UNIT-4

Chapter 10

Attractor Neural Networks

Neural Networks: A Classroom Approach
Satish Kumar

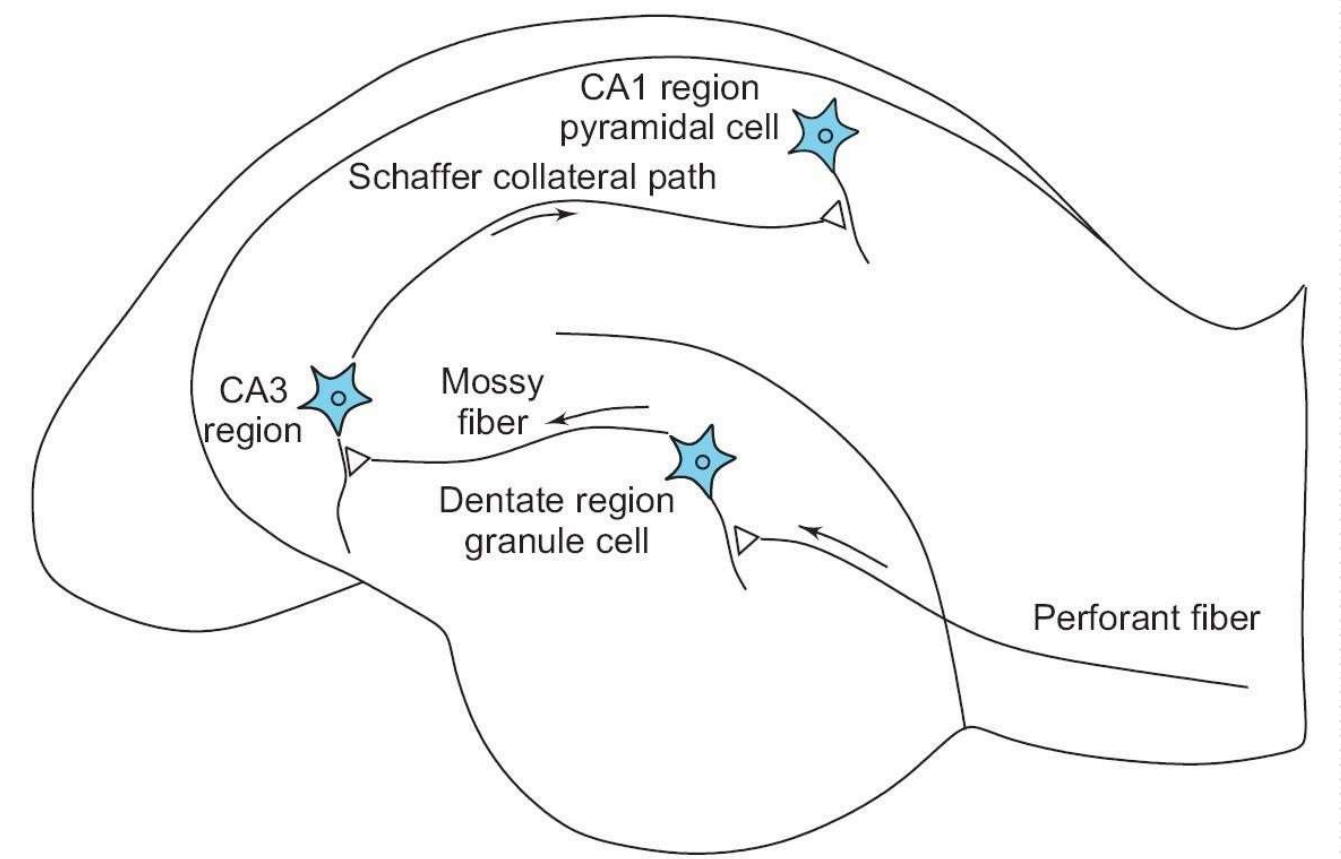
Human Memory is Associative

- We recall facts and information by invoking a chain of associations within a complex web of embedded knowledge.
- Chains of thought can be generated through conscious or unconscious reasoning
  - Fragrance of a flower evokes a pleasant memory of the past,
  - Name of a friend suddenly create a surge of emotions
- Entire experiences are recalled in complete richness of fact
  - on a simple cue
  - within fractions of a second

An Unanswered Question

- How do we accomplish such complex computations so effortlessly?
- Largely motivates the research currently being conducted on associative memory

Hippocampal Pathways Involved in Long-term Potentiation (LTP)



Three Major Pathways

- *Perforant fiber pathway* runs to granule cells in the *dentate region*.
- Granule cells in turn send axons that form the *mossy fiber pathway* to cells in the CA3 region.
- CA3 cells project axons onto dendrites of pyramidal cells in the CA1 region (*Schaffer collateral path*)

LTP

- Extensive research reveals that a brief high frequency stimulus train to any one of these three pathways leads to an enduring increase in the excitatory postsynaptic potential (EPSP) of CA1 pyramidal neurons.
- The strength of the EPSP increases in response to the stimulus train, and this change can last for anything from hours to weeks.

Three Important Properties of CA1 LTP

- Cooperativity:
  - CA1 LTP cannot be produced by activating only one fiber.
  - A minimum number of fibers must be activated together to achieve LTP.
- Associativity:
  - When both strong and weak excitatory inputs arrive in the same dendritic region of a pyramidal cell, the weak input gets potentiated if it is activated in association with the strong one.
- Specificity:
  - LTP is specific to the dendrites where it is produced.

Hebb's Postulate

- "When an axon of cell A excite[s] cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells so that A's efficacy as one of the cells firing B is increased."

Attractor Neural Network Associative Memory

- Hebbian learning principle implies that synaptic dynamics are governed by a conjunction of pre and postsynaptic activities.
- Leads to powerful associative memories that relate or *associate* one concept or idea with another

Neural Network Dynamics

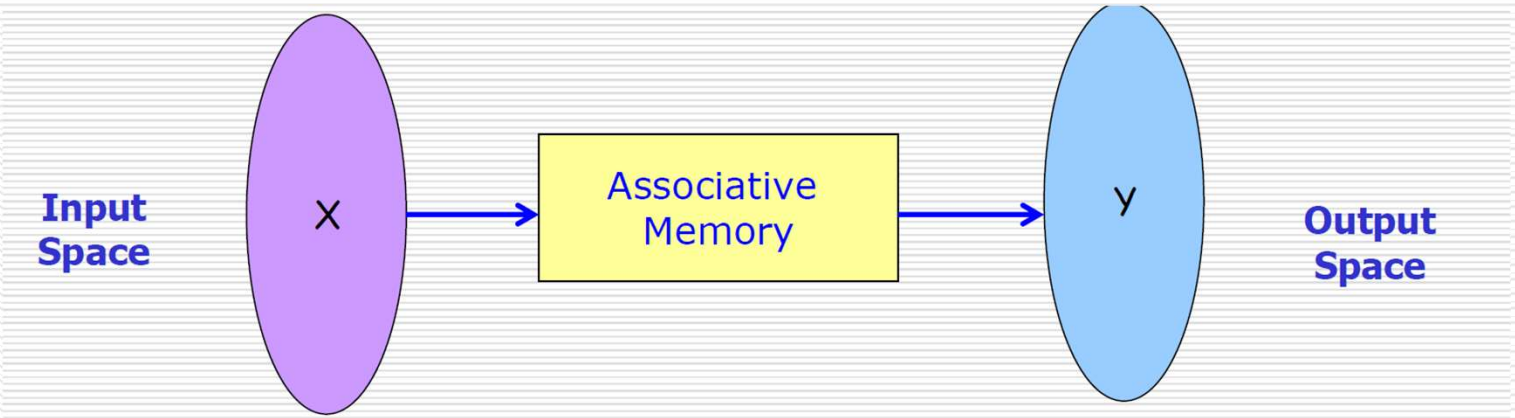
- Vector field governed by
  - structure of connections
  - nature of the signal function
- Network states evolve in time until the local minimum is reached, after which the states stop evolving further

Broad Objective

- Point of local minimum at which the network state stabilizes is to be interpreted as a memory
- Given a cue or an initial input vector, the initial state can be represented by a point in the energy surface
- Network state evolves in time till the system hits the energy minimum
- Final state is the recalled memory vector

