Neurocomputers

Joanna Matuszak, Joanna Wojciechowicz

Seminar on Analysis, Differential Equations and Modelling

2024

Contents

Neurocomputers

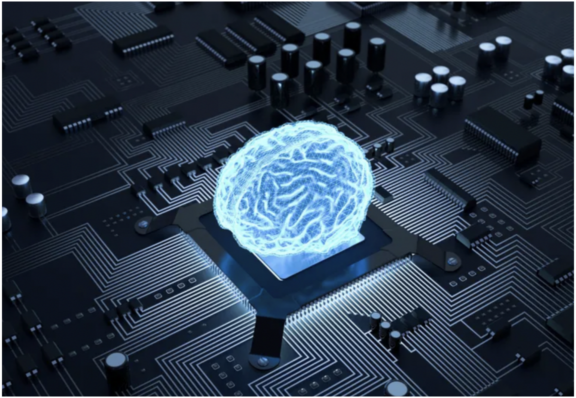

Figure: National Geographic

Neurocomputers

- Employs principles of the human brain.
- Consists of many interconnected units - so called neurons - performing simple nonlinear transformations in parallel.
- Main aim: not general-purpose computation, but pattern recognition using associative memory.
- Examples: feedforward neural networks, Hopfield model, oscillatory neural network.

Hopfield model - introduction

The Hopfield network finds inspiration in the intricate connections among biological neurons in the human brain, mimicking nature's efficiency in information processing. Similar to the synaptic communication in biological systems, the Hopfield network utilizes connections to store and retrieve patterns. Hopfield network also provides a model for understanding human memory.

Hopfield model - dynamics

The Hopfield model consists of N neurons - each labeled by a lower index i, where $1 \leq i \leq N$. Neurons in this model can have only two states. A neuron i is ON if its state variable equals $S_i = +1$ and neuron is OFF if it equals $S_i = -1$. The dynamics evolves in the discrete time with time steps Δt.

Hopfield model - dynamics

Neurons interact with each other with weights w_{ij}. We define the input potential of neuron i, influenced by the activity of other neurons as

$$h_i\left(t\right) = \sum_j w_{ij} S_j\left(t\right).$$

Hopfield model - dynamics

This input potential of the neuron at time t influences the probability of update of the state variable S_i into specific state in the next time step:

$$P\left(S_i\left(t+\Delta t\right)=+1|h_i\left(t\right)\right)=g\left(h_i\left(t\right)\right)=g\left(\sum_j w_{ij}S_j\left(t\right)\right),$$

where g - a monotonically increasing gain function with values between 0 and 1.

Hopfield model - dynamics

In our simulations we chose commonly used g function:

$$g(h) = \frac{1}{2}[1 + \tanh(\beta h)]$$

with a parameter β. For chosen $\beta \to \infty$, we have

$$g(h) = \begin{cases} 1 \ \mathit{for} \ h \geq 0 \\ 0 \ \mathit{for} \ h \leq 0. \end{cases}$$

Hopfield model - pattern recognition

The task for Hopfield model is to store and recall M different patterns. Patterns are labeled by the index μ with $1 \leq \mu \leq M$. Each pattern μ is defined as a desired configuration $\{p_i^\mu = \pm 1;\ 1 \leq i \leq N\}$. The network of N neurons correctly represents pattern μ, if the state of all neurons $1 \leq i \leq N$ is $S_i(t) = S_i(t + \Delta t) = p_i^\mu$. In other words, patterns must be fixed points of the dynamics.

Hopfield model - pattern recognition

At the very beginning of working with Hopfield model for patterns, a random number generator generates, for each pattern μ string of N independent binary numbers $p_i^\mu = \pm 1;\ 1 \leq i \leq N$ with expected value $< p_i^\mu >= 0$. Strings of different patterns are independent. The weights are chosen as

$$w_{ij} = c \sum_{\mu=1}^{M} p_i^\mu p_j^\mu$$

with positive constant $c \geq 0$. The network has full connectivity. The standard choice of the constant c is $c = \frac{1}{N}$.

Hopfield model - pattern recognition

In order to mimic memory retrieval in the Hopfield model, an input is given by initializing the network state $S(t_0) = S_i(t_0)\,;\ 1 \leq i \leq N$. After initialization, the network evolves freely under dynamics. Ideally the dynamics should converge to a fixed point corresponding to the pattern μ which is most similar to the initial state.

