Progress Report
November 13 to December 13, 2010

Raj K. Kumar, 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)

December 13, 2010

1 Introduction

In this report we will explain our activities in the past three weeks. We start by summarizing our discussions with a number of people at EPFL in order to list different ideas suggested by experts in the field of neuroscience or coding theory. We then proceed with discussing the results on applications of Gold sequences in increasing the storage capacity of Hopfield networks. We show that Gold sequences are fantastic in terms of the stability condition and one can memorize much more Gold patterns than purely random ones, *in absence of noise*. However, Gold sequences indicate a bad performance in presence of noise and that comes from their particular correlation properties, as well as inherent structure of Hopfield networks.

We then show that Hopfield networks are unable to learn particular functions, including XOR function. This is not surprising as we have seen Hopfield networks can not memorize codewords of a linear codeword. As a result, we are currently considering a more powerful structure, called Boltzmann machines, which we will briefly introduce in the end of this report. More details will be provided in the following reports.

2 Meetings and Presentations

In the past weeks, we had a number of discussions with various people at EPFL as well as the KTN-student talk. Here is a brief summary of our discussions and ideas suggested by other people.

2.1 Meeting with Mr. Ali Ajdari

As suggested by Masoud Alipour, we had a discussion with Mr. Ali Ajdari, a 4^{th} year PhD student at Laboratory of Nonlinear Systems (LANOS). He works on information processing in complex networks under supervision of Prof. Martin Hasler. He suggested a number of approaches for our problem, applications of coding theory in neural networks. We talked about echostate networks, which constitute of a set of input/output layers plus a hidden *random* layer in between. It is shown that the network can learn the relationship between input and output patterns even if the weights in the hidden layer are picked randomly. He confirmed that echostate networks and kernel techniques may help us with our idea as kernel techniques are used to map a set

of inputs to a higher dimension. Also, he mentioned that echostate networks, in the format he has used to work with, are more suitable for time series, i.e. when you have time-varying input. So he told me that for our case, we either have to come up with a time-series version of the messages (to be mapped to a higher dimension) or consider other options. Nevertheless, in a later meeting he told us that there is a recent paper that solves the issue of time series and we will look into this paper in future.

2.2 KTN Student Talks

On the 2^{nd} of December 2010, we gave a talk on applications of coding theory in neural systems in Know Thy Neighbor (KTN) student talks. As a result, we received a number of suggestion from the audience as topics to consider in our future research. In particular, Prof. Rudiger Urbanke suggested pursuing ideas in code on real field (lattice codes specially). This is one of the topics that we are going to consider in future since it makes more biological sense to consider coding schemes over real fields.

3 Gold Sequences and Hopfield Networks

In this section, we will explain our results on applications of Gold sequences to increase the storage capacity of Hopfield networks. We start by reviewing how Gold sequences are produced.

3.1 Gold Sequences

Let m be an *odd* integer and l be another integer such that l and m are relatively prime. Let $d = 2^l + 1$. Furthermore, assume α to be a primitive element of $F(2^m)$. The family of Gold sequences are generated as follows:

$$g_i^a = (-1)^{T(d\alpha^i)+T(\alpha^{id})} \tag{1}$$

In which $a \in F(2^m)$, g_i^a is the i^{th} bit of the Gold pattern $\underline{g}^a$ and $T(.)$ is the finite field trace function from $F(2^m)$ to $F(2)$. The length of each pattern is equal to $n = 2^m - 1$, i.e. i varies from 1 to n. For each $a \in F(2^m)$ we will get a binary sequence with length N, which gives us 2^m of such sequences. By considering their cyclic shifts, we can have $2^{2m} - 1$ of Gold sequences in total [1].

3.2 Stability of Gold Sequences as Memorized Patterns in Hopfield Networks

Suppose we select the first $M = 2^m$ patterns of the Gold family as the set of patterns which we would like to memorize using a Hopfield network. Here, we assume that the traditional Hopfield weighting scheme is used, as shown in equation (2), in which where x_i^μ is the i^{th} bit of the μ^{th} Gold pattern

$$w_{ij} = \frac{1}{n} \sum_{\mu=1}^{M} x_i^\mu x_j^\mu \tag{2}$$

Furthermore, we assume that neurons use equation (3) to update their states, s_i.

$$s_i = \begin{cases} 1, & \sum_{j=1}^{n} w_{ij} s_j > \theta \\ -1, & \text{Otherwise} \end{cases} \tag{3}$$

