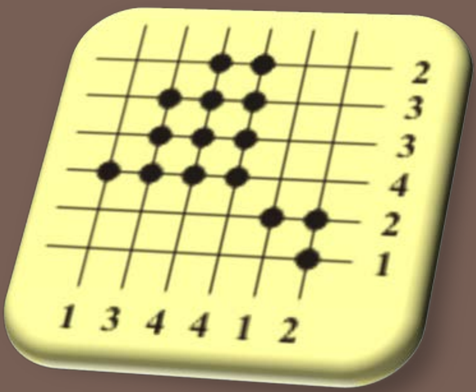
EXTRACTING GEOMETRICAL FEATURES OF DISCRETE IMAGES FROM THEIR PROJECTIONS



TAMÁS S. TASI, PHD STUDENT
DR. PÉTER BALÁZS, ASSISTANT PROFESSOR
DEPARTMENT OF IMAGE PROCESSING AND COMPUTER GRAPHICS

2012. 06. 30.

Conference of PhD Students in Computer Science

Outline

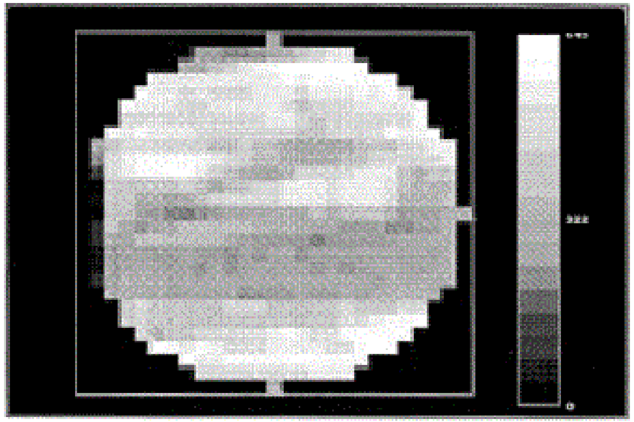
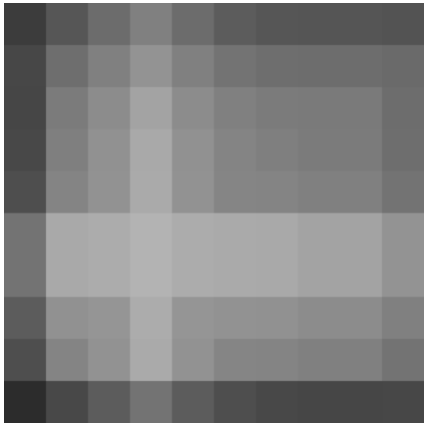
- Discrete tomography
- Geometrical properties of discrete sets
- Neural networks
- 3 investigated problems:
  1) Determining *connectedness* and *convexity* from two projections in binary images
  2) *Perimeter estimation* from two projections in binary images
  3) *Estimating the number of different intensities* in discrete images from two projections

Discrete tomography

- We assume, that the image only contains intensity values known beforehand:

