Storing messages with multipartite neural cliques

Nissim Zerbib[1], Vincent Gripon[2], and ?[3]

[1] *Département d'Informatique, École normale supérieure, Paris, France*
[2] *Département d'Électronique, Télécom Bretagne, Brest, France*

Abstract

We extend recently introduced associative memories based on clustered cliques to multipartite cliques. We propose a variant of the classic retrieving rule. We study its performance relatively to the former one for retrieving partially erased or corrupted messages. We provide both analytical and simulation results showing improvements in both networks capacities and resilience. We compute asymptotic capacities of these networks.

Index terms— associative memory, error correcting code, cliques, multipartite cliques, neural networks

I Introduction

II Networks of neural cliques

1 Learning messages

Let $\mathcal{M}$ be a set of messages of length c over the alphabet $\mathcal{A} = \{1, 2, \dots, \binom{l}{a}\}$ and $m = |\mathcal{M}|$. Messages in $\mathcal{M}$ are stored in an undirected unweighted graph, or neural network, of $c \cdot l$ vertices or neurons.

Each message $x = (x_i)_{1 \leqslant i \leqslant c}$ is mapped to $y = (y_i)_{1 \leqslant i \leqslant c}$ where y_i is the binary representation of the x_i-th a-combination of l elements (the mapping doesn't matter as long as it is fixed).

Storing a message x in the network amounts to consider the network in state y, that is for all $1 \leqslant j \leqslant l$ and $1 \leqslant i \leqslant c$, the neuron j in the cluster i is activated if and only if $y_{i,j} = 1$; then adding all connections between activated neurons excepting for neurons sharing the same cluster. Thus we obtain a multipartite clique between activated neurons representing a message. The parameter a describing the number of activated neurons per cluster will be called number of activities.

2 Retrieving messages

Retrieving messages occurs by message passing.

a The "winner takes all" rule

take only maximum score

b The "a winners take all" rule

To fully exploit the local regularity of the code (constant weight of a), we make a small modification the classic retrieving rule by keeping activated all neurons that achieve a score equal or

greater than the score appearing in w-th position in the ordered array of scores (where w is a positive integer parameter)

Although it is a less biologically plausible rule, it has the same algorithmic complexity, since selecting the w-th element in an array of n elements is $O(n)$ in complexity.

Intuitively the choice $w = a$ is optimal (it it is the most natural with $w = 1$). The following is a description of the algorithm[1] :

Algorithm 1 w-sum of sum

Input: v a binary array of length $n = c \cdot l$, W the weight matrix representing the network

1: **procedure** SUM OF SUM(v, W, γ, S, w)
2: **for** s from 1 to S **do**
3: $s \leftarrow \gamma v + Wv$
4: **for** i from 1 to c **do**
5: treshold $\leftarrow$ Select(s[i], w) ▷ Select(t, w) returns the element that would be in rank w if the array t was sorted
6: **for** j from 1 to l **do**
7: $v[i][j] \leftarrow s[i][j] \geqslant$ treshold
8: **end for**
9: **end for**
10: **end for**
11: **end procedure**

III Retrieval performance

1 Retrieving partially erased messages

a Analytical result

Suppose we are trying to recover a message where c_e clusters have been erased.

[1] No optimizations are described here (early stopping etc.)

As the probability for i and j to be active in their respective clusters for one message is $\frac{a}{l}$ providing messages are identically distributed, it follows that the probability for the edge (i, j) of not being added in the network for one message is $1 - \left(\frac{a}{l}\right)^2$. Since messages are independent, the density d, which is the probability for an edge to be in the network, can be expressed like this :

$$d = 1 - \left(1 - \left(\frac{a}{l}\right)^2\right)^m \tag{1}$$

Thanks to the memory effect γ, activated neurons in a non-erased cluster stay activated and are the only ones doing so in their cluster. For neurons in erased clusters, the maximum attainable score is $a(c - c_e)$. Neurons corresponding to the original message achieve this score. Besides the probability for a neuron v to achieve the maximum score in such a cluster, that is to mean $a(c - c_e)$ is $d^{a(c-c_e)}$.

Since neurons representing messages are independent (as messages are themselves independent), the probability for none of the vertices of one erased cluster, excepting the correct ones, to achieve the maximum is $\left(1 - d^{a(c-c_e)}\right)^{l-a}$.

Scores in clusters being also independent, the probability for none of the vertices in any erased cluster, excepting the correct ones, to achieve the maximum in their respective clusters is $\left(1 - d^{a(c-c_e)}\right)^{c_e(l-a)}$.

