Neocognitron

Marek Kukačka

MFF UK

March 3, 2011

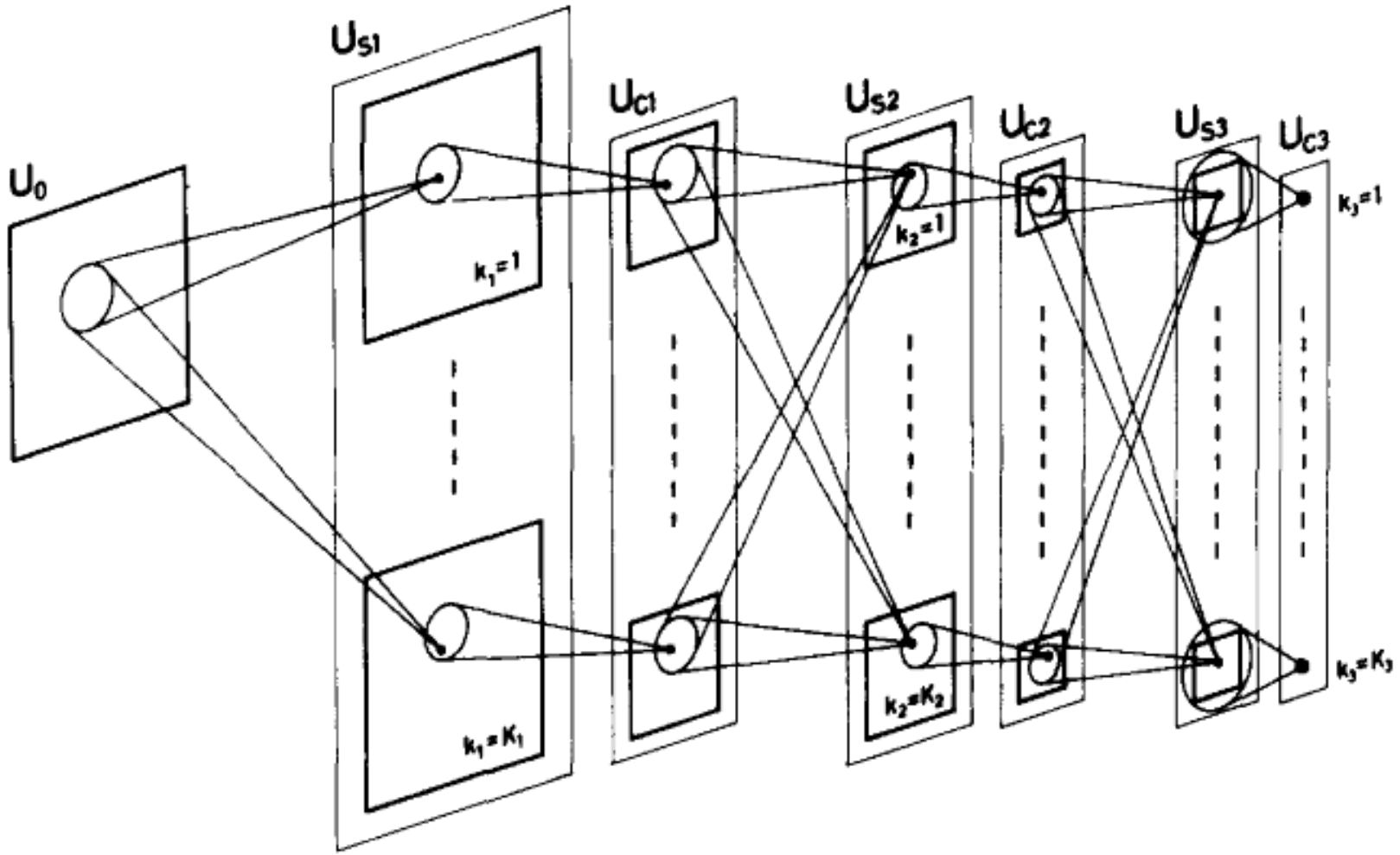
Content

Introduction

Neocognitron model:

- introduced by Kunihiko Fukushima in 1980
- a multi-layered hierarchical neural network
- inspired by the mammalian visual cortex
- improvement over older Cognitron model, adding robustness

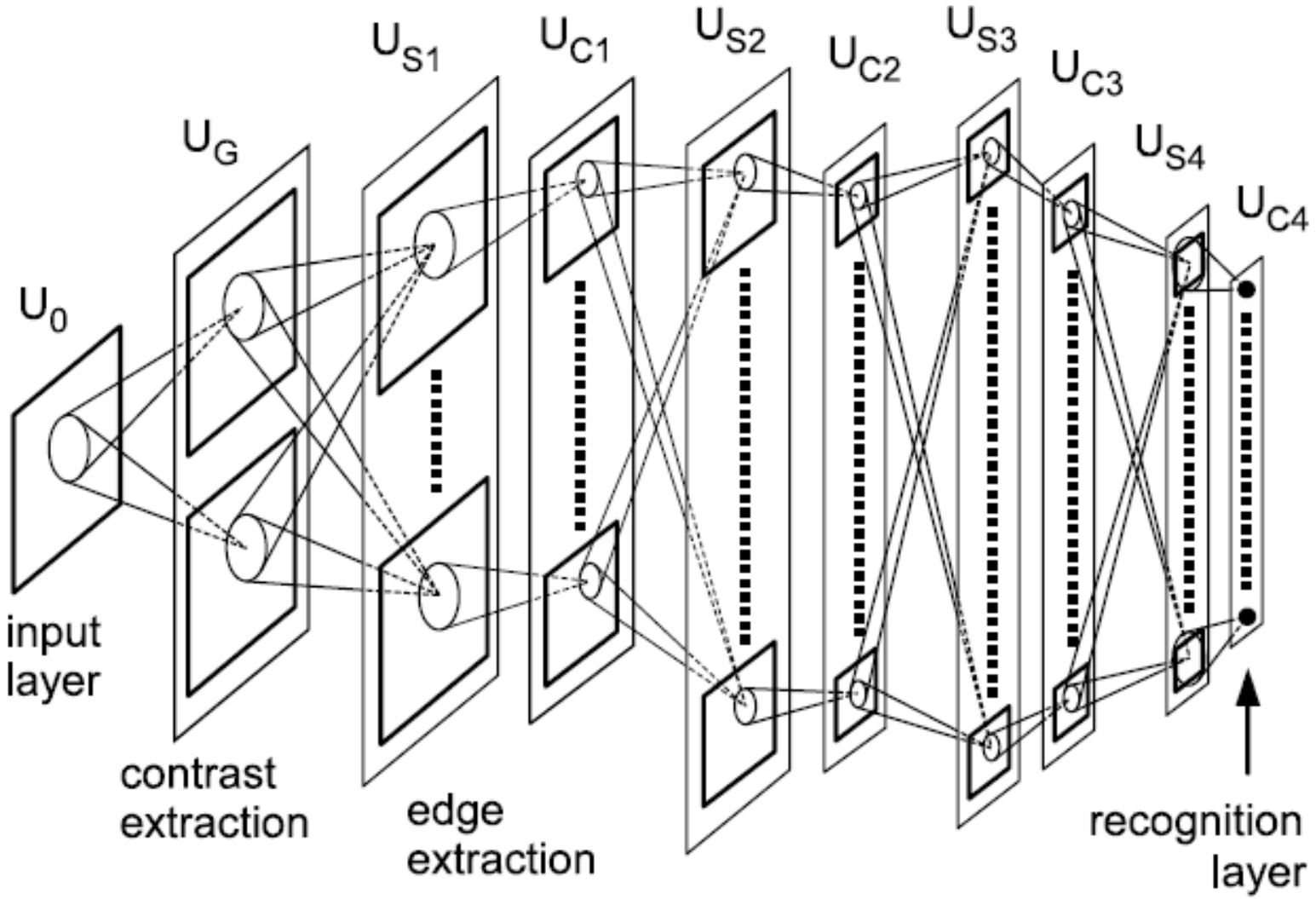
Network structure-1982



Network structure-1982

- alternating S-layers and C-layers
- 3 stages
- limited number of cell-planes in each layer

