Nearly-optimal associative memories based on distributed constant weight codes

Goals

- Design a nearly-optimal associative memory in terms of efficiency,
  - Associative memory: device able to retrieve previously learned messages from part of their content,
  - Efficiency: ratio of the amount of bits learned to the amount of bits used,
- Use for that an architecture based on distributed constant weight codes.

Binary constant weight codes

Three parameters

- Length n,
- Weight w,
- Overlapping r.

Binary

$$code = \{m\} \subset \{0;1\}^n$$

Constant weight

$$\forall m, \sum_{i=1}^{n} m_i = w$$

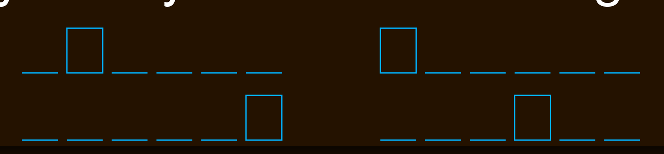
Code

$$\forall m \neq m', \sum_{i=1}^{n} m_i m'_i \leq r$$

Thrifty code

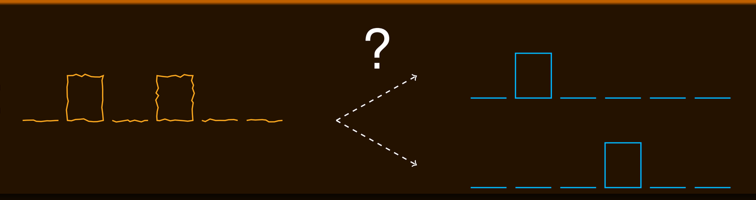
A particular constant weight code with weight 1

Code containing only binary words with a single “1”:



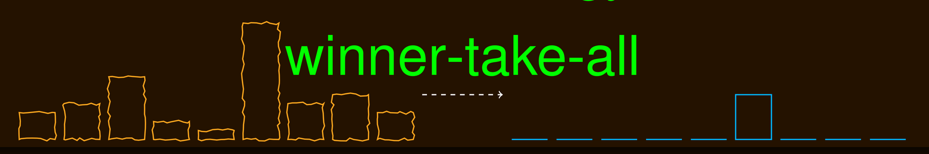
Drawback

$d_{min} = 2$:

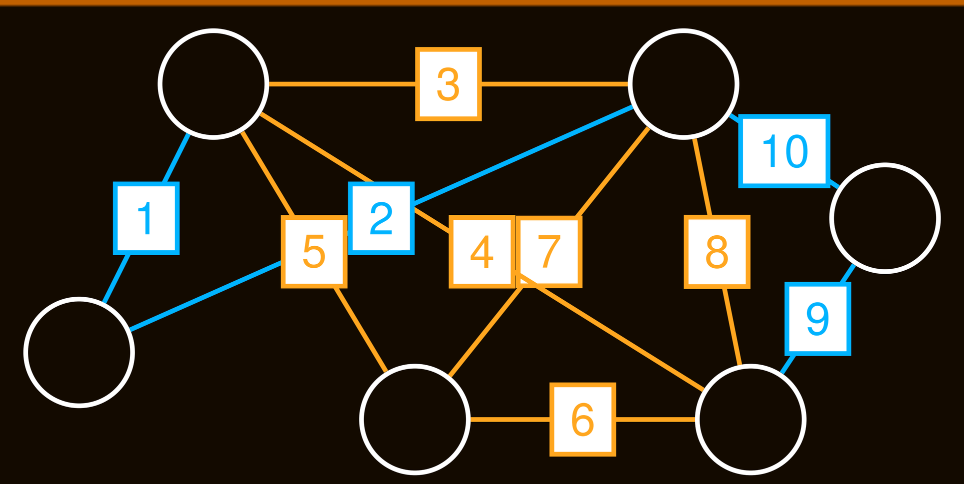


Pros

Easy to decode and minimise the energy:



Clique code



Another binary constant weight code: the clique code

- Cliques of constant size,
- Idea = use connections instead of vertices,
  - Example: $\{3;4;5;6;7;8\}$ caracterizes a clique,
- Other representation: 0 0 1 1 1 1 1 1 0 0 (3 4 5 6 7 8) → constant weight code.

