WARSAW UNIVERSITY OF TECHNOLOGY

FACULTY OF MATHEMATICS

AND INFORMATION SCIENCE

Neural Networks

Lecture 16

Cellular neural networks

Cellular Neural Networks

Cellular Neural Networks (CNN) are a parallel computing paradigm similar to neural networks, with the difference that communication is allowed between neighbouring units only.

CNN can be viewed as a special case of a continuous-time Hopfield network.

It differs from the analog Hopfield network in its *local connectivity* property

Cellural Neural Networks – CNN

Cellural Neural Networks – CNN are built from identical nonlinear units called cells. It is a multi-input, dynamical system, and the behavior of the overall system is driven primarily through the weights of the processing unit’s linear interconnect.

Cellular Neural Networks

From an architecture standpoint, CNN processors are a system of a finite, fixed-number, fixed-location, fixed-topology, locally interconnected, multiple-input, single-output, nonlinear processing units.
Cells are defined in a normed space, commonly a two-dimensional Euclidean geometry, like a grid.

Cellural Neural Networks – CNN

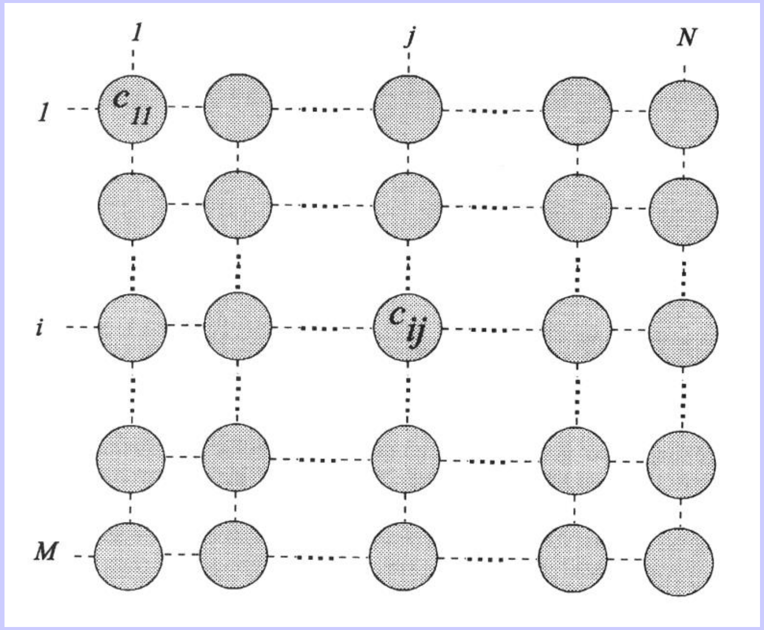
Cells are defined in a normed space, commonly a two-dimensional geometry, like a grid. The cells are not limited to two-dimensional spaces however; they can be defined in an arbitrary number of dimensions and can be square, triangle, hexagonal, or any other spatially invariant arrangement. Topologically, cells can be arranged on an infinite plane or on a toroidal space. Cell interconnect is local, meaning that all connections between cells are within a specified radius (with distance measured topologically). Connections can also be time-delayed to allow for processing in the temporal domain.

Network topology

Typically the two-dimensional network is organized in an eight-neighbour rectangular grid

Each cell c_{ij} situated in *i-th* row and *j-th* column interacts directly only with the cells within its *radius of neighbourhood r*.

When r = 1, which is a common assumption, the neighbourhood includes the cell itself and its eight nearest neighbouring cells



Network topology

A cell $c_{i+k,j+l}$ situated in $i+k$ row and $j+l$ column belongs to the cell neighbourhood, when

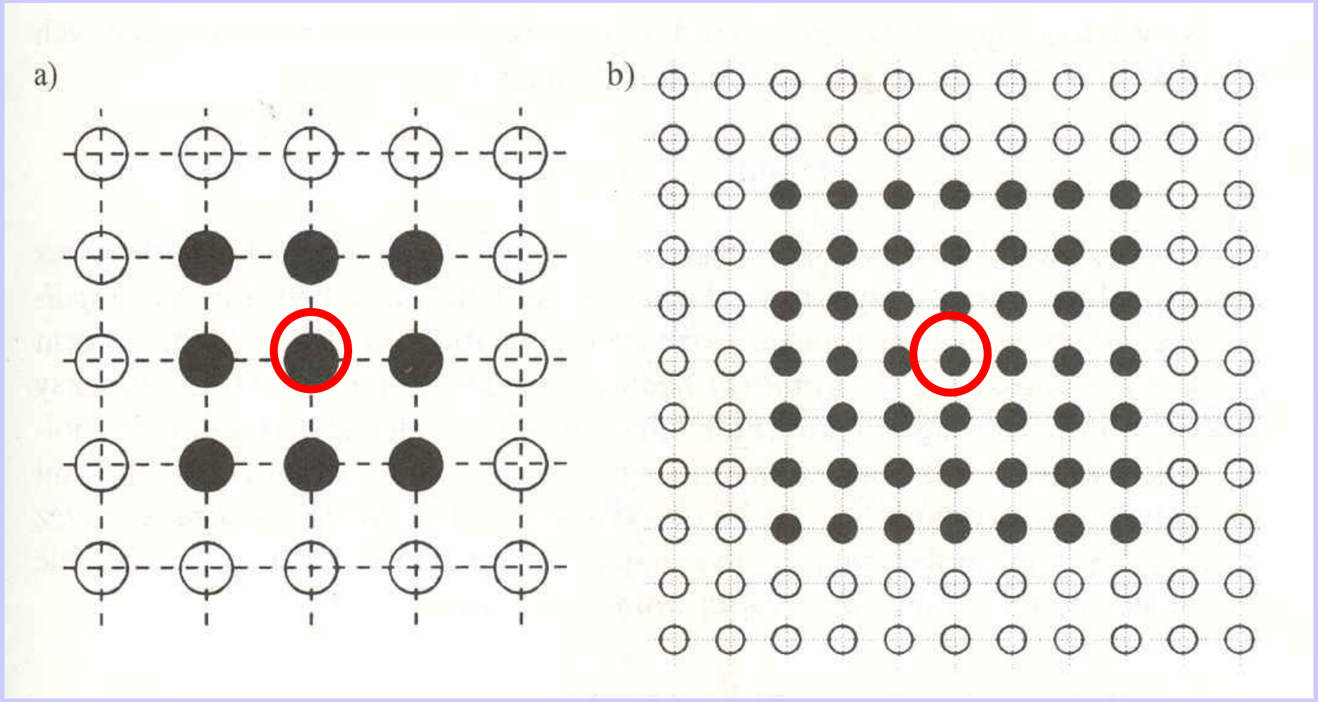
$$c_{i+k,\,j+l} \in N_r(i,\,j) \Leftrightarrow |k| \le r, |l| \le r$$