Network structure-2010

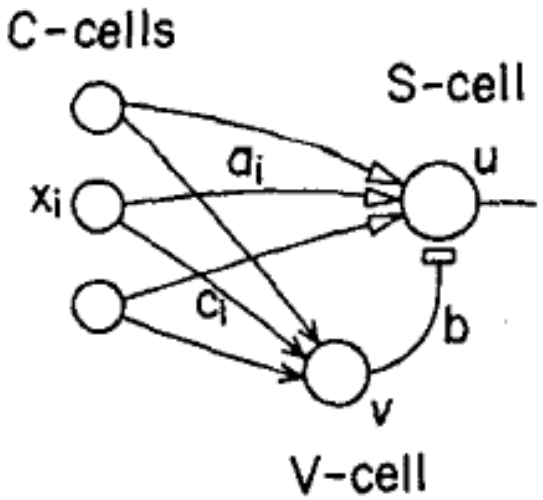


Network structure-2010

- 4 stages of S-C layer pairs
- new layer U_G for contrast extraction
- number of planes in intermediate S-layers determined by self-organization

S-cells

- only cells in the network with *flexible* weights
- limited perception window, defined by perception radius
- grouped in *cell-planes*, in which all cells have the same weights
- their response function has developed since the introduction of the model, but stays approximately the same



S-cell's output function (1982)

$$u = r \cdot \varphi\left[\frac{1 + \sum_i a_i x_i}{1 + \frac{r}{1+r} bv} - 1\right]$$

where

- $\varphi(x) = max(x, 0)$
- r is cell's saturation parameter
- a_i are cell's excitatory weights, with i ranging over cell's perception radius and all cell-planes in the lower layer
- b is the inhibitory weight
- v is the input of accompanying *V-cell*

S-cell's output function (1997)

$$u = \frac{\theta}{1-\theta}\varphi\left[\frac{1+\sum_i a_i x_i}{1+\theta bv} - 1\right]$$

where

- $\varphi(x) = max(x, 0)$
- θ is cell's threshold
- a_i are cell's excitatory weights, with i ranging over cell's perception radius
- b is the inhibitory weight
- v is the input of accompanying *V-cell*

S-cell's output function (2010)

$$u = \frac{1}{1-\theta}\varphi\left[\frac{\sum_i a_i x_i}{bv} - \theta\right]$$

where

- $\varphi(x) = max(x, 0)$
- θ is cell's threshold
- a_i are cell's excitatory weights, with i ranging over cell's perception radius
- b is the inhibitory weight
- v is the input of accompanying *V-cell*

V-cells

- accompany each S-cell
- have the same perception area as the S-cell
- have fixed weights c_i, which decrease with distance from the perception area's center

V-cell's output function:

$$v = \sqrt{\sum_i c_i x_i^2}$$

C-cells

- input connections only from a single S-plane (1-to-1 relationship)
- generally supposed to be active whenever at least one S-cell in their perception are is active
- various output functions in the past, e.g. 1988, where fixed weights $d(i)$ decrease in the same manner as c_i:

$$u_c = \Psi\Big(\sum_i d(i)u_s(i)\Big)$$

$$\Psi(x) = \frac{\varphi(x)}{1+\varphi(x)}$$

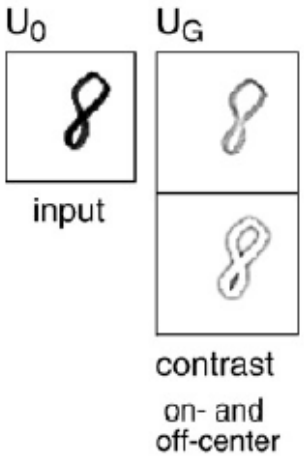
- or from 2010, where inhibitory surround is included in the weights $d(i)$:

$$\Psi(x) = \sqrt{x}$$

G-cells

- introduced in 2003
- form two cell-planes (k=1,2) in the contrast-extracting layer U_G, which is the first layer above input
- perform on-center and off-center contrast extraction
- their fixed weights g_i form the shape of the Mexican hat, centered on the perception area of the cell

$$u_G = \varphi\left[(-1)^k \sum_i g_i u_i\right]$$



Learning

- first version of Neocognitron (1982) used only *self-organization* for learning
- in later versions, several forms of *supervised learning* were added

Learning - Self-organization

Excitatory weights of an S-cell:

$$\Delta a_i = q \cdot c_i \cdot x_i$$

In later articles, the learning parameter q has been removed.
Inhibitory weight of an S-cell (older):

$$\Delta b = q \cdot v$$

Inhibitory weight of an S-cell (newer - directly computed):

$$b = \sqrt{\sum_k \sum_i \frac{a_{i,k}^2}{c_i}}$$

Learning - vector notation

Another way to look at how the network is learning.

Let $\vec{X} = \sum_p \vec{x}^{(p)}$ be the sum of training vectors for one cell-plane, called the *reference vector* of this plane.

Lets define *weighted inner product* as $(\vec{x}, \vec{y}) = \sum_i c_i x_i y_i$ and norm $||x|| = \sqrt{(x, x)}$.

Since $\Delta a_i = c_i x_i$, we get $a_i = \sum_p c_i X_i^{(p)}$, therefore $\sum_i a_i x_i = (\vec{X}, \vec{x})$.

Also, $b = \sqrt{\sum_i \frac{a_i^2}{c_i}} = ||X||$ and $v = \sqrt{\sum_i c_i x_i^2} = ||x||$.

Learning - vector notation contd.

We can then rewrite the (2010) S-cell output function as

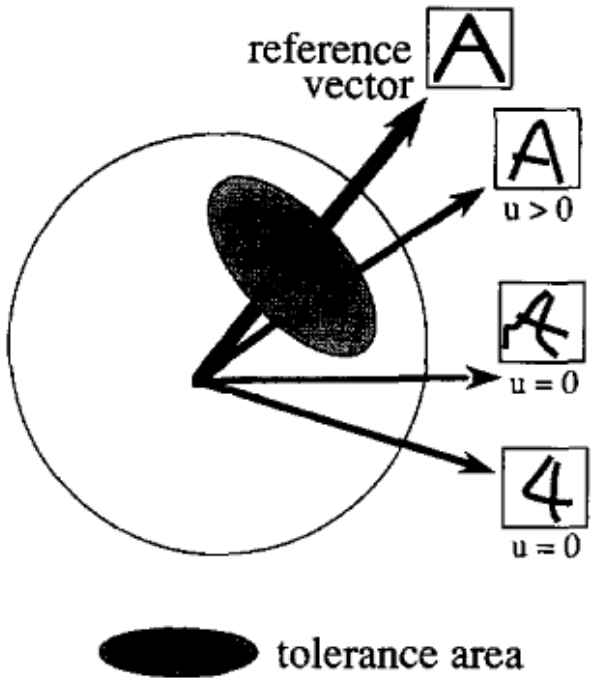
$$u = \frac{1}{1-\theta}\varphi\left[\frac{\sum_i a_i x_i}{bv} - \theta\right] = \frac{1}{1-\theta}\varphi(s-\theta)$$

where

$$s = \frac{\sum_i a_i x_i}{bv} = \frac{(\vec{X}, \vec{x})}{||X|| \cdot ||x||}$$

is the *resemblance* of the cell's reference vector and the presented pattern.

Learning - vector notation contd.



