Networks of Neural Cliques

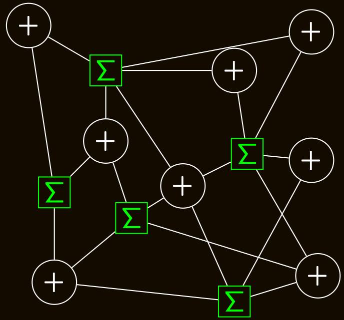
Vincent Gripon Claude Berrou

Télécom Bretagne, Lab-STICC

Jan. 31, 2011

Context: crossbreeding between information theory and associative memories

LDPC decoder



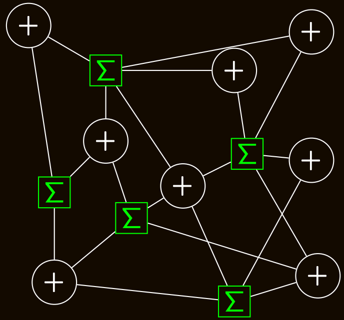
Neocortical "decoder"

Strong analogy

But where are parity relations?

Context: crossbreeding between information theory and associative memories

LDPC decoder



Neocortical "decoder"

Strong analogy

But where are parity relations?

Context: crossbreeding between information theory and associative memories

Analogies

Error correcting decoding		Neural decoding
Fixed point	$\leftrightarrow$	Uniqueness of thought
Minimum distance	$\leftrightarrow$	Separable memories
Huge diversity of combinations	$\leftrightarrow$	Large memory capacity
Low density of graphs	$\leftrightarrow$	Low density of neocortex

Resilience, homeostasis, synchronization, noise impact...

Dissimilarities

Maximum girth	$\leftrightarrow$	Random girth
Bipartite graph	$\leftrightarrow$	Random graph
Chosen codewords	$\leftrightarrow$	Random messages
Linearly dependent codewords	$\leftrightarrow$	Independence

Context: crossbreeding between information theory and associative memories

Analogies

Error correcting decoding		Neural decoding
Fixed point	$\leftrightarrow$	Uniqueness of thought
Minimum distance	$\leftrightarrow$	Separable memories
Huge diversity of combinations	$\leftrightarrow$	Large memory capacity
Low density of graphs	$\leftrightarrow$	Low density of neocortex

Resilience, homeostasis, synchronization, noise impact. . .

Dissimilarities

Maximum girth	$\leftrightarrow$	Random girth
Bipartite graph	$\leftrightarrow$	Random graph
Chosen codewords	$\leftrightarrow$	Random messages
Linearly dependent codewords	$\leftrightarrow$	Independence

Outline

Outline

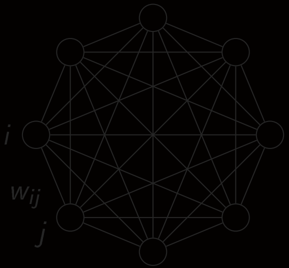
Associative memories, state of the art

Principle

Two operations:

- Learn a message,
- Retrieve a previously learnt message in presence of erasures or errors.

An example: the Hopfield network



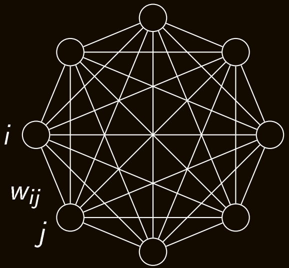
- Learning: M binary messages $\mathbf{d}^m$:
$$w_{ij} = \sum_{m=1, i \neq j}^{M} d_i^m d_j^m,$$
- Retrieving: repeat
$$\forall i, v_i \leftarrow \mathrm{sgn}\Big(\sum_{j \neq i} v_j w_{ij}\Big).$$

Principle

Two operations:

- Learn a message,
- Retrieve a previously learnt message in presence of erasures or errors.

An example: the Hopfield network



- Learning: M binary messages $\mathbf{d}^m$:
$$w_{ij} = \sum_{m=1, i \neq j}^{M} d_i^m d_j^m,$$
- Retrieving: repeat
$$\forall i, v_i \leftarrow \mathrm{sgn}\left(\sum_{j \neq i} v_j w_{ij}\right).$$

Performance and bounds

Hopfield Network

- Diversity (number of learnt messages): $\frac{n}{log(n)}$,
- Capacity: $\frac{n^2}{log(n)}$,
- Binary information stored: $\frac{n(n-1)}{2} log_2(M+1)$,
- $\Rightarrow$ Efficiency $\approx \frac{2}{log(n) log_2(M+1)}$.
- Sensitive connections, negative values, diversity and message length $= f$(network size), no distinction between a message and its opposite...

Theoretical bounds for a Hopfield-like network

- Memory used (connections over P values): $\approx \frac{n^2}{2} log_2(P)$,
- For efficiency $= 1$: $\approx \frac{n}{2} log_2(P)$ messages of length n,
- If length $= k$: $\approx \frac{n^2 log_2(P)}{2k}$.