Whence error rate in retrieving messages is :

$$P_{err} = 1 - \left(1 - d^{a(c-c_e)}\right)^{c_e(l-a)} \tag{2}$$

We define the efficiency of the network relatively to the hardness of the problem (how much it is efficient compared to a perfect associative memory). A perfect associative memory would

use exactly the same number of bits than the original messages and recover messages by maximum likelihood (ambiguities causing errors). Efficiency is the amount of information retrieved divided by the hardness of the problem.

The network is a binary symmetric matrix with null diagonal and can therefore be stored in memory as $\frac{c(c-1)l^2}{2}$ bits. Since messages are not stored in an ordered fashion, the information stored is the set of messages, thereby it amounts to $\log_2(\frac{(\binom{l}{a}^c)^m}{m!}) \approx m(c\log_2\binom{l}{a} - \log_2(m) + 1)$. The information effectively stored in the network is $P_{retrieve} \times m\frac{2\left(c\log_2\binom{l}{a}-\log_2(m)+1\right)}{c(c-1)l^2}$.

Ambiguities would occur in a perfect associative memory if at least two messages were identical over non-erased letters. The probability it doesn't happen is $(1-\binom{l}{a}^{c-c_e})^{m-1}$.

$\eta_m = P_{retrieve}m\frac{2\left(c\log_2\binom{l}{a}-\log_2(m)+1\right)}{c(c-1)l^2} \times (1-(\frac{1}{\binom{l}{a}})^{c-c_e})^{-(m-1)}$

The maximum information that can be stored in the network is therefore $\eta = \max_x \eta_x$.

b Simulations

Theoretical and empirical densities match very closely, see Figure 1.

Simulations agree with the analytical results for error rate, see Figure 2.

For one iteration (for erasures, not for errors), "winner takes all" is the same as "a-winners take all".

For multiple iterations a-winners take all is a significant improvement.

Figure 3 shows improvements in capacities over unipartite cliques network. The more the problem is difficult, the more redundancies introduced by multipartite cliques are useful.

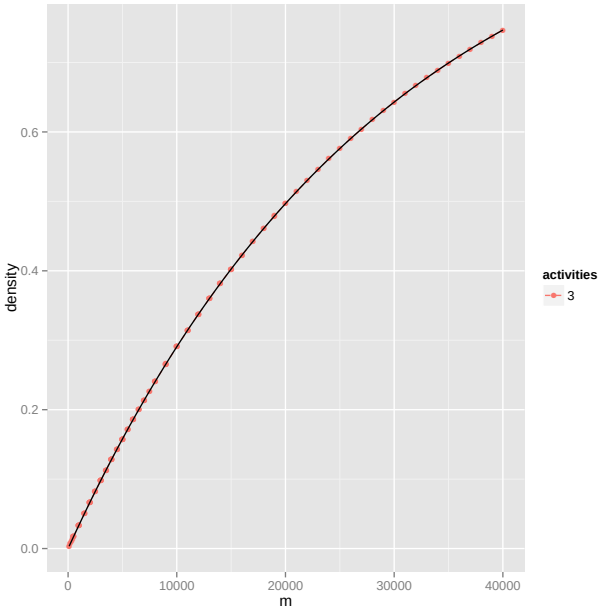

Figure 1: Theoretical and empirical densities

2 Retrieving corrupted messages

a Analytical result

Messages can be corrupted, our models for errors is : fix c_e the number of erroneous clusters, we draw at random c_e clusters and in each of these draw at random, uniformly, a new letter in the alphabet that will replace the old one (it could be the same).

Knowing the density of the network, which follows formula (1), we can compute the laws of probabilities for scores achieved by different type of neurons in different clusters. We can divide neurons in 5 groups. Subscript g will indicate a neuron being part of the message (it should be activated), subscript b the contrary (the neuron should not be activated). Subscript a means the neuron is already activated. Subscript c corre-