Learning - Seed-cell selection

Cells can be thought of as organized into *hypercolumns*, with cells sharing their perception area being in the same hypercolumn. Selection of cells for adaptation, the so-called *seed-cells*, is as follows:

- in each hypercolumn, the cell with the highest output value is selected as the *seed candidate*
- in each cell-plane, the seed candidate with the highest output is selected to be the seed-cell. Therefore, there can be at most one seed-cell in each cell-plane.
- the selected seed-cells' weights are adapted, and along with them, all cell in their cell-plane have their weights changed in the same manner
- if there is no seed-candidate in a cell-plane, then cells in this cell-plane are not adapted in this step

Learning - adding new cell-planes

Adding new cell-planes to a layer was proposed in (1996,1997) with the following procedure:

- if there is a location in the layer's input, with non-zero values, for which all cells are silent, new cell-plane is added to the layer
- weights of cell in the new cell-plane are set to react to the pattern found at this location
- a so-called *seed-selection* plane has been introduced to simplify the search for such locations

In (1996), the addition of a new cell-plane was done by utilizing not-yet-adapted cell-plane.
The final number of cell-planes is influenced by the selectivity threshold during training (more detail later).

Winner-kill-loser

Slightly different adaptation rule proposed in (2010):

- seed-cells are selected in the order of their descending output
- a cell is selected as a seed only if no cell was previously selected from the same cell-plane, and only if it has the highest output in its hypercolumn
- if there are non-silent cells on the location of the selected seed, they are *removed along with their cell-plane*

Together with adding new cell-planes, this rule moderates the resulting size of the layer, ensuring that each feature is recognized by only a single cell-plane.

Learning with a teacher

Supervised learning is applied in several forms during the training of the network:

- *edge-extracting layer* U_{S1} (1988) - teacher presents an edge with certain orientation and points out the position of the feature, by which a seed-cell is selected
- *highest stage S-layer* (2003) - if the winner of the competition has the wrong label, new S-plane is added with the right label for this particular pattern
- in older versions of the model - connecting S-planes belonging to the same feature to one C-plane

Learning - general comments

- adaptation of a layer is performed only after adaptation of all lower layers has finished
- only a small number of presentations of the training set is necessary (e.g. 4 repetitions)

No-response detector

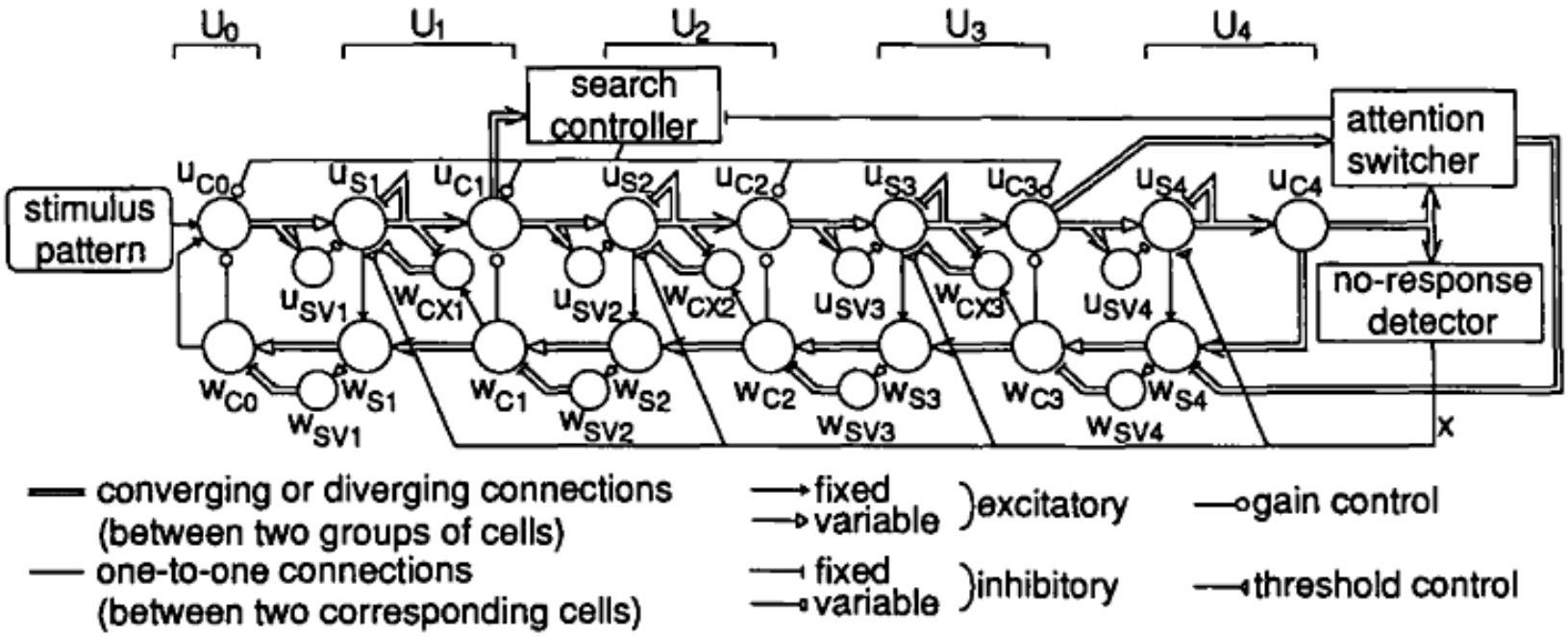
- introduced in (1988), along with selective attention
- in all cells in the output layer are silent, this detector starts to gradually lower thresholds of all cells in the network, until at least one output cell responds
- used to recognize even very deformed or noisy patterns

