Progress Report
20-30 September 2011

Amir Hesam Salavati
E-mail: raj.kumar@epfl.ch, hesam.salavati@epfl.ch

Supervisor: Prof. Amin Shokrollahi
E-mail: amin.shokrollahi@epfl.ch

Algorithmics Laboratory (ALGO)
Ecole Polytechnique Federale de Lausanne (EPFL)

October 22, 2011

1 Summary

In the past 10 days and after my vacation, I got back to reading some new papers to find a way to implement the learning phase in our neural associative memory with exponential capacity. These papers include a paper about learning the basis vector for the null space of a set of patterns as well as one on compressive sensing plus two more on various topics related to associative memory. I also worked on implementing one of the ideas to implement the learning phase which was partially successful. I also defined two semester projects on our two neural projects, one of which was taken by a master student.

In this report, you will find the summary of the papers I read as well as some details on the idea that I have been working on.

2 The subspace learning problem

In [1], the author addresses the problem of implementing neural networks capbale of performing Principle Component Analysis (PCA) as well as minor component analysis. Four different learning rules are considered for PCA and it was proven analytically that these approaches converge to the desired principle components. The considered rules are very similar and only differ in small details, although they some times have considerable effects on the output and complexity of implementation. The four learning rules are discussed in some more details below. More importantly for our work is equation (6) which specified the learning rule for identifying minor components.

Furthermore, the authors have also considered a neural network which is able identify minor components, i.e. those along which data does vary much. These components are very similar to the basis vectors of the dual space. It was also analytically proven that the suggested approach converges to the minor components.

The general PCA problem is that we have M patterns $x^1, \dots, x^M$, which are n dimensional vectors, and would like to find the first p largest eigenvectors of the matrix $C = \sum_{\mu=1}^{M} (x^\mu)^T x^\mu$. Let $w^1, \dots, w^p$ denote these eigenvectors and W be the $n \times p$ matrix whose columns are composed by w^i vectors.

2.1 Stochastic Gradient Ascent approach

In the Stochastic Gradient Ascent (SGA) method, we have a network of one input layer, one output layer (the state of which is denoted by vector y) and p parallel layers, one for each component. See figure 1 for more details.

The *learning rule* for the SGA model is given by equation (1).

$$\Delta w^j = \gamma^\mu y_j^\mu \left[x^\mu - y_j^\mu w^j - 2 \sum_{i<j} y_i^\mu w^i \right] \tag{1}$$

where

$$y_j^\mu = (w^j)^T x^\mu \tag{2}$$

In eqution (1), γ^μ is the learning gain for pattern μ. Furthermore, the first term, namely, $y_j^\mu x^\mu$, is the Hebbian term which measures the projection of x^μ onto the current weight vector (component). The rest of the terms are implicit orthonormality constraints, i.e. they make the components orthonormal.

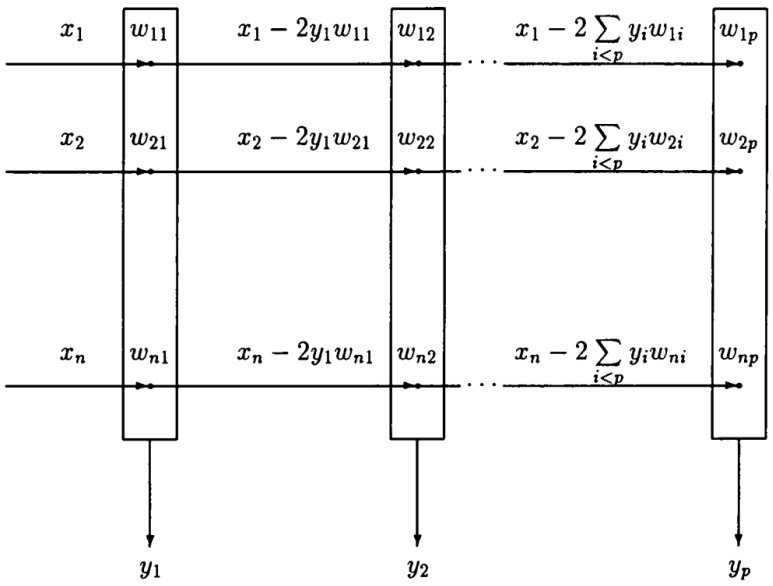

Figure 1: The Stochastic Gradient Ascent (SGA) network [1].