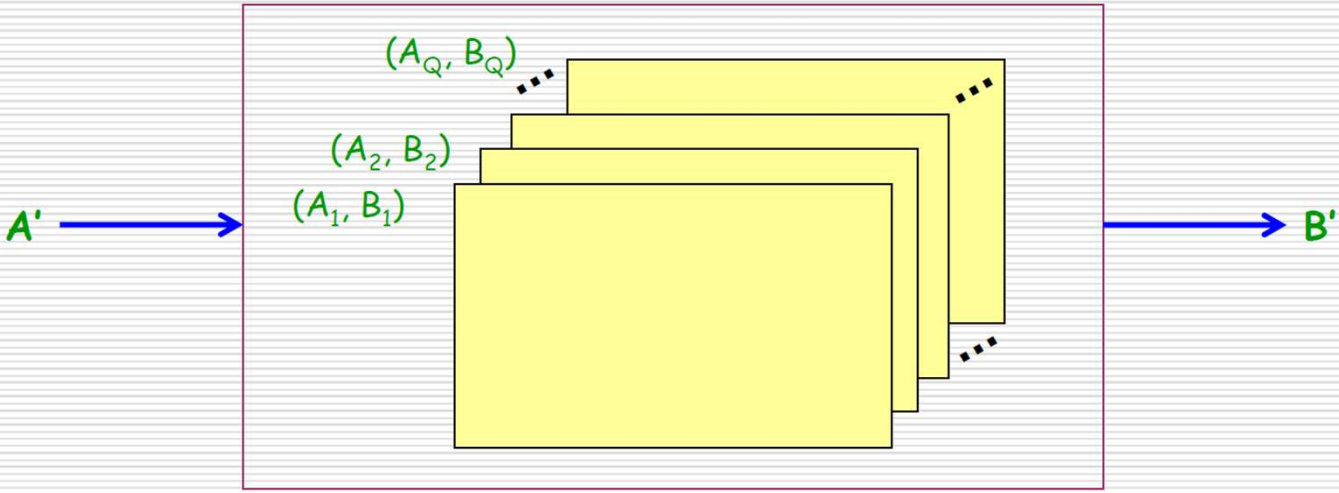
The Associative Memory Model

- Mapping from unknown domain points to known range points



- The memory learns an underlying association from a training data set

Matrix Associative Memory

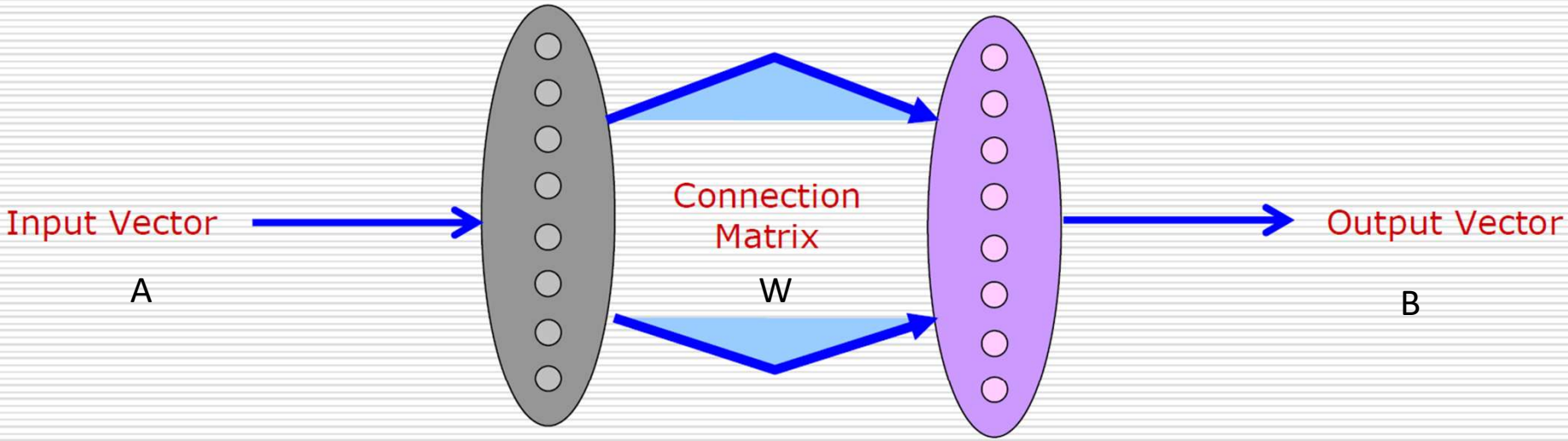


Encodes associations $\{A_i, B_i\}_{i=1}^{Q}$ $A_i \in B^n$, $B_i \in B^m$ into a connection matrix

Hetero- and Auto-associative Memory

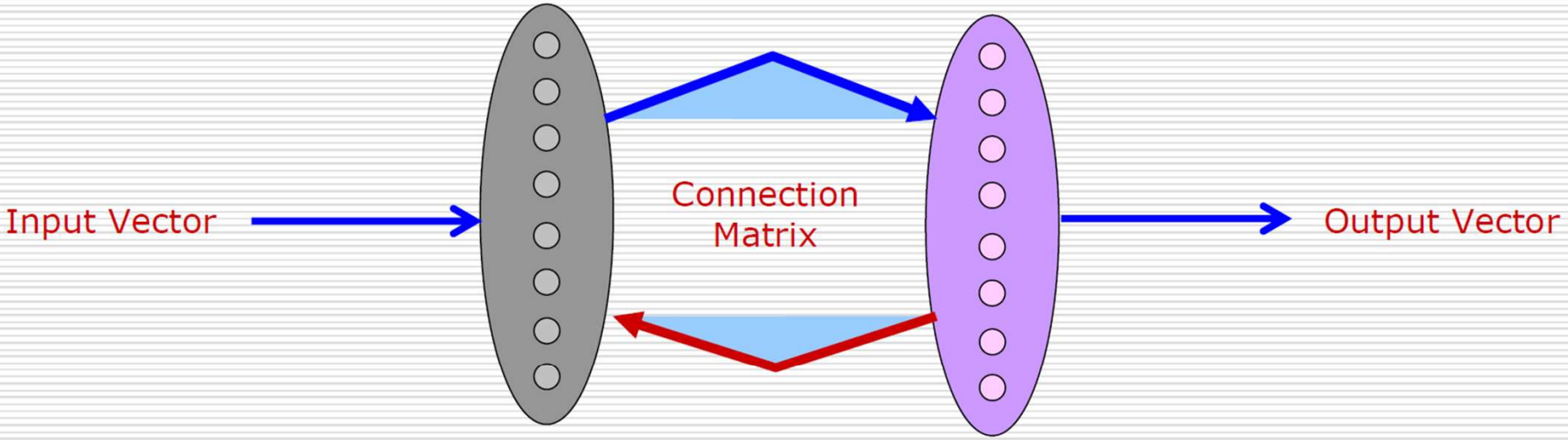
- When A_i, B_i are in different spaces $A_i \in B^n$, $B_i \in B^m$ the memory is heteroassociative
- When A_i, B_i are in the same space $A_i, B_i \in B^n$ the memory is autoassociative

