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 efficiencies and resilience to damages. We compute asymptotic capacities of these networks.**

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

I. Introduction

Associative memories

Recently, a new type of associative memories was proposed by Gripon and Berrou [2].

GBNN can also be used to retrieve messages from erroneous versions of them. We derive a formula for error rate in case of errors on messages.

Fault tolerance of those networks was extensively studied in [4]. We extend this to multipartite cliques and show significant improvements in resilience to faults.

II. Networks of neural cliques

A. Learning messages

Let $\mathcal{M}$ be a set of messages of length c over the alphabet $\mathcal{A} = \{1, 2, \dots, \binom{l}{a}\}$ (where a is a positive integer) and $m = |\mathcal{M}|$. Messages in $\mathcal{M}$ are stored in a neural network, of $c \cdot l$ units.

Our goal is to store messages so as to be able to retrieve them from partial or erroneous versions of them.

Each message $x = (x_j)_{1 \leqslant j \leqslant c}$ is mapped to $y = (y_j)_{1 \leqslant j \leqslant c}$ where y_j 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).[1]

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

Units k in cluster j will be designated by index $l = (j-1)c + k$.

The state of the network corresponds to a message.

[1]This is a local code of constant-weight a.

$\tilde{W} = \max_{1 \leqslant i \leqslant m} x^i x^{i\intercal}$

$$W_{ll'} = \begin{cases} 0 & \text{if } \exists j \in [1, c],\ (j-1)c + 1 \leqslant l, l' \leqslant jc \\ \tilde{W}_{ll'} & \text{otherwise} \end{cases}$$

B. Retrieving messages

Retrieving messages occurs by message passing. We adapt the rule first developed in [2] to multipartite cliques.

We make a small modification the classic retrieving rule to fully exploit the local regularity of the code (constant weight of a). We keep activated all units that achieve a score equal or greater than the w-th score. (where w is a positive integer parameter)

Intuitively the choice of $w = a$ is optimal (it it is the most natural with $w = 1$ as well). Algorithm 1 is a description of the algorithm[2] :

x

$\tilde{x}$

$s^t = \gamma \tilde{x^t} + W \tilde{x^t}$

select(w, v) is equal to the w-th element of v (counting repetitions).

$$\tilde{x}^{t+1}_{(j-1)c+k} = \begin{cases} 1 & \text{if } s^t_{(j-1)c+k} \geqslant \text{select}(w, (s^t_{(j-1)c+k'})_{1 \leqslant k' \leqslant l}) \\ 0 & \text{otherwise} \end{cases}$$

Algorithm 1 w-sum of sum

Input: v a binary array of length $n = c \cdot l$, W the binary weight matrix representing the network, S the number of iterations, γ memory effect

```
procedure SUM OF SUM(v, W, γ, S, w)
    for s from 1 to S do
        s ← γv + Wv
        for i from 1 to c do
            treshold ← Select(s[i], w)                ▷ Select(t, w) returns the element that would be in rank w if the array t was sorted
            for j from 1 to l do
                v[i][j] ← s[i][j] ⩾ treshold
            end for
        end for
    end for
end procedure
```

III. Retrieval performance

A. Retrieving partially erased messages

1) Analytical result: We generalise results on erasures obtained in [2] to multipartie cliques.

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

Suppose we are trying to recover a message where c_e clusters have been erased (i.e. all units in those clusters are set to 0).

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 as :

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

Thanks to the memory effect γ, activated units in a non-erased cluster stay activated and are the only ones doing so in their cluster. For units in erased clusters, the maximum attainable score is $a(c-c_e)$. Units corresponding to the original message achieve this score. Besides the probability for a unit 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 unit activations 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 :

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

In order to compare different networks using different alphabets, 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 $\log_2(\frac{(\binom{l}{a}^c)^m}{m!}) =_{m\to+\infty} m(c\log_2\binom{l}{a} - \log_2(m) + \frac{1}{\log 2} + o(1))$ (thanks to Stirling's formula). The information effectively stored in the network is $P_{retrieve} \times \frac{2m\left(c\log_2\binom{l}{a} - \log_2(m) + \frac{1}{\log 2}\right)}{c(c-1)l^2}$.

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 $(1-\binom{l}{a}^{c-c_e})^{m-1}$.

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

The maximum information that can be stored in the network is therefore $\eta = \max_x \eta_x$, which we will define as the efficiency of the network.

2) 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.