Selective attention and restoring patterns

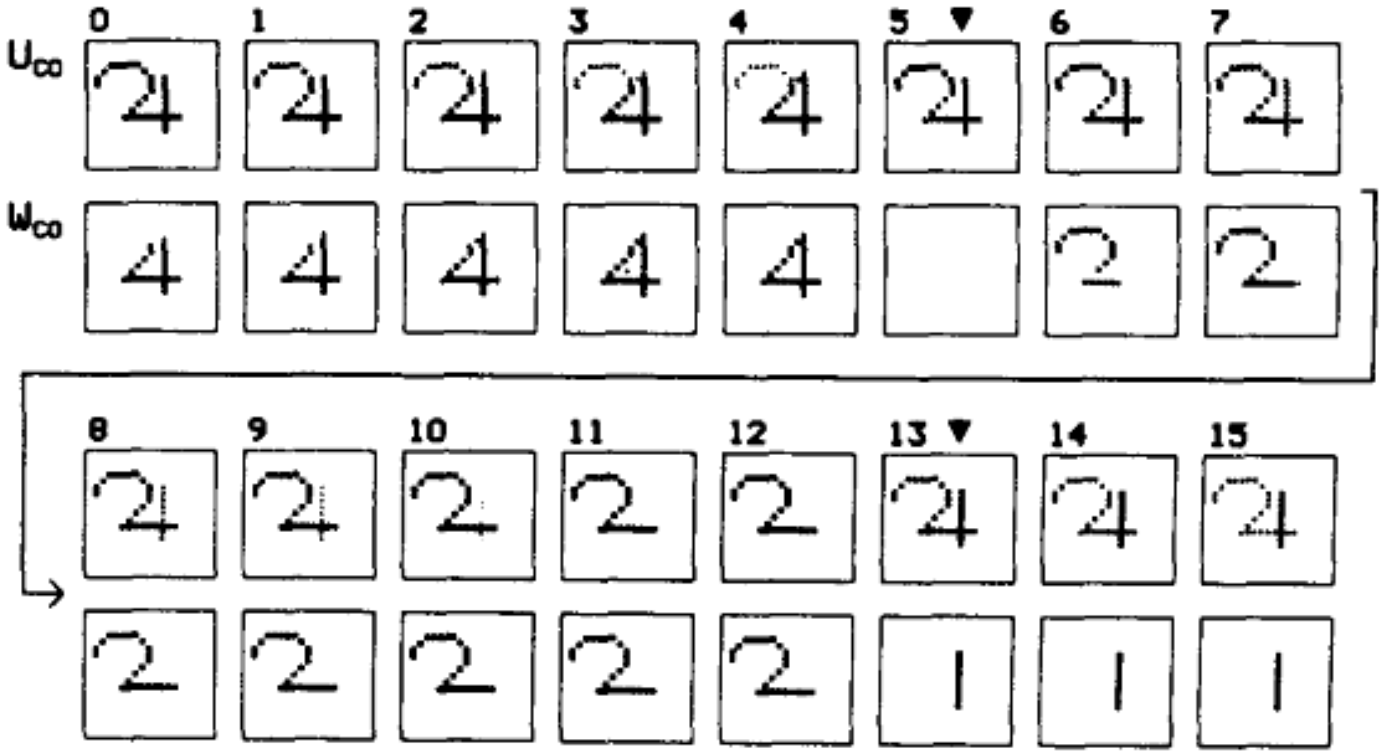
- introduced in 1986, more detail provided in (1988,1993), similar idea used in (2005) for restoring occluded patterns
- backward (or top-down) pathways are added to the network
- the backward flow, gated by the activated forward pathways, directs the network's attention to one (learned) pattern
- also, the backward flow helps the network to recognize deformed features or add missing features to the pattern