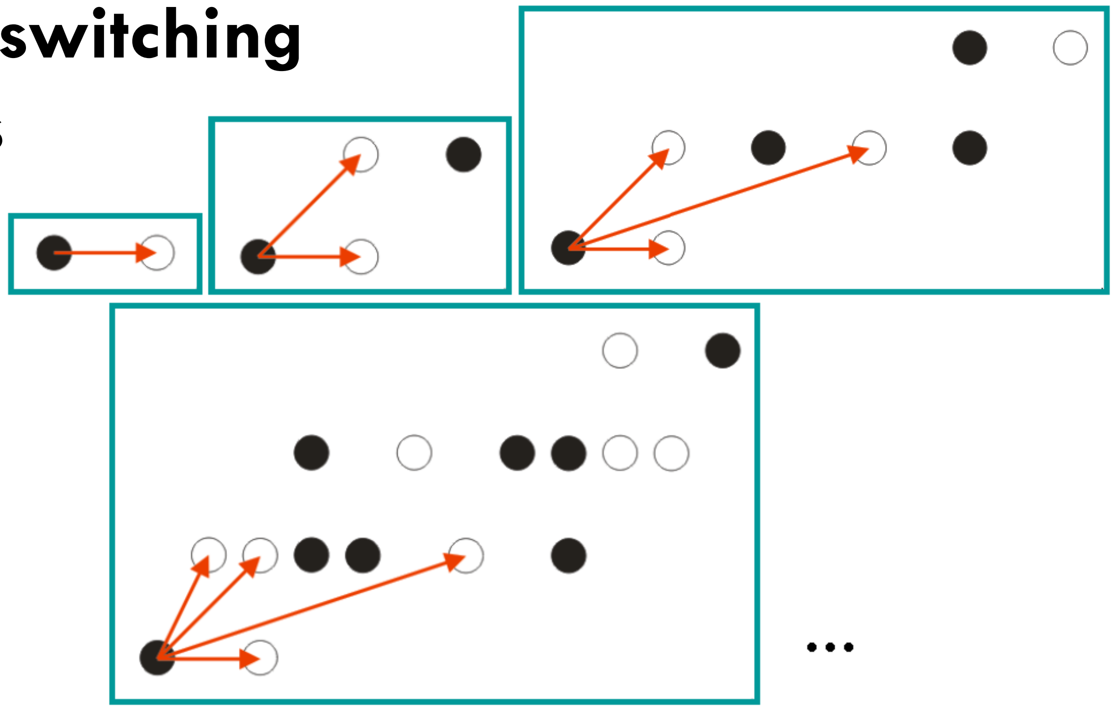
$$f : \mathbb{R}^2 \to S$$

- In case $S = \{0,1\}$, then we are dealing with *binary tomography*

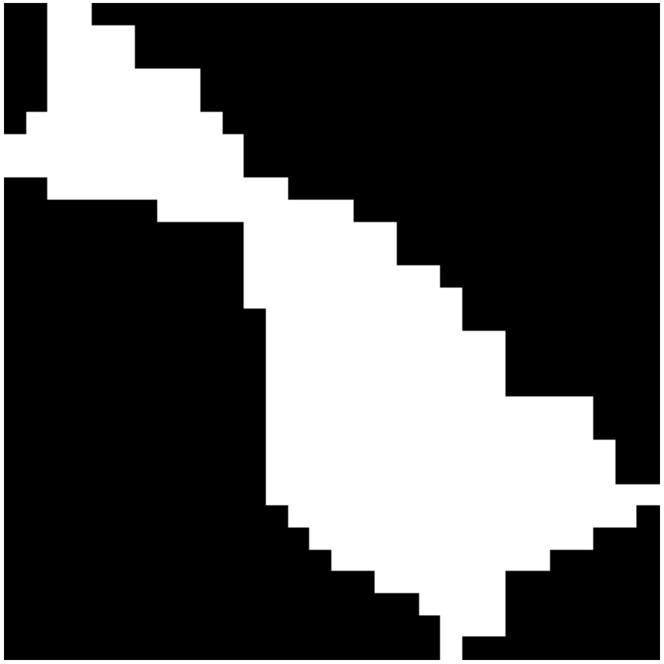

Discrete tomography

- **Small number** (<10) **of projections** are available
  - ⇨ the problem is usually underdetermined
  - ⇨ more than one possible solutions
- Presence of **switching components**



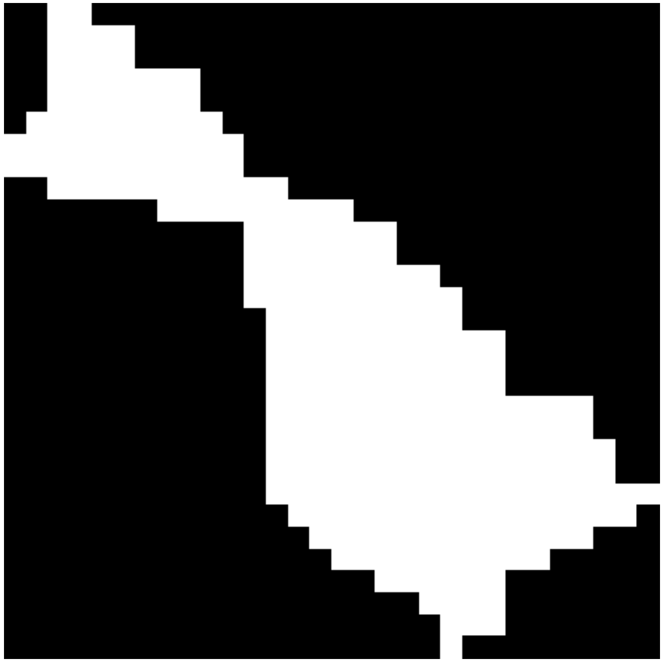
Discrete tomography

- We have to reduce the number of possible solutions:
  - with the help of a *model image*, or
  - with the aid of *a priori geometrical/topological information*



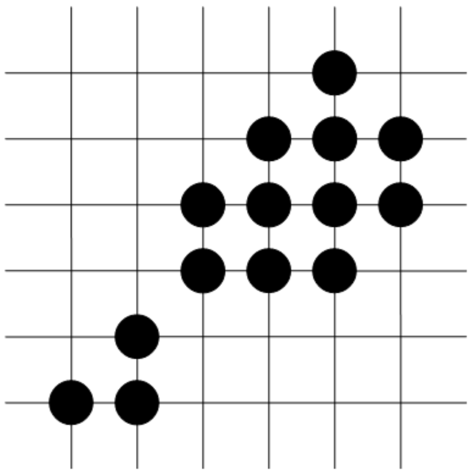

Discrete tomography

- We have to reduce the number of possible solutions:
  - with the help of a *model image*, or
  - with the aid of *a priori geometrical/topological information*

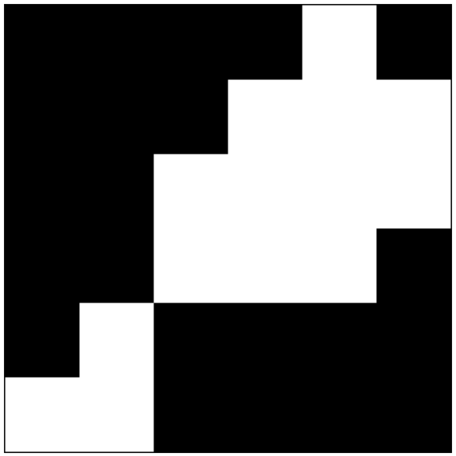



Two projections of a discrete set

- Let $F_1 \subseteq \mathbb{Z}^2$ be a so-called discrete set



$$\begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

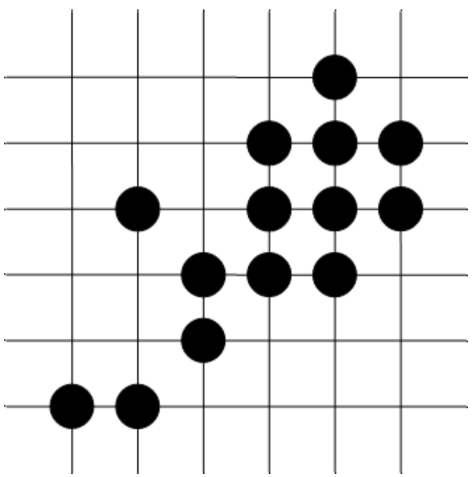


$$\mathcal{H}(F_1) = \mathbf{H}_1 = (1,3,4,3,1,2)$$

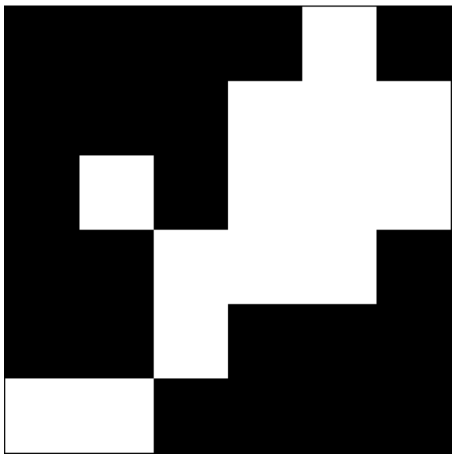
$$\mathcal{V}(F_1) = \mathbf{V}_1 = (1,2,2,3,4,2)$$