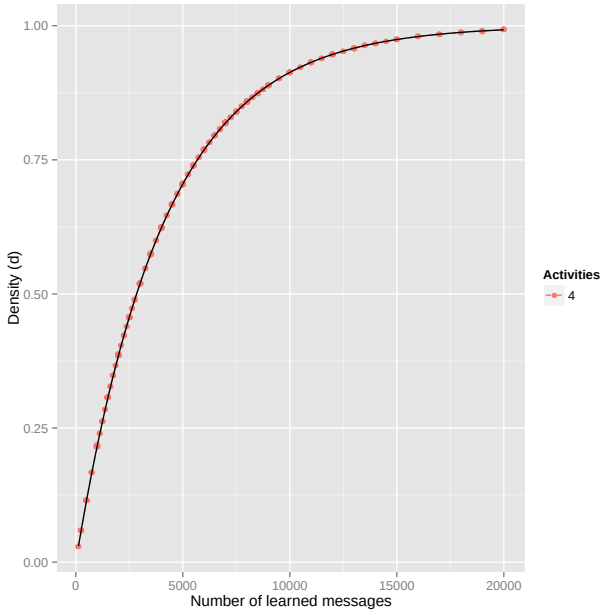

Figure 1. Theoretical and empirical densities for a network of 8 clusters, 256 units per cluster and 4 activated units per cluster

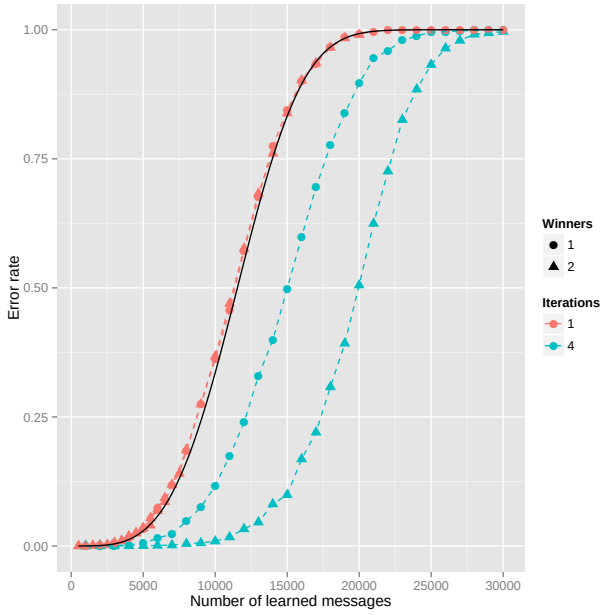

Figure 2. Evolution of error rate for increasing number of stored messages, 2 activities per cluster, 4 clusters, 512 units per cluster, two erasures, each point is the mean of 5 networks with 1000 sampled messages, analytical result (for one iteration) in continuous black

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.

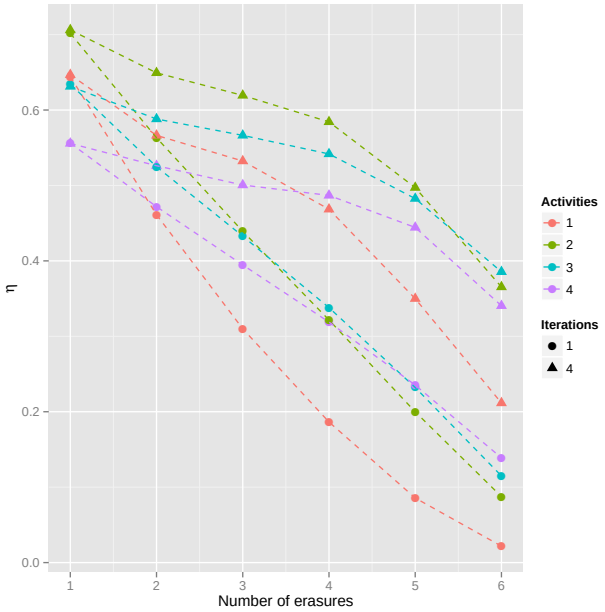

Figure 3. Comparison between different number of activities per cluster : 8 clusters, 256 units per cluster, 1000 sampled messages, pools of 5

B. Retrieving corrupted messages

1) Analytical result: We provide a formula for error rate in case of messages with corrupted clusters.

Messages can be corrupted, our models for errors is : given 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 units in different clusters. We can divide units in 5 groups. Subscript g will indicate a unit being part of the message (it should be activated), subscript b the contrary (the unit should not be activated). Subscript a means the unit is already activated. Subscript c corresponds to a correct cluster, e to an erroneous cluster. Correct units in uncorrupted clusters achieve at least a score of $a(c-c_e-1)+\gamma$. Being connected to a given activated unit in a corrupted cluster occurs with probability d, since errors are identically distributed. Moreover corrects units see their scores shift, as do all activated units 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 unit 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 unit 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 units 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 units 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 units achieve strictly the best scores.