Performance and bounds

Hopfield Network

- Diversity (number of learnt messages): $\frac{n}{log(n)}$,
- Capacity: $\frac{n^2}{log(n)}$,
- Binary information stored: $\frac{n(n-1)}{2} log_2(M+1)$,
- $\Rightarrow$ Efficiency $\approx \frac{2}{log(n) log_2(M+1)}$.
- Sensitive connections, negative values, diversity and message length $= f$(network size), no distinction between a message and its opposite...

Theoretical bounds for a Hopfield-like network

- Memory used (connections over P values): $\approx \frac{n^2}{2} log_2(P)$,
- For efficiency $= 1$: $\approx \frac{n}{2} log_2(P)$ messages of length n,
- If length $= k$: $\approx \frac{n^2 log_2(P)}{2k}$.

Performance and bounds

Hopfield Network

- Diversity (number of learnt messages): $\frac{n}{log(n)}$,
- Capacity: $\frac{n^2}{log(n)}$,
- Binary information stored: $\frac{n(n-1)}{2} log_2(M+1)$,
- $\Rightarrow$ Efficiency $\approx \frac{2}{log(n) log_2(M+1)}$.
- Sensitive connections, negative values, diversity and message length $= f$(network size), no distinction between a message and its opposite...

Theoretical bounds for a Hopfield-like network

- Memory used (connections over P values): $\approx \frac{n^2}{2} log_2(P)$,
- For efficiency $= 1$: $\approx \frac{n}{2} log_2(P)$ messages of length n,
- If length $= k$: $\approx \frac{n^2 log_2(P)}{2k}$.

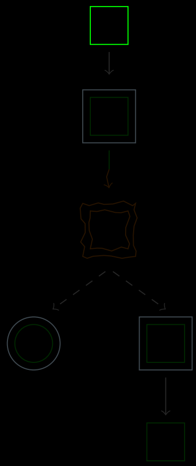
Outline

Systematic coding

- One wants to transmit k bits : 01100..10010,
- He adds $n-k$ bits : 1110..011,
- The added bits are a function of the initial ones,
- The codeword is the concatenation of both: 01100..100101110..011.

Error correcting decoding

- Noise has been added to the codeword,
- The closest known one is chosen,
- A larger distance between codewords leads to better probability of success,
- $\Rightarrow$ minimum distance $d_{\min}$.



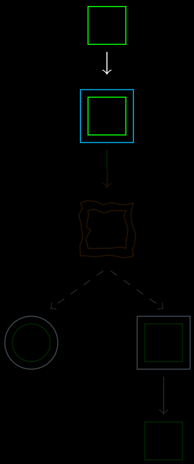
Error correcting codes, classical approach

Systematic coding

- One wants to transmit k bits : 01100..10010,
- He adds $n - k$ bits : 1110..011,
- The added bits are a function of the initial ones,
- The codeword is the concatenation of both: 01100..100101110..011.

Error correcting decoding

- Noise has been added to the codeword,
- The closest known one is chosen,
- A larger distance between codewords leads to better probability of success,
- $\Rightarrow$ minimum distance $d_{\min}$.

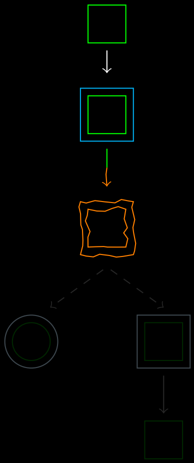


Systematic coding

- One wants to transmit k bits : 01100..10010,
- He adds $n - k$ bits : 1110..011,
- The added bits are a function of the initial ones,
- The codeword is the concatenation of both: 01100..100101110..011.

Error correcting decoding

- Noise has been added to the codeword,
- The closest known one is chosen,
- A larger distance between codewords leads to better probability of success,
- $\Rightarrow$ minimum distance d_{min}.



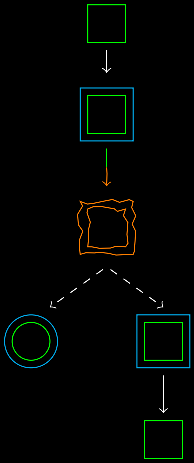
Error correcting codes, classical approach

Systematic coding

- One wants to transmit k bits : 01100..10010,
- He adds $n-k$ bits : 1110..011,
- The added bits are a function of the initial ones,
- The codeword is the concatenation of both: 01100..100101110..011.

Error correcting decoding

- Noise has been added to the codeword,
- The closest known one is chosen,
- A larger distance between codewords leads to better probability of success,
- $\Rightarrow$ minimum distance d_{min}.



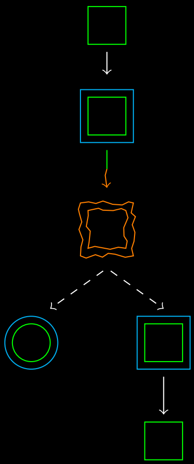
Error correcting codes, classical approach

Systematic coding

- One wants to transmit k bits : 01100..10010,
- He adds $n - k$ bits : 1110..011,
- The added bits are a function of the initial ones,
- The codeword is the concatenation of both: 01100..100101110..011.