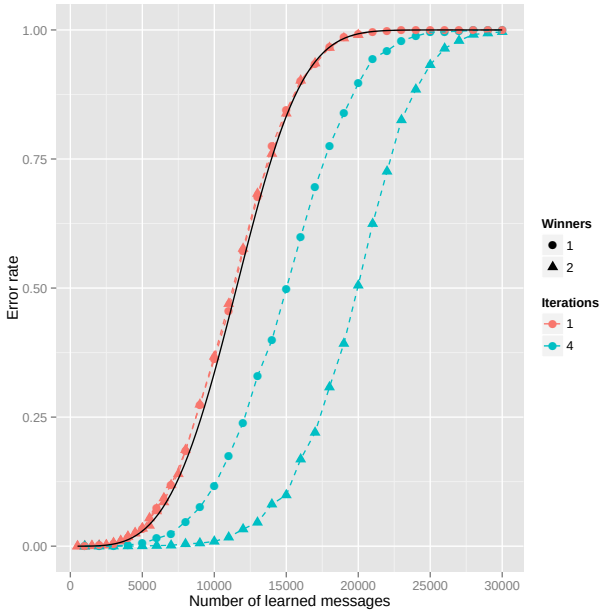

Figure 2: 2 activities per cluster, 4 clusters, 512 neurons per cluster, two erasures, each point is the mean of 5 networks with 1000 sampled messages, analytical result in continuous black

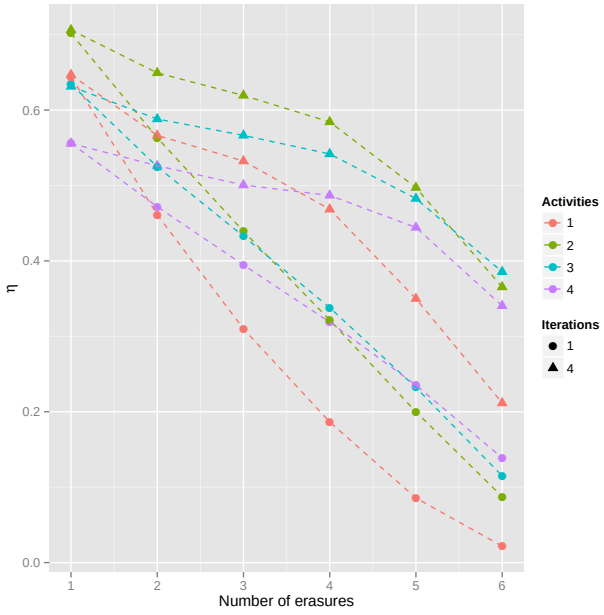

Figure 3: Comparison between different number of activities per cluster : 8 clusters, 256 neurons per cluster, 1000 sampled messages, pools of 5

sponds to a correct cluster, e to an erroneous cluster. Correct neurons in uncorrupted clusters achieve at least a score of $a(c-c_e-1)+\gamma$. Being connected to a given activated neuron in a corrupted cluster occurs with probability d, since errors are identically distributed. Moreover corrects neurons see their scores shift, as do all activated neurons respectively thanks to being part of the original message and the memory effect γ. We can express those probability laws as : For example an **a**ctivated **g**ood neuron in a **c**orrect cluster starts with a base score of $a(c-c_e-1)+\gamma$. The probability it has to be connected to a random activated neuron in an erroneous cluster is d. Thereby its law is :

$$P(n_{agc} = a(c-c_e-1)+\gamma+x) = \binom{ac_e}{x} d^x (1-d)^{ace-x}$$

Correct neurons in corrupted clusters achieve at least a score of $a(c-c_e)$. $P(n_{ge} = a(c-c_e)+x) = \binom{a(c_e-1)}{x} d^x (1-d)^{a(c_e-1)-x}$

Activated neurons benefits of the memory effect γ :
$P(n_{abe} = \gamma + x) = \binom{a(c-1)}{x} d^x (1-d)^{a(c-1)-x}$

Finally $P(n_{be} = x) = P(n_{bc} = x) = \binom{a(c-1)}{x} d^x (1-d)^{a(c-1)-x}$

A message is retrieved after one iteration, if and only if correct neurons achieve strictly the best scores.

$$P_{retrieve} = \left[\sum_{n=1}^{a(c-1)+\gamma} \left[\sum_{k=1}^{a} \binom{a}{k} P(n_{agc} = n)^k P(n_{agc} > n)^{a-k} \right] P(n_{bc} < n)^{l-a} \right]^{c-c_e} \tag{3}$$

$$\times \left[\sum_{n=1}^{a(c-1)+\gamma} \left[\sum_{k=1}^{a} \binom{a}{k} P(n_{ge} = n)^k P(n_{ge} > n)^{a-k} \right] P(n_{be} < n)^{l-2a} P(n_{abe} < n)^a \right]^{c_e} \tag{4}$$

$$P_{err} = 1 - P_{retrieve}$$

b Simulations

See Figure 4 compares the two rules. "a-winners take all" better than "winner takes all" with errors instead of erasures, even for one iteration.