Idea

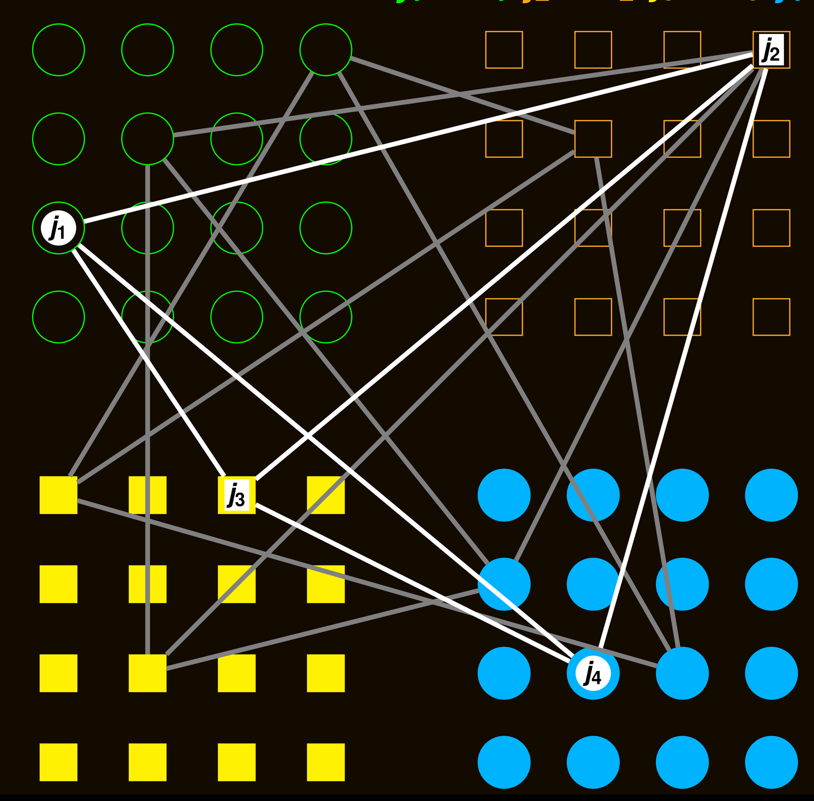
Associative memory ≈ erasure channel associated decoder.

Learning

Learning the $l = 16$-ary message $\underbrace{8}_{j_1 \text{ in } c_1}\ \underbrace{3}_{j_2 \text{ in } c_2}\ \underbrace{2}_{j_3 \text{ in } c_3}\ \underbrace{9}_{j_4 \text{ in } c_4}$,

Locally: thrifty code

Globally: clique code



Static parameters

- n neurons,
- c clusters,
- $l = \frac{n}{c}$ neurons per cluster,
- Memory effect γ,
- W the binary adjacency matrix.

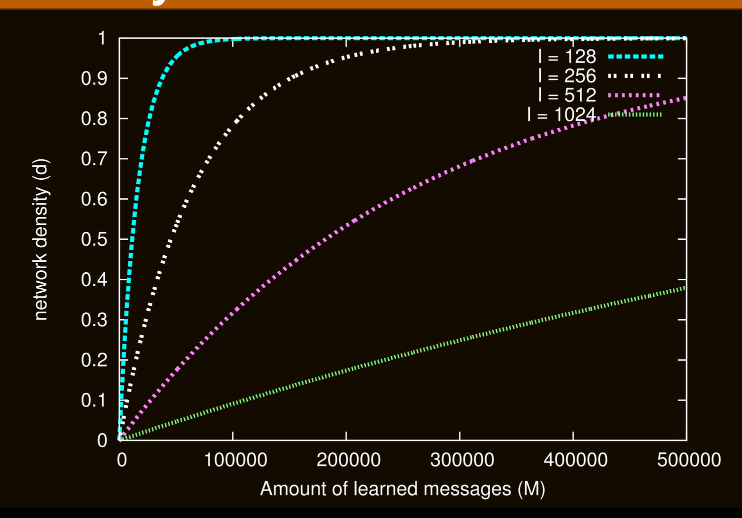
Dynamic parameters

$v^t_{(c_i,l_j)}$ the value of neuron l_j of cluster c_i at iteration t.