Two Layer Feedforward Neural Network Associative Memory



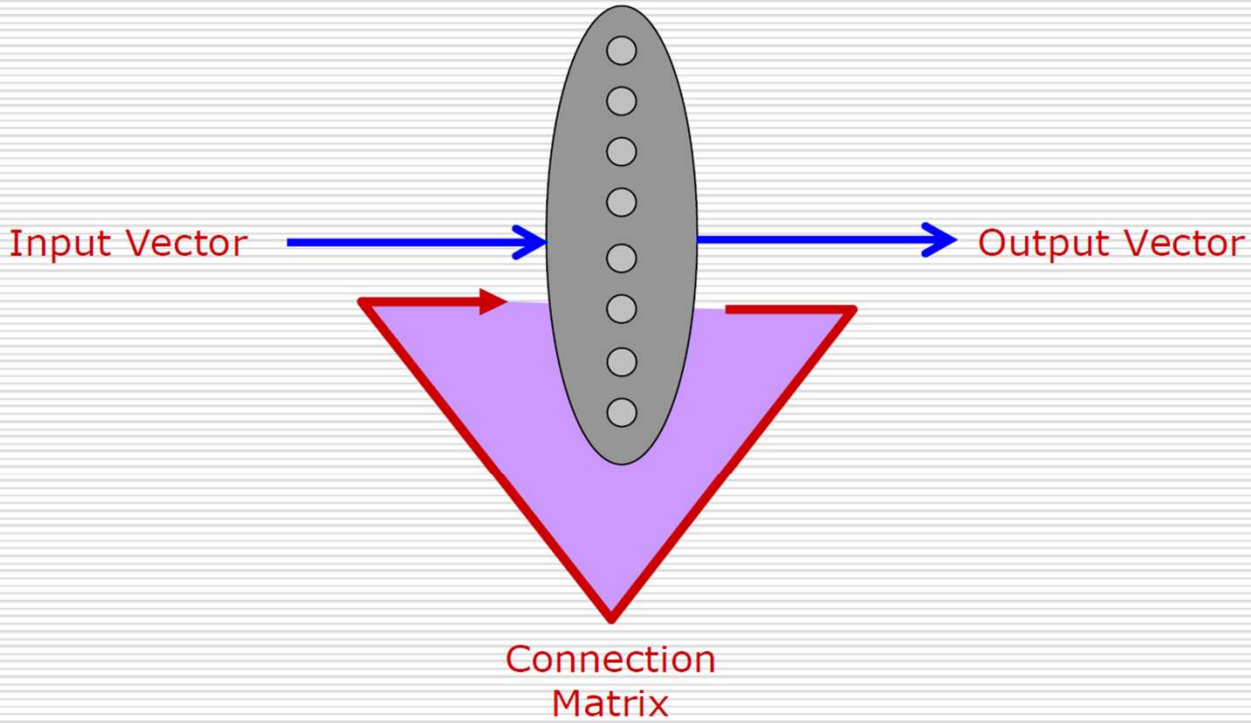
- The two layers are assumed to be connected through n*m connection matrix W.
- If all neurons are assumed linear, then the signal vector $B=A^TW$

Two Layer Feedback Neural Network Associative Memory



- Reverse set of connections allows signals from output to flowback to input layer
- Introducing dynamics in the network operation

One Layer Feedback Neural Network Associative Memory



- A single layer of neurons feeds back to itself generating auto associative architecture

Hebb Encoding Scheme

- Consider two layers F_x, F_y of n,m neurons with connection matrix W
- Weight matrix is constructed using Generalized Hebb Rule
  - modifies synapses in accordance with the product of the presynaptic activity with the activity of the postsynaptic neuron

$$\delta w_{ij} = \eta a_i b_j$$

$$\mathbf{W} = \begin{pmatrix} a_1 b_1 & \cdots & a_1 b_m \\ \vdots & \ddots & \vdots \\ a_n b_1 & \cdots & a_n b_m \end{pmatrix} = A B^T$$

Linear Associative Memory

- Assume that a single association (A,B) has been encoded into the connection matrix W, using the outer product

$$\mathbf{W} = A B^T$$

- Presentation of the vector A yields perfect recall

$$(B')^T = A^T \mathbf{W} = A^T A B^T = \|A\|^2 B^T$$

Linear Associative Memory

- Encoding more than one association…
  - superpose the individual association matrices.
- For Q associations
  - $\{A_i, B_i\}$, $A_i \in \mathbf{B}^n$, $B_i \in \mathbf{B}^m$

$$\mathbf{W} = \sum_{k=1}^{Q} A_k B_k^T$$

Orthogonal Linear Associative Memory (OLAM)

- A special (and very restrictive) case arises if the vectors A_k are *orthonormal*
  - orthogonal to one another
  - normalized to unity magnitude

$$
\begin{aligned}
(B')^T &= A_i^T \mathbf{W} \\
&= A_i^T \sum_{k=1}^{Q} A_k B_k^T \\
&= A_i^T A_i B_i^T + \sum_{k \neq i} A_i^T A_k B_k^T \\
&= \|A_i\|^2 B_i^T + 0 \\
&= B_i^T
\end{aligned}
$$

Note- Orthonormality implies: $A_i^T A_i = 1$ and $A_i^T A_j = 0$ $i \neq j$

Orthogonal Linear (One-shot) Associative Memories: Example

- Encode the vector associations
  - $A_1 = (1\ 0\ 0\ 0)$ $B_1 = (1\ 2\ 3)$
  - $A_2 = (0\ 1\ 0\ 0)$ $B_2 = (-2\ 3\ 1)$
  - $A_3 = (0\ 0\ 1\ 0)$ $B_3 = (4\ 0\ 4)$

$$\mathbf{W} = A_1 B_1^T + A_2 B_2^T + A_3 B_3^T$$

$$= \begin{pmatrix} 1 & 2 & 3 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 \\ -2 & 3 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 4 & 0 & 4 \\ 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 \\ -2 & 3 & 1 \\ 4 & 0 & 4 \\ 0 & 0 & 0 \end{pmatrix}$$

Orthogonal Linear (One-shot) Associative Memories: Example

- It is easy to verify that A_1, A_2, A_3 are fixed points of the system

$$A_1^T \mathbf{W} = (1\ 0\ 0\ 0) \begin{pmatrix} 1 & 2 & 3 \\ -2 & 3 & 1 \\ 4 & 0 & 4 \\ 0 & 0 & 0 \end{pmatrix} = (1\ 2\ 3) = B_1^T$$

- Verify for A_2, A_3

Orthogonal Linear (One-shot) Associative Memories: Example

- The addition of input vectors yields the addition of corresponding associants

$$(A_1 + A_2)^T \mathbf{W} = (1\ 1\ 0\ 0) \begin{pmatrix} 1 & 2 & 3 \\ -2 & 3 & 1 \\ 4 & 0 & 4 \\ 0 & 0 & 0 \end{pmatrix}$$

$$= (-1\ 5\ 4) = B^T = (B_1 + B_2)^T$$

Orthogonal Linear (One-shot) Associative Memories: Example

- Input vectors close to a memory recall a vector close to its associant

$$A^T\mathbf{W} = (.9\ .1\ .1\ .1)\begin{pmatrix} 1 & 2 & 3 \\ -2 & 3 & 1 \\ 4 & 0 & 4 \\ 0 & 0 & 0 \end{pmatrix}$$

$$= (1.1\ \ 2.1\ \ 3.2) = B^T \simeq B_1^T$$

More Questions…

- What is the nature of the dynamics that underlies these models?
- How do we analyze the stability of these models?
- What are the important pathologies that arise in Hebb encoded matrix memories?
- What are the capacities of these memories?

