Neural Memories Based on Faulty Elements

I. The Model

The network is modled by a weighted bipartite graph. We have n pattern nodes as well as m constraint nodes, with $m < n$. Pattern and constraint neurons are denoted by $b_1, \ldots, b_n$ and $c_1, \ldots, c_m$, respectively. The connectivity matrix W is selected such that it is orthogonal to patterns in a dataset, i.e.

$$W.x = 0, \forall x \in \mathcal{X},$$

where $\mathcal{X}_{\mathcal{C}\times n}$ is the database of $\mathcal{C}$ patterns of length n. Furthermore, the patterns are assumed to have non-negative integers as their elements.

The goal of this network is to correct one input error with faulty network elements. The decoding is perfromed by iteratively passing messages over the links. However, in contrast to the message passing in decoding LDPC codes, the messages accross outgoing links of a pattern (constraint) node are the same, i.e. whatever a node sends, goes to all its neighbors.

With some abuse of notations, let us denote the messages transmitted by pattern node i and constraint node j at round t with $x_i(t)$ and $y_j(t)$, respectively. In round 0, we assume the pattern nodes are initialized with a noise version of one of the patterns in the dataset, i.e.

$$x_i(0) = x + z,$$

where z is the noise vector. In this work, we only focus on ± 1 noises and since at this point the goal is to correct only one input error, we assume to have $\|z\|_0 = \|z\|_1 = 1$. Without loss of generality, throughout this paper we assume $z_1 = +1$ and $z_i = 0$ for $1 < i \leq n$.

In round t, the pattern and constraint neurons update their state according to the feedback they receive from their neighbors. However, since they are not fault-free themselves, some random noise parameters affect their decision. More specifically, the decision making criteria for pattern node i is

$$x_i(t+1) = \begin{cases} x_i(t) + \text{sign}(g_i(t)), & \text{if } |g_i(t)| \geq \gamma \\ x_i(t), & \text{o.w.,} \end{cases} \tag{1}$$

where γ is the update threshold and g_i is given by

$$g_i(t) = \frac{\text{sign}(W_i^c) \cdot y(t)}{d_i} + u_i. \tag{2}$$

Here, d_i is the degree of pattern node i, W_i^c is the i^{th} column of the matrix W, $y(t) = [y_1(t), \ldots, y_m(t)]$ is the vector of messages transmitted by the constraint neurons and u_i is the random noise parameter affecting pattern node i.

In this work, we consider a bounded noise model, i.e. we assume that u_i is uniformly distributed in the interval $[-\epsilon, \epsilon]$, with $\epsilon < 1$.

On the constraint side, the update rule is

$$y_i(t) = \begin{cases} +1, & \text{if } h_i(t) \geq \theta \\ 0, & \text{if } -\theta \leq h_i(t) \leq \theta \\ -1, & \text{o.w.,} \end{cases} \tag{3}$$

where θ is the update threshold and h_i is defined as

$$h_i(t) = W_i^r \cdot x(t) + v_i, \tag{4}$$

in which W_i^r is the i^{th} colurowmn of the matrix W, $x(t) = [x_1(t), \ldots, x_n(t)]$ is the vector of messages transmitted by the pattern neurons and v_i is the random noise parameter affecting pattern node i. As before, we consider a

Algorithm 1 Intra-Module Error Correction

Input: Training set $\mathcal{X}$, threshold φ, iteration $t_{\max}$
Output: $x_1^{(\ell)}, x_2^{(\ell)}, \ldots, x_{n_\ell}^{(\ell)}$

1: **for** $t = 1 \rightarrow t_{\max}$ **do**
2: *Forward iteration:* Calculate the weighted input sum $h_i^{(\ell)} = \sum_{j=1}^{n_\ell} W_{ij}^{(\ell)} x_j^{(\ell)}$, for each neuron $y_i^{(\ell)}$ and set $y_i^{(\ell)} = \text{sign}(h_i^{(\ell)})$.
3: *Backward iteration:* Each neuron $x_j^{(\ell)}$ computes

$$g_j^{(\ell)} = \frac{\sum_{i=1}^{m_\ell} \text{sign}(W_{ij}^{(\ell)}) y_i^{(\ell)}}{\sum_{i=1}^{m_\ell} \text{sign}(|W_{ij}^{(\ell)}|}).$$

4: Update the state of each pattern neuron j according to

$$x_j^{(\ell)} = x_j^{(\ell)} + \text{sign}(g_j^{(\ell)})$$

only if $|g_j^{(\ell)}| > \varphi$.
5: $t \leftarrow t + 1$
6: **end for**

bounded noise model and assume v_i's to be independently and identically distributed in the interval $[-\delta, \delta]$ for some $\delta < 1$.

The whole error correcting procedure is formally given by Algorithm 1, which is replicated from our work in [1]. There, $y_i^{(\ell)}$ and $x_j^{(\ell)}$ represent the messages transmiteed by pattern neuron j and cluster neuron i in cluster ℓ.