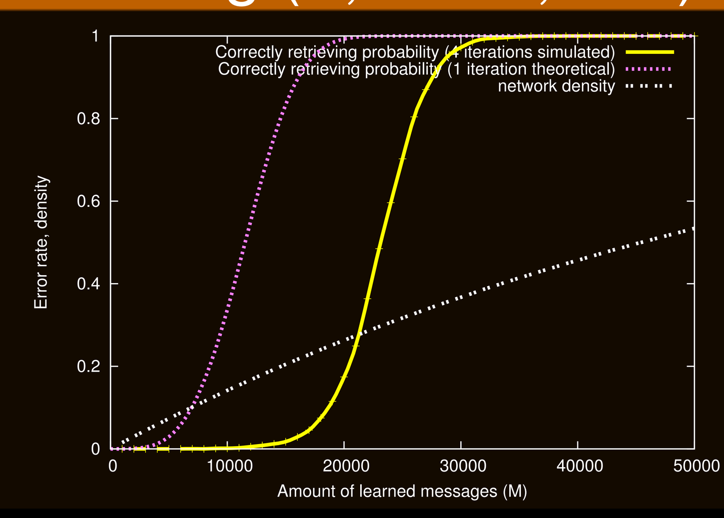
Retrieving equations

$$v^{t+1}_{(c_i,l_j)} = \begin{cases} 1 & \text{if } \gamma v^t_{(c_i,l_j)} + \sum_{c_{i'}=1}^{c} \max_{1\leq l_{j'} \leq l} v^t_{(c_{i'},l_{j'})} \cdot W_{(c_{i'},l_{j'})(c_i,l_j)} = \gamma + c - 1 \\ 0 & \text{otherwise} \end{cases}$$

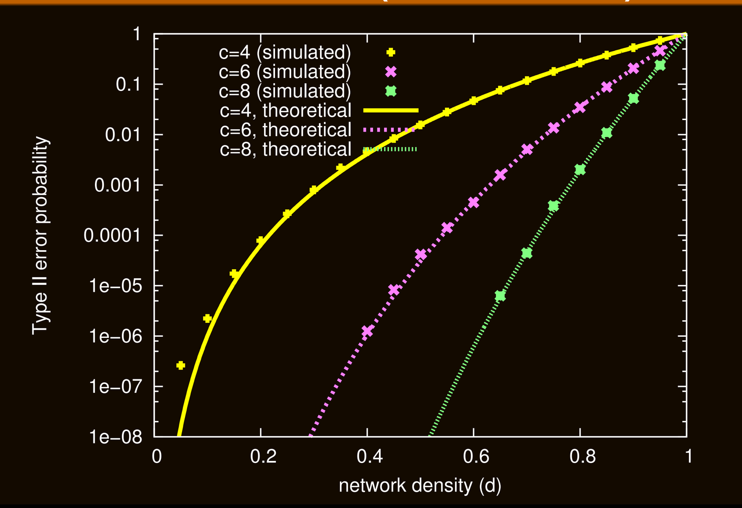
Density



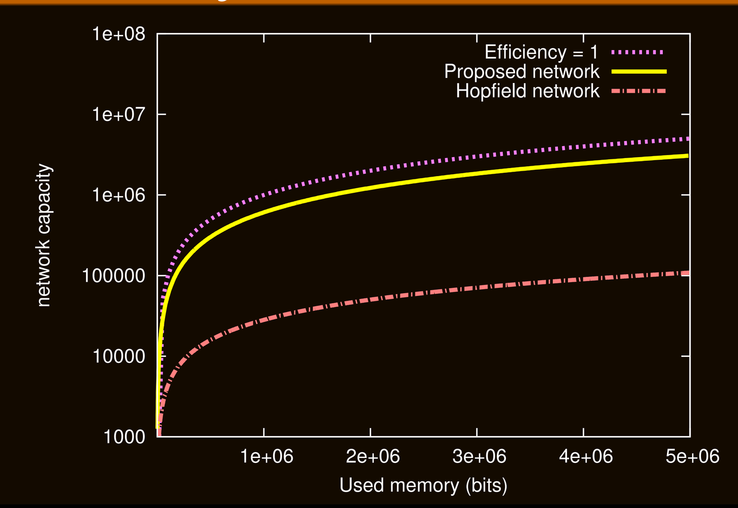
Retrieving ($c, l = 8, 256$)