Error correcting decoding

- Noise has been added to the codeword,
- The closest known one is chosen,
- A larger distance between codewords leads to better probability of success,
- $\Rightarrow$ minimum distance $d_{\min}$.



Error correcting codes, general approach

Definition

- A code is a set of codewords,
- No need of systematic part, the association message↔codeword is arbitrary.

An example: constant weight codes

- Codewords contain exactly ω 1 and do not share more than α 1 at the same locations,
- These are called constant weight codes with parameters ω, α and with length n: $C^n(\omega, \alpha)$,
- For instance, $C^n(1, 0)$ contains all codewords with a single 1 $(C^3(1, 0) = \{100, 010, 001\})$,
  - Weak $d_{\text{min}} = 2$,
  - but easily decodable, low energy consumption,
  - Can be aggregated as in distributed codes...

Error correcting codes, general approach

Definition

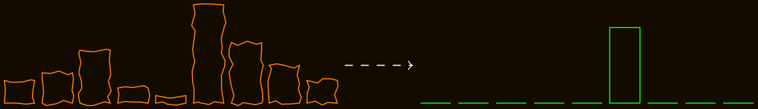
- A code is a set of codewords,
- No need of systematic part, the association message↔codeword is arbitrary.

An example: constant weight codes

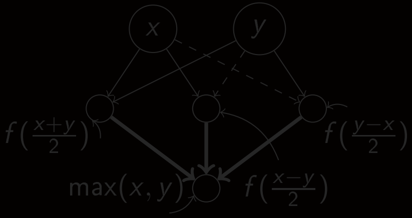
- Codewords contain exactly ω 1 and do not share more than α 1 at the same locations,
- These are called constant weight codes with parameters ω, α and with length n: $C^n(\omega, \alpha)$,
- For instance, $C^n(1, 0)$ contains all codewords with a single 1 ($C^3(1, 0) = \{100, 010, 001\}$),
  - Weak $d_{\text{min}} = 2$,
  - but easily decodable, low energy consumption,
  - Can be aggregated as in distributed codes...

Constant weight codes ($\omega = 1$) and neural decoding

Decoding



Neural decoding

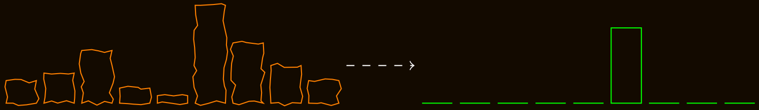


$\Rightarrow$

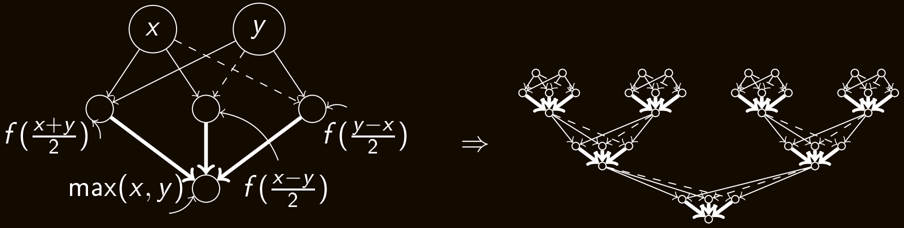
$$f(i) = \max(0, i)$$

Constant weight codes ($\omega = 1$) and neural decoding

Decoding



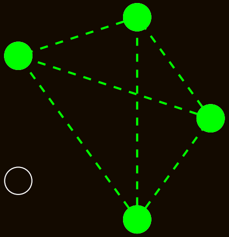
Neural decoding



$$f(i) = \max(0, i)$$

Coding using c-cliques

Example: 4-cliques



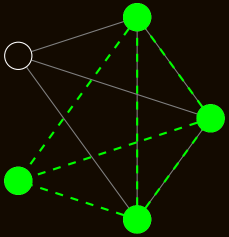
- 1 maximum 4-clique

Cliques and codes

- Minimum distance (edges): $2(c-1) \approx 2c$,
- Rate of the code $\approx \left\lceil \frac{c}{2} \right\rceil \frac{2}{c(c-1)} \approx \frac{1}{c} \Rightarrow$ Merit factor $= 2$.

Coding using c-cliques

Example: 4-cliques



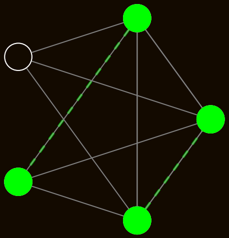
- 2 maximum 4-cliques

Cliques and codes

- Minimum distance (edges): $2(c-1) \approx 2c$,
- Rate of the code $\approx \left\lceil \frac{c}{2} \right\rceil \frac{2}{c(c-1)} \approx \frac{1}{c} \Rightarrow$ Merit factor $= 2$.