2.2 Subspace Network learning algorithm

The Subspace Network algorithm is very similar to SGA but with different orthonormalization method. The *learning rule* for the Subspace Network model is given by equation (3).

$$\Delta w^j = \gamma^\mu y_j^\mu \left[x^\mu - \sum_{i=1}^{p} y_i^\mu w^i \right] \tag{3}$$

where y_j^μ is given by equation (2). It has been shown that the final results of the above learning rule will not converge to the actual eigenvectors but a rotated basis in the sub-space spanned by them.

2.3 Generalized Hebbian Algorithm

The Generalized Hebbian Algorithm (GHA) was proposed by Sanger [10]. The *learning rule* for the GHA algorithm is extremely similar to that of Subspace Networks except for a small change in the summation index. The rule is given by equation (4).

$$\Delta w^j = \gamma^\mu y_j^\mu \left[x^\mu - \sum_{i \leq j} y_i^\mu w^i \right] \tag{4}$$

The advantage of rule GHA learning rule compared to that of subspace network is that the resulting weight vectors tend to the correct eigenvectors and not a rotated basis of the sub-space spanned by them.

2.4 Weighted Subspace Algorithm

The weighted subspace algorithm is also similar to the above (specially to the subspace network approach), except fot the addition of p scalars $\theta_1, \ldots, \theta_p$, as shown in the *learning rule* given by equation (5) [11], [12].

$$\Delta w^j = \gamma^\mu y_j^\mu \left[x^\mu - \theta_j \sum_{i=1}^{p} y_i^\mu w^i \right] \tag{5}$$

If the scalar are chosen such that $0 < \theta_1 < \ldots < \theta_p$, then because of *broken symmetry*, the resulting weights will tend to true eigenvectors.

2.5 Learning minor components

Minor components are the eigenvectors corresponding to the smallest eigenvalues of the correlation matrix. The proposed *learning rule* for the minor components is given by equation (6).

$$\Delta w^j = \gamma^\mu \left[-y_j^\mu x^\mu + \left((y_j^\mu)^2 + 1 - (w^j)(w^j) \right) w^j + \alpha y_j^\mu \sum_{i>j} y_i^\mu w^i \right] \tag{6}$$

where α is a parameter which must be chosen suitably.

- The first term in equation (6) is the anti-Hebbian learning.
- The second term is the *forgetting term* times w^j.
- One disadvantage of the suggested learning rule is that the update rule for the weight vecotor w^i depends on other weight vectors as well.
  - We might be able to overcome this issue by sequentially updating the weights using a similar scheme to that of our bipartite neural constraint enforcing graph.
- The third term takes care of the orthonormality.

This approach will yield the p eigenvectors of the correlation matrix which correspond to the p smallest eigenvalues if:

- Assuming the eigenvalues be ordred as $\lambda_1 > \ldots > \lambda_n$ and $\lambda_p < 1$, then $\alpha > \lambda_p/\lambda_n - 1$.
- All p smallest eigenvalues must be positive and less than 1.

Then the algorithm converges to either the minor components or the negative of them. Note that if all eigenvalues are less than 1 then the suggested approach will yield all the eigenvectors. Best convergence is obtained when α is large.

3 Compressive sensing an neural associative memories

In [2] the authors address the problem of compressive sensing, i.e. the case where we have to reconstruct a sparse vector x from its *noisy* linear measurements. An efficient approach to solve this problem was proposed by the authors which has a rather low complexity while maintaining good recovery performances. The authors use game theoretic arguments, which resembles those of

message passing-based algorithms, to perform ℓ_1 minimizations. The approach is very similar to our algorithm in [3] to perform neural error correction with the difference that theu authors repeat this method several times with different parameters to get the best and most sparse approximation. The authors obtain worst-case computational complexity but using simulations they show that the emprical complexity is much better than the theoretical one, which is still efficient.