Classification ($l = 512$)



Efficiency

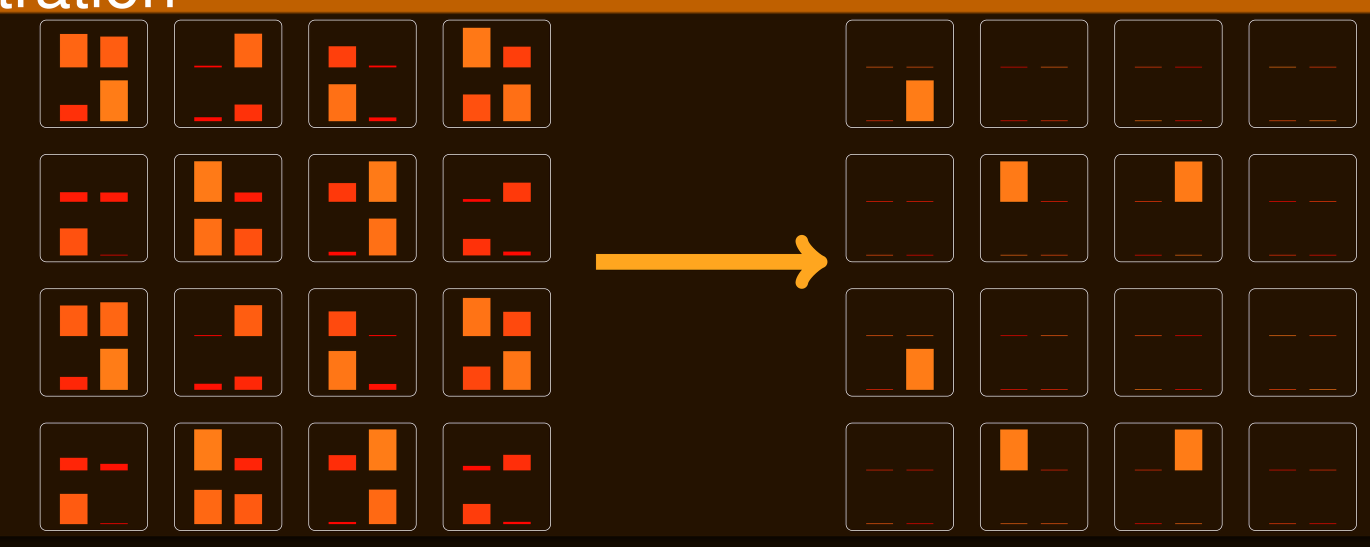


Direction I: Sparse messages

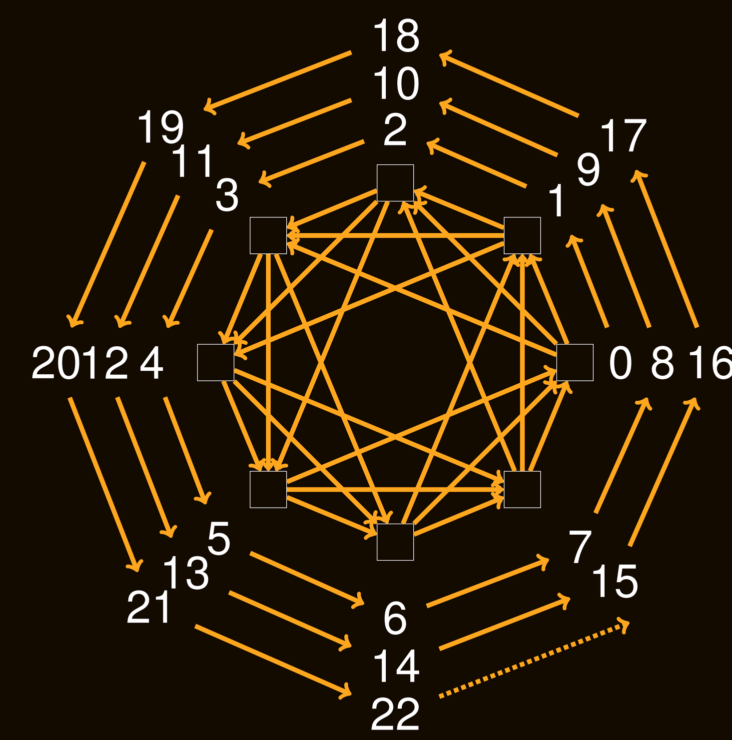
Learn sparse messages

- Idea: use limited number of clusters,
- Retrieving: add a global winner-take-all rule.