We would like to obtain $P_c^{(1)}$, the probability of correcting this single error. To this end, we will need the following definitions as well:

- $\Pi^{(0)}$: The probability that a constraint neuron makes a wrong decision due to its internal noise when there are no external noise introduced to the network, i.e. $\|z\|_0 = 0$.
- $P_e^{(0)}$: The probability that a pattern neuron makes a wrong decision due to its internal noise when there are no external noise introduced to the network, i.e. $\|z\|_0 = 0$.
- $\Pi^{(1)}$: The probability that a constraint neuron makes a wrong decision due to its internal noise when we have one input error (external noise), i.e. $\|z\|_0 = \|z\|_1 = 1$.
- $P_e^{(1)}$: The probability that a pattern neuron makes a wrong decision due to its internal noise when we have one input error (external noise), i.e. $\|z\|_0 = \|z\|_1 = 1$.

So we have $P_c^{(1)} = 1 - P_e^{(1)}$.

In the next section, we will calculate the probability of making wrong decisions in the absence and presence of external noise. We then select appropriate update threshold parameters, θ and γ, to minimze the probability of making mistakes. Finally, we make and attempt to obtain anlytical upper bounds on this probability and compare it with the exact values, derived from numerical analysis. Later, we will compare these values with those obtained from simulations.

II. Deriving Probability of Making Wrong Decisions

A. Calculating $\Pi^{(0)}$

At first, let us calculate the probability that a constraint node makes a mistake when there are no external noise introduced in the network. To this end, consider constraint node i whose decision parameter will be

$$h_i = W_i^r \cdot x(0) + v_i = v_i$$

Therefore, the probability of making a mistake will be

$$\begin{aligned} \Pi^{(0)} &= \Pr\{|v_i| > \theta\} \\ &= \max\left(0, \frac{\delta - \theta}{\delta}\right). \end{aligned} \tag{5}$$

Thus, to make $\Pi^{(0)} = 0$ we will select $\theta > \delta$.[1] So from now on, we assume

$$\Pi^{(0)} = 0 \tag{6}$$

B. Calculating $P_e^{(0)}$

Knowing that the constraint will not send any non-zero messages in absence of external noise, we focus on the pattern neurons in the same circumstance. A given pattern node b_j will receive a zero from all its neighbors among the constraint nodes. Therefore, its decision parameter will be $g_j = u_j$. As a result, a mistake could happen if $|u_j| \geq \gamma$. The probability of this event is given by

$$\begin{aligned} P_e^{(0)} &= \Pr\{|u_i| > \gamma\} \\ &= \max\left(0, \frac{\epsilon - \gamma}{\gamma}\right). \end{aligned} \tag{7}$$

Therefore, to make $P_e^{(0)}$ go to zero, we must select $\gamma \geq \epsilon$. This choice, however, may adversely affect the probability of correcting one input error as we will see later. So we leave it as it is for the moment and in section III we will find a γ that minimizes the overall error probability.

C. Calculating $\Pi^{(1)}$

It's now time to consider the situation in which we have one external error. Without loss of generality, we assume it is the first pattern node, b_1, that is corrupted with noise whose value is $+1$. Now we would like to calculate the probability that a constraint node makes a mistake in such circumstances. Furthermore, we will only the constraint neuorns that are connected to b_1. Because for the other constraint neurons, the situation is the same as the previous cases where there were no external noise.

for a constraint neuron j that is connected to b_1, the decision parameter is

$$h_j = W_j^r.(x + z) + v_j = 0 + W_j^r.z + v_j = w_{j1} + v_j$$

We consider two error events:

1) A constraint node j makes a mistake and do not send a message at all. The probability of this event is denoted by $\Pi_1^{(1)}$.
2) A constraint node j makes a mistake and send a message with the opposite sign. The probability of this event is denoted by $\Pi_2^{(1)}$.

We first calculate the probability of $\Pi_2^{(1)}$. Without loss of generality, assume the $w_{j1} > 0$ so that the probability of an error of type two is as follows (the case for $w_{j1} < 0$ is exactly the same):

$$\begin{aligned} \Pi_2^{(1)} &= \Pr\{w_{ji} + v_j < -\theta\} \\ &= \max\left(0, \frac{\delta - (\theta + w_{j1})}{2\delta}\right). \end{aligned} \tag{8}$$

However, since $\theta > \delta$ and $w_{j1} > 0$, then $\delta - (\theta + w_{j1}) < 0$ and $\Pi_2^{(1)} = 0$. Therefore, the constraint neurons will never send a message that has an opposite sign to what it should have. All remains to do is to calculate the probability that they remain silent by mistake.

To this end, we will have

$$\begin{aligned} \Pi_1^{(1)} &= \Pr\{|w_{ji} + v_j| < \theta\} \\ &= \max\left(0, \frac{\delta + \min(\theta - w_{j1}, \delta)}{2\delta}\right). \end{aligned} \tag{9}$$

[1]Note that this might not be possible in all cases since, as we will see later, the minimum absolute value of network weights should be at least θ. Therefore, if θ is too large we might not be able to find a proper set of weights.

The above equation can be simplifed if we assume that the absolute value of all weights in the network is bigger than a constant $\eta > \theta$.Then, the above equation will simplify to

$$\Pi_1^{(1)} \leq \max\left(0, \frac{\delta - (\eta - \theta)}{2\delta}\right). \tag{10}$$

Putting the above equations together, we will obtain

$$\Pi^{(1)} \leq \max\left(0, \frac{\delta - (\eta - \theta)}{2\delta}\right). \tag{11}$$

In case $\eta - \theta > \delta$, we could even manage to make this probability equal to zero. However, we will leave it as it is and use equation (11) to calculate $P_e^{(1)}$.