where r jest natural number called *radius of neighbourhood*, $N_r(i,j)$ denotes neighbourhood of the c_{ij} of the radius r.

Network topology

Part of the cellural neural network with the radius of neighbourhood

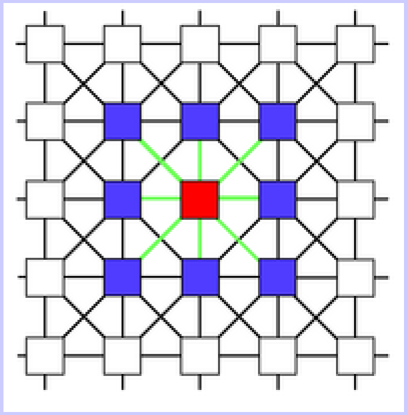
$r = 1$ $r = 3$



Network topology

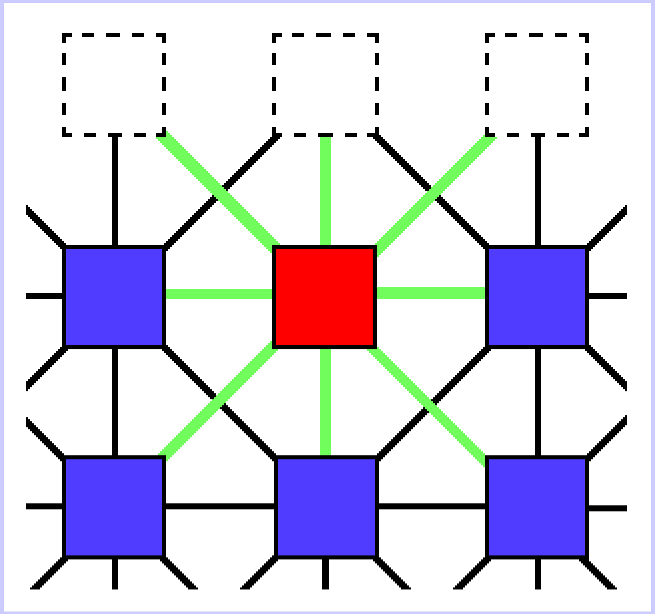
Part of the cellular neural network with the radius of neighbourhood   r = 1

The figure shows the emphasized cell (red) connected to the nearest neighbours (blue). The cells marked in grey represent the neighbourhood cells of the black cell. The neighbourhood includes the black cell itself.



Network topology

In order to calculate the state of the cells on the boundary, it is necessary to define the boundary conditions of the network, as shown in the figure.



Local connections of an *edge cell*. Observe that three of its neighbors are *boundary cells* (dashed).

Network topology

In the standard model, the boundary conditions can be:

- Fixed (or Dirichlet), if the value of the *boundary cells* is a prescribed constant;
- Zero-flux (or Neumann), if the value of the *boundary cells* is the same as the *edge cells*;
- Periodic (or toroidal), if the value of the *boundary cells* is the same as the *edge cells* on the opposite side (e.g., top *boundary cells* have the value of bottom *edge cells*).