Two projections of a discrete set

- Let $F_2 \subseteq \mathbb{Z}^2$ be another discrete set



$$\begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

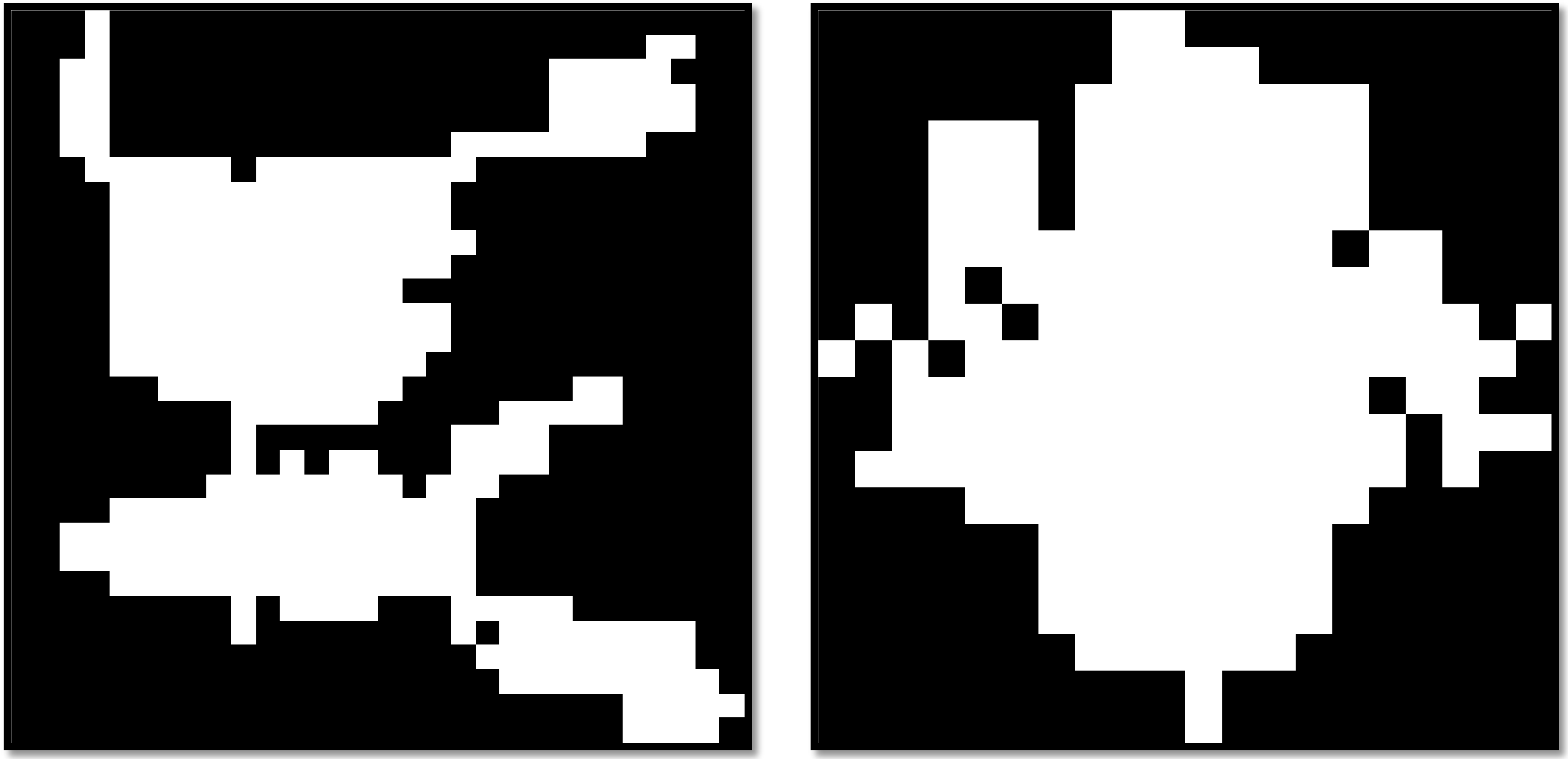


$$\mathcal{H}(F_2) = \mathbf{H}_2 = (1,3,4,3,1,2)$$

$$\mathcal{V}(F_2) = \mathbf{V}_2 = (1,2,2,3,4,2)$$

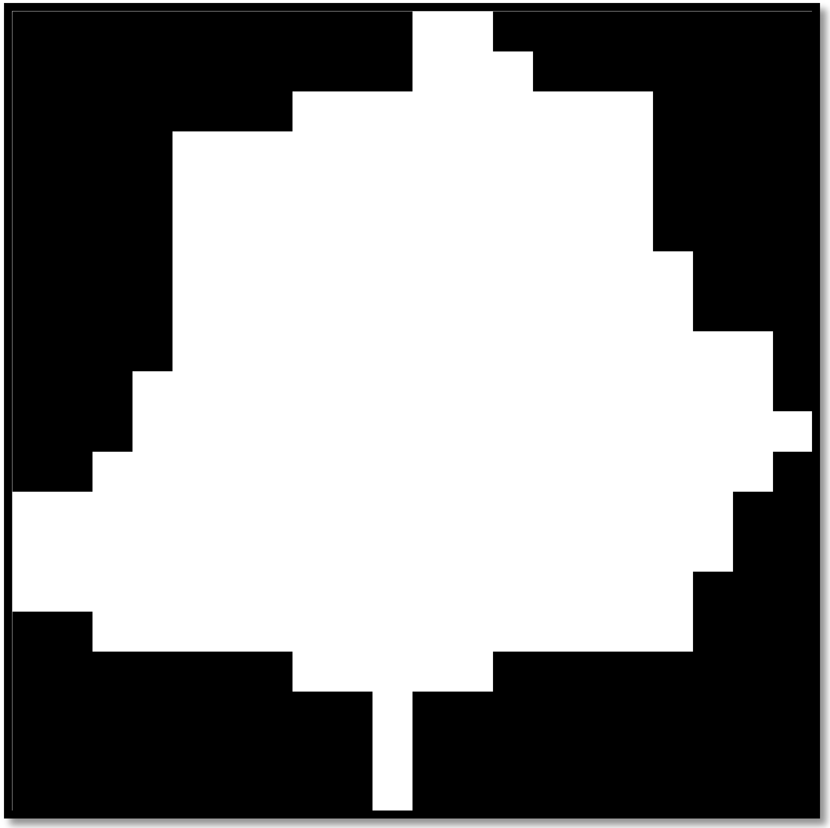
Properties of discrete sets

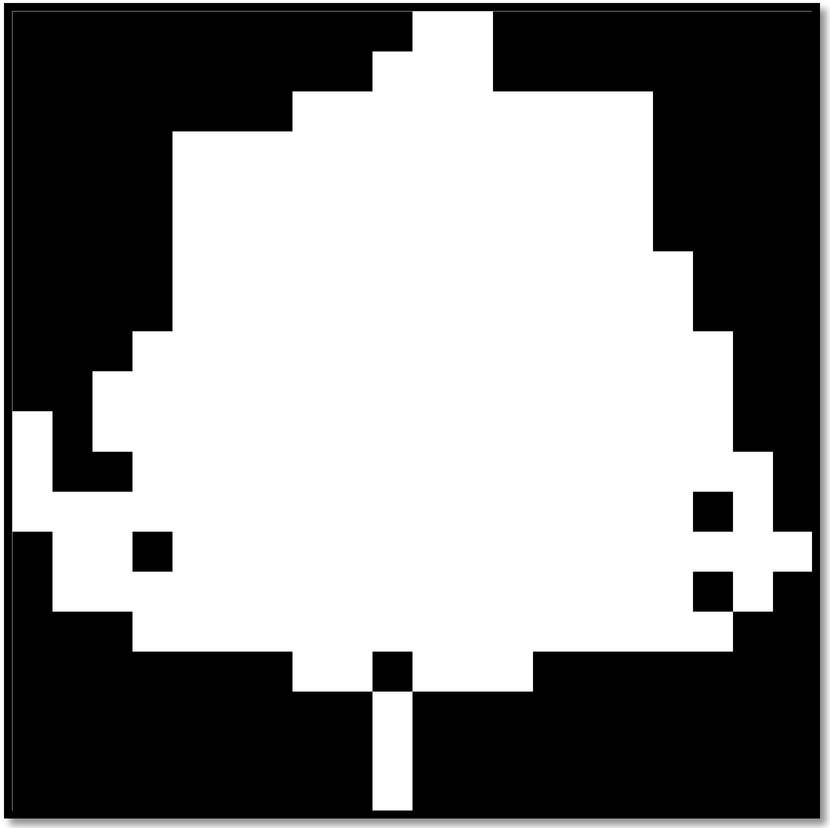
- 4/8-connectedness



Properties of discrete sets