Coding using c-cliques

Example: 4-cliques



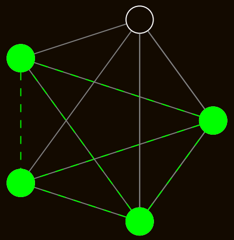
- 2 maximum 4-cliques

Cliques and codes

- Minimum distance (edges): $2(c-1) \approx 2c$,
- Rate of the code $\approx \left\lceil \frac{c}{2} \right\rceil \frac{2}{c(c-1)} \approx \frac{1}{c} \Rightarrow$ Merit factor $= 2$.

Coding using c-cliques

Example: 4-cliques



- 1 maximum 5-clique

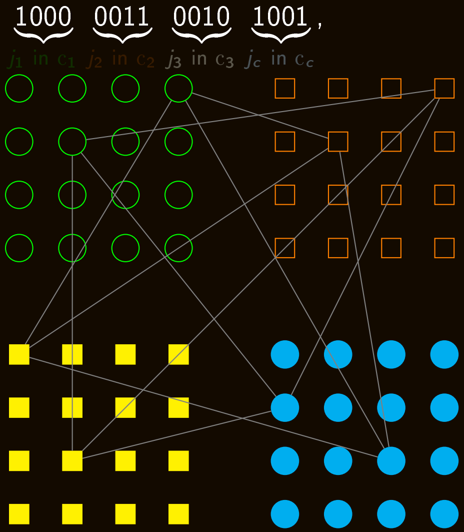
Cliques and codes

- Minimum distance (edges): $2(c-1) \approx 2c$,
- Rate of the code $\approx \left\lceil \frac{c}{2} \right\rceil \frac{2}{c(c-1)} \approx \frac{1}{c} \Rightarrow$ Merit factor $= 2$.

Outline

1. Associative memories and error correcting codes
   - Hopfield network
   - Error correcting codes
2. Neural cliques networks
   - Model
   - Performance
3. Further work
   - More clusters and better performance
   - Correlated entries
4. Conclusion
   - Biological plausibility, applications
   - Openings

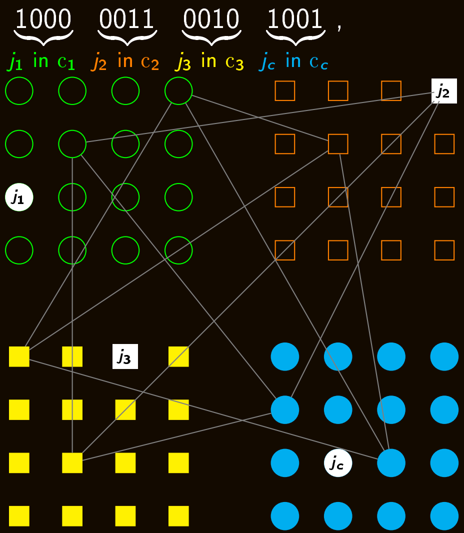
Idea



- n neurons (fanals),
- c clusters,
- κ bits to address a specific cluster,
- $l = \frac{n}{c} = 2^\kappa$ neurons in each cluster,
- $k = c\kappa$ bits in learnt messages,
- Sparsity : A unique fanal is active in each cluster.

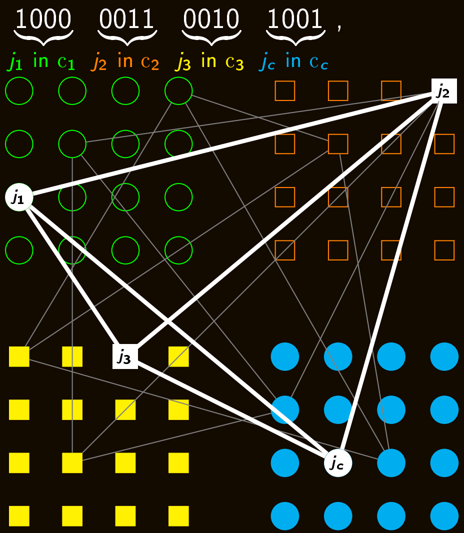
Neural network with sparse coding

Idea



- n neurons (fanals),
- c clusters,
- κ bits to address a specific cluster,
- $l = \frac{n}{c} = 2^{\kappa}$ neurons in each cluster,
- $k = c\kappa$ bits in learnt messages,
- Sparsity : A unique fanal is active in each cluster.

Idea



- n neurons (fanals),
- c clusters,
- κ bits to address a specific cluster,
- $l = \frac{n}{c} = 2^{\kappa}$ neurons in each cluster,
- $k = c\kappa$ bits in learnt messages,
- Sparsity : A unique fanal is active in each cluster.

Equations and bounds

Learning

- Fanal value: $\mu_{bj}^{m} = 1$ if neuron j of cluster b is associated with message m,
- $W_{b_1 j_1 b_2 j_2} = \min\left(\sum_{m=1, b_1 \neq b_2}^{M} \mu_{b_1 j_1}^{m} \mu_{b_2 j_2}^{m}, 1 \right)$

Density

- After M random messages: $d \approx 1 - (1 - \frac{1}{l^2})^M$,
- A density close to 1 corresponds to an overloaded network.