In words, neuron i decides to fire or remain silent based on the weighted sum over its input links, which we denote by h_i. Suppose we have fixed the weights according to equation (2) and would like to recall pattern p in absence of noise. In other words, we are interested in assessing the stability of the network for all patterns $p = 1, \ldots, M$. The ultimate goal would be to see how much we can increase M and still maintaining the stability condition. If we expand h_i, and assuming the firing threshold θ be equal to zero, we will get:

$$\begin{aligned} h_i &= \sum_{j=1}^{n} w_{ij} x_j^p = \frac{1}{n} \sum_{j=1}^{n} \sum_{\mu=1}^{M} x_i^m x_j^m x_j^p \\ &= \frac{1}{n} \sum_{\mu=1}^{M} x_i^m \sum_{j=1}^{n} x_j^m x_j^p = x_i^p + \frac{1}{n} \sum_{\mu=1, \neq p}^{M} x_i^m < \underline{x}^m, \underline{x}^p > \end{aligned} \tag{4}$$

In which $< \underline{x}^m, \underline{x}^p >$ is the inner product of patterns p and m. In the above equation, the first term, i.e. x_i^p, is the desired one because we would like the sign of h_i to be equal to the sign of x_i^p if our goal is to recall pattern p. The second term is the interference term and it depends on the correlation between pattern p and all the other patterns. Ideally, we would like the interference term be as small as possible such that it does not affect the sign of x_i^p.

Now let's replace the formulation of Gold sequences in computing the interference part. Without loss of generality, we assume that the pattern p, which we would like to recall at the moment, corresponds to the Gold pattern with $a = 0$ (see equation (1)). We will have:

$$\begin{aligned} I_i^p &= \frac{1}{n}\sum_{\mu=1,\neq p}^{M}(-1)^{T(a_\mu\alpha^i)+T(\alpha^{id})}\sum_{j=1}^{n}(-1)^{T(\alpha^{jd})}(-1)^{T(a_\mu\alpha^j)+T(\alpha^{jd})} \\ &= \frac{1}{n}\sum_{a=1}^{2^m}(-1)^{T(a\alpha^i)+T(\alpha^{id})}\sum_{j=1}^{n}(-1)^{T(a\alpha^j)} = \frac{(-1)^{T(\alpha^{id})}}{n}\sum_{j=1}^{n}\sum_{a=1}^{2^m}(-1)^{T(a\alpha^i)+T(a\alpha^j)} \\ &= \frac{M-1}{n}(-1)^{T(\alpha^{id})} + \frac{(-1)^{T(\alpha^{id})}}{n}\sum_{j=1,\neq i}^{n}\sum_{a=1}^{2^m}(-1)^{T(a\alpha^i+a\alpha^j)} \\ &= \frac{M-1}{n}(-1)^{T(\alpha^{id})} - \frac{n-1}{n}(-1)^{T(\alpha^{id})} = \frac{M-n}{n}(-1)^{T(\alpha^{id})} \\ &= \frac{M-n}{n}x_i^p \end{aligned} \tag{5}$$

Because the trace function is linear. Furthermore, since α is primitive element of $F(2^m)$, $\alpha^i + \alpha^j$ for $i \neq j$ is equal to some α^k with $k \neq 0$. Hence, the summation $\sum_{a=1}^{2^m}(-1)^{T(a\alpha^i+a\alpha^j)} = -1$ as it runs over all the finite field $F(2^m)$ except for $a = 0$. Since we have selected $M = 26m = n + 1$, it is obvious that $I_i^p = \frac{1}{n}x_i^p$ and would not affect the sign of x_i^p.

Combining equation (5) and (4), we will see that (for $M = n + 1$):

$$h_i = \frac{M}{n}x_i^p \tag{6}$$

In fact, following a similar approach, one can show that a similar result holds if we choose M carefully. In particular, selecting M to be multiple integers of $n + 1$. In the next section we will show this fact for several values of M using numerical simulations. We use a particularly intuitive graphical tool to analyze the behavior of Gold sequences and compare them to that of purely random case.

3.2.1 Numerical Results for Larger Number of Memorized Patterns

In this section, we will demonstrate some numerical results on the behavior of the interference term when we increase the number of memorized patterns,

M, beyond the size of network, n. To do so, we calculate $-x_i^p I_i^p$ because to have stability we would like to see $sign(h_i x_i^p) = sign(1 + x_i^p I_i^p) = 1$. this requires $-x_i^p I_i^p < 1$ *for all patterns* p *and all bits* i.