- *h*-, *v*- and *hv*-convexity



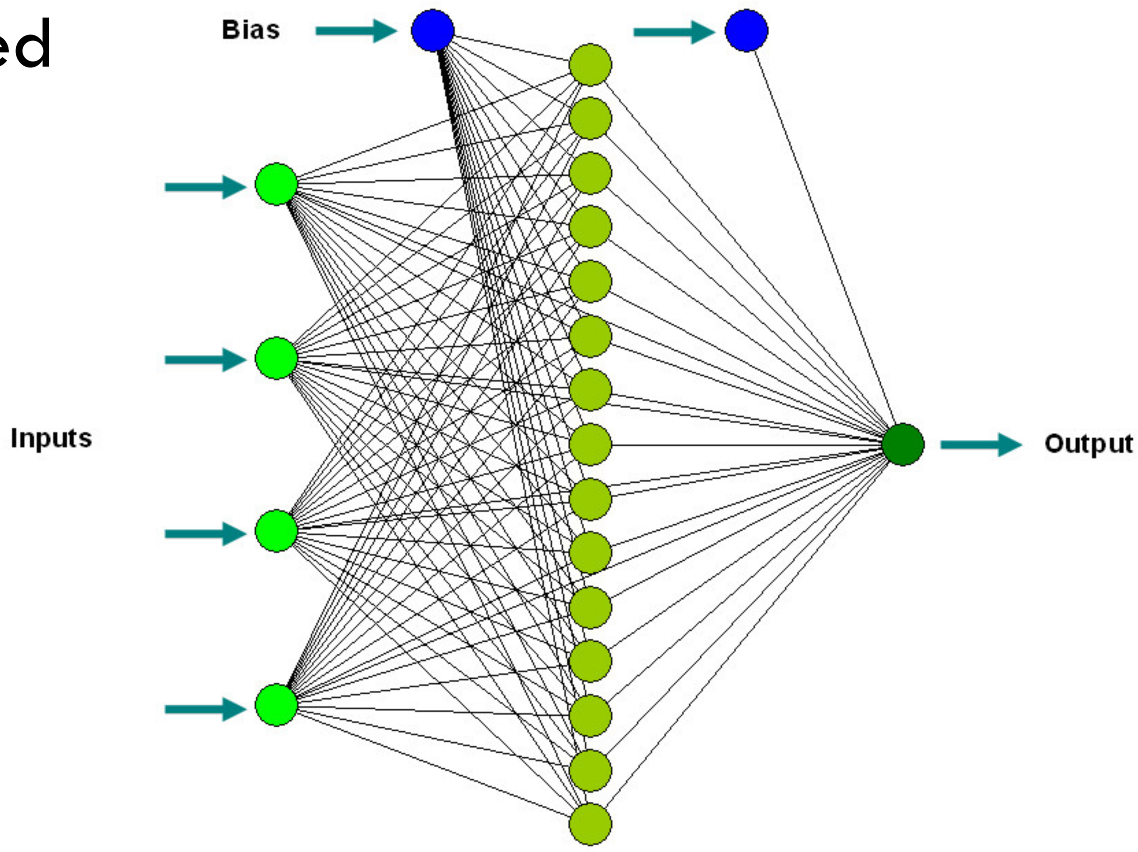


Properties of discrete sets

- Several reconstruction algorithms rely on the **prior knowledge** of these geometrical features
- Problems:
  - these are quite strict terms
  - the prior knowledge is often uncertain
  - which reconstruction method to choose is questionable
- Let’s use data, which is available <u>before</u> the reconstruction process begins
  ⇨ i.e. the **projections values**