Network operation

All cells are transforming signals the same way, generating **output signal**

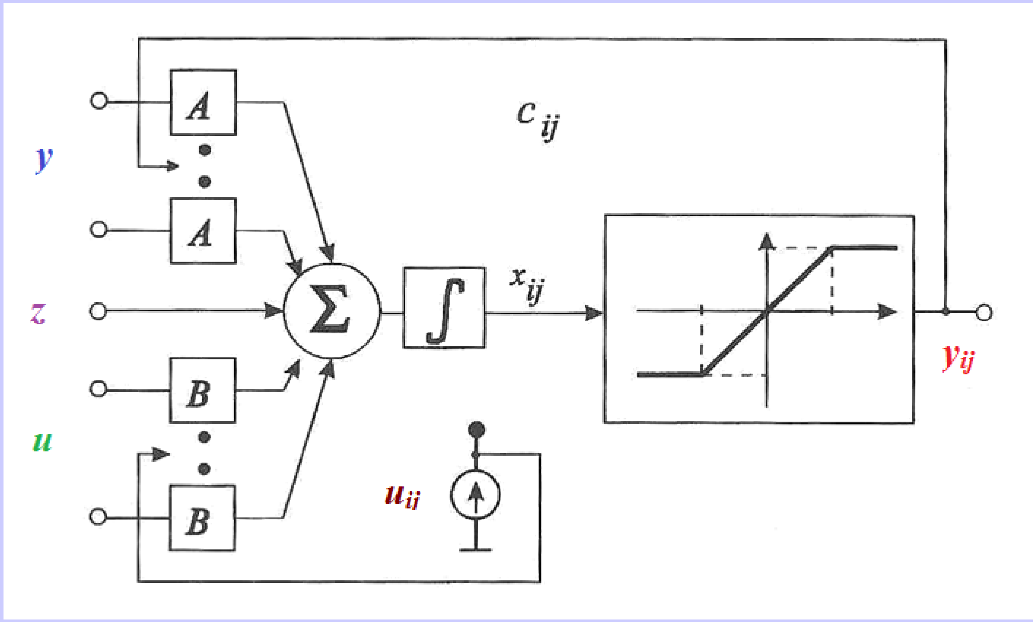
The output signal y_{ij} depends from **cells' state** x_{ij}.

Cell state is determined by the integration of the sum of cell **control signals** multiplied by the proper coefficients

Network operation

Control signals:

a) output signals $y_{i+k,j+l}$ the cells in the neighborhood
b) own output signal of the c_{ij} cell y_{ij}
c) input signals $u_{i+k,j+l}$ the cells in the neighborhood
d) owninput signal of the cell c_{ij}
e) bias z



Network operation

For input signals u_{ij} and initial conditions of a cell state x_{ij} for $t=0$, following conditions are imposed

$$\left|x_{ij}(0)\right| \le 1, \qquad \left|u_{ij}(t)\right| \le 1$$

Network operation

Weights of the output signals are called feedback operators and denoted

$$A_{i,j;i+k,j+l}$$

subscripts i,j refer to the cell c_{ij} being controlled
subscripts $i+k,j+l$ refer to the cells controlling the cell c_{ij}

Network operation

Weights of input signals are called control operators and denoted

$$B_{i,j;i+k,j+l}$$

where

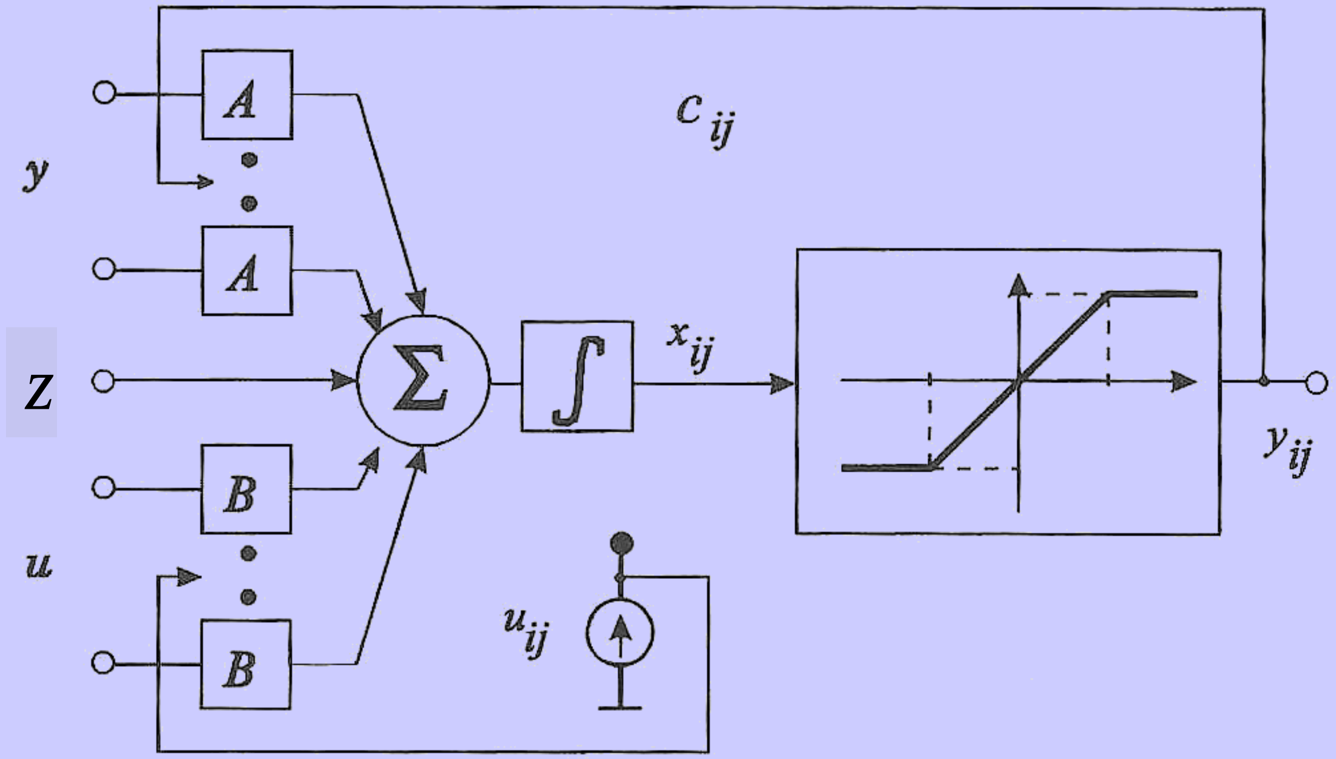
subscripts i,j refer to the cell c_{ij} being controlled

subscripts $i+k,j+l$ refer to the cells controlling the cell $c_{ij.}$

Bias signal z has usually constant value but not necessary identical for every cell in the network.