D. Calculating $P_e^{(1)}$

We start by first calculating the probability that a non-corrupted pattern node b_j makes a mistake, which is to change its state in round 1. Let us denote this probability by $P_{e_1}^{(1)}$. Now to calculate $P_{e_1}^{(1)}$ assume b_j has degree d_j and it has a common neighbors with b_1, the corrupted pattern node.

Out of these a common neighbors, a_1 will send ± 1 messages and the others will, mistakenly, send nothing. Thus, the decision making parameter of pattern node j, g_j, will be bounded by

$$g_j(1) = \frac{\text{sign}(W_j^c) \cdot y(0)}{d_j} + u_j. \leq \frac{a_1}{d_j} + u_j.$$

We will denote $\text{sign}(W_j^c) \cdot y(0)$ by f_j for brevity from this point on.

In this circumstances, a mistake happens when $|g_j| \geq \gamma$. Thus

$$\begin{aligned} P_{e_1}^{(1)} &= \Pr\{|g_j| \geq \gamma | \deg(b_j) = d_j \& |\mathcal{N}(b_1) \cap \mathcal{N}(b_j)| = a\} \\ &= \Pr\{\frac{f_j}{d_j} + u_j \geq \gamma\} + \Pr\{\frac{f_j}{d_j} + u_j \leq -\gamma\}, \end{aligned} \tag{12}$$

where $\mathcal{N}(b_i)$ represents the neighborhood of pattern node b_i among constraint nodes.

By simplifying equation (12) we will get

$$P_{e_1}^{(1)}(f_j) = \begin{cases} +1, & \text{if } |f_j| \geq (\epsilon + \gamma) d_j \\ \max(0, \frac{\epsilon - \gamma}{\epsilon}), & \text{if } |f_j| \leq |\epsilon - \gamma| d_j \\ \frac{\epsilon - (\gamma - f_j/d_j)}{2\epsilon}, & \text{if } |f_j - \gamma d_j| \leq \epsilon d_j \\ \frac{\epsilon - (\gamma + f_j/d_j)}{2\epsilon}, & \text{if } |f_j + \gamma d_j| \leq \epsilon d_j. \end{cases} \tag{13}$$

We should now average the above equation over f_j, a_1, a and d_j. To start, suppose out of the a_1 non-zero messages the node b_j receives, e of them have the same sign as the link they are being transmitted over. Thus, we will have $f_j = e - (a_1 - e) = 2e - a_1$. Assuming the probability of having the same sign for each message is $1/2$, the probability of having e equal signs out of a_1 elements will be $\binom{a_1}{e}(1/2)^{a_1}$. Thus, we will get

$$\bar{P}_{e_1}^{(1)} = \sum_{e=0}^{a_1} \binom{a_1}{e} (1/2)^{a_1} P_{e_1}^{(1)}(2e - a_1). \tag{14}$$

Now note that the probability of having $a - a_1$ mistakes from the constraint side is given by $\binom{a}{a_1}(\Pi^{(1)})^{a-a_1}(1 - \Pi^{(1)})^{a_1}$. thus, and we some abuse of notations we will get

$$\bar{P}_{e_1}^{(1)} = \sum_{a_1=0}^{a} \binom{a}{a_1} (\Pi^{(1)})^{a-a_1}(1 - \Pi^{(1)})^{a_1} \sum_{e=0}^{a_1} \binom{a_1}{e} (1/2)^{a_1} P_{e_1}^{(1)}(2e - a_1). \tag{15}$$

Finally, the probability that b_j and b_1 have a common neighbors can be approximated by $\binom{d_j}{a}(1 - \bar{d}/m)^{d_j - a}(\bar{d}/m)^a$, where $\bar{d}$ is the average degree of pattern nodes. Thus, and again with some abuse of notation, we will obtain

$$\bar{P}_{e_1}^{(1)} = \sum_{a=0}^{d_j} \binom{d_j}{a} (1 - \bar{d}/m)^{d_j - a}(\bar{d}/m)^a \sum_{a_1=0}^{a} \binom{a}{a_1} (\Pi^{(1)})^{a-a_1}(1 - \Pi^{(1)})^{a_1} \sum_{e=0}^{a_1} \binom{a_1}{e} (1/2)^{a_1} P_{e_1}^{(1)}(2e - a_1), \tag{16}$$

where $P_{e_1}^{(1)}(2e-a_1)$ is given by equation (12). We will not simplify the above equation any further and use it as it is in our numerical analysis in order to obtain the best parameter γ.