Neural networks

- Inspired by the neural system of the human brain
- Learning algorithms that learn from a set of samples presented beforehand



Neural networks

- Chosen implementations:
  - Bobby Anguelov's C++ realization

    http://takinginitiative.net/category/artificial-intelligence/neural-networks/
  - WEKA Data Mining Software – MLP

    http://www.cs.waikato.ac.nz/ml/weka/index.html
- Common features of both:
  - backpropagation learning
  - momentum technique
  - feed-forward, 3-layer architecture
  - activation function g is a sigmoid

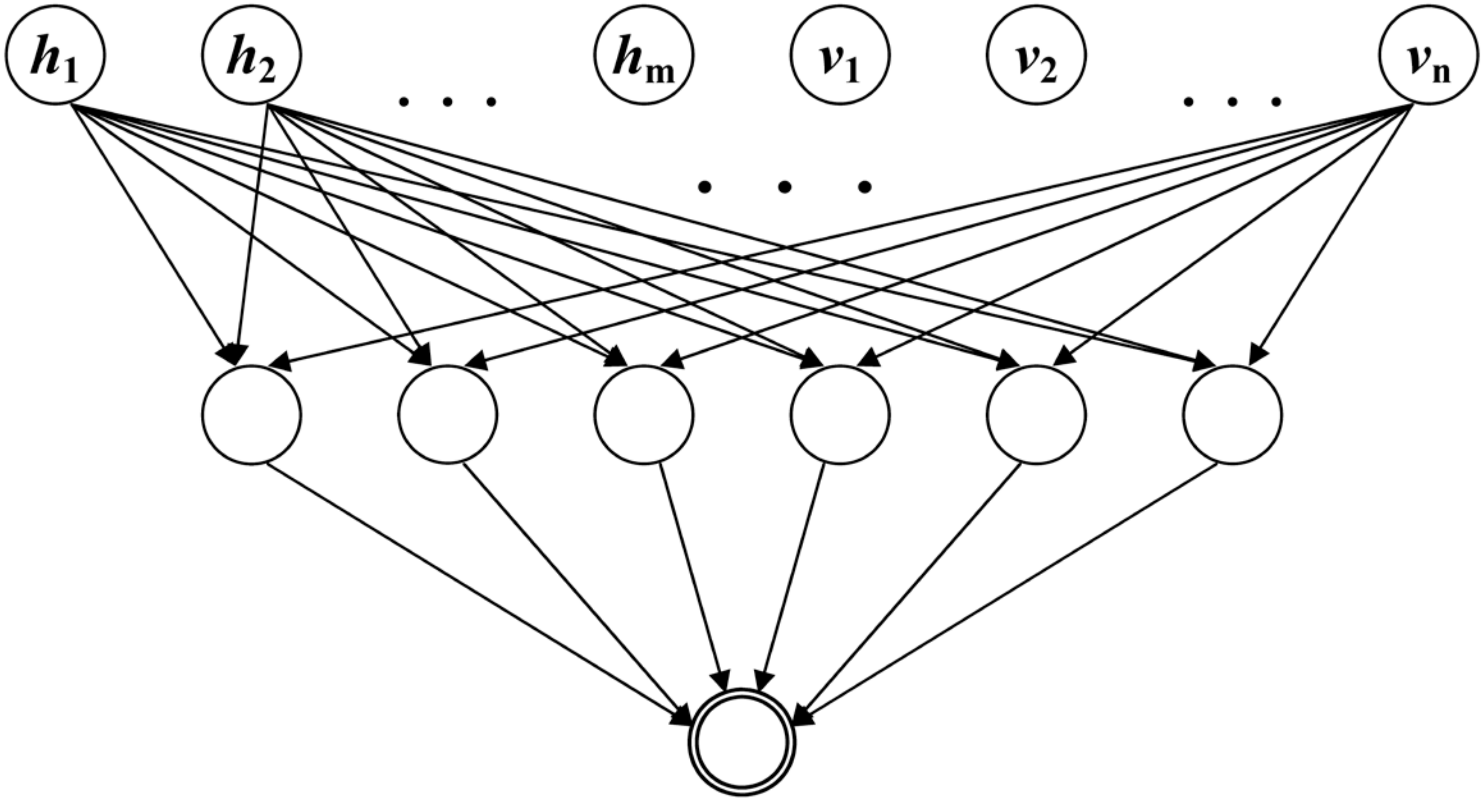
Application of neural networks

- Mostly for **reconstruction purposes** in discrete tomography
- Drawbacks:
  - one neuron often corresponds to one pixel(!)

    ⇨ network size is close to being unmanageable
  - *several million* learning samples are needed
  - 10-20 projections from different directions are necessary to obtain results of sufficient quality
- Instead of actually reconstructing the image, we try to aid the reconstruction process