More specifically, the problem of interest is the following: given the connectivity matrix $\Phi_{M\times N}$ ($M < N$) of an *expander* graph and an M dimensional vector $y = \Phi x + \mu$, we would like to find a vector x^* with at most k non-zero entries such that:

$$x^* = argmin_{\|x\|_0 \leq k} \|y - \Phi x\|_1 \tag{7}$$

This problem is called the *expander-based compressive sensing* (e-CS) problem.

The authors formulate the e-CS problem in the form of a min-max problem. This problem is then solved *approximately* using a game-theoretic approach since the exact problem is intractable. The suggested method is repeated several times with different parameters to obtain the best answer.

Although not mentioned specifically, their approach features a message passing algorithm in its core which is extremely similar to what we do in [3] in order to correct errors using a neural network. The main difference in the GAME algorithm and that of ours in [3] is the use of soft threshold function (8), instead of a hard one which we have been using, and the addition of update step η in the algorithm shown in figure 2. Furthermore, the GAME algorithm does not need the connectivity matrix to be expander (The e-GAME algorithm does require that on the other hand).

$$\mathcal{S}(\alpha, \theta) = \begin{cases} \theta & \text{if } \alpha \geq \theta; \\ -\theta & \text{if } \alpha < -\theta \\ \alpha & \text{otherwise.} \end{cases} \tag{8}$$

Algorithm 1 The GAME algorithm

Inputs: $\boldsymbol{y}$, Φ, and parameters T, τ and $\eta > 0$.
Output: A T-sparse approximation $\hat{\boldsymbol{x}}$ to $\boldsymbol{x}^*$.

1: Set $\mathbf{P}^1 = [\mathbf{0}]_{1\times 2M}$.
2: **for** $t = 1, \cdots, T$ **do**
3: Calculate $\boldsymbol{\xi}^t \doteq -\Phi^\top \mathbf{P}^t$.
4: Find the index $i^t \doteq \arg\max_{i\in[M]} |\xi_i^t|$.
5: Set $\boldsymbol{x}^t = \tau \cdot \text{sign}(\xi_{i^t}^t) \cdot \boldsymbol{e}_{i^t}$. (Note that $\boldsymbol{x}^t \in \Delta(\tau, 1)$.)
6: Update $\mathbf{Q}^{t+1} = \mathbf{P}^t + \eta(\Phi \boldsymbol{x}^t - \boldsymbol{y})$.
7: For each $j \in [M]$, let $\mathrm{P}_j^{t+1} = \mathcal{S}\left(\mathrm{Q}_j^{t+1}, 1\right)$.
8: **end for**
9: Output $\hat{\boldsymbol{x}} \doteq \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{x}^t$.

Figure 2: The GAME algorithm to solve the min-max expander-based compressive sensing problem [2].

e-GAME is the extended version of GAME which uses expander graphs to obtain good compressed sensing estimations in a short time. In a nutshell and in terms of our approach in [3], the e-GAME algorithm works as follows: let the neural network run for a pre-determined number of iterations. If an answer was found that satisfies the constraints, repeat the whole process with a smaller permitted degree of sparsity. Otherwise, increase the number of steps and *allowed* degree

of sparsity. Repeat this process as many times as required according the desired accuracy. Theorem V.1 guarantees the convergence to the correct result based on the required accuracy. It is shown that the e-GAME algorithm is able to find a k-sparse solution with ℓ_1 norm of the error proportional to $\|\mu\|_1$. So in case of no noise, the solution is exact [2].