Hopfield model - pattern recognition

In order to measure the similarity between the current state $S(t) = S_i(t);\ 1 \leq i \leq N$ and a pattern μ, we introduce the overlap

$$m^{\mu}(t) = \frac{1}{N} \sum_i p_i^{\mu} S_i(t) .$$

The overlap takes a maximum value of 1, if $S_i(t) - p_i^{\mu}$, i.e., if the pattern is retrieved. It is close to zero if the current state has no correlation with pattern μ. The minimum value $m^{\mu}(t) = -1$ is achieved if each neuron takes the opposite value to that desired in pattern μ.

Hopfield model - pattern recognition

1. Generate M patterns.
2. Initialize the model by giving the random state on the input.
3. Compute the input potential h of each neuron.
4. Compute the probability of each neuron to be ON or OFF - $g(h)$ function.
5. Change the state of each neuron.
6. Repeat 3-5 until convergence.

Hopfield model - important note

- In our Hopfield network all the updates of neurons are done in parallel, but an update scheme where only one neuron is updated per time step is also possible.

Hopfield model - numerical simulations

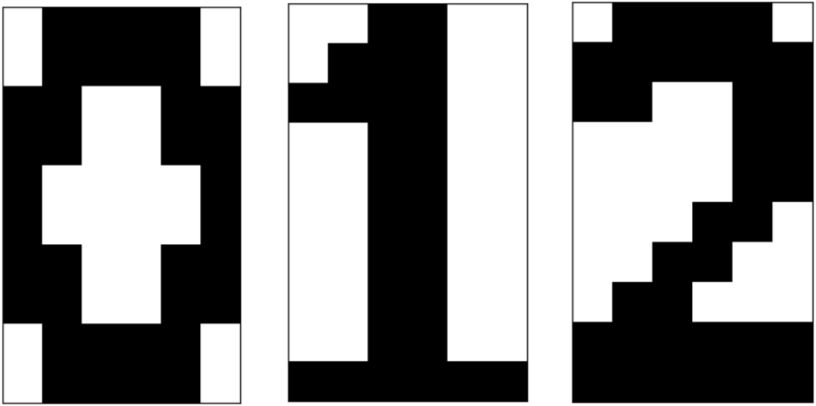

Figure: Memorized patterns

Hopfield model - example 1

We start with totally random input and try to converge to one of memorized patterns.

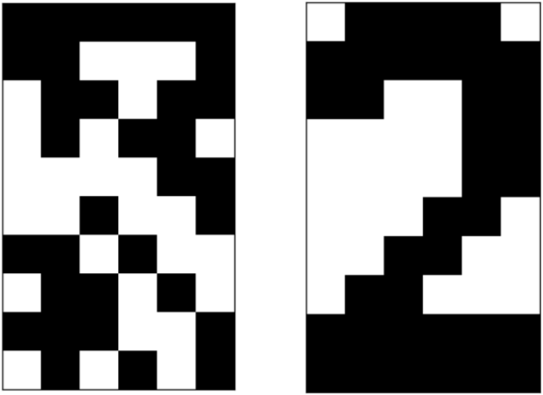

Figure: Hopfield model recognition for random initial pattern. Convergence in 1 step.

Hofield model - example 2

We take one of memorized patterns, add noise to it, use it as an input for our model and try to converge to one of memorized patterns.

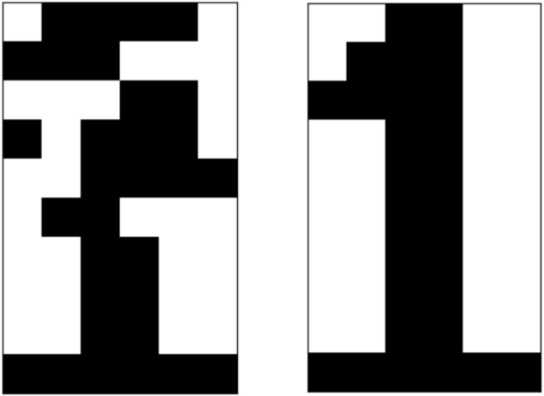

Figure: Hopfield model recognition for initial pattern close to one of the memorized patterns. Convergence in 1 step.

Hopfield model - conclusions

- The model always converges, but sometimes to patterns that were not memorized.
- In the case of random initial state in 37.4 % of cases we converged exactly to one of the memorized patterns, 36.1 % to inverted memorized pattern and 26.5 % to different pattern.
- In the case of the initial state close to one of the memorized patterns in 99.2 % of cases we converged exactly to one of the memorized patterns, 0.1 % to inverted memorized pattern and 0.7 % to different pattern.