Now we will turn our attention to the probability that the corrupted node, b_1, makes a mistake, which is either not to update at all or update its itself in the wrong direction. Recalling that we have assume the external noise term in b_1 to be a $+1$ noise, the wrong direction would be for node b_1 to increase its current vlaue instead of decreasing it. Furthermore, we assume that out of d_1 neighbors of b_1, some j of them have made a mistake and will not send any messages to b_1. Thus, the decision parameter of b_1, will be $g_1 = u + (d_1 - j)/d_1$. Denoting the probability of making a mistake at b_1 by $P_{e_2}^{(1)}$ we will get

$$\begin{aligned} P_{e_2}^{(1)} &= \text{Pr}\{g_1 \leq \gamma | \deg(b_1) = d_1 \& j \text{ errors on constraint side}\} \\ &= \text{Pr}\{\frac{d_1 - j}{d_1} + u < \gamma\}, \end{aligned} \tag{17}$$

which simplifies to

$$P_{e_2}^{(1)}(j) = \begin{cases} +1, & \text{if } |j| \geq (1+\epsilon-\gamma)d_1 \\ \max(0, \frac{\epsilon-\gamma}{\epsilon}), & \text{if } |j| \leq (1-\epsilon-\gamma)d_1 \\ \frac{\epsilon+\gamma-(d_1-j)/d_1}{2\epsilon}, & \text{if } |\gamma d_1 - (d_1 - j)| \leq \epsilon d_1. \end{cases} \tag{18}$$

Noting that the probability of making j mistakes on the constraint side is $\binom{d_1}{j}(\Pi^{(1)})^j(1-\Pi^{(1)})^{d_1-j}$, we will get

$$\bar{P}_{e_2}^{(1)} = \sum_{j=0}^{d_1} \binom{d_1}{j} (\Pi^{(1)})^j (1-\Pi^{(1)})^{d_1-j} P_{e_2}^{(1)}(j), \tag{19}$$

where $P_{e_2}^{(1)}(j)$ is given by equation (18).

Putting the above results together, the overall probability of making a mistake on the side of pattern neurons when we have one bit of external noise is given by

$$P_e^{(1)} = \frac{1}{n}\bar{P}_{e_2}^{(1)} + \frac{n-1}{n}\bar{P}_{e_1}^{(1)} \tag{20}$$

We will use this equation in order to find the best update threshold γ.

III. Choosing the best γ

In this section, we apply numerical methods to equation (20) in order to find the best γ for different values of noise parameter, ϵ. The following figures show the best choice for the parameter γ. The update threshold on the constraint side is chosen such that $\theta > \delta$. In each figure, we have illustrated the final probability of making a mistake, $P_e^{(1)}$ as well for comparision.

Figure 5 illustrates the behavior of the error probability as a function of γ for different values of ϵ and for $\Pi^{(1)} = 0.02$.

The interesting trend here is that in all cases, $\gamma_{\min}$, the update threshold that gives the best result, is chosen such that it is quite large. This actually is in line with our expectation because a small γ will results in non-corrupted nodes to update their states more frequently. On the other hand, a very large γ will prevent the corrupted nodes to correct their states, specially if there are some mistakes made on the constraint side, i.e. $\Pi^{(1)} > 0$. Therefore, since we have much more non-corrupted nodes to the corrupted nodes, it is best to choose a rather high γ but not too high.

Please also note that when $\Pi^{(1)}$ is very high, there are no values of ϵ for which error-free storage is possible.

IV. Correcting Error Outside Clusters

Now that we have an estimate on the probability of correcting one error, let us move to error correcting algorithm outside the cluster. The algorithm we use is very similar to the one we proposed in [1], given by Algorithm 2, in which $v^{(\ell)}$ is the ℓ^{th} cluster. The only difference is the analysis. To this end, let $\tilde{\lambda}$ and $\tilde{\rho}$ be the degree distribution of edges adjacent to pattern neurons and clusters in the contracted graph, $\tilde{G}$. Furthermore, let p_e be the probability that a pattern neuron is corrupted with external ± 1 noise. Then, the following theorem gives us a condition to check whether our error correcting algorithm works.

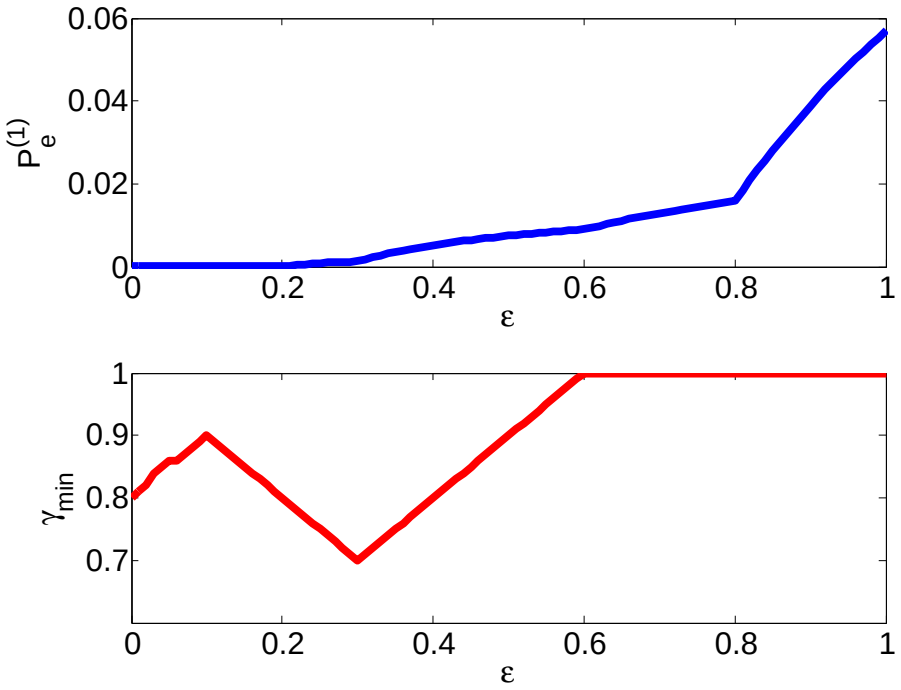

Fig. 1: The best update threshold γ for different values of noise parameter, ϵ as well as the final probability of making a mistake. In this figure, we have $\Pi^{(1)} = 0$.