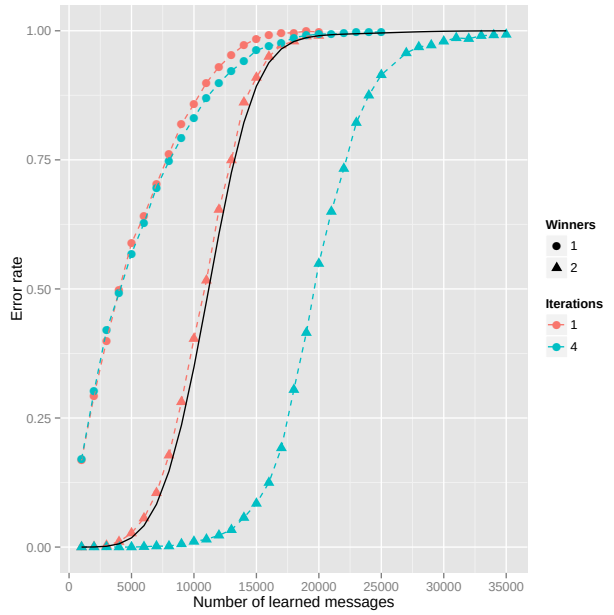

Figure 4: 2 activities per cluster, 4 clusters, 512 neurons per cluster, one erroneous cluster, each point is the mean of 5 networks with 1000 sampled messages, analytical result in continuous black

$$\eta_m = P_{retrieve} m^{\frac{2\left(c\log_2\binom{l}{a}-\log_2(m)+1\right)}{c(c-1)l^2}} \times \left[\sum_{k=c_e+1}^{c}\left(1-\frac{1}{\binom{l}{a}}\right)^k\left(\frac{1}{\binom{l}{a}}\right)^{c-k}\right]^{-(m-1)}$$

IV Resilience

As information is shared accross connections with multiple activated neurons per cluster, they are more resilient to network damages.

$P(n_{v_c} = n_0 + \gamma) = \binom{a(c_k-1)}{n_0}(1-\psi)^{n_0}\psi^{a(c_k-1)-n_0}$

$P(n_{v_c} = n_0) = \binom{ac_k}{n_0}(1-\psi)^{n_0}\psi^{ac_k-n_0}$

$P_+ = \psi(1-d) + (1-\psi)d$

$P(n_v = x) = \binom{ac_k}{x}P_+^x(1-P_+)^{ac_k-x}$

$P(\text{no other vertex activated in this cluster}) = \sum_{n_0=1}^{a(c-c_e)} P(n_{v_c} \geqslant n_0)^a \left[\sum_{x=0}^{n_0-1} P(n_v = x)\right]^{l-a}$

Sans tenir compte du choix au hasard si plus sont activés. Pour une itération $\gamma = \infty$. Pour plusieurs $\gamma = 1$.

a priori for noise on edges :

$P(\text{no other vertex activated in this cluster}) = \sum_{n=1}^{a(c-c_e)}\left[\sum_{k=1}^{a}\binom{a}{k}P(n_{v_c}=n)^k P(n_{v_c}>n)^{a-k}\right] P(n_v < n)^{l-a}$

$P(\text{no other vertex activated in any cluster}) = \left(\sum_{n=1}^{a(c-c_e)}\left[\sum_{k=1}^{a}\binom{a}{k}P(n_{v_c}=n)^k P(n_{v_c}>n)^{a-k}\right]\left[\sum_{x=0}^{n-1} P(n_v =\right.\right.$

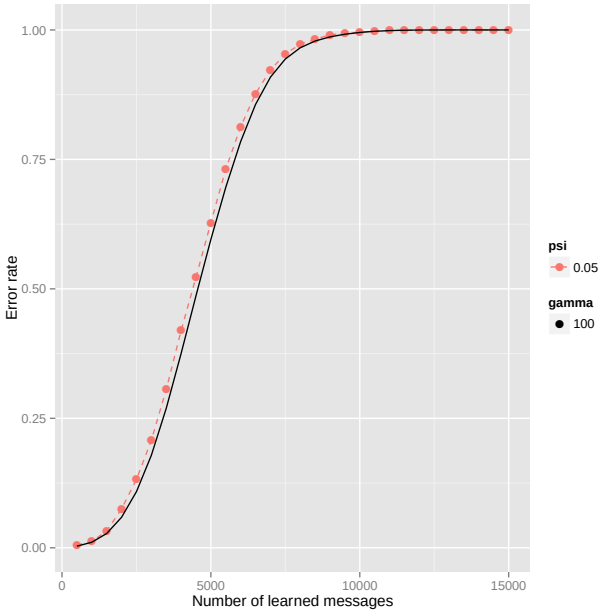

Figure 5: 2 activities per cluster, 8 clusters, 256 neurons per cluster, 4 erasures, each point is the mean of 10 networks with 1000 sampled messages, analytical result in continuous black

We compute values represented on figure 6 by scanning on the number of messages, so as to find the maximum efficiency.

