acknowledgments

Fig. 3: The final pattern error probability for the constrained and unconstrained coupled neural systems.

References

[1] J. J. Hopfield, *Neural networks and physical systems with emergent collective computational abilities*, Proc. Natl. Acad. Sci., Vol. 79, 1982, pp. 2554-2558.

[2] S. S. Venkatesh, D. Psaltis, *Linear and logarithmic capacities in associative neural networks*, IEEE Trans. Inf. Theory, Vol. 35, No. 3, 1989, pp. 558-568.

[3] S. Jankowski, A. Lozowski, J.M., Zurada, *Complex-valued multistate neural associative memory*, IEEE Tran. Neur. Net., Vol. 1 , No. 6, 1996, pp. 1491-1496.

[4] M. K. Muezzinoglu, C. Guzelis, J. M. Zurada, *A new design method for the complex-valued multistate Hopfield associative memory*, IEEE Trans. Neur. Net., Vol. 14, No. 4, 2003, pp. 891-899.

[5] K.R. Kumar, A.H. Salavati and A. Shokrollahi, *Exponential pattern retrieval capacity with non-binary associative memory*, Proc. IEEE Information Theory Workshop, 2011.

[6] D. O. Hebb, *The organization of behavior*, New York: Wiley & Sons, 1949.

[7] R. McEliece, E. Posner, E. Rodemich, S. Venkatesh, *The capacity of the Hopfield associative memory*, IEEE Trans. Inf. Theory, Jul. 1987.

[8] V. Gripon, C. Berrou, *Sparse neural networks with large learning diversity*, IEEE Trans. on Neural Networks, Vol. 22, No. 7, 2011, pp. 10871096.

[9] C. Berrou, V. Gripon, *Coded Hopfield Networks*, Proc. Symp. on Turbo Codes and Iterative Information Processing, pp. 15, 2010.

[10] P. Peretto, J. J. Niez, *Long term memory storage capacity of multiconnected neural networks*, Biological Cybernetics, Vol. 54, No. 1, 1986, pp. 53-63.

[11] A. H. Salavati, A. Karbasi, *Multi-Level Error-Resilient Neural Networks*, IEEE Int. Symp. Inf. Theory (ISIT 2012), 2012.

[12] J. Hertz, A. Krogh, R. G. Palmer, *Introduction to the theory of neural computation*, USA: Addison-Wesley, 1991.

[13] S. Kudekar, T. Richardson, R. Urbanke, *Threshold saturation via spatial coupling: why convolutional LDPC ensembles perform so well over the BEC*, IEEE Trans. Inf. Theory, Vol. 57, No. 2, 2012, pp. 803-834.

[14] A. Yedla, Y. Jian, P. S. Nguyen, H. D. Pfister, *A simple proof of threshold saturation for coupled scalar recursions*, To appear in Int. Symp. Turbo codes and Itr. Info. Processing (ISTC), 2012

[15] Y. Jian, H. D. Pfister, K. R. Narayanan, *Approaching capacity at high rates with iterative hard-decision decoding*, Int. Symp. Inf. Theory (ISIT), 2012.

[16] A. Karbasi, A. H. Salavati, A. Shokrollahi, *Iterative Learning and Denoising in Neural Associative Memories*, To appear in ICML 2013.

[17] D. S. Modha, R. Ananthanarayanan, S. K. Esser, A. Ndirango, A. J. Sherbondy, R. Singh "Cognitive computing," Communications of the ACM, Vol. 54, No. 8, 2011, pp. 62-71.

[18] P. Dayan, L. F. Abbott, *Theoretical neuroscience: computational and mathematical modeling of neural systems*, MIT Press, 2004.

[19] A. Karbasi, A. H. Salavati, A. Shokrollahi, *Coupled Neural Associative Memories*, Technical Report, arXiv, 2013, http://arxiv.org/abs/1301.1555