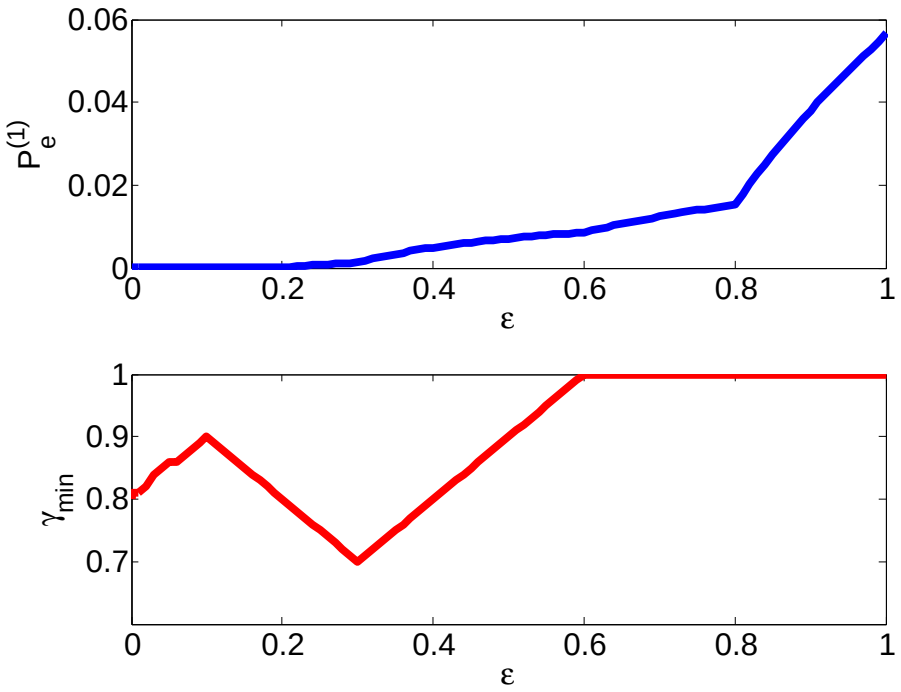

Fig. 2: The best update threshold γ for different values of noise parameter, ϵ as well as the final probability of making a mistake. In this figure, we have $\Pi^{(1)} = 0.01$.

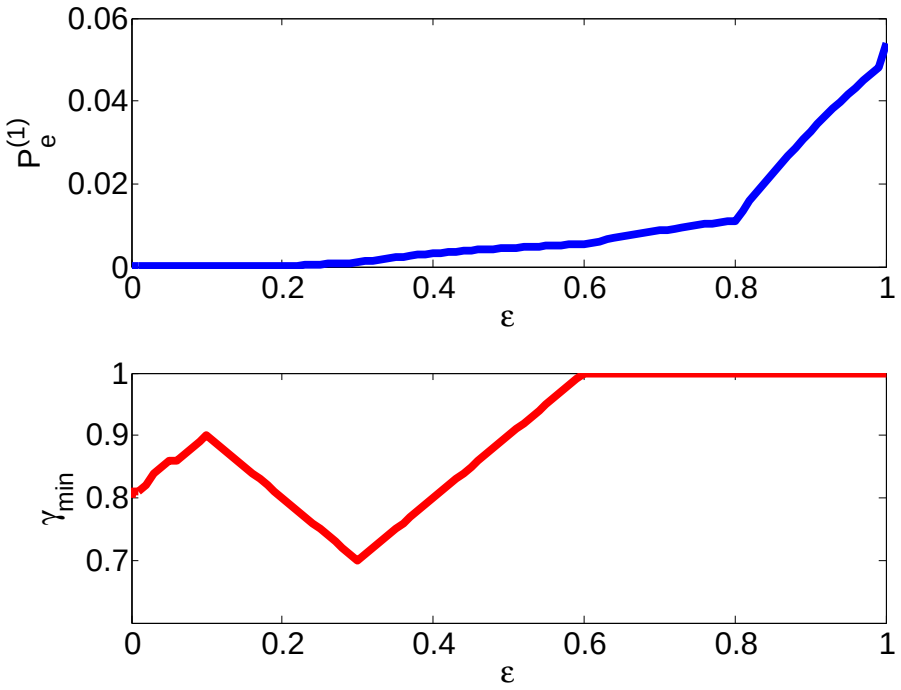

Fig. 3: The best update threshold γ for different values of noise parameter, ϵ as well as the final probability of making a mistake. In this figure, we have $\Pi^{(1)} = 0.1$.