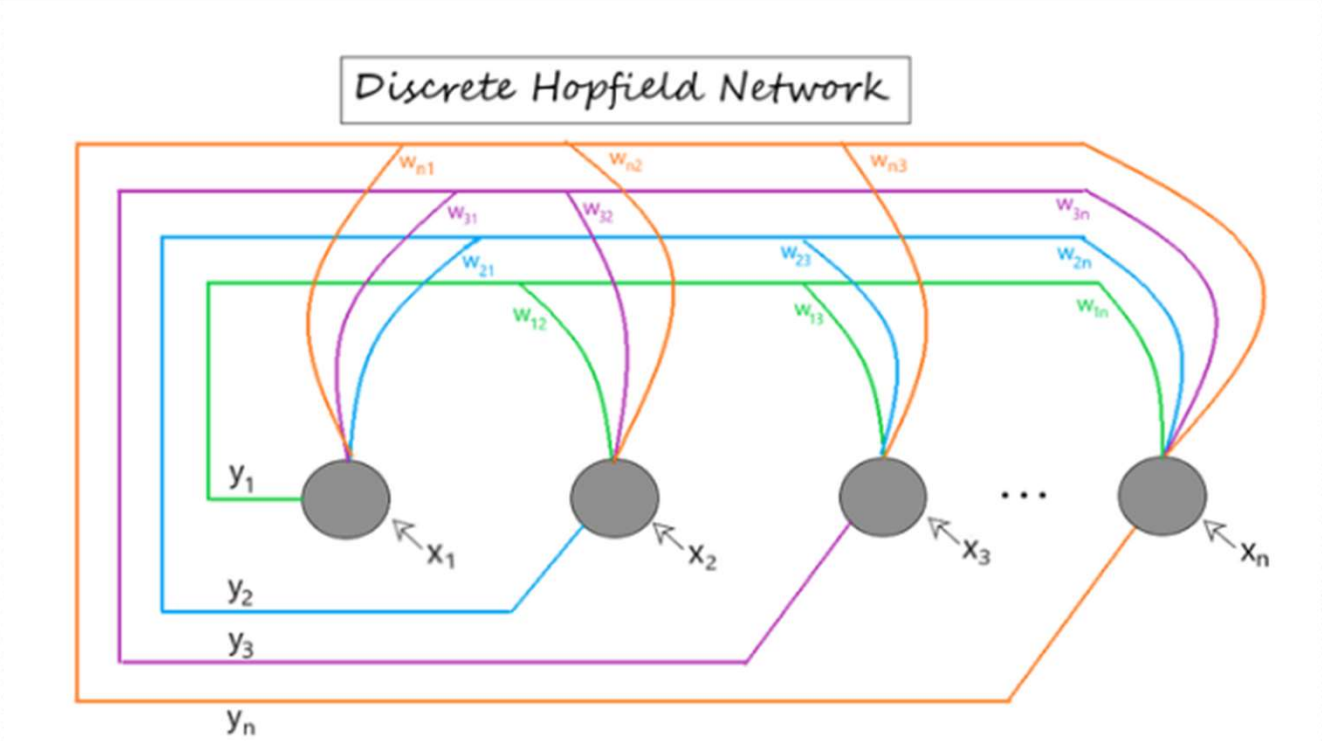
Hopfield Network

- Popularized by Nobel Laureate John Hopfield
- Basic idea: **It is a Recurrent or feedback Network.**
  - Map stable states to correspond to certain desired memory vectors or to solutions of an optimization problem
  - Then the time evolution of dynamics leads to a stable state where the outputs of the network correspond to the answer

Fundamental Issues

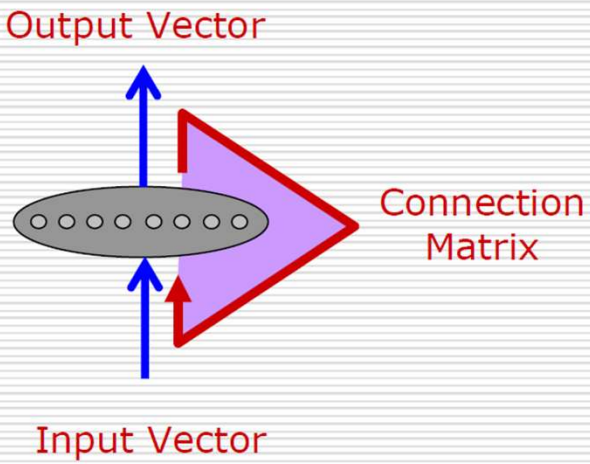
- How do we "program" the solutions of the problem into stable states of the network?
- How do we ensure that the feedback system designed is stable?

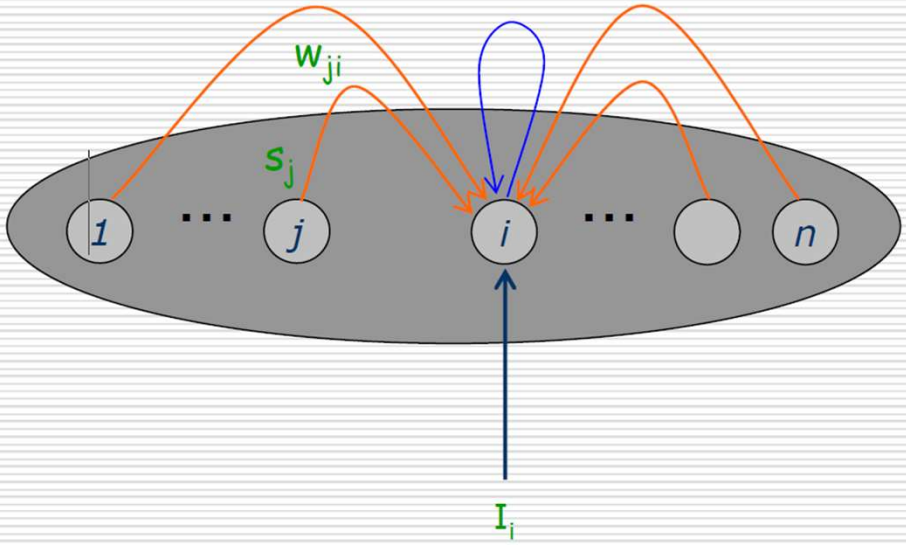
Discrete Hopfield Network Architecture



https://www.geeksforgeeks.org/hopfield-neural-network/

Hopfield Network Architecture

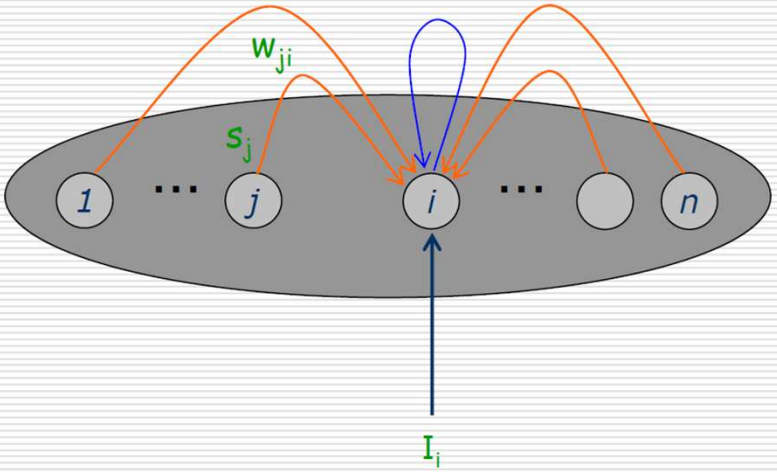




$$\dot{x}_i = \underbrace{-A_i x_i}_{\text{Passive decay}} + \underbrace{\sum_{j=1}^{n} w_{ji} \mathcal{S}_j(x_j)}_{\text{Signal feedback}} + \underbrace{I_i}_{\text{External input}}$$

Hopfield Network Architecture

- It comprises *n* neurons that receive external inputs I_i
- Each neuron feeds back signals($\delta_i(x_i)$) ,
- through weights w_{ij}



$$\dot{x}_i = \underbrace{-A_i x_i}_{\text{Passive decay}} + \underbrace{\sum_{j=1}^{n} w_{ji} \delta_j(x_j)}_{\text{Signal feedback}} + \underbrace{I_i}_{\text{External input}}$$

Neuron Characteristic and Inverse

- Assume that individual neurons have distinct sigmoidal characteristics

$$\mathcal{S}_i(x) = \frac{1 - e^{-\lambda_i x}}{1 + e^{-\lambda_i x}}$$

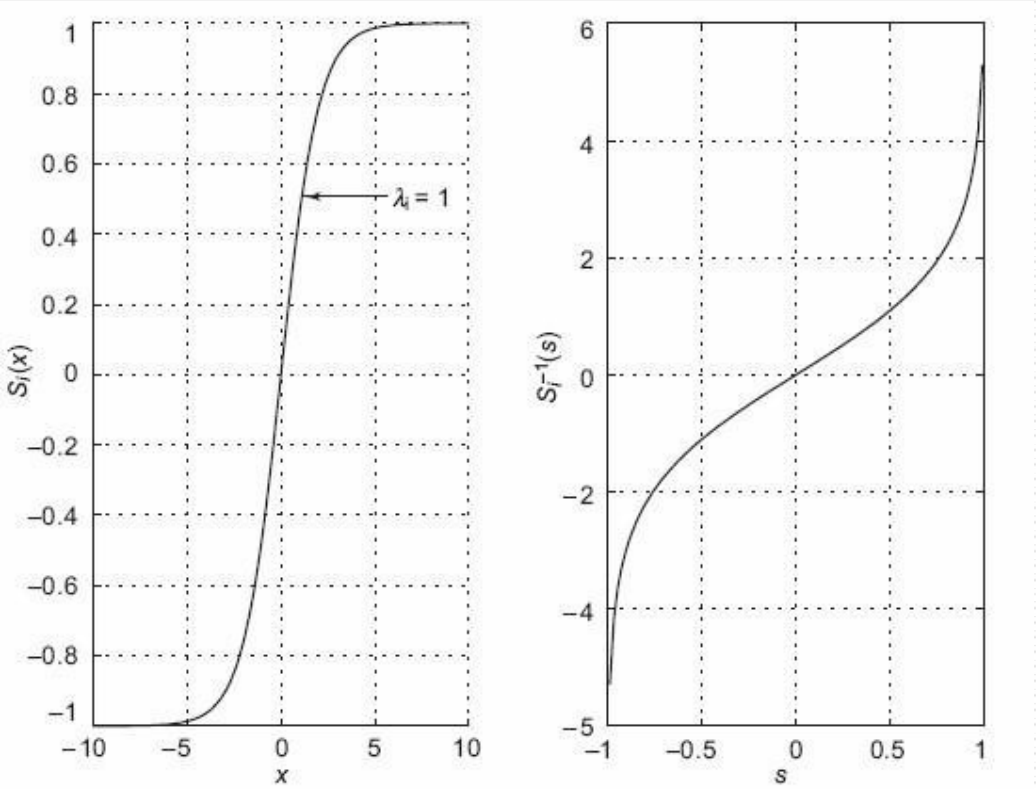
- Bipolar sigmoid is used
- λ is Gain scale factor common for all neurons