This allows the network to recognize and restore damaged patterns. Also, the network can segment and recognize several patterns from a single input, even if the patterns overlap.

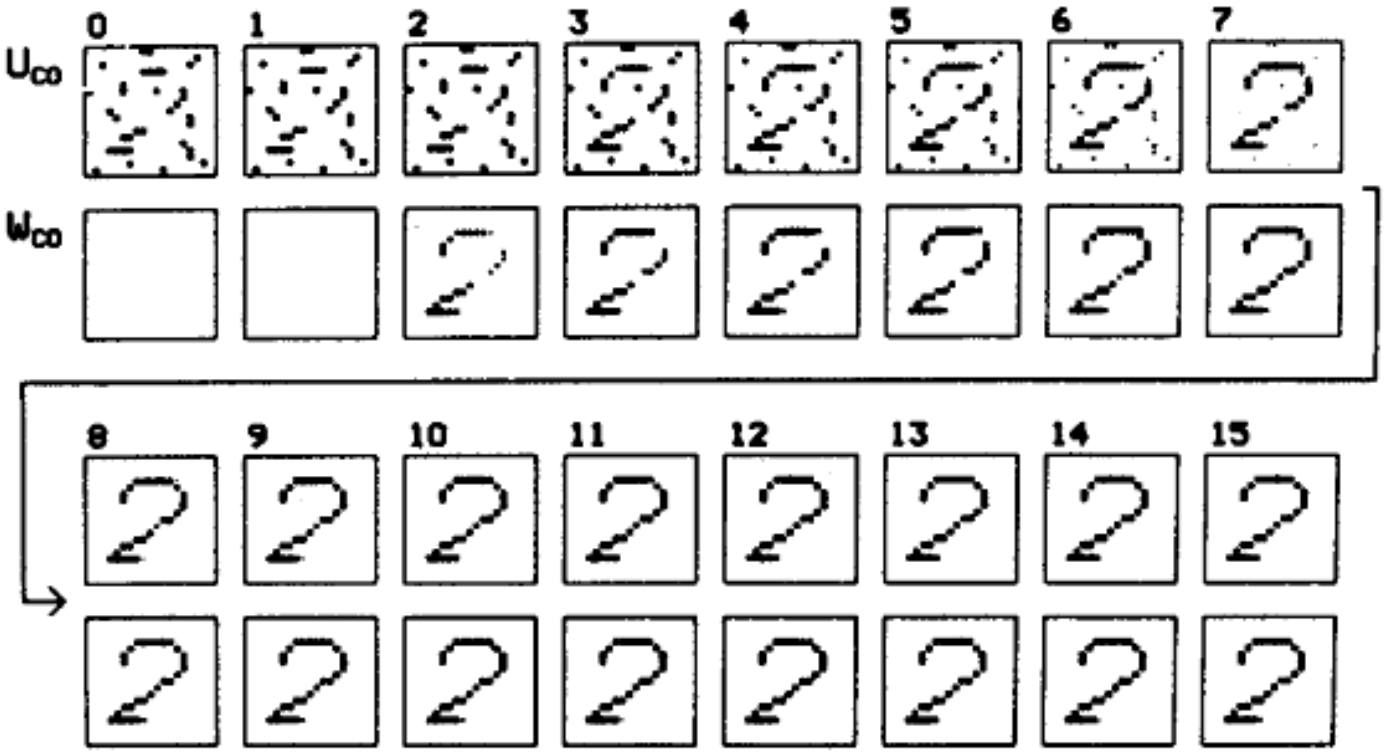
Selective attention - network structure



Selective attention - segmentation



Selective attention - restoration



Double thresholding

- proposed in (1996)
- using different threshold for learning and recognition phase
- higher thresholds for learning: higher selectivity causes more feature-extracting cells to be created
- lower threshold for recognition: lower selectivity, higher deformation tolerance, more robust network

Different layers use different thresholds, their optimal values are difficult to determine.

Inhibitory surround

- proposed in (2003)
- previously, the fixed weights of C-cells were all excitatory
- here, inhibitory rim around the excitatory weights is proposed
- this helps C-cells to better recognize bends, end-points and identical features placed close together
- this modification allowed the *bend-extracting* layer used in (1996,1997) to be removed

Computer simulations

- in the early articles, only a few patterns were used for a training set (e.g. digits 0-4)
- due to the limited size of the network (memory limits!), the network was not able to correctly learn more patterns
- in (1988), the computer used for the simulation had 16-bit processor 8086 with 348 kB memory and co-processor 8087
- in more recent articles, the network was trained with ETL1 database (3000 samples for training, 3000 for testing)