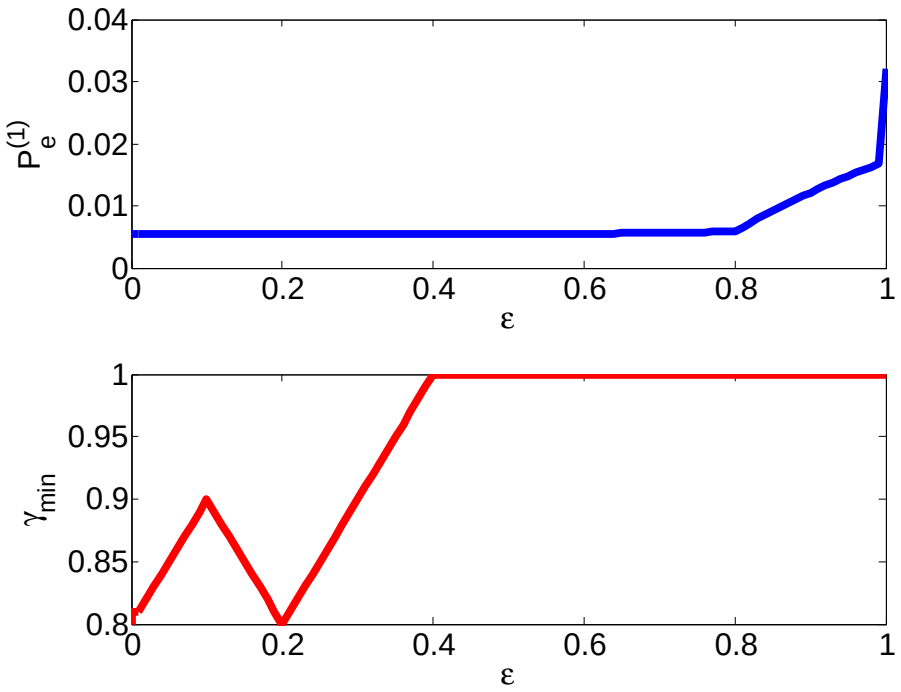

Fig. 4: The best update threshold γ for different values of noise parameter, ϵ as well as the final probability of making a mistake. In this figure, we have $\Pi^{(1)} = 0.8$.

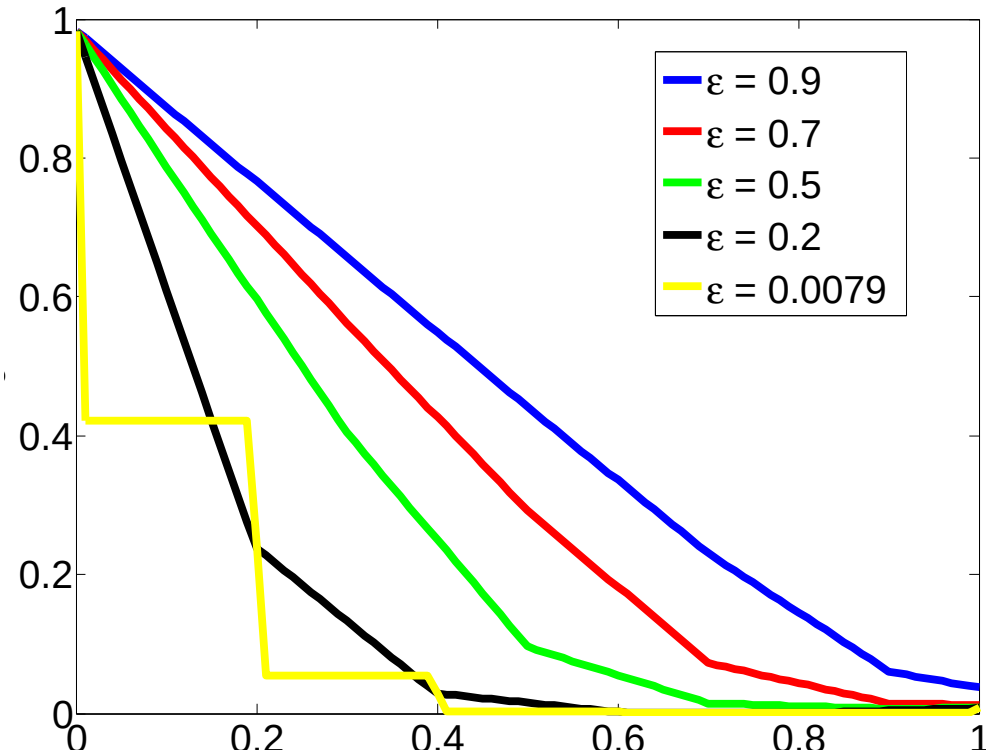

Fig. 5: The behavior of $P_e^{(1)}$ as a function of γ for different values of noise parameter, ϵ. Here, we have $\Pi^{(1)} = 0.02$.

Algorithm 2 Sequential Peeling Algorithm

Input: $\widetilde{G}, G^{(1)}, G^{(2)}, \ldots, G^{(L)}$.
Output: $x_1, x_2, \ldots, x_n$

1: **while** there is an unsatisfied $v^{(\ell)}$ **do**
2: **for** $\ell = 1 \to L$ **do**
3: If $v^{(\ell)}$ is unsatisfied, apply Algorithm 1 to cluster $G^{(l)}$.
4: If $v^{(\ell)}$ remained unsatisfied, revert the state of pattern neurons connected to $v^{(\ell)}$ to their initial state. Otherwise, keep their current states.
5: **end for**
6: **end while**
7: Declare $x_1, x_2, \ldots, x_n$ if all $v^{(\ell)}$'s are satisfied. Otherwise, declare failure.