Therefore, we calculate $-x_i^p I_i^p$ for all bits of a pattern p. This provides us with a *pdf* of interference terms: **for each pattern** p, on the horizontal axis we will have the amount of $-x_i^p I_i^p$, and on the vertical axis the number of bits with that amount of interference (see figures below). We would like to see that such a graph never overpasses the critical point 1 on the horizontal axis, *for all patterns* p.

In the following figures, we have computed the interference *pdf* for the case of $n = 127$ and three different values of M: $M = 128$, $M = 200$ and $M = 500$. We have plotted the pdf for both Gold sequences and random sequences. In each case, the average pdf and the *worst* pdf is shown, where the worst pdf is defined to be the one which is closest to the critical point 1 on the horizontal axis or overpasses it.

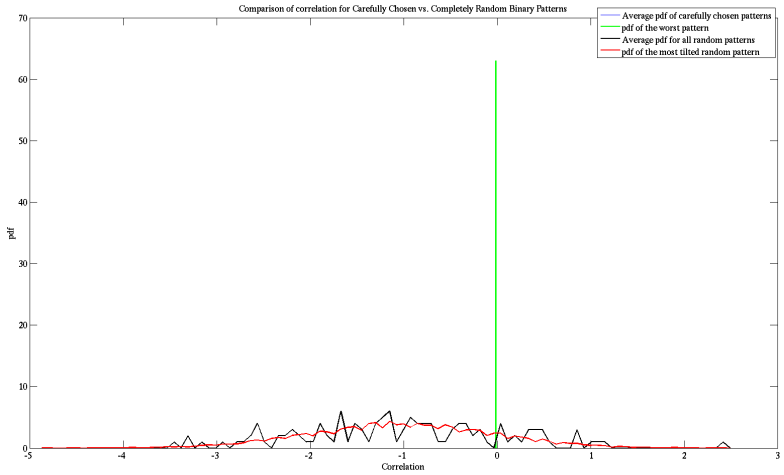

Figure 1: Interference pdf for $n = 127$ and $M = 128$

Based on the figures we can make the following conclusions:

1. For $M = 128$, Gold sequences are stable and do not result in any stability errors. This is not the case for $M = 200$ as for some of the

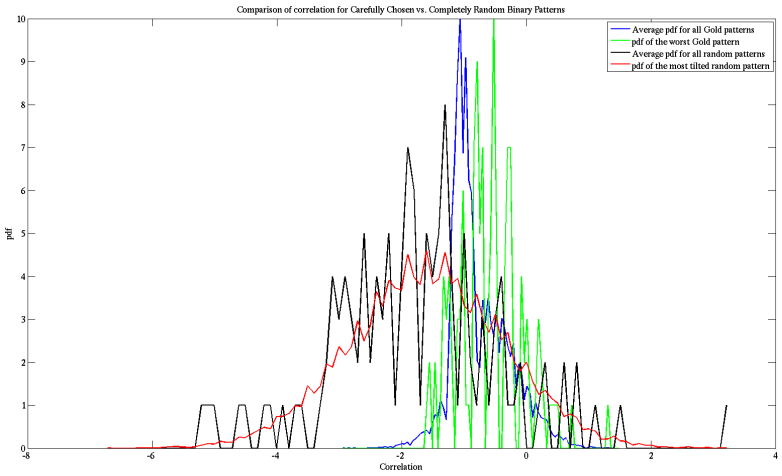

Figure 2: Interference pdf for $n = 127$ and $M = 200$

patterns, the pdf passes the point 1. However, if we keep increasing M to 500 we see that the set of Gold sequences become stable again. Thus, one can memorize much more patterns using Gold sequences with the guarantee that they are at least stable.

2. Random patterns are not even stable if $M > n$, i.e. even in absence of noise, they result in recovery errors.

3.3 Performance of Gold Sequences in Presence of Noise

In the previous section, we showed that Gold sequences are much better than random patterns in terms of stability. In this section we investigate their performance in presence of noise and show that Gold sequences do not demonstrate good performance in presence of noise. To prove that, we consider the simplest case that only one bit at position k is flipped and we would like to recall pattern p which corresponds to the Gold sequence with $a = 0$ (see equation (1)). In other words, our initial pattern is $\underline{s} = \underline{x}^p + \underline{e}$, where $\underline{e}$ is the error vector and is equal to zero in all of its elements except