Bounds

$$M_{\max} = \frac{(c-1)n^2}{2c^2 log_2(\frac{n}{c})}$$

Equations and bounds

Learning

- Fanal value: $\mu_{bj}^{m} = 1$ if neuron j of cluster b is associated with message m,
- $W_{b_1 j_1 b_2 j_2} = \min\left(\sum_{m=1, b_1 \neq b_2}^{M} \mu_{b_1 j_1}^{m} \mu_{b_2 j_2}^{m}, 1\right)$

Density

- After M random messages: $d \approx 1 - (1 - \frac{1}{l^2})^M$, $P(\text{connection})$
- A density close to 1 corresponds to an overloaded network.

Bounds

$$M_{\max} = \frac{(c-1)n^2}{2c^2 log_2(\frac{n}{c})}$$

Equations and bounds

Learning

- Fanal value: $\mu_{bj}^{m} = 1$ if neuron j of cluster b is associated with message m,
- $W_{b_1 j_1 b_2 j_2} = \min(\sum_{m=1, b_1 \neq b_2}^{M} \mu_{b_1 j_1}^{m} \mu_{b_2 j_2}^{m}, 1)$

Density

- After M random messages: $d \approx 1 - (1 - \frac{1}{l^2})^M$, $P(\text{no connection}|\text{M})$
- A density close to 1 corresponds to an overloaded network.

Bounds

$$M_{\max} = \frac{(c-1)n^2}{2c^2 log_2(\frac{n}{c})}$$

Equations and bounds

Learning

- Fanal value: $\mu_{bj}^{m} = 1$ if neuron j of cluster b is associated with message m,
- $W_{b_1 j_1 b_2 j_2} = \min\left(\sum_{m=1, b_1 \neq b_2}^{M} \mu_{b_1 j_1}^{m} \mu_{b_2 j_2}^{m}, 1\right)$

Density

- After M random messages: $d \approx 1 - (1 - \frac{1}{l^2})^M$,
- A density close to 1 corresponds to an overloaded network.

Bounds

$$M_{\max} = \frac{(c-1)n^2}{2c^2 log_2(\frac{n}{c})}$$

Equations and bounds

Learning

- Fanal value: $\mu_{bj}^{m} = 1$ if neuron j of cluster b is associated with message m,
- $W_{b_1 j_1 b_2 j_2} = \min(\sum_{m=1, b_1 \neq b_2}^{M} \mu_{b_1 j_1}^{m} \mu_{b_2 j_2}^{m}, 1)$

Density

- After M random messages: $d \approx 1 - (1 - \frac{1}{l^2})^M$,
- A density close to 1 corresponds to an overloaded network.

Bounds

$$M_{\max} = \frac{(c-1)n^2}{2c^2 log_2(\frac{n}{c})}$$

Equations and bounds

Learning

- Fanal value: $\mu_{bj}^{m} = 1$ if neuron j of cluster b is associated with message m,
- $W_{b_1 j_1 b_2 j_2} = \min(\sum_{m=1, b_1 \neq b_2}^{M} \mu_{b_1 j_1}^{m} \mu_{b_2 j_2}^{m}, 1)$

Density

- After M random messages: $d \approx 1 - (1 - \frac{1}{l^2})^M$,
- A density close to 1 corresponds to an overloaded network.

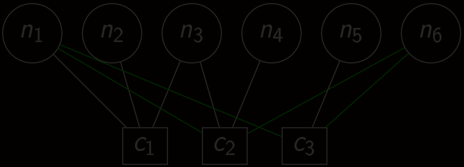
Bounds

$$M_{\text{max}} = \frac{(c-1)n^2}{2c^2 log_2(\frac{n}{c})}$$

Iterative process

- Globally, using neurons as adders (clique code):
  $$\forall b_1, j_1\, v_{b_1 j_1} \leftarrow \sum_{j_2, b_2 \neq b_1} W_{b_1 j_1 b_2 j_2} \mu_{b_2 j_2} + \gamma \mu_{b_1 j_1},$$
- Locally, winner-take-all (constant weight code):
  - $\forall b, S^b_{\max} = \max_j v_{bj},$
  - $\forall b, j, \mu_{bj} \leftarrow \begin{cases} 1 & \text{if } v_{bj} = S^b_{\max} \text{ and } S^b_{\max} \geq \sigma \\ 0 & \text{otherwise} \end{cases}.$

Correlation *or not* correlation



Retrieving process

Iterative process

- Globally, using neurons as adders (clique code):
  $$\forall b_1, j_1\, v_{b_1 j_1} \leftarrow \sum_{j_2, b_2 \neq b_1} W_{b_1 j_1 b_2 j_2} \mu_{b_2 j_2} + \gamma \mu_{b_1 j_1},$$
- Locally, winner-take-all (constant weight code):
  - $\forall b, S^b_{\max} = \max_j v_{bj},$
  - $\forall b, j, \mu_{bj} \leftarrow \begin{cases} 1 & \text{if } v_{bj} = S^b_{\max} \text{ and } S^b_{\max} \geq \sigma \\ 0 & \text{otherwise} \end{cases}.$