- With inverse

$$\mathcal{S}_i^{-1}(s) = \frac{1}{\lambda_i} \ln\left(\frac{1+s}{1-s}\right) = \frac{1}{\lambda_i}\mathcal{S}^{-1}(s)$$

$$s = \mathcal{S}(x) = \frac{1 - e^{-x}}{1 + e^{-x}}$$

Neuron Characteristic and Inverse



Note: Both signal function and its inverse are smooth and monotonic increasing.

-this property leads to stability of Hopfield network.

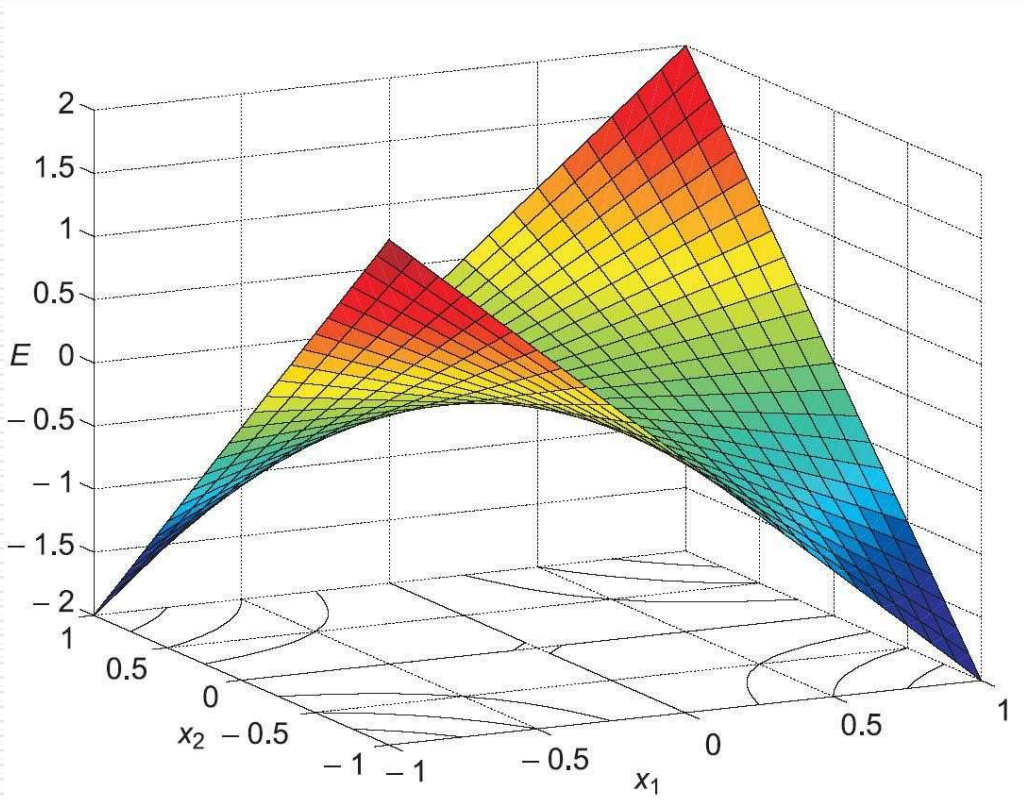
Notes on the Hopfield Network

- The system represents a high dimension cross-coupled non-linear dynamical system
  - Difficult to find find a closed form solution.
- In the original Hopfield network the weight w_{ii} is usually set to zero
  - Neurons do not feed back signals to themselves
- The model under consideration has Cohen-Grossberg form which necessarily dictates that connections be symmetric for stability

$$W_{ij} = W_{ji}$$

Example

- Energy function for a Hopfield network that encodes two vectors (-1,1) and (1,-1)
- *The network relaxes to an attractor in whose basin of attraction the initial state lies.*



Network state starts out at some initial value (denoted by a point on energy landscape), its states evolve to settled down to closest local minimum[Basin of attraction]

Applications of Hopfield Network

- Content Addressable Memory
  - Pattern Completion
- 3D Object Detection
- TSP optimization

Application of Hopfield Networks:

Content Addressable Memory

- Example: recall a binary number 11001010 based on input 11001000 with one bit distortion
- Key to the correct memory vector is the partially distorted input data itself
- Recall is essentially a clean up operation on a partially distorted input
- CAMs solve the completion problem

Hamming Distance Based Recall

- If
  - the Hamming distance of the input is such that the point lies within the basin of attraction of the memory which is represented by a distorted input
- Then
  - we expect that the correct memory should get recalled

Encoding Memories via Outer Products

- Bipolar outer product encoding

$$\mathbf{W} = \sum_{k=1}^{Q} X_k X_k^T$$

- Where $X_k = 2A_k - 1$, weight matrix **W** is symmetric, $\mathbf{W} = \mathbf{W}^T$
- Original model

$$\mathbf{W} = \sum_{k=1}^{Q} X_k X_k^T - Q\mathbf{I}$$

Zero off the diagonal elements

Avoids identity operator type behaviour

Asynchronous Neuron Update

- Standard Hopfield network update is always asynchronous *and* *deterministic*
  - A neuron is randomly selected for update at an instant of time.
  - The neuron activation transforms to the new signal using the deterministic signal function
- The number of times any neuron gets a chance to update is the same *on average*
- Serial asynchronous update guarantees that the network converges to a fixed point equilibrium

Synchronous Neuron Update

- Activations of all neurons are calculated at an instant of time
- All neurons update their signal values together to generate the signal vector at the next instant of time
- Under a synchronous update the Hopfield network approaches either a fixed point equilibrium or a two-step limit cycle

Operational Summary of the Hopfield CAM algorithm

Given	A set of binary vectors $\{A_i\}_{i=1}^{Q}$ to be encoded into a Hopfield CAM using bipolar encoding and decoding.
Encode	$\rightarrow \mathbf{W} = \sum_{k=1}^{Q} X_k X_k^T - Q\mathbf{I}$
Initialize	$\rightarrow$ Set-up neuron signals vector to the probe vector X_0.

Operational Summary of the Hopfield CAM algorithm

Iterate ↻Repeat

{

⇝ Compute the neuron activations $y_i^{k+1} = \sum_{j=1}^{n} w_{ji} s_j^k \quad i = 1, \ldots, n$

⇝ Select a neuron I at random from the n neurons in the field

⇝ Update the neuron in accordance with the signal function

$$s_I^{k+1} = \mathcal{S}(y_I^{k+1}) = \begin{cases} 1 & \text{if} \quad y_I^{k+1} > 0 \\ -1 & \text{if} \quad y_I^{k+1} < 0 \\ \mathcal{S}(y_I^k) & \text{if} \quad y_I^{k+1} = 0 \end{cases}$$

} until (signals of all neurons no longer change.)