Theorem 1. *Under the assumptions that graph $\widetilde{G}$ grows large and it is chosen randomly with degree distributions given by $\widetilde{\lambda}$ and $\widetilde{\rho}$, Algorithm 2 is successful if $p_e \cdot \widetilde{\lambda}(1 - P_c^{(1)}\widetilde{\rho}(1-z)) < z$ for $z \in (0, p_e)$.*

Proof: The proof is similar to Theorem 3.50 in [2]. Each cluster node receives an error message from its neighboring pattern nodes with probability z. Now consider a given *noisy* pattern neuron which is connected to a given cluster $v^{(\ell)}$. Let $\pi^{(\ell)}(t)$ be the probability that the cluster node $v^{(\ell)}$ with degree $\widetilde{d}_\ell$ sends an error message during iteration t of Algorithm 2. This event happens if

1) the cluster node $v^{(\ell)}$ receives at least one error message from its other neighbors among pattern neurons along its input edges, i.e. if it is connected to more than one noisy pattern neuron. Then, with probability one it send an error message.
2) the cluster node $v^{(\ell)}$ does not receive any other error messages from its other neighbors. In this case, it will send an error message with probability at most $P_e^{(1)} = 1 - P_c^{(1)}$.

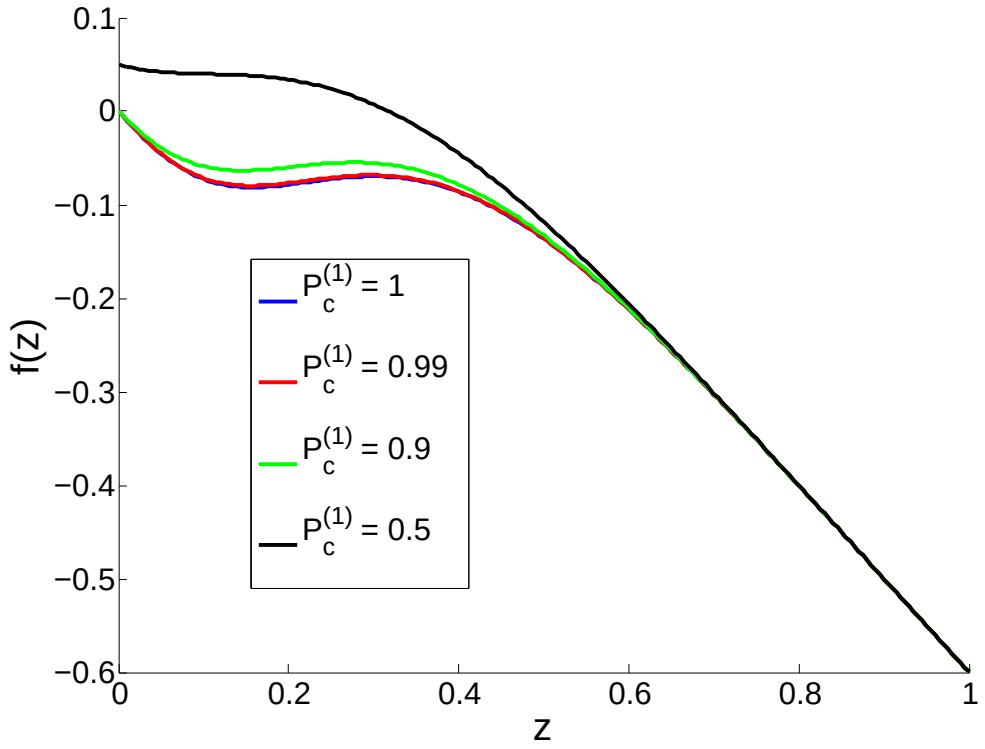

Fig. 6: The behavior of the function $f(z) = z - p_e \cdot \widetilde{\lambda}(1 - P_c^{(1)}\widetilde{\rho}(1-z))$ as a function of z and for different values of $P_c^{(1)}$. In all cases, $p_e = 0.1$.

Therefore,

$$\pi^{(\ell)}(t) = 1 - P_c^{(1)}(1 - z(t))^{\widetilde{d}_\ell - 1} \tag{21}$$

As a result, if $\pi(t)$ shows the average probability that a cluster node sends a message declaring the violation of at least one of its constraint neurons, we will have,

$$\pi(t) = \mathbb{E}_{\widetilde{d}_\ell}\{\pi^{(\ell)}(t)\} = \sum_i \widetilde{\rho}_i(1 - P_c^{(1)}(1 - z(t))^{\widetilde{d}_\ell - 1}) = 1 - P_c^{(1)}\widetilde{\rho}(1 - z(t)) \tag{22}$$

Now consider a given pattern neuron x_i with degree d_i. This node will remain noisy in iteration $t+1$ of Algorithm 2 if it was noisy in the first place and in iteration $t+1$ all of its neighbors among constraint neurons send a violation message. Therefore, the probability of this node being noisy will be $z(0)\pi(t)^{d_i}$. As a result, noting that $z(0) = p_e$, the average probability that a pattern neurons remains noisy will be

$$z(t+1) = p_e \cdot \sum_i \widetilde{\lambda}_i \pi(t)^i = p_e \cdot \widetilde{\lambda}(\pi(t)) = p_e \cdot \widetilde{\lambda}(1 - P_c^{(1)}\widetilde{\rho}(1 - z(t))) \tag{23}$$