Cell

Block diagram of a single cell



CNN dynamics

The CNN dynamics is described by a system of nonlinear differential equations. Using the simplest first-order cell dynamics and linear interactions, the state equation of a cell in position *(i,j)* is as follows

$$C\frac{dx_{ij}}{dt} = -\frac{x_{ij}}{R_x} + \sum_{k=-r}^{r}\sum_{l=-r}^{r} A_{ij;i+k,j+l}\, y_{i+k,j+l} +$$

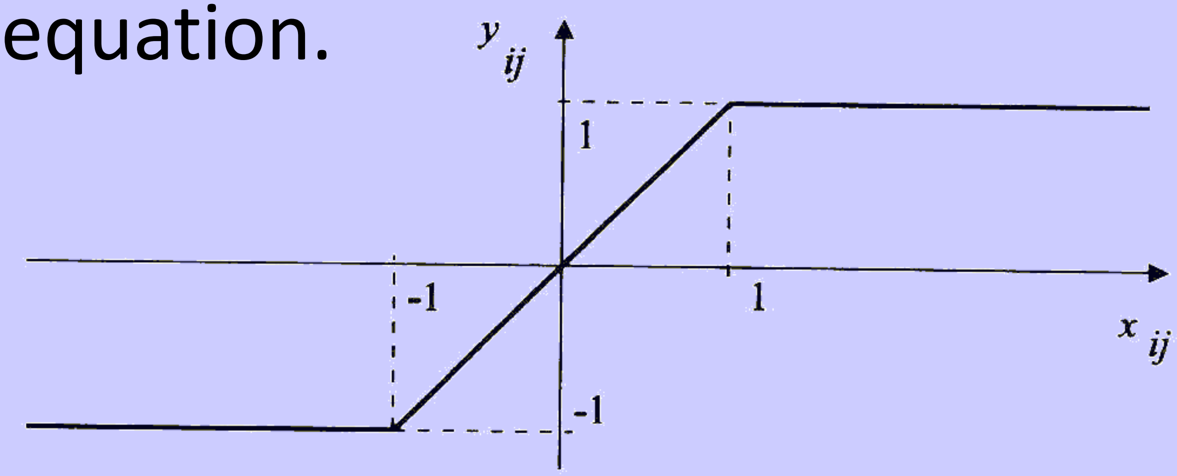
$$+ \sum_{k=-r}^{r}\sum_{l=-r}^{r} B_{ij;i+k,j+l}\, u_{i+k,j+l} + z$$

CNN dynamics

The expression for the output y_{ij} defined by the piece-wise linear function is

$$y_{ij} = \mathrm{f}(x_{ij}) = 0{,}5 * (|x_{ij} + 1| - |x_{ij} - 1|)$$

Where f(*) is the standard nonlinearity for the output equation.



Network operation

The set of matrices *{A,B}*, which contains the weights of the network, is called the *cloning template* and it defines the operation performed by the network. When the values of the *cloning template* do not depend on the position of the cell, the CNN is called *space-invariant*.

$A_{ij;i+k,j+l} = A_{IJ;I+k,J+l}$ and $B_{ij;i+k,j+l} = B_{IJ;I+k,J+l}$

where $0 \leq i, I, i+k, I+k < M$

$0 \leq j, J, j+l, J+l < N$

Network operation

We can neglect subscripts ij and IJ leaving only kl, where $-r \le k,l \le r$.

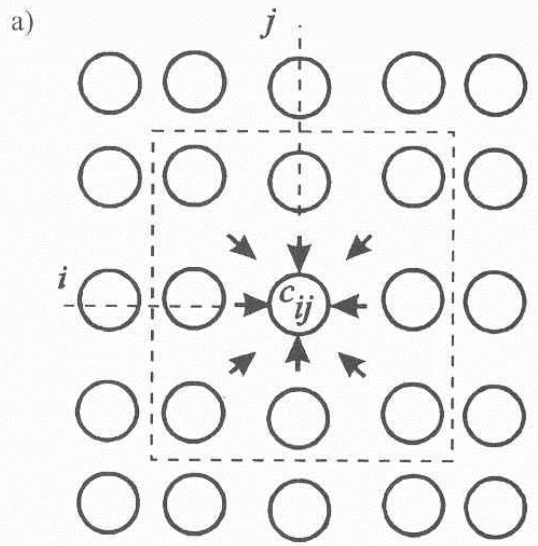
Weights A_{kl} i B_{kl} are usually represented in a matrix form **A** (**feedback operator**) and **B** (**control operator**) with dimension $(2r+1)x(2r+1)$.

The central element of neighborhood has the indexes $k = 0$ and $l = 0$, and A_{00} denotes power of the cell c_{ij} self-control.

Network operation

A_{kl} defines the power of control by a cell distant by l columns and k rows.