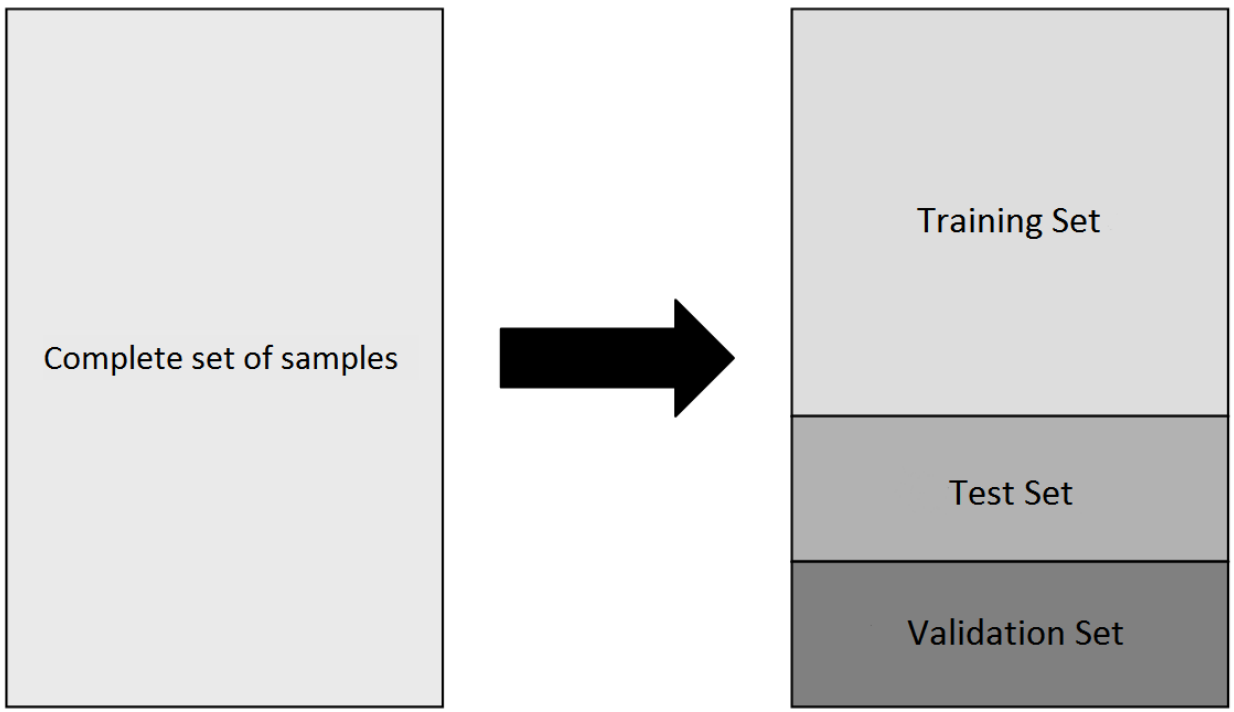
Application of neural networks

- The I/O of the neural network in case of two orthogonal projections:



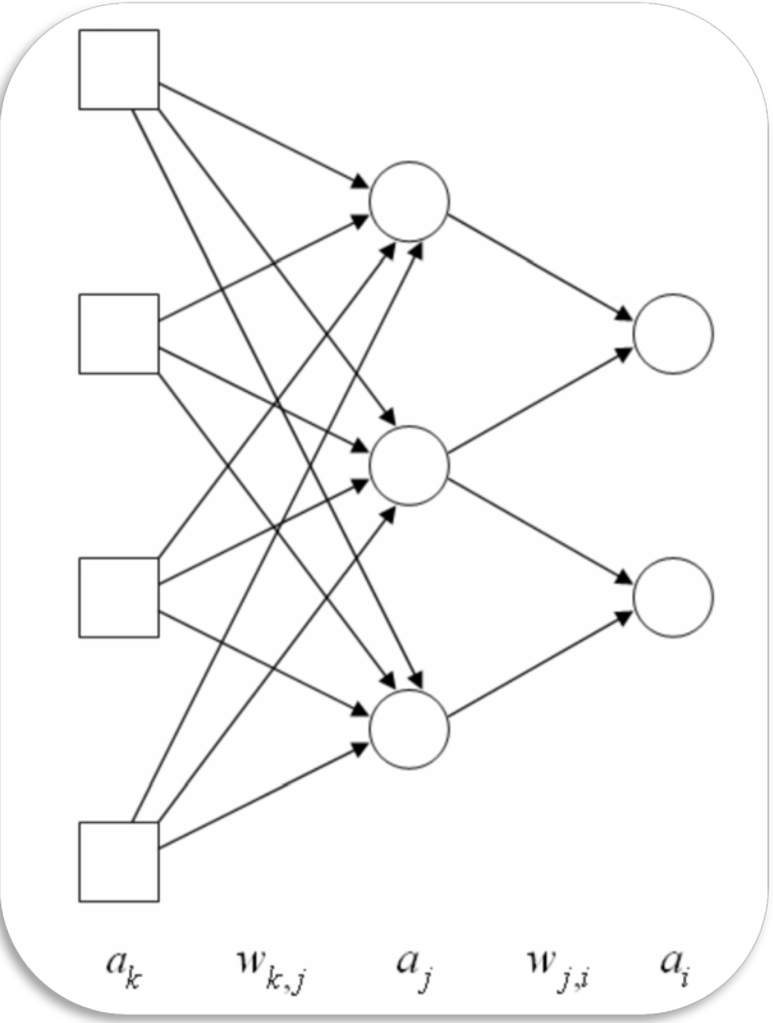
Application of neural networks

- For the learning phase:
  - generate a huge dataset of samples
  - divide it to get a training-, a test- and a validation set



Application of neural networks

- Modification of the weights of connections during the learning phase



$$w_{j,i} = w_{j,i} + \Delta w_{j,i}(t)$$

$$\Delta w_{j,i}(t) = \alpha a_j \Delta_i + \beta \Delta w_{j,i}(t-1)$$

$$\Delta_i = \mathbf{Err}_i \, g'(in_i)$$

$$w_{k,j} = w_{k,j} + \Delta w_{k,j}(t)$$

$$\Delta w_{k,j}(t) = \alpha a_k \Delta_j + \beta \Delta w_{k,j}(t-1)$$

$$\Delta_j = g'(in_j) \sum_i w_{j,i} \Delta_i$$

Application of neural networks

- Parameters to set:
  - learning rate (α)
  - momentum constant (β)
  - number of hidden neurons
  - number of epochs
  - number of training- and test samples
  - advanced data partitioning methods to use
  - how to decrease α
  - etc.