The probability for all good units of a correct cluster (resp. of an erroneous cluster) to achieve a score equal or greater than x, with at least one having a score of x is $P_{ac}(x) = \sum_{k=1}^{a} \binom{a}{k} P(n_{agc} = x)^k P(n_{agc} > x)^{a-k}$ (resp. $P_e(x) = \sum_{k=1}^{a} \binom{a}{k} P(n_{ge} = x)^k P(n_{ge} > x)^{a-k}$)

$$P_{retrieve} = \left[\sum_{x=1}^{a(c-1)+\gamma} P_{ac}(x) P(n_{bc} < x)^{l-a} \right]^{c-c_e} \times \left[\sum_{x=1}^{a(c-1)+\gamma} P_e(x) P(n_{be} < x)^{l-2a} P(n_{abe} < x)^a \right]^{c_e} \quad (3)$$

with the approximation that there is few activated units that should be kept active in an erronenous cluster.

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

2) 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.

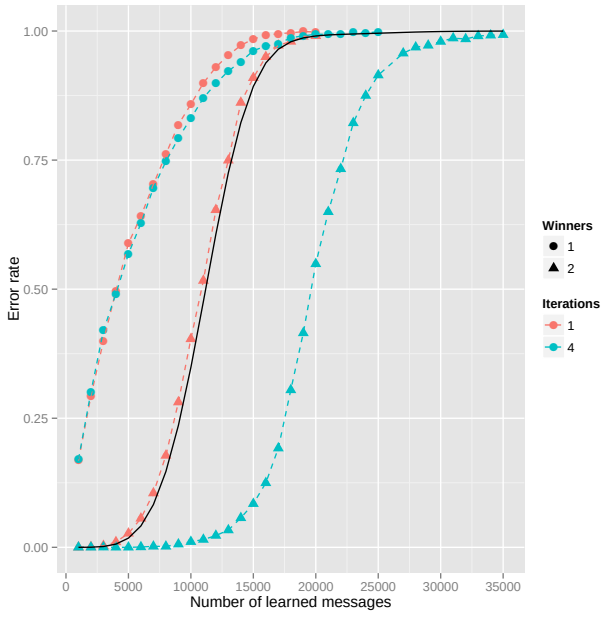

Figure 4. 2 activities per cluster, 4 clusters, 512 units per cluster, one erroneous cluster, each point is the mean of 5 networks with 1000 sampled messages, analytical result in continuous black

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

IV. Resilience

A. Analytical result

As information on a message is shared across connections with multiple activated units per cluster, such networks are more resilient to damages (for one message information is carried by $a^2\binom{c}{2}$ edges. v_c will indicate a correct unit that should be activated, v the other type of units.

Our deviation model is the same as [4][3], that is random binary noise on edges. We generalise results of [4] to multipartite cliques.

We assume $\gamma = \infty$ for one iteration as it prevents activated neurons to be deactivated. With probability $1-\psi$, one correct vertex is still connected to a given activated unit in an intact cluster. Thus $P(n_{v_c} = x) = \binom{a(c_k-1)}{x}(1-\psi)^x\psi^{a(c-c_e-1)-x}$

The probability for an edge to be present in the network after damages is : $P_+ = \psi(1-d) + (1-\psi)d$. So for other units $P(n_v = x) = \binom{a(c-c_e)}{x}P_+^x(1-P_+)^{a(c-c_e)-x}$.

The probability for all correct units to achieve a score equal or greater than x, with at least one achieving a score of x is $P_c(x) = \sum_{k=1}^{a}\binom{a}{k}P(n_{v_c} = x)^k P(n_{v_c} > x)^{a-k}$

It follows that $P(\text{no other vertex activated in one cluster}) = \sum_{x=1}^{a(c-c_e)} P_c(x)P(n_v < x)^{l-a}$.

$$P_{retrieve} = \left(\sum_{x=1}^{a(c-c_e)} P_c(x)P(n_v < x)^{l-a}\right)^{c_e} \tag{4}$$

B. Simulations

With random noise on edges, hardness of the problem must take into account the capacity of the binary symmetric channel (the closer ψ is of 0.5, the harder the problem is). Our new is efficiency is written as : $\eta_{m,\psi} = P_{retrieve}m\frac{2\left(c\log_2\binom{l}{a}-\log_2(m)+\frac{1}{\log 2}\right)}{c(c-1)l^2} \times (1-\binom{l}{a}^{c-c_e})^{-(m-1)} \times \frac{1}{1+\psi\log_2(\psi)+(1-\psi)\log_2(1-\psi)}$

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

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 usage of multipartite cliques improve the resilience of the network.

Figure 6 seems to indicate a linear relationship (after a transition for multiple activities) between $\eta_\psi - \eta_0$ and ψ.