Network fragment for $r = 1$ and matrix **A**

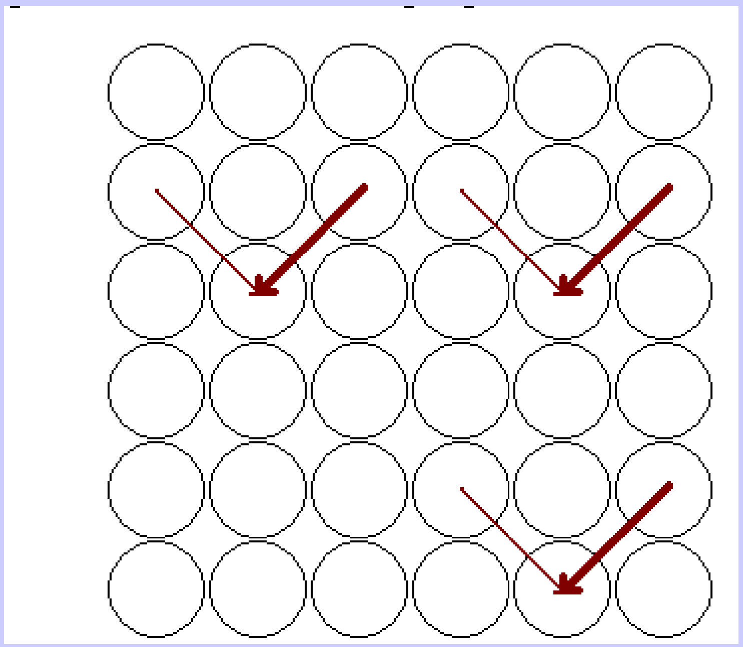


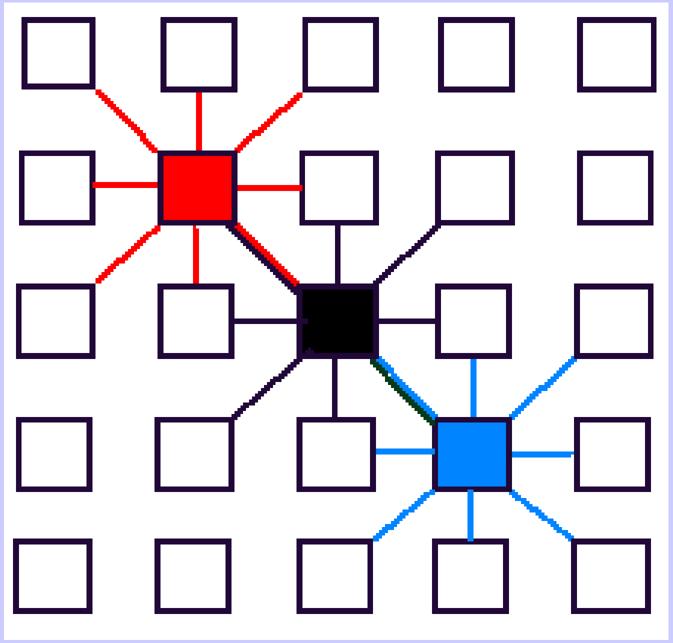
b)

$$\mathbf{A} = \begin{bmatrix} A_{-1-1} & A_{-1\,0} & A_{-1\,1} \\ A_{0-1} & A_{0\,0} & A_{0\,1} \\ A_{1-1} & A_{1\,0} & A_{1\,1} \end{bmatrix}$$

Network operation

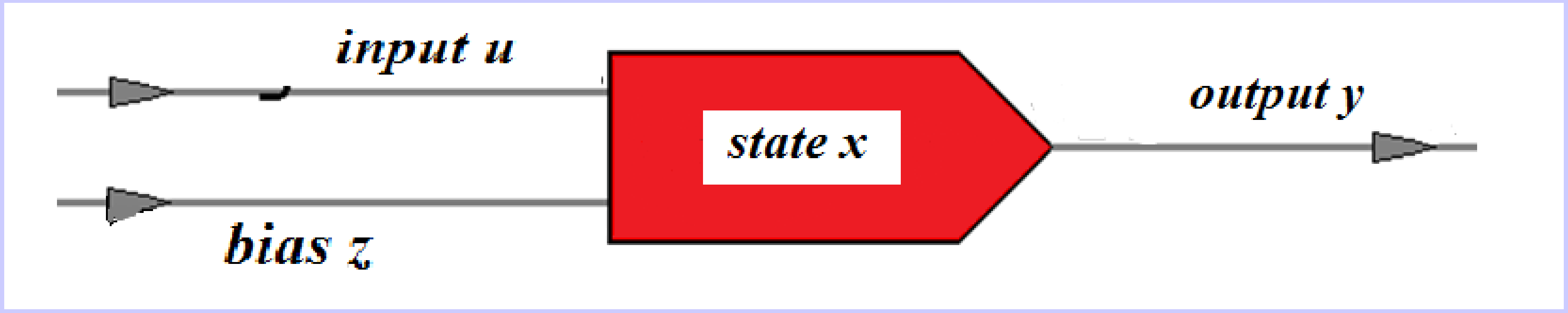
The template cloning is identical for every cell





Network operation

The simplified cell



output $y(t) = \mathrm{f}(x(t), u(t), z)$

state $x(t+1) = \mathrm{g}(x(t), u(t), z)$

$$x, y, u, z \in \mathbf{R}$$

Network operation

To describe the network structure it is enough to define the cloning template:

- bias term ***z*** (usually identical for each cell),
- control operator **A**, describing the weights in the feedback connections,
- feedback operator **B**, describing the feedforward connections,
- initial conditions.