4 Error correcting codes and neural networks

I read the ideas mentioned in [6]. The paper uses combinatorial approaches to check if neurons use error correcting codes. In the end, the authors find evidence in favor of the existence of such codes, which they call receptive field codes. The idea is that each neurons receptive field have some common intersections together which results in redundancy that can be used for error correction. More interestingly, the authors show that although such receptive field codes are terrible in terms of block error rate, they will result in a pattern which is close to the correct one in some metric space. This behavior is acceptable in neural networks as most of the time 100% accuracy is not required. We might use the same idea for our work [3] as well.

Another paper that I read on this topic was [7] by Gripon and Berrou. In this paper the authors address the problem of increasing the storage capacity of neural associative memories. They use graph cliques as a means of memorizing binary patterns and propose a simple scheme based on winner-take-all methods to recover the correct pattern in presence of erasures[1]. More specifically, each pattern is broken into several sub-patterns and, in a similar manner, the network is divided into several clusters. The bits in each sub-pattern then correspond to a cluster in the network. Furthermore, inside each cluster and for a given pattern only one neuron fires. In other words, in each sub-pattern only one bit is equal to one at a time and all the rest are equal zero. As a result, patterns are stored in connections among clusters, i.e. graph cliques. For each pattern, all the firing neurons are connected to each other in the learning phase. In the recall phase and in case of an error, the feedback from the neurons in other clusters help the erroneous neurons correct themeselves.

Mathematical analysis and the simulations performed by the authors show that the storage capacity of the proposed solution increases by a couple of times compared to Hopfield networks while maintaining similar pattern filling capabilities in case on an erasure. Unfortunately, other types of errors are not considered in this paper. The technical parts of the paper are very similar to the authors journal paper [8].

5 Futureworks

Regarding e-GAME and varying sparsity sizes, We might be able to do the same with our current approach in [3] in order to increase the accuracy. The ideas mentioned in [6] might also turns out helpful specially in cases where 100% accuracy is not required.

References

[1] E. Oja, *Principal components, minor components, and linear neural networks*, Neural Net., Vol. 5, 1992, pp. 927-935.

[1] Neural cliques seem to be biologically relevant as well since in [9], the authors relate neural cliues to memory.

[2] S. Jafarpour, V. Cevher, R. E. Schapire, *A game theoretic approach to expander-based compressive sensing*, Proc. IEEE Int. Symp. Inf. Theory (ISIT), 2011.

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

[4] B. A. Olshausen, D. J. Field, *Emergence of simple-cell receptive field properties by learning a sparse code for natural images*, Nature Vol. 381, 1996, pp. 607-609.

[5] E. Oja, *Principal components, minor components, and linear neural networks*, Neur. Net., Vol. 5, No. 6, 1992, pp. 927-935.

[6] J. L. Walker, C. Curto, V. Itskov, K. Morrison, Z. Roth,*A coding theory perspective on combinatorial neural code*, Algebraic Coding Theory Workshop, Sep. 2011, EPFL.

[7] V. Gripon, C. Berrou, *A simple and efcient way to store many messages using neural cliques*, IEEE Symp. Computational Intelligence, Cognitive Algorithms, Mind, and Brain (CCMB), 2011.

[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] L. Lin, R. Osan, J. Z. Tsien, *Organizing principles of real-time memory encoding: neural clique assemblies and universal neural codes*, Trends Neurosci., Vol. 29, No. 1, 2006, pp. 4857.

[10] T. D. Sanger, *Optimal unsupervised learning in a single-layer linear feedforward neural network*, Neural Net., Vol. 2, 1989, pp. 450-473.

[11] E. Oja, H. Ogawa, J. Wangviwattana, *Principal component analysis by homogeneous neural networks, part I: the weighted subspace criterion*, IEICE Tran. Inf. Sys., Vol. E75-D, No.3, 1992, pp.366-375.

[12] E. Oja, H. Ogawa, J. Wangviwattana, *Principal component analysis by homogeneous neural networks, part II: analysis and extensions of the learning algorithms*, IEICE Tran. Inf. Sys., Vol. E75-D, No. 3,pp. 366-375.