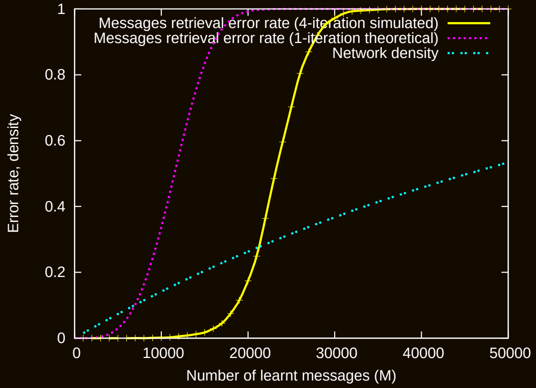
Correlation *or not* correlation

Outline

1. Associative memories and error correcting codes
   - Hopfield network
   - Error correcting codes
2. Neural cliques networks
   - Model
   - Performance
3. Further work
   - More clusters and better performance
   - Correlated entries
4. Conclusion
   - Biological plausibility, applications
   - Openings

Performance

Associative memories

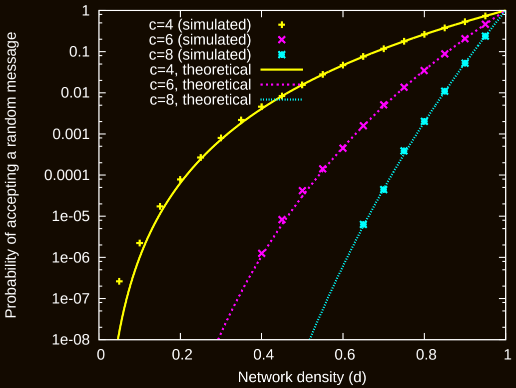


Error probability while retrieving learnt messages with 4 out of 8 clusters with no provided information and with $l = 256$.

- Gain in comparison with the Hopfield network: 130 in diversity, 12 in capacity, and 11 in efficiency ($4.9\% \rightarrow 53.3\%$). Performance depends mainly of l.

Performances (suite)

Classification

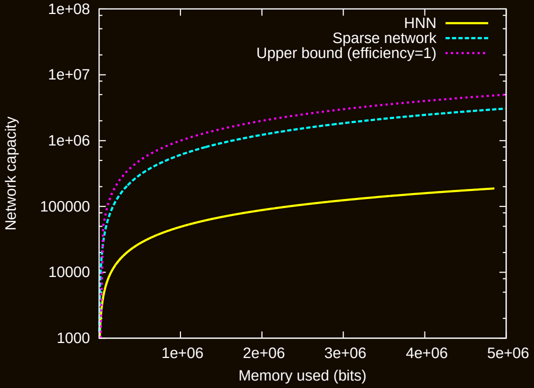


Error probability of second kind (probability to accept a non learnt message) with $l = 512$.

- No first kind error (a learnt message is always recognized),
- A very good second kind error, depending on c.

Comparison in capacity

Capacity



Capacity of Hopfield neural networks and sparse coding neural networks in function of the amount of stored information.

- Near the optimum,
- Huge gain compared to the Hopfield model.

Outline

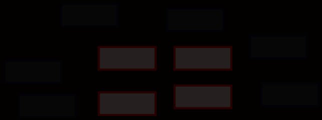
Goal

Increase performance keeping l constant.

1,2,3. . . and 4

- Messages of length $k \leq n$,
- A unique fanal in each cluster,
- Sparse network. . .
- . . . Sparse messages.

Illustration



To a fourth level of sparsity

Goal

Increase performance keeping l constant.

1,2,3. . . and 4

- Messages of length $k \leq n$,
- A unique fanal in each cluster,
- Sparse network. . .
- . . . Sparse messages.

Illustration

To a fourth level of sparsity

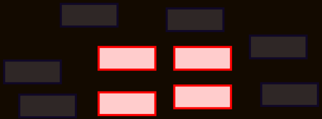
Goal

Increase performance keeping l constant.

1,2,3. . . and 4

- Messages of length $k \leq n$,
- A unique fanal in each cluster,
- Sparse network. . .
- . . . Sparse messages.

Illustration



To a fourth level of sparsity

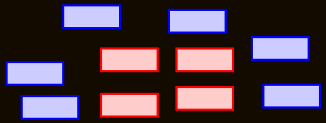
Goal

Increase performance keeping l constant.