Example 1

- Encode, recall, analyze stability
  - 110, 001

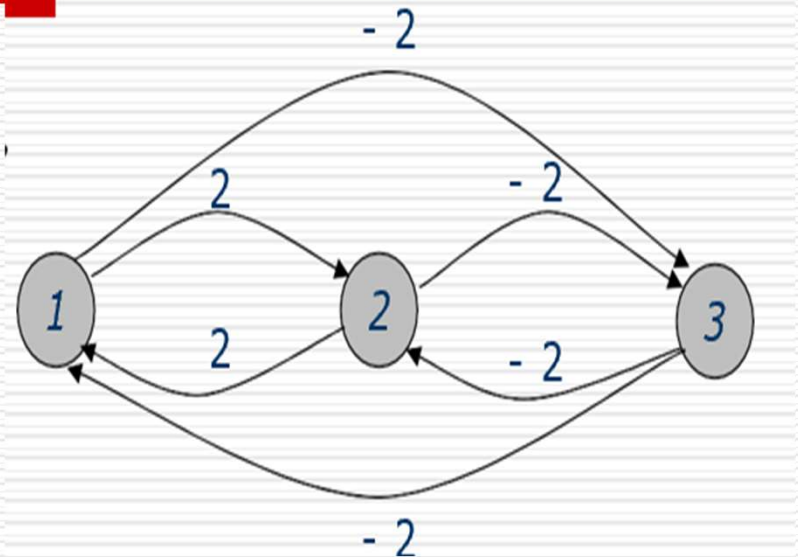
$$\mathbf{W} = \sum_{k=1}^{2} X_k X_k^T - 2\mathbf{I}$$

$$= \begin{pmatrix} 1 \\ 1 \\ -1 \end{pmatrix} \begin{pmatrix} 1 & 1 & -1 \end{pmatrix} + \begin{pmatrix} -1 \\ -1 \\ +1 \end{pmatrix} \begin{pmatrix} -1 & -1 & +1 \end{pmatrix} - 2 \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

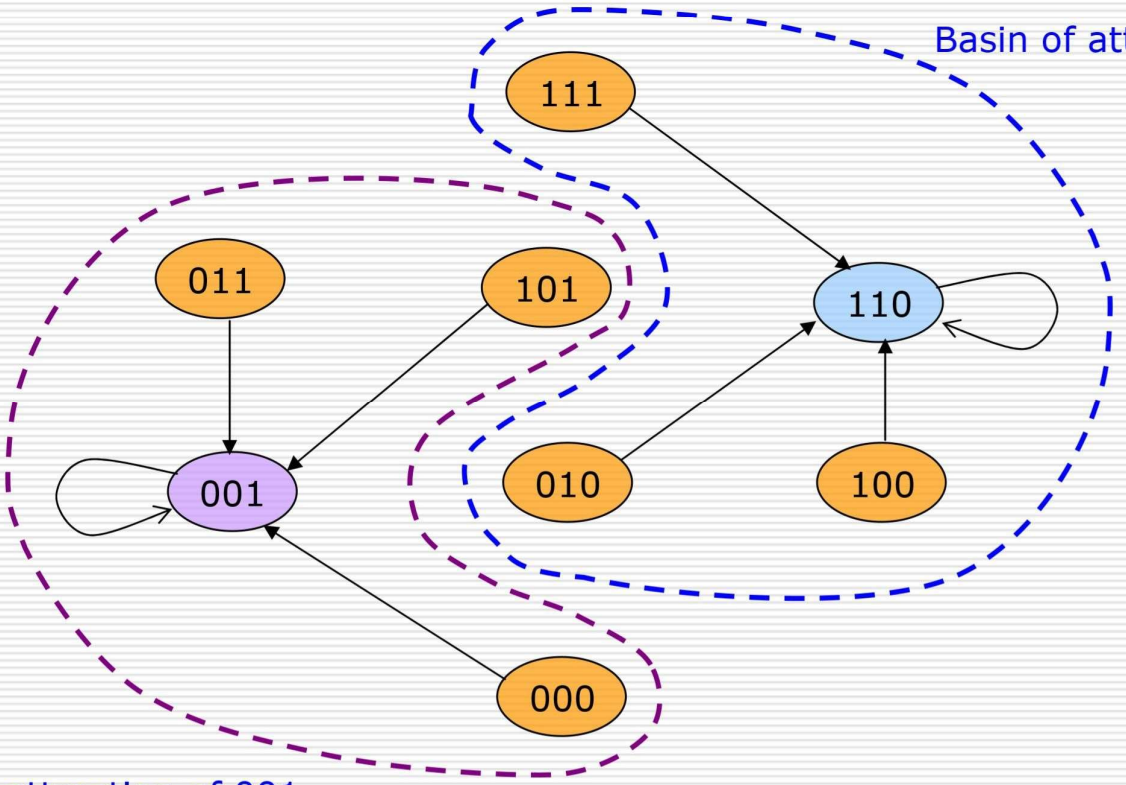
$$= \begin{pmatrix} 0 & 2 & -2 \\ 2 & 0 & -2 \\ -2 & -2 & 0 \end{pmatrix}$$

Example 1

Network architecture

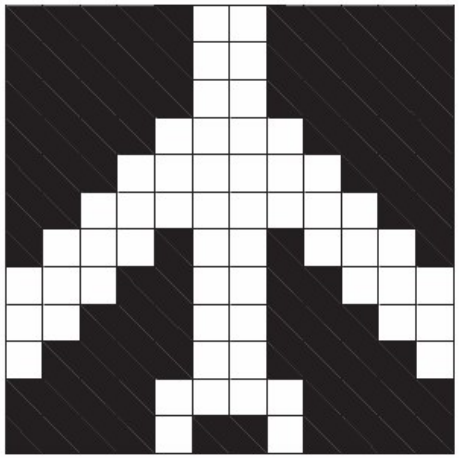


Basins of Attraction

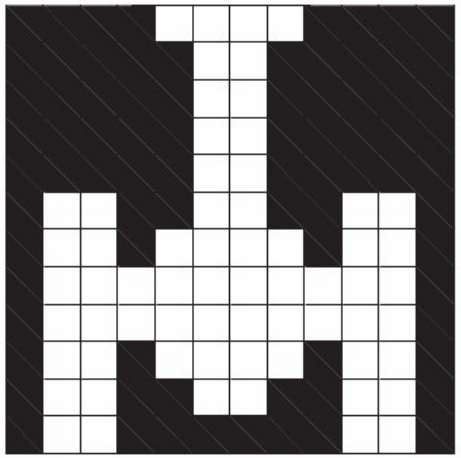


Application of Hopfield Model: Pattern Completion

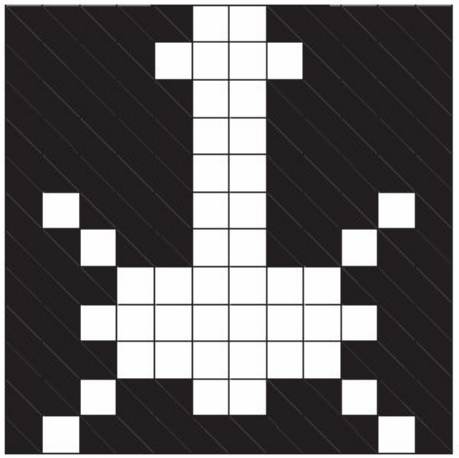
- Consider three 12×12 pixel based images