To start the calculations it is necessary to define:

- initial state ***x*** for each cell,
- input signal ***u*** for each cell.

Network operation

The g function is define by:

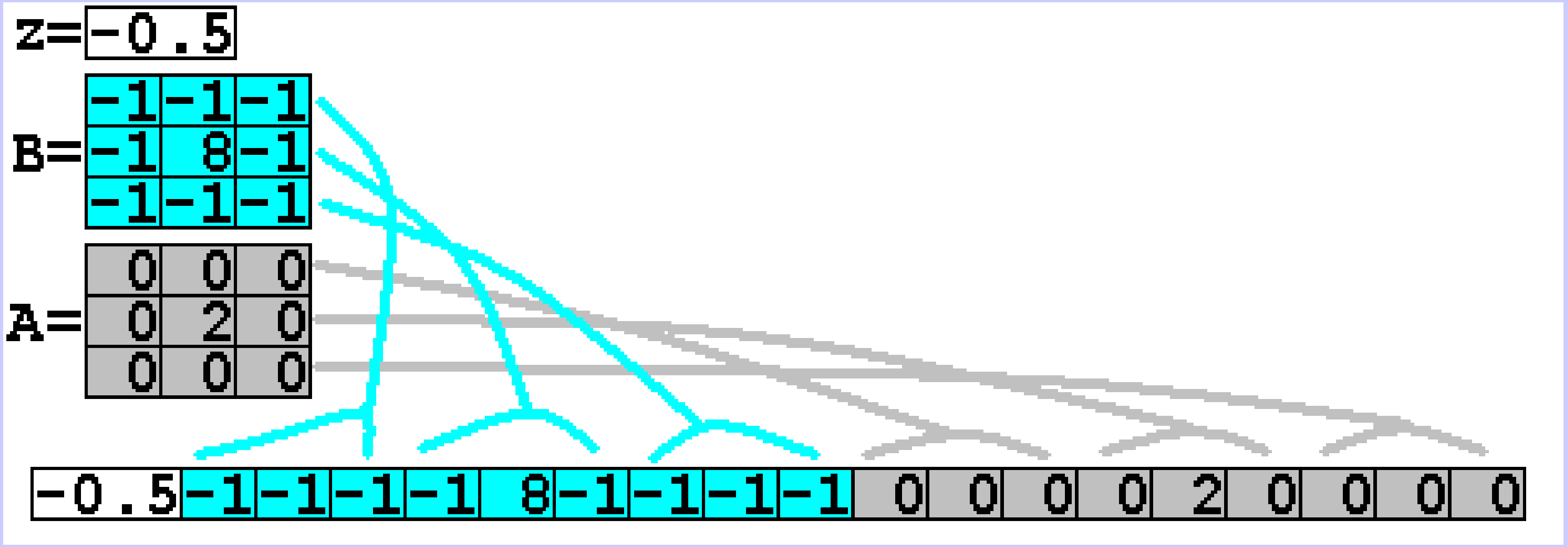
$$x_{ij}(t+1) = x_{ij}(t) + \sum_{k=-r}^{r}\sum_{l=-r}^{r} A_{ij;i+k,j+l}\, y_{i+k,j+l} +$$

$$+ \sum_{k=-r}^{r}\sum_{l=-r}^{r} B_{ij;i+k,j+l}\, u_{i+k,j+l} + z$$

Weights A and B are invariable and their selection depends on the problem to be solved.

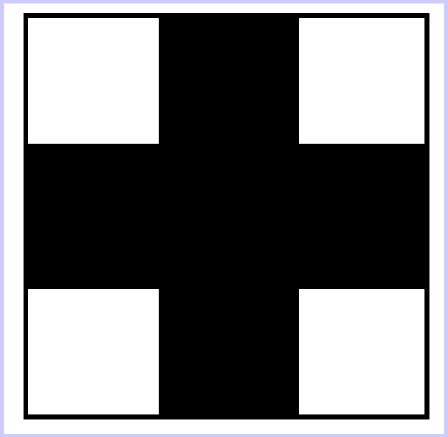
Representation

Matrix and vector (genotype) notation of the weights



Representation

Detection of composed binary pattern in the input.

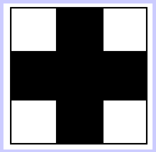


Representation

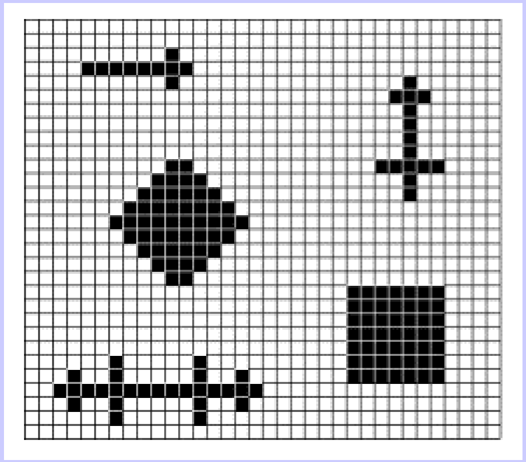
Rectangular CNN grid, weight matrices and bias