- in these cases, the layers trained with self-organization grew to the size of approx. 100 cell-planes
- with fine-tuned parameters, the network was able to reach 100% recognition rate on the training set, and 98.6% on the training set

References

- K. Fukushima, S. Miyake: "Neocognitron: A new algorithm for pattern recognition tolerant of deformations and shifts in position", *Pattern Recognition*, 15[6], pp. 455-469 (1982).
- K. Fukushima: "Neocognitron: A hierarchical neural network capable of visual pattern recognition", *Neural Networks*, 1[2], pp. 119-130 (1988).
- K. Fukushima, T. Imagawa: "Recognition and segmentation of connected characters with selective attention", *Neural Networks*, 6[1], pp. 33-41 (1993).
- K. Fukushima, M. Tanigawa: "Use of different thresholds in learning and recognition", *Neurocomputing*, 11[1], pp. 1-17 (1996).
- K. Fukushima: "Neocognitron for handwritten digit recognition", *Neurocomputing*, 51, pp. 161-180 (2003).
- K. Fukushima: "Neocognitron trained with winner-kill-loser rule", *Neural Networks*, 23[7], pp. 926-938 (Sept. 2010).

References contd.

- K. Fukushima: "A neural network for visual pattern recognition", *IEEE Computer*, 21[3], pp. 65-75 (1988).
- K. Fukushima, K. Nagahara, H. Shouno: "Training neocognitron to recognize handwritten digits in the real world", *Proceedings, The Second Aizu International Symposium on Parallel Algorithms/Architectures Synthesis*, March 17-21, 1997, Aizu-Wakamatsu, Japan, pp. 292-298 (March 21, 1997), IEEE Computer Society Press, Los Alamitos, CA.
- K. Fukushima: "Restoring partly occluded patterns: a neural network model", *Neural Networks*, 18[1], pp. 33-43 (2005).