Hopfield model - convergence

- In the context of the Hopfield model, convergence to one of the stored patterns is not guaranteed. When the network receives an input, it tries to move towards a stable state, which ideally corresponds to one of the stored patterns. However, it may also converge to a state that is a combination of multiple stored patterns or a state that is not exactly one of the stored patterns. This is known as spurious states or spurious attractors.
- Spurious states are patterns $x_s \notin M$, where M is the set of patterns to be memorized. In other words, they correspond to local minima in the energy function. They can be composed of various combinations of the original patterns or simply the negation of any pattern in the original pattern set.

Hopfield model - convergence

- Whether the network converges to one of the stored patterns or to a spurious state depends on various factors including the network architecture, the number and nature of the stored patterns, the initialization of neuron states, and the dynamics of updating.
- In general if we input in to the network the pattern that is close enough to one of memorized patterns (memorized pattern + noise) it will convergence to the proper pattern.

Oscillatory Neural Network - introduction

As mentioned before there are many neural network models that can be used as a theoretical basis for a neurocomputer. The most promising are oscillatory neural networks because they take into account rhythmic behavior of the brain.

Oscillatory Neural Network

Looking at the brain we treat the cortex as being a network of weakly connected autonomous oscillators forced by the thalamic input.

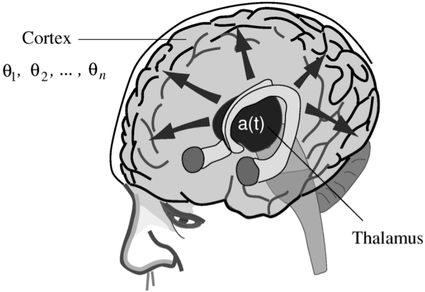

Figure: We treat the cortex as being a network of weakly connected autonomous oscillators forced by the thalamic input $a(t)$.

Oscillatory Neural Network

The fact whether these oscillators communicate with each other or not depends on their frequencies.

- If two oscillators have approximately equal frequencies, then they do communicate in the sense that the phase (timing) of one of them is sensitive to the phase of the other.
- When they have different frequencies, their phases uncouple.

It means that the oscillator can interact with other oscillator only if it has appropriate frequency.

Oscillatory Neural Network

This suggest specific architecture of the oscillatory neural network model. It consists of oscillators having different frequencies and connected homogeneously and weakly to a common medium.

Oscillatory Neural Network

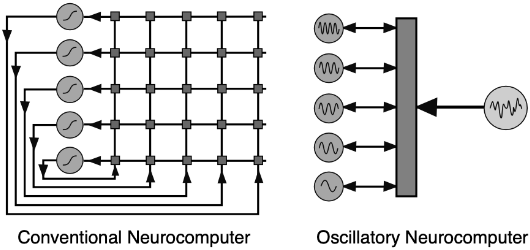

Figure: A conventional neurocomputer having n neurons (circles) would have n^2 connections (squares). An oscillatory neurocomputer with dynamic connectivity imposed by the external input (large circle) needs only n connections: between each neuron (circle) and a common medium (rectangle).

Oscillatory Neural Network

We will ilustrate the model using Kuramoto's phase model:

$$\dot{\vartheta}_i = \omega_i + \epsilon a\left(t\right) \sum_{j=1}^{n} \sin\left(\vartheta_j - \vartheta_i\right),$$

where ϑ_i is the phase of the i-th oscillator, $a\left(t\right)$ is the external input and $\epsilon \ll 1$ is the strength of connection.

Oscillatory Neural Network

Let

$$\vartheta_i\left(t\right) = \Omega_i t + \varphi_i,$$

then

$$\dot{\varphi}_i = \epsilon a\left(t\right)\sum_{j=1}^{n} \sin\left(\{\Omega_j - \Omega_i\}t + \varphi_j - \varphi_i\right).$$

Oscillatory Neural Network

Suppose we are using a quasiperiodic external input. We are given a matrix of connections $C = (c_{ij})$. Let

$$a(t) = a_0 + \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} \cos(\{\Omega_j - \Omega_i\} t)$$

be a time dependent external input, which is a quasiperiodic function of t.