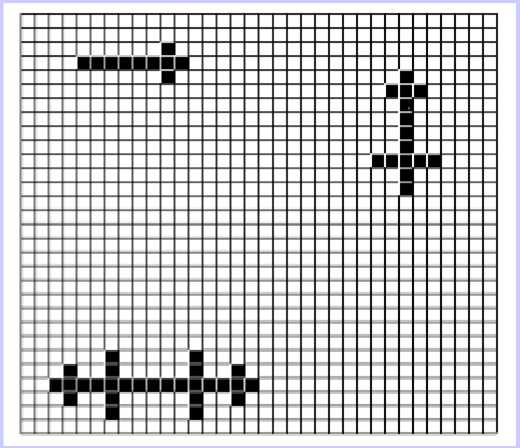
$$A = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 3{,}7 & 1 \\ 1 & 1 & 1 \end{bmatrix} \qquad B = \begin{bmatrix} -0{,}1 & 0{,}1 & -0{,}1 \\ 0{,}1 & 10{,}0 & 0{,}1 \\ -0{,}1 & 0{,}1 & -0{,}1 \end{bmatrix} \qquad z = 0$$

Representation

Extraction of the shapes containing 

Input pattern Output after processing





Applications

Cellular neural networks are used in many areas, but mainly in:

- image processing
- feature extraction
- modeling physical phenomena

Applications

But also:

missile tracking, flash detection, level and gain adjustments, color constancy detection, contrast enhancement, deconvolution, image compression, motion estimation, image encoding, image decoding, image segmentation, orientation preference maps, pattern learning/recognition, multi-target tracking, image stabilization, resolution enhancement, image deformations and mapping, image inpainting, optical flow, contouring, moving object detection, axis of symmetry detection, and image fusion.

Applications

- Are relatively simple for realization (small number of connections).
- Are able to perform parallel and distributed processing which yields to high computational power.

Multi-Layer CNN

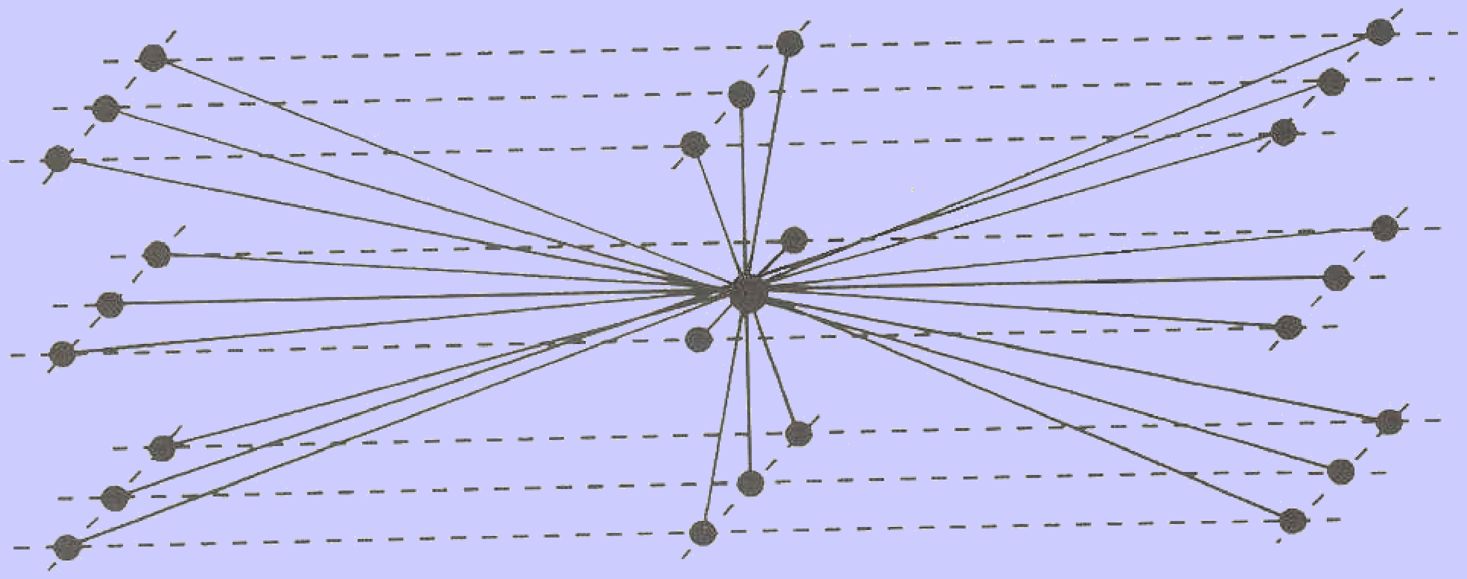
In multi- layer structure instead of scalar values :

- – state x_{ij},
- – output signal y_{ij},
- – input signal u_{ij},

we have vectorial: $\boldsymbol{X}_{ij}$, $\boldsymbol{Y}_{ij}$ i $\boldsymbol{U}_{ij}$.