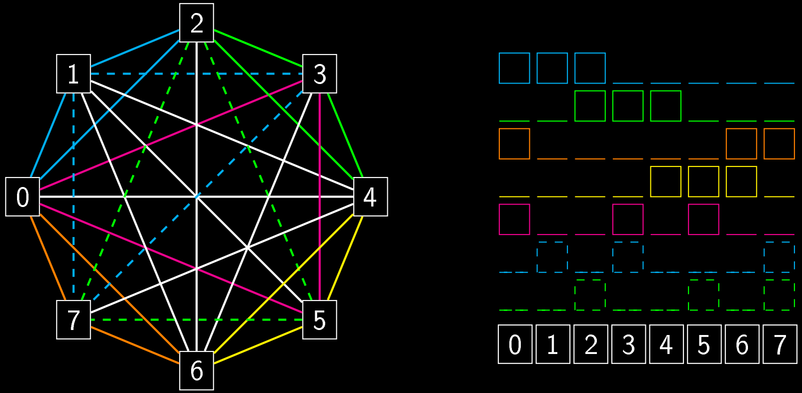
1,2,3...and 4

- Messages of length $k \leq n$,
- A unique fanal in each cluster,
- Sparse network...
- ...Sparse messages.

Illustration



Illustration



Sub-networks

Controlled sparsity

- To avoid epilepsy, sparsity must be guided,
- For instance, force messages to address clusters with some properties,
- $\Rightarrow$ sub-networks are a code $C^{cc'}(c,1)$,
- Interest: the global density is directly given by local densities.

Diversity

- The fully addressed network learns up to $\approx \alpha \left(\frac{n}{c}\right)^2$ messages,
- Considering that the number of clusters has been increased by a factor c':
  - We have c'^2 sub-networks of c clusters,
  - Each one learns up to $\approx \alpha \left(\frac{n}{cc'}\right)^2$ messages,
- Conclusion: $\approx \alpha \left(\frac{n}{c}\right)^2$.

Sub-networks

Controlled sparsity

- To avoid epilepsy, sparsity must be guided,
- For instance, force messages to address clusters with some properties,
- $\Rightarrow$ sub-networks are a code $C^{cc'}(c,1)$,
- Interest: the global density is directly given by local densities.

Diversity

- The fully addressed network learns up to $\approx \alpha \left(\frac{n}{c}\right)^2$ messages,
- Considering that the number of clusters has been increased by a factor c':
  - We have c'^2 sub-networks of c clusters,
  - Each one learns up to $\approx \alpha \left(\frac{n}{cc'}\right)^2$ messages,
- Conclusion: $\approx \alpha \left(\frac{n}{c}\right)^2$.

Retrieving process and other sparsity controls

Coincidences

- The retrieving process is assured this way:
  - Each couple of cluster is colored by a unique propagation time,
  - Neurons switch activated only if there is temporal coincidence,
  - The threshold σ controls the epilepsy.

Other sparsity controls

- One can allow $\alpha \geq 1$ recoverings,
- The number of sub-networks becomes $c'^{\alpha+1}$,
- On the other hand, density is more complex to estimate,
- It is still an open question to find the best compromise.

Retrieving process and other sparsity controls

Coincidences

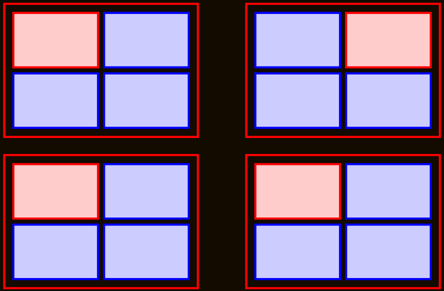
- The retrieving process is assured this way:
  - Each couple of cluster is colored by a unique propagation time,
  - Neurons switch activated only if there is temporal coincidence,
  - The threshold σ controls the epilepsy.

Other sparsity controls

- One can allow $\alpha \geq 1$ recoverings,
- The number of sub-networks becomes $c'^{\alpha+1}$,
- On the other hand, density is more complex to estimate,
- It is still an open question to find the best compromise.

Another approach: a fractal sparsity

winner-take-all between clusters



Performance and remarks

- Performance is the same as before in comparable states,
- Performance is similar in case of partial erasures,
- Messages are coding their own physical location.

Outline

1. Associative memories and error correcting codes
   - Hopfield network
   - Error correcting codes
2. Neural cliques networks
   - Model
   - Performance
3. Further work
   - More clusters and better performance
   - Correlated entries
4. Conclusion
   - Biological plausibility, applications
   - Openings

Correlated entries

Problem

The learning process produces artificial correlation but suffers from that of messages.

Correlations

- There are two types of correlation:
  - Intrinsic: *brain* and *train* are learnt → ambiguity on ∗*rain*.
  - Caused by the model:
    - If the network learns *jam*, *jet* and *cat*, it also learns *jat*...
    - Idea: adding hidden signatures to learnt messages:
    - For instance: "jama", "jetb" and "catc".

Example

- Learning the French words with 6 letters,
- Performance in retrieving process increase from 30% to 80%.

Correlated entries

Problem

The learning process produces artificial correlation but suffers from that of messages.

Correlations

- There are two types of correlation:
  - Intrinsic: *brain* and *train* are learnt → ambiguity on $*$*rain*.
  - Caused by the model:
    - If the network learns *jam*, *jet* and *cat*, it also learns *jat*...
    - Idea: adding hidden signatures to learnt messages:
    - For instance: "jama", "jetb" and "catc".