After a transition, efficiencies seem to follow a linear decreasing low according to ψ.

Absolute values of slopes diminish for increasing number of activities per cluster. We can conclude multipartite cliques improve the resilience of the network.

$$\eta_{m,\psi} = P_{retrieve} m \frac{2\left(c\log_2\binom{l}{a} - \log_2(m) + 1\right)}{c(c-1)l^2} \times \left(1 - \binom{l}{a}^{c-c_e}\right)^{-(m-1)} \times \frac{1}{1+\psi\log_2(\psi)+(1-\psi)\log_2(1-\psi)}$$

$$\eta_\psi = max_x\, \eta_{x,\psi}$$

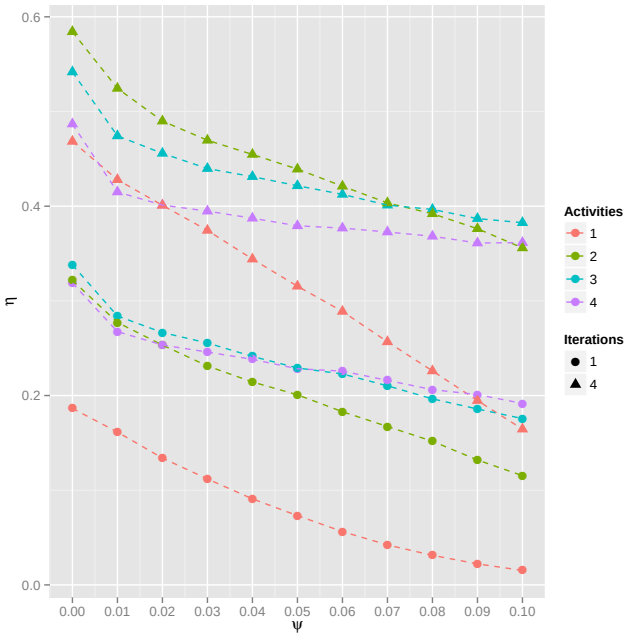

Figure 6: Comparison of efficiencies between different number of activities per cluster : 8 clusters, 256 neurons per cluster, 1000 sampled messages, pools of 5 networks

V Maximum theoretical efficiency

It was shown in [?], asymptotic capacities of clique networks is $\log 2$. We generalise this result to multipartite cliques. The entropy of the messages set can be approximated as $H(M) \approx mc \log_2 \binom{l}{a}$. From $d = 1 - \left(1 - \left(\frac{a}{l}\right)^2\right)^m$, we can deduce $m \sim \frac{\log_2(1-d)}{\log_2(1-(\frac{a}{l})^2)} \sim -\frac{l^2 \log_2(1-d) \log 2}{a^2}$. We set $\log_2 \binom{l}{a} = kc$ (otherwise density approaches 0 or 1, and efficiency 0). It follows that $\eta \sim c \log_2 \binom{l}{a} \log 2(1-d) \sim \frac{kc^2 \log_2(1-d) \log 2}{a^2 \binom{c}{2}}$.

$\eta \sim -2k \log_2((1-d)a^{-2} \log 2$.

We can bound the probability $P_{\exists e}$ for a message being "present" in the network, that is to mean adding it will not add any edge to the network. By union-bound $P_{\exists e} \leqslant \binom{l}{a}^c d^{a^2\binom{c}{2}} = 2^{c \log_2 \binom{l}{a} + a^2 \binom{c}{2} \log_2(d)} \approx 2^{c^2(k + \frac{a^2}{2} \log_2(d))}$.

Fixing $k^* = -\frac{a^2}{2} \log_2(d)$ gives us $\eta \sim \log_2 d \log_2(1-d) \log 2$ and therefore

$$\eta_{max} = \log 2$$

Multipartite cliques don't improve theoretical efficiency (for not accepting messages that weren't previously seen by the network)

VI Conclusion

References

[LPGRG14] François Leduc-Primeau, Vincent Gripon, Michael Rabbat, and Warren Gross. Cluster-based associative memories built from unreliable storage. In *ICASSP*, May 2014. To appear.