V. Maximum theoretical efficiency

It was shown in [5] that maximal asymptotic capacity of Willshaw networks is $\log 2$. We adapt this result to multipartite clique clustered networks.

The entropy of the messages set can be approximated as $H(\mathcal{M}) \approx mc\log_2\binom{l}{a}$. 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

[3]Compared to this article, we do not draw at random a letter in case of ambiguity, which hurts performance a bit.

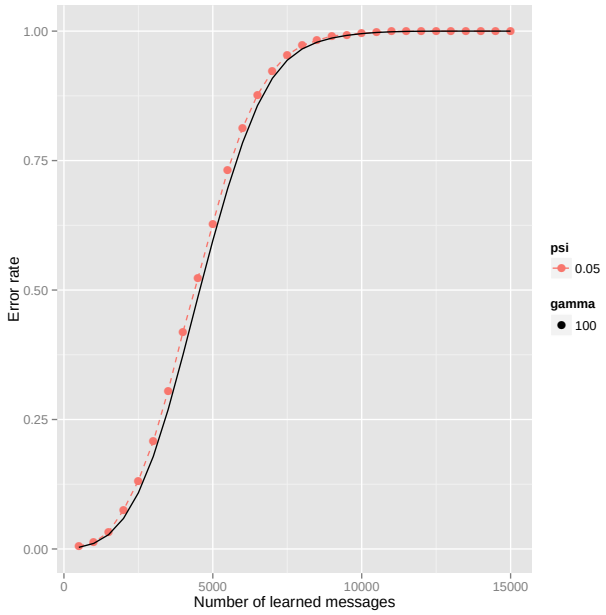

Figure 5. $\psi = 0.05$, 2 activities per cluster, 8 clusters, 256 units per cluster, 4 erasures, each point is the mean of 10 networks with 1000 sampled messages, analytical result in continuous black

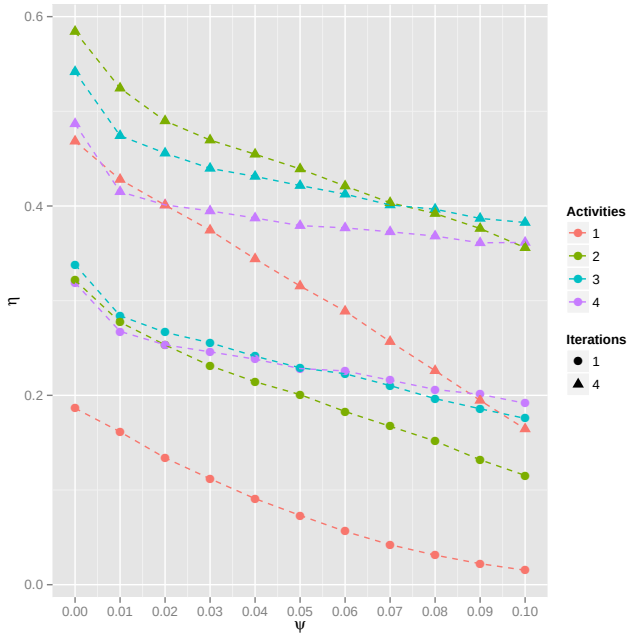

Figure 6. Comparison of evolutions of efficiency with increasing damages between different number of activities per cluster : 8 clusters, 256 units per cluster

$\log_2\binom{l}{a} = kc$ (otherwise density approaches 0 or 1, and efficiency 0). It follows that $\eta \sim \frac{mc\log_2\binom{l}{a}}{l^2\binom{c}{2}} \sim \frac{kc^2\log_2(1-d)\log 2}{a^2\binom{c}{2}}$.

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

We can bound the probability $P_{\exists e}$ for a message being "wrongly present" in the network, that is to mean adding it will not add any edge to the network albeit it is not in $\mathcal{M}$. 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 $P_{\exists e}$ and $\eta \sim \log_2 d \log_2(1-d)\log 2$ and therefore

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

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

VI. Conclusion

References

[1] Vincent Gripon and Claude Berrou. A simple and efficient way to store many messages using neural cliques. In *Proceedings of IEEE Symposium on Computational Intelligence, Cognitive Algorithms, Mind, and Brain*, pages 54–58, Paris, France, April 2011.

[2] Vincent Gripon and Claude Berrou. Sparse neural networks with large learning diversity. *IEEE Transactions on Neural Networks*, 22(7):1087–1096, July 2011.

[3] Vincent Gripon and Claude Berrou. Nearly-optimal associative memories based on distributed constant weight codes. In *Proceedings of Information Theory and Applications Workshop*, pages 269–273, San Diego, CA, USA, February 2012.

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

[5] Günther Palm. On associative memories. 1980.