Example

- Learning the French words with 6 letters,
- Performance in retrieving process increase from 30% to 80%.

Correlated entries

Problem

The learning process produces artificial correlation but suffers from that of messages.

Correlations

- There are two types of correlation:
  - Intrinsic: *brain* and *train* are learnt $\rightarrow$ ambiguity on $*$*rain*.
  - Caused by the model:
    - If the network learns *jam*, *jet* and *cat*, it also learns *jat*...
    - Idea: adding hidden signatures to learnt messages:
    - For instance: "jama", "jetb" and "catc".

Example

- Learning the French words with 6 letters,
- Performance in retrieving process increase from 30% to 80%.

Correlated entries

Problem

The learning process produces artificial correlation but suffers from that of messages.

Correlations

- There are two types of correlation:
  - Intrinsic: *brain* and *train* are learnt $\rightarrow$ ambiguity on $*$*rain*.
  - Caused by the model:
    - If the network learns *jam*, *jet* and *cat*, it also learns *jat*...
    - Idea: adding hidden signatures to learnt messages:
    - For instance: "jama", "jetb" and "catc".

Example

- Learning the French words with 6 letters,
- Performance in retrieving process increase from 30% to 80%.

Outline

Plausibility, applications

Biological plausibility

- Positive neurons, binary connections ⇒ strong resilience (≠ Hopfield),
- Low global density, strong local interactions (small world philosophy),
- Biologically plausible operations: sum and winner-take-all,
- Partition into clusters,
- Neuron specialization. . .

Applications

- Associative memories,
- Classification (*go no-go*),
- Sort,
- Associating pieces of information: independent messages share the same material.

Plausibility, applications

Biological plausibility

- Positive neurons, binary connections $\Rightarrow$ strong resilience ($\neq$ Hopfield),
- Low global density, strong local interactions (small world philosophy),
- Biologically plausible operations: sum and winner-take-all,
- Partition into clusters,
- Neuron specialization...

Applications

- Associative memories,
- Classification (*go no-go*),
- Sort,
- Associating pieces of information: independent messages share the same material.

Outline

What is already done

- Sparse coding:
  - Important gains on the diversity and the capacity,
  - Increase in the network efficiency,
- Distributed coding: more learnt messages than neurons,
- Biological plausibility,
- Perspectives in the design of intelligent machines,
- Immediate applications: associative memories and classification.

Ongoing work

- Noise influence, retrieving blurred messages,
- Partial erasures of clusters,
- Networks of networks.

Results and openings

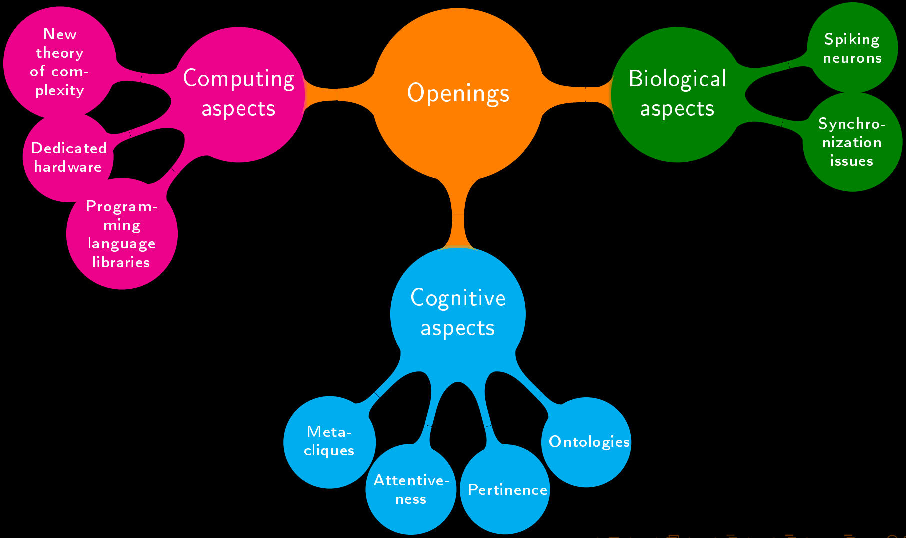
What is already done

- Sparse coding:
  - Important gains on the diversity and the capacity,
  - Increase in the network efficiency,
- Distributed coding: more learnt messages than neurons,
- Biological plausibility,
- Perspectives in the design of intelligent machines,
- Immediate applications: associative memories and classification.

Ongoing work

- Noise influence, retrieving blurred messages,
- Partial erasures of clusters,
- Networks of networks.

Openings


Openings
Computing aspects
New theory of com- plexity
Dedicated hardware
Program- ming language libraries
Biological aspects
Spiking neurons
Synchro- nization issues
Cognitive aspects
Meta- cliques
Attentive- ness
Pertinence
Ontologies

End of the lecture

Thank you for listening, I am at your disposal if you have any question.