Therefore, the decoding operation will be successful if $z(t+1) < z(t),\ \forall t$. As a result, we must look for the maximum p_e such that we will have $p_e \cdot \widetilde{\lambda}(1 - P_c^{(1)}\widetilde{\rho}(1-z)) < z$ for $z \in [0, p_e]$. ■

Theorem 1 gives us a nice insight about the role of $P_c^{(1)}$: if it is equal to 1, we will get the same result as we had in [1]. However, as $P_c^{(1)}$ moves towards 0, $z(t)$ grows towards p_e, which means we can not correct any input error. Another important trait is that once $P_c^{(1)} < 1$, $z(t)$ will be bounded away from 0 because $\widetilde{\lambda}(x)$ is an increasing function of x. So given that $1 - P_c^{(1)}\widetilde{\rho}(1-z) \geq 1 - P_c^{(1)}$, we have $z(t+1) \geq \widetilde{\lambda}(1 - P_c^{(1)})$. Figure 6 illustrates how $z - p_e \cdot \widetilde{\lambda}(1 - P_c^{(1)}\widetilde{\rho}(1-z))$ behaves as a function of z and for different values of $P_c^{(1)}$. Error-free storage is possible in places where the relationship has negative values.

Figure 7 illustrates the final error probability (i.e. $z(\infty)$) as a function of p_e for different values of $P_c^{(1)}$. Note that as $P_c^{(1)}$ becomes smaller, the final error probability increases. Furthermore, although not visible in the graph, when $P_c^{(1)} > 0$, the final error probability is strictly bigger than 0.

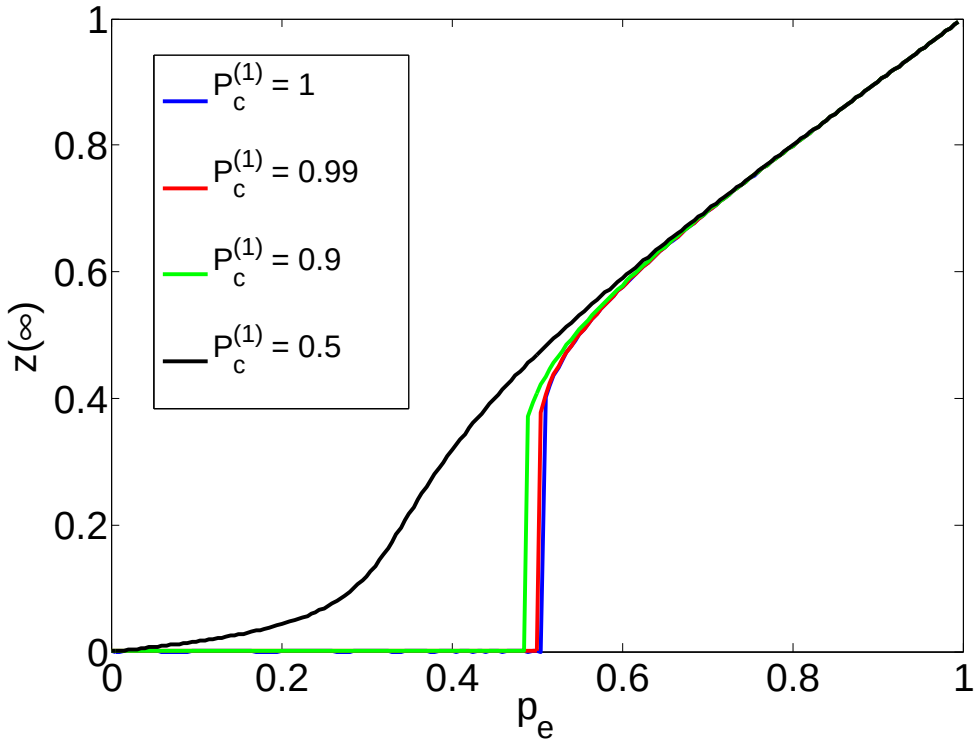

Fig. 7: The final error probability $z(\infty)$ as a function of p_e and for different values of $P_c^{(1)}$.

V. Conclusion and Future Works

From the arguments mentioned in this document, it is quite clear that the best choice for update threshold at pattern neurons is a rather large γ. In fact, our results show that the best γ is always larger than the noise parameter ϵ, which is good news since it means $P_e^{(0)}$, the probability of making a mistake in absence of external noise, is equal to zero. So from now on, we will assume not making a mistake in absence of external noise.

In presence of a single external error, we could obtain a lower bound on the probability that such an error gets corrected ($P_c^{(1)}$). This probability is a function of γ of course and finding their relationship is our next task.

Finally, having obtained $P_c^{(1)}$, we could plug it into our already available results in order to obtain the recursion on the probability of error for the whole network (recall that our arguemnts in this document corresponds to only a single cluster).

References

[1] A. Karbasi, A. H. Salavati, and A. Shokrollahi, "Iterative learning and denoising in convolutional neural associative memories," in *Proc. Int. Conf. on Machine Learning (ICML)*, 6 2013.

[2] M. Luby, M. Mitzenmacher, A. Shokrollahi, and D. A. Spielman, "Efficient erasure correcting codes," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 569–584, 2001.