Illustration



Direction II: Learning sequences

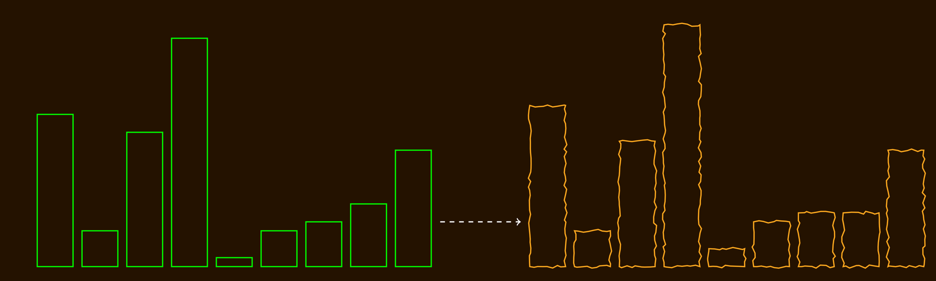


Performance

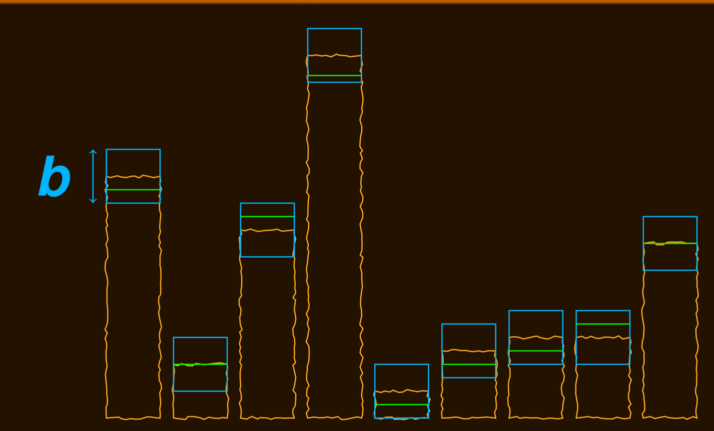
- $c = 50$ clusters,
- $l = 256$ neurons/cluster,
- $L = 1000$ symbols in sequences,
- $m = 1823$ learned sequences,
- $P_e \leq 0.01$.

Direction III: Soft decoding

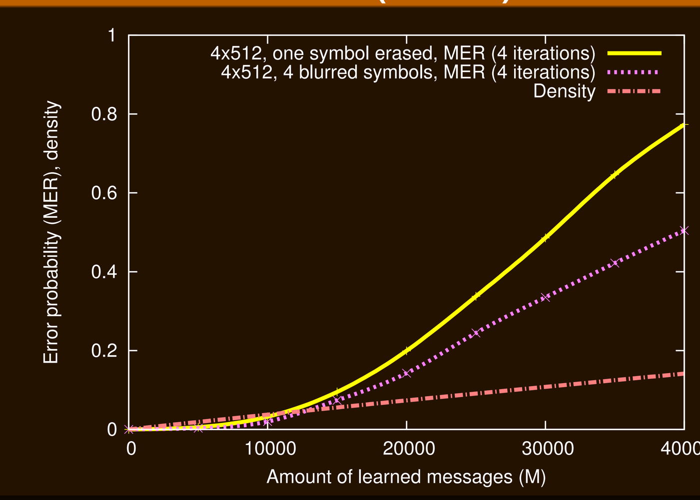
Noise model



Soft decoding



Performance (b=5)



References

- V. Gripon and C. Berrou, “Sparse neural networks with large learning diversity,” *IEEE transactions on neural networks*, vol. 22, pp. 1087 - 1096, Jul. 2011.
- V. Gripon and C. Berrou, “A simple and efficient way to store many messages using neural cliques,” *Proc. of IEEE Symposium on Computational Intelligence, Cognitive Algorithms,* Mind, and Brain, pp. 54 - 58, Apr. 2011.
- V. Gripon, Networks of neural cliques. PhD thesis, Télécom Bretagne, July 2011.