(a) Plane

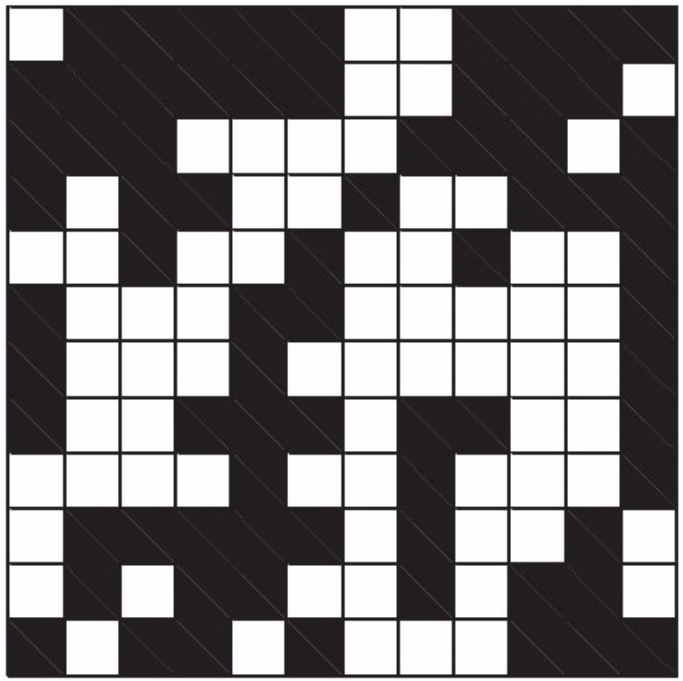


(b) Tank

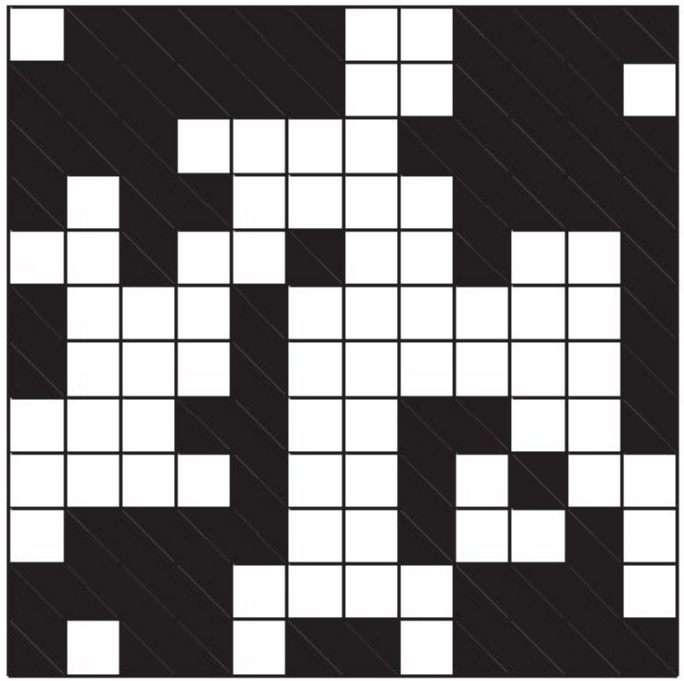


(c) Helicopter

Application: Pattern Completion



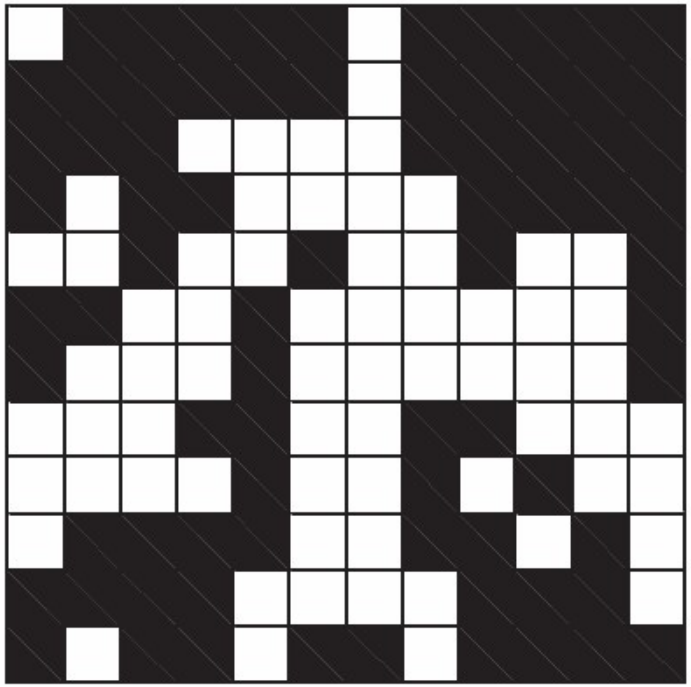
(a) 40 % Distorted Plane



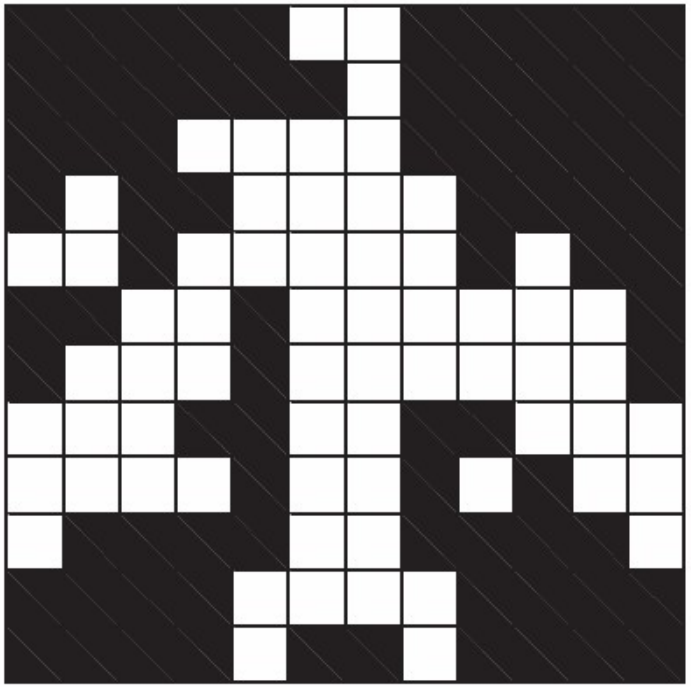
(b) Iteration: 48

Contd.

Application: Pattern Completion



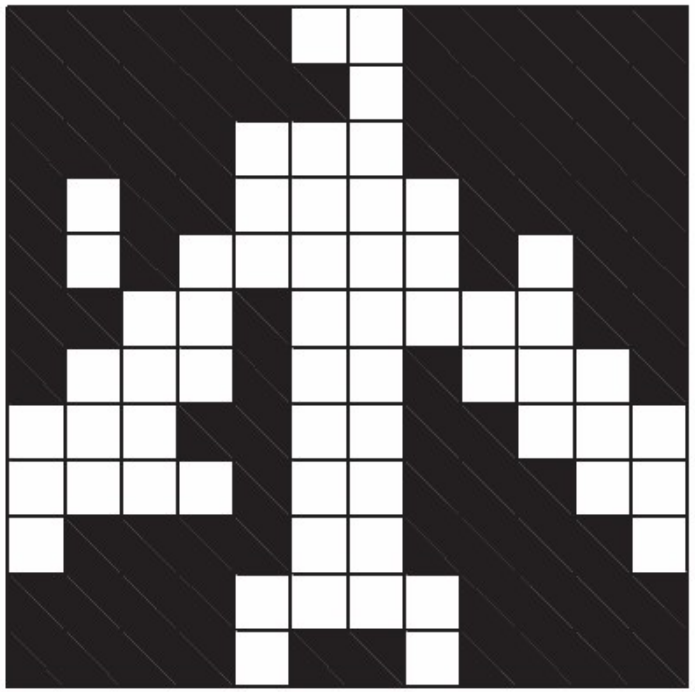
(c) Iteration: 72



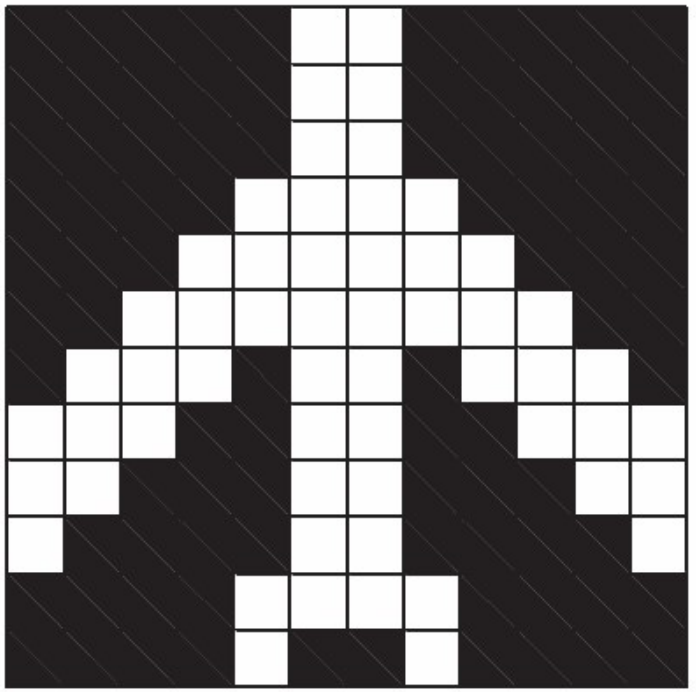
(d) Iteration: 96

Contd.

Application: Pattern Completion



(e) Iteration: 120



(f) Iteration: 144

Brain-State-in-a-Box (BSB) Neural Network

- Predecessor of the Hopfield network
- Extends the linear associator
- Similar to the Hopfield network
  - autoassociative model
  - connection matrix computed using outer products
- Operation of both models is also very similar
- Differences arising primarily
  - In the way activations are computed in each iteration
  - Signal function
- BSB stands apart from other models in its use of the linear-threshold signal function.