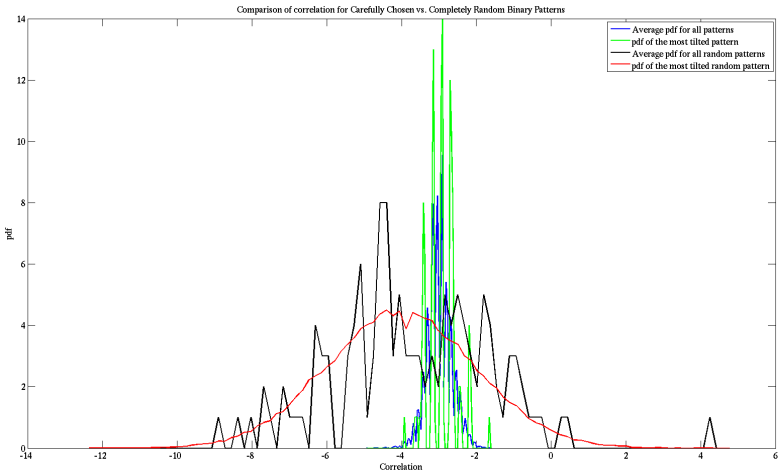

Figure 3: Interference pdf for $n = 127$ and $M = 500$

for the k^{th} element e_k which is equal to $e_k = -2x_k^p$. Replacing this pattern in equation (4) and using equation (5) we obtain:

$$\begin{aligned} h_i &= \sum_{j=1}^{n} w_{ij} s_j = \sum_{j=1}^{n} w_{ij} x_j^p + \sum_{j=1}^{n} w_{ij} e_j \\ &= \frac{M}{n} x_i^p - \frac{2}{n} \sum_{\mu=1}^{M} x_i^\mu x_k^\mu x_k^p \end{aligned} \tag{7}$$

The term $\sum_{\mu=1}^{M} x_i^\mu x_k^\mu x_k^p$ reduces to $M x_i^p$ if $k = i$. Otherwise:

$$\begin{aligned} x_k^p \sum_{\mu=1}^{M} x_i^\mu x_k^\mu &= (-1)^{T(\alpha^{kd})} \sum_{a=0}^{2^m} (-1)^{T(a\alpha^i)+T(\alpha^{id})} (-1)^{T(a\alpha^k)+T(\alpha^{kd})} \\ &= (-1)^{T(\alpha^{id})} \sum_{a=0}^{2^m} (-1)^{T(a\alpha^i)} (-1)^{T(a\alpha^k)} \\ &= (-1)^{T(\alpha^{id})} \sum_{a=0}^{2^m} (-1)^{T(a\alpha^i + a\alpha^k)} = 0 \end{aligned} \tag{8}$$

Because $k \neq i$ and the summation $\sum_{a=0}^{2^m}(-1)^{T(a\alpha^i+a\alpha^k)}$ runs over all the finite field. Putting equations (7) and (8) we obtain:

$$h_i = \begin{cases} -\frac{M}{n}x_i^p, & \text{if } i = k \\ \frac{M}{n}x_i^p, & \text{Otherwise} \end{cases} \tag{9}$$

Which means that the sign of h_i is different from x_i^p when the i^{th} bit has been erroneous from the beginning. In other words, all flipped bits remain erroneous and all correct bits remain correct forever and the above scheme with Gold sequences is unable to correct any errors.

4 Future Works

In this report, we saw that although Gold sequences are quite good in terms of stability, their error performance is very poor. There are a number of possible reasons for this phenomenon: the traditional Hopfield weighting scheme, the special correlation properties of Gold sequences and network structure itself. However, one can show that even if we use patterns with zero interference, a single bit flip can not be corrected if $M > n/2$. Therefore, we are concentrating on other network structures and weighting scheme.

One particular structure of interest is a Boltzmann machine. Boltzmann machines are very similar to Hopfield networks in structure but contain additional hidden layers, i.e. internal layers without having access to input/output data directly. The reason they are so interesting from a coding theoretical point of view is that one can show that Hopfield networks are not able to learn particular functions, including *XOR*. However, by adding at most two additional hidden layers to the network, i.e. considering a Boltzmann machine, one can learn *any* function. Therefore, we are currently studying Boltzmann machines. The weighting scheme as a result would be different as well which might be helpful.

Considering lattice codes as well as other ideas mentioned by Mr. Ajdari would be other topics we will look into in future.

References

[1] R. Gold, "Optimal binary sequences for spread spectrum multiplexing", IEEE Transactions on Information Theory, Vol. 13, No. 4, 1967, pp. 619 −621.