Oscillatory Neural Network

If we denote $s_{ij} = \frac{c_{ij}+c_{ji}}{2}$, use the slow time $\tau = \epsilon t$ then we can rewrite system as

$$\varphi_i' = \sum_{j=1}^{n} s_{ij} \sin(\varphi_j - \varphi_i).$$

We can see that taken external input can dynamically connect any two oscillators provided that the corresponding c_{ij} is not zero.

Oscillatory Neural Network

Hebbian learning rule:
Suppose we are given a set of m key vectors to be memorized

$$\xi^k = \left(\xi_1^k, \xi_2^k, \ldots, \xi_n^k\right), \ \xi_i^k = \pm 1, \ k = 1, \ldots, m,$$

where $\xi_i^k = \xi_j^k$ means that the ith and jth oscillators are in-phase ($\varphi_i = \varphi_j$), and $\xi_i^k = -\xi_j^k$ means they are antiphase ($\varphi_i = \varphi_j + \pi$).

Oscillatory Neural Network

A Hebbian learning rule of the form

$$s_{ij} = \frac{1}{n} \sum_{k=1}^{m} \xi_i^k \xi_j^k$$

is the simplest one among many possible learning algorithms. It suffices to apply the quasiperiodic external input with $c_{ij} = s_{ij}$ for all i and j.

Oscillatory Neural Network - simulation

Suppose we are given a vector $\xi^0 \in \mathbb{R}$ to be recognized. Let us apply the external input $a(t)$ with $c_{ij} = \xi_i^0 \xi_j^0$ for certain period of time. This results in the phase deviation system of the form

$$\varphi_i' = \sum_{j=1}^{n} \xi_i^0 \xi_j^0 \sin(\varphi_j - \varphi_i).$$

We can see that if $\xi_i^0 \xi_j^0 = 1$, then $\varphi_i(t) - \varphi_j(t) \to 0$, and if $\xi_i^0 \xi_j^0 = -1$, then $\varphi_i(t) - \varphi_j(t) \to \pi$ for all i and j.

Oscillatory Neural Network - simulation

When we restore the original input $a(t)$, which induces the desired dynamic connectivity, the recognition starts from the input image ξ^0 with added noise.

Oscillatory Neural Network - simulation

1. Initialize the network. Let ξ^0 be the pattern that has to be recognized. Then, we define a matrix of connections $C = (c_{ij}) = \left(\xi_i^0 \xi_j^0\right)$, and we run the model until convergence.
2. Pattern recognition. Now we use the seed given by the previous step as a initial condition of the network, with some noise. Then, we simulate the system with the matrix s_{ij} defined beofre, until convergence.

Oscillatory Neural Network - simulation

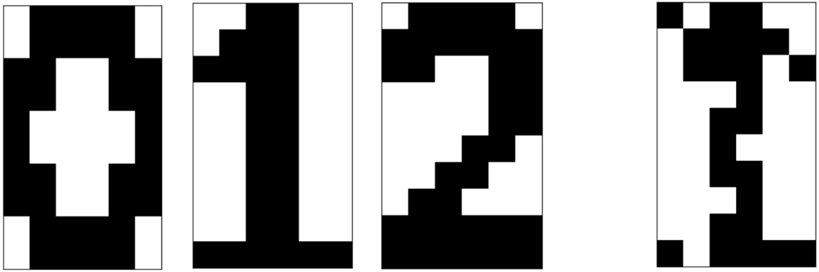

Figure: Memorized patterns and the input pattern.

Oscillatory Neural Network - simulation

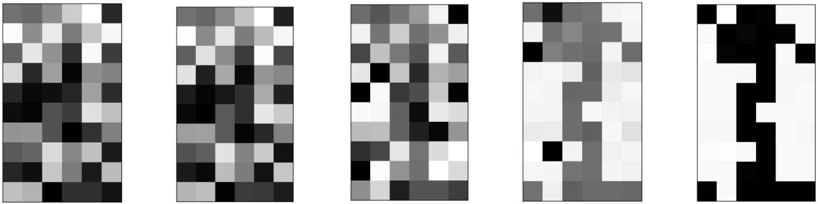

Figure: Initialization stage of the simulation. States are shown for t=0, t=2.5, t=5, t=7.5, t=10.

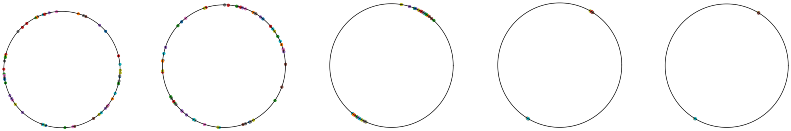

Figure: Evolution of $\varphi_i(t)$ during initialization stage of the simulation. States are shown for t=0, t=2.5, t=5, t=7.5, t=10.