BSB Applications

- Speech perception and probability learning
- Multistable perception
- Cognitive computation
- Radar signal categorization

Operational Details

- **Activation Function**

$$X_{k+1}^T = \gamma S_k^T + \alpha S_k^T \mathbf{W} + \delta S_0^T$$

- **Signal Function**

$$S_{k+1} = \mathcal{S}(X_{k+1})$$

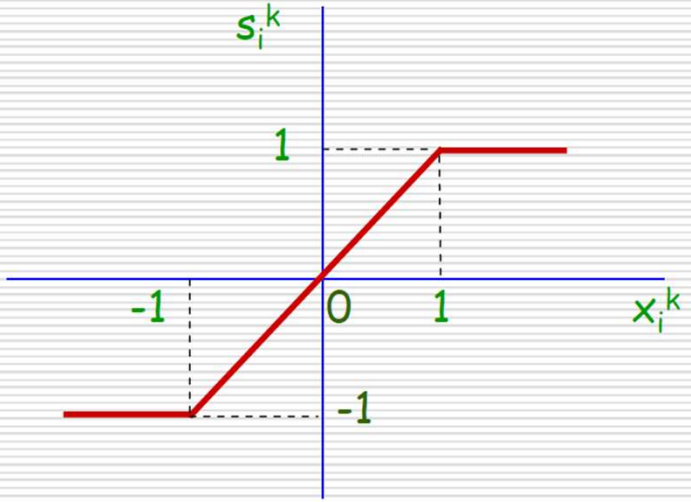
- **Weight Matrix**

$$\mathbf{W} = \sum_{k=1}^{Q} \lambda_k A_k A_k^T$$

  - Arranged so that A_i is an eigenvector of W with eigenvalue λ_i

Operational Details: Signal Function

- Piecewise linear signal function.
- States are boxed into the hypercube: *brain-state-in-a-box!*



$$s_i^k = \mathcal{S}(x_i^k) = \begin{cases} -1 & x_i^k < -1 \\ x_i^k & -1 \le x_i^k \le 1 \\ 1 & x_i^k > 1 \end{cases}$$

Operational Summary of the BSB Model

Given	A set of binary vectors $\{A_i\}_{i=1}^{Q}$ to be encoded into a BSB network using binary encoding.
Encode	$\hookrightarrow \mathbf{W} = \sum_{k=1}^{Q} \lambda_k A_k A_k^T$
Initialize	$\hookrightarrow$ Set up neuron signals vector to the probe vector S_0. $\hookrightarrow$ Set γ, α, δ.
Iterate	$\circlearrowleft$Repeat { $\rightsquigarrow$ Compute the neuron activations $X_{k+1}^T = \gamma S_k^T + \alpha S_k^T \mathbf{W} + \delta S_0^T$ $\rightsquigarrow$ Update the neurons synchronously $S_{k+1} = \mathcal{S}(X_{k+1})$, where $\mathcal{S}(x_i^k) = \begin{cases} -1 & x_i^k < -1 \\ x_i^k & -1 \le x_i^k \le 1 \\ 1 & x_i^k > 1 \end{cases}$ } until $(S_{k+1} = S_k)$

Rework Signal Update Equation

- Neuron-specific scalar form of the BSB vector update equation with $\gamma = 1$ and $\delta = 0$

$$s_i^{k+1} = \mathcal{S}\left(s_i^k + \alpha \sum_{j=1}^{n} w_{ji} s_j^k \right)$$

$$= \mathcal{S}\left(\sum_{j=1}^{n} (\delta_{ji} + \alpha w_{ji}) s_j^k \right)$$

$$= \mathcal{S}\left(\sum_{j=1}^{n} B_{ji} s_j^k \right)$$

- BSB is a special case of Cohen-Grossberg dynamics

$$B_{ji} = \delta_{ji} + \alpha w_{ji}$$

Kronecker delta

Simulated Annealing

- Provides a general framework for the optimization of the behaviour of complex systems
- Operates by introducing noise in a controllable fashion into the operational dynamics of the system
- Robust iterative search

Configuration, Temperature, Ground State

- A specific combination of neuron states is a *configuration* of the network
  - In an n node network there are 2^n configurations
- Basic Idea:
  - Generate different configurations of the system at various values of a control parameter called the *temperature*
  - Gradually reduce the value of this parameter to search for an optimal or *ground state* solution to the problem

Simulated Annealing Procedure

- Randomize neuron states once in the beginning, and initialize the temperature to a high value.
- Choose a neuron I randomly from the network
- Compute the energy E_A of the present configuration A
- Flip the state of neuron I to generate a new configuration B
- Compute energy E_B of configuration B

Simulated Annealing Procedure

- If $E_B < E_A$ accept configuration B
- Else accept configuration B with probability $\exp(-\Delta E/T)$, $\Delta E = E_B - E_A$
- Continue selecting and testing neurons randomly, and set their states several times in this way until a *thermal equilibrium* is reached.
- Finally, lower the temperature and repeat the procedure.

Stochastic Simulated Annealing Algorithm: Hopfield Network

Given	A set of binary vectors $\{A_i\}_{i=1}^{Q}$ to be encoded into a Hopfield CAM using bipolar encoding.
Encode	$\hookrightarrow \mathbf{W} = \sum_{k=1}^{Q} X_k X_k^T - Q\mathbf{I}$
Initialize	$\hookrightarrow T_0, k = 0, k_{\max}$ (temperature, iteration, iteration limit) S_0, c (signal vector, temperature contraction)

Stochastic Simulated Annealing Algorithm

Iterate ↻Repeat

{

↻Repeat

{

$\rightsquigarrow$ Select neuron I randomly.

$\rightsquigarrow$ Compute: $\Delta E^{(I)} = 2 s_I \sum_{j=1}^{n} w_{jI} s_j$

$\rightsquigarrow$ if $\Delta E^{(I)} < 0$, $s_I = -s_I$

else if $e^{-\Delta E^{(I)}/T_k} >$ rand [0,1), $s_I = -s_I$

} until(all nodes are polled several times)

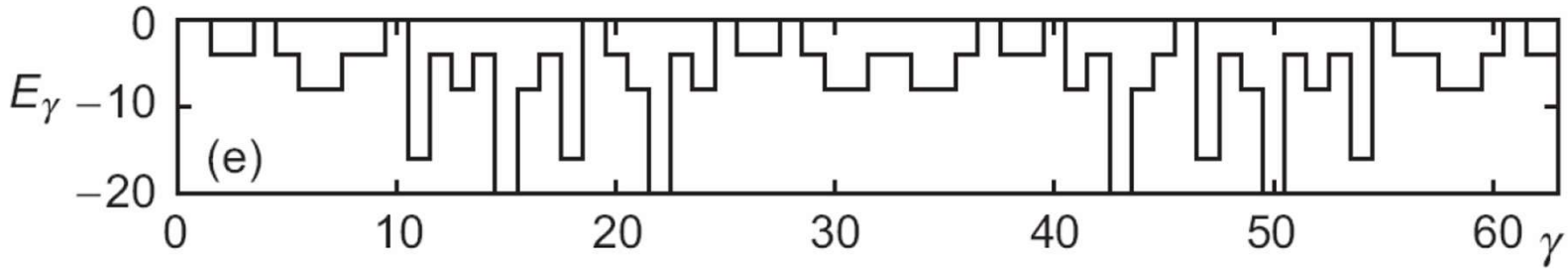
$\rightsquigarrow$ Reduce temperature: $T_{k+1} = cT_k$

} until ($k = k_{\max}$ or stopping criterion met)

Example

- Consider a Hopfield network encoding vectors $A_1 = (110001)$ $A_2 = (101010)$

$$\mathbf{W} = \begin{pmatrix} 0 & 0 & 0 & -2 & 0 & 0 \\ 0 & 0 & -2 & 0 & -2 & 2 \\ 0 & -2 & 0 & 0 & 2 & -2 \\ -2 & 0 & 0 & 0 & 0 & 0 \\ 0 & -2 & 2 & 0 & 0 & -2 \\ 0 & 2 & -2 & 0 & -2 & 0 \end{pmatrix}$$



Hopfield Net SA: Simulation Result