Multi-Layer CNN

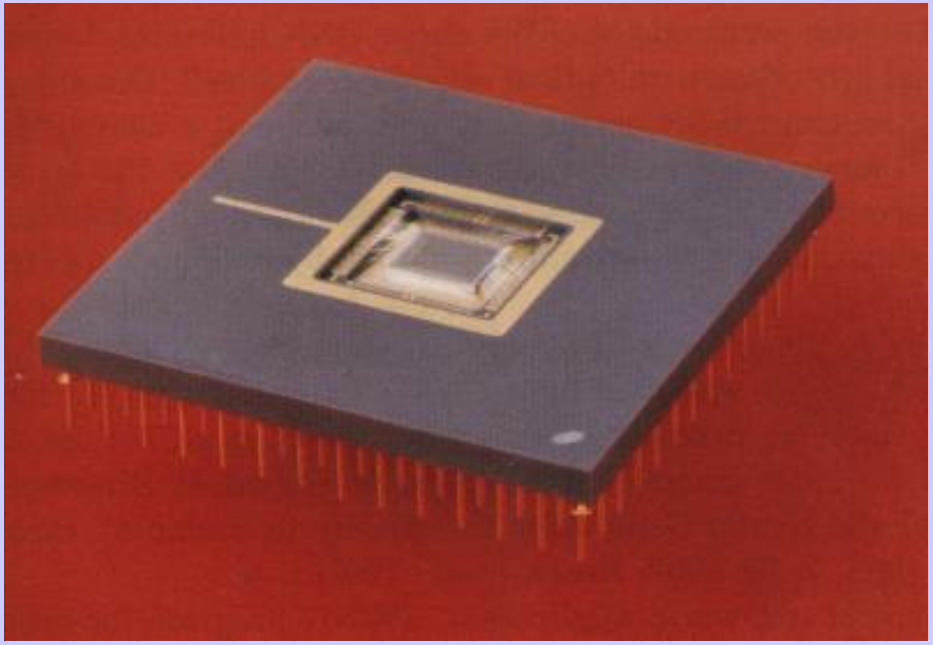
Example of connections of single cell in the middle layer of three layer cellular network of radius r = 1



source: T.Kacprzak, K.Ślot, *Sieci neuronowe komórkowe*, PWN 1995.

Technical realisation

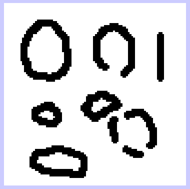
- 5.5 mm x 4.7 mm
- 0.3 W
- 60 000 transistors
- 16 x 16 cells
- Programmable weights

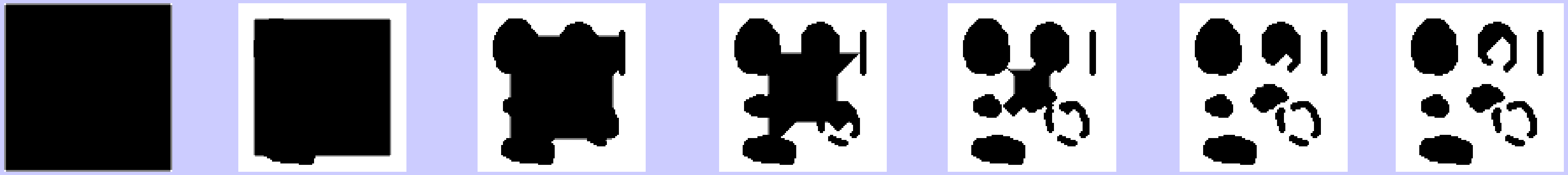


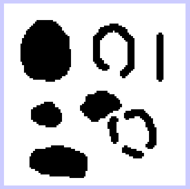
For 100 x 100 cell expected calculation speed **10^{12}** instructions per second

Example

Filling of closed region


input signal (100 x 100)


t = 0 t = 10 t = 20 t = 30 t = 40 t = 50 t = 60


final output signal, t = 71

Potential use

Linear templates B 3x3, A 1x1, binary patterns

image segmentation	separate points removal	one-side edge detection
vertical translation	NOT operation	corner detection
horizontal translation	AND operation	image erosion
diagonal translation	OR operation	image fusion
separate points remaining	edge detection	image compression

Potential use

Linear templates B 3x3, A 3x3, binary patterns

shadow casting in define direction	lines detection parallel to diagonal
vertical gap detection	detection of fully filled objects
diagonal gap detection	filling of closed shapes
separate points remaining	edge detection

Potential use

Linear templates B 3x3, A 3x3, gray-levels patterns

half-toning
inverse half-toning
texture extraction of similar shade

Nonlinear templates B 3x3, A 3x3, binary patterns

histogram creation
pixel-wise parity detection (XOR for neighbor pixels)
row-wise parity detection

Potential use

Nonlinear templates B 3x3, A 3x3, gray-levels patterns

contour detecton
region detection of similar gray-levels

Potential use

Linear templates B 3x3, A 3x3, with time-delay

object movement detection (8 basic directions)
object movement detection with filtering of stationary objects
object movement detection with predefined speed or slower

Linear templates B 5x5, A 5x5

de-blurring

CNNs scientific applications

- feature extraction & classification
- motion detection & estimation
- collision avoidance
- object counting & size estimation
- path tracking
- detecting minima and maxima
- detecting area with gradients that exceed a given threshold
- thermographic maps
- antenna-array images
- medical maps and images

References

- Chua, Leon O. and Yang, Lin (1988). *Cellular neural networks: theory*. IEEE Transactions on Circuits and Systems 35(10): 1257-1272.
- Chua, Leon O. and Roska, Tamas (1993). *The CNN paradigm*. IEEE Transactions on Circuits and Systems 40(3): 147-156.
- L. O. Chua, *CNN: A paradigm for complexity*, World Scientific 1998.