Oscillatory Neural Network - simulation

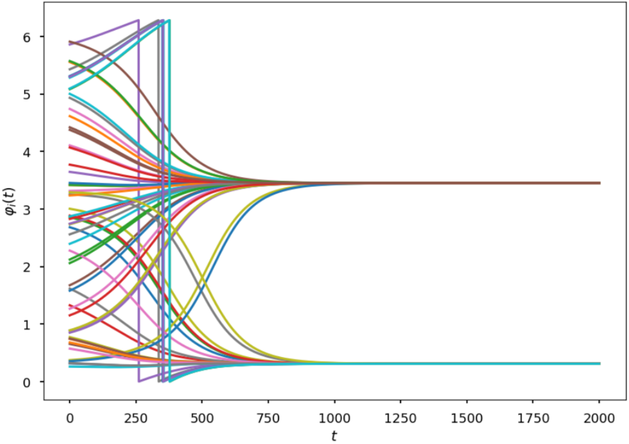

Figure: Evolution of $\varphi_i(t)$ during the initial phase.

Oscillatory Neural Network - simulation

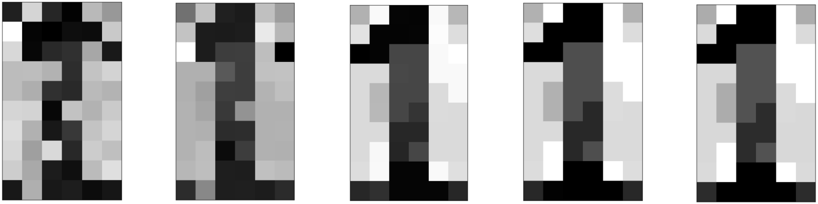

Figure: Recognition stage of the simulation. States are shown for t=0, t=2.5, t=5, t=7.5, t=10.

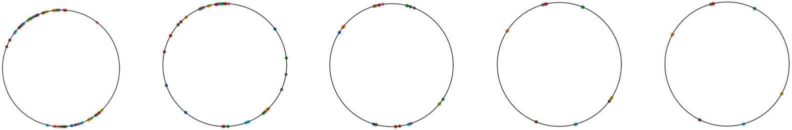

Figure: Evolution of $\varphi_i(t)$ during recognition stage of the simulation. States are shown for t=0, t=2.5, t=5, t=7.5, t=10.

Oscillatory Neural Network - simulation

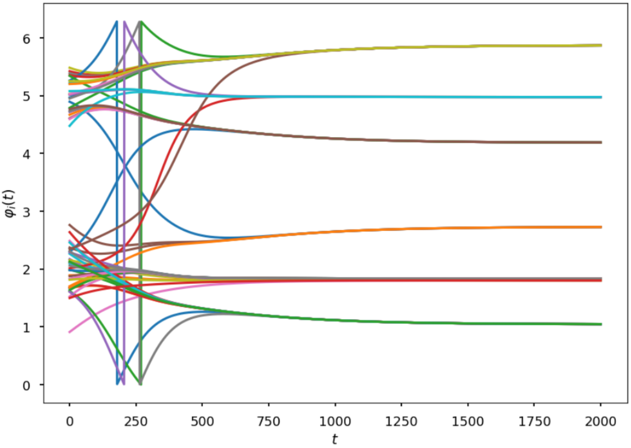

Figure: Evolution of $\varphi_i(t)$ during the recognition phase.

Summary

- Neurocomputers focuses on pattern recognition using associative memory, not general-purpose computatipons. Neurocomputers consist of so called neurons.
- Hopfield model can converge to the closest pattern to the input pattern from the set of memorized patterns. The knowledge about memorized patterns is stored in weights w_{ij}.
- Oscillatory Neural Networks mimic oscillatory behaviour of the brain - we discussed only one proposed architecture.

Bibliography

1. Hoppensteadt, Frank C., and Eugene M. Izhikevich. "Oscillatory neurocomputers with dynamic connectivity." Physical Review Letters 82.14 (1999): 2983.
2. Gerstner, Wulfram, et al. Neuronal dynamics: From single neurons to networks and models of cognition. Cambridge University Press, 2014.

Thank you!