1. Connectedness and convexity

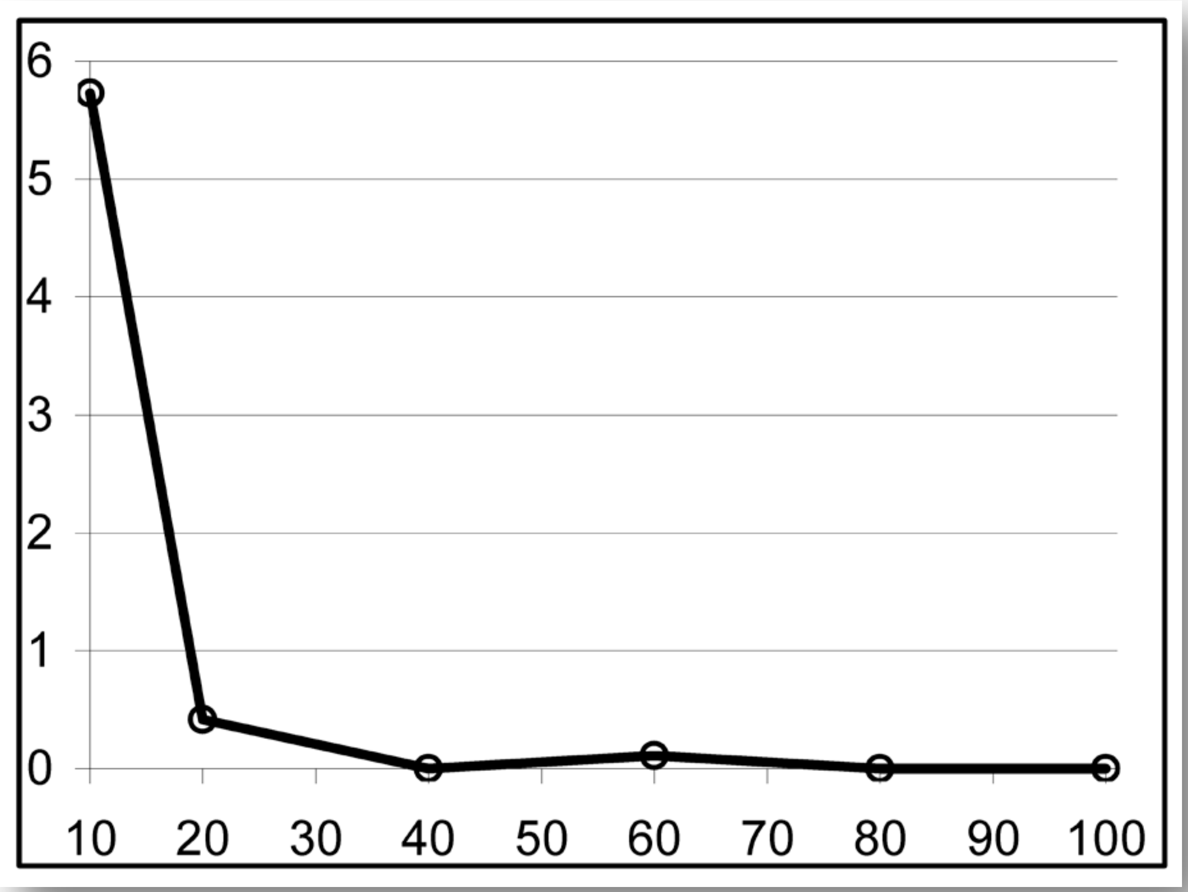
- *hv*-convex 4-conn. sets vs. random binary images
- 2880-960-960 samples in each set

Size	Hidden neurons	TSA(%)	GSA(%)	VSA(%)	Err(%)
10	4	93.819	94.167	94.271	5.729
20	6	99.931	99.688	99.583	0.417
40	8	100.0	99.896	100.0	0.0
60	8	100.0	99.792	99.792	0.108
80	8	100.0	100.0	100.0	0.0
100	8	100.0	100.0	100.0	0.0

- proved to be an easy task

1. Connectedness and convexity

- Classification error depending on the size:



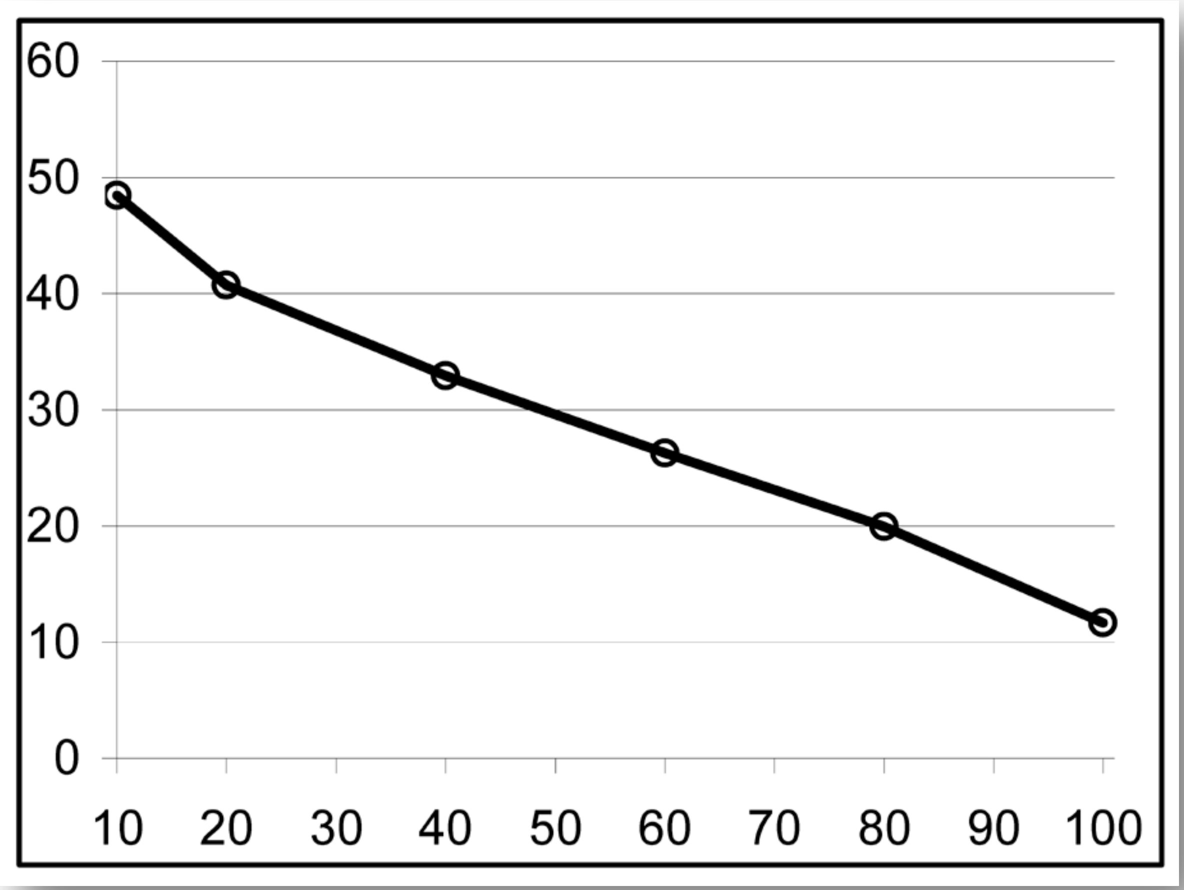
1. Connectedness and convexity

- *hv*-convex 4-conn. sets vs. discrete sets up to 4% different from these
- 2880-960-960 samples in each set

Size	Epochs	Hidden neurons	α	VSA(%)	Err(%)
10	30000	30	10^{-3}	51.5625	48.4375
10	40000	40	$10^{-3} \rightarrow \ldots \rightarrow 1.25\times10^{-4}$	51.1458	48.8542
20	30000	40	10^{-3}	59.2708	40.7202
40	3000	120	10^{-4}	67.0833	32.9167
60	2500	100	$10^{-4} \rightarrow 5\times10^{-5}$	73.7152	26.2848
80	2500	120	$10^{-4} \rightarrow 5\times10^{-5}$	80.0347	19.9653
80	2500	160	$10^{-4} \rightarrow 5\times10^{-5}$	79.9306	20.0694
100	2000	175	$5\times10^{-5} \rightarrow 10^{-5}$	88.3333	11.6667

1. Connectedness and convexity

- Classification error depending on the size:



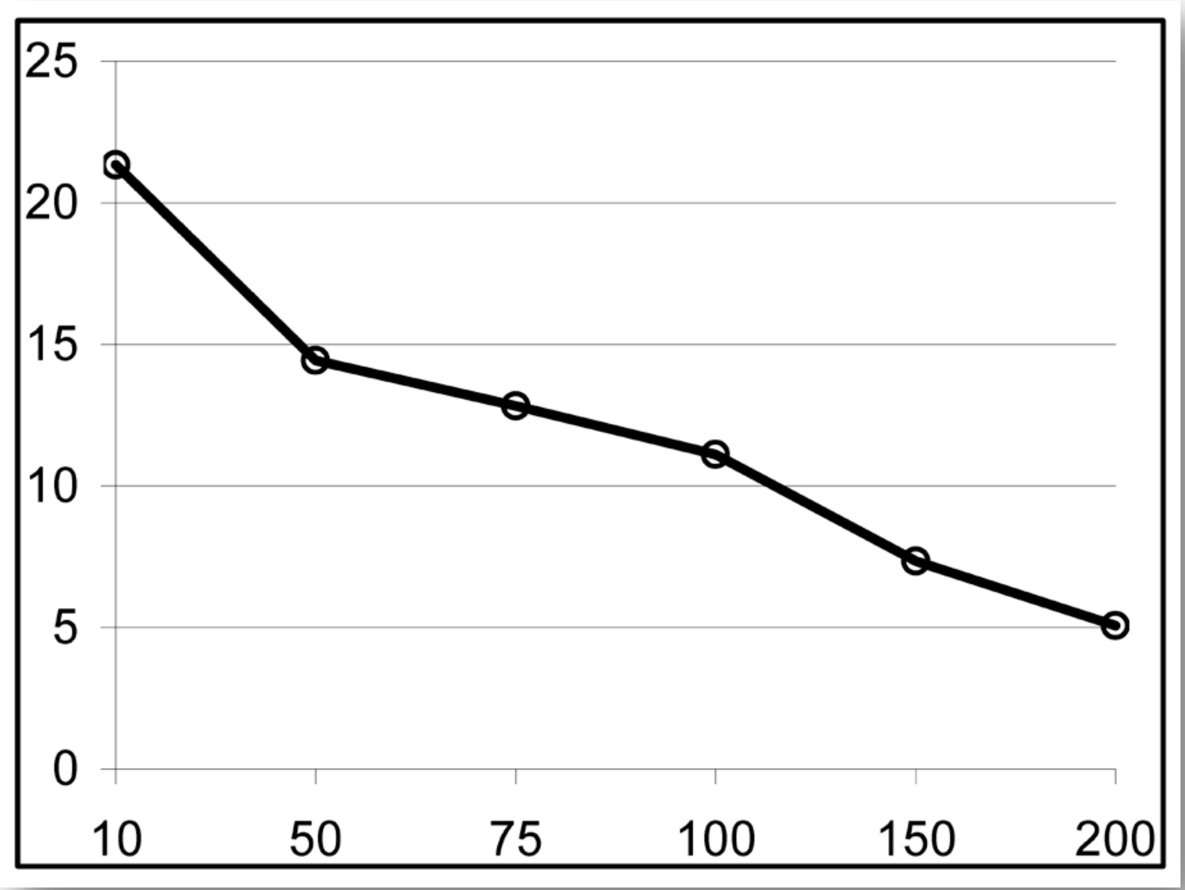
1. Connectedness and convexity

- *hv*-convex 8-, but not 4-conn. sets vs. *hv*-convex 4-conn. discrete sets
- 1800-600-600 samples in each set
- continuously growing training set

Size	Epochs	Hidden neurons	α	VSA(%)	Err(%)
10	50000	30	10^{-4}	78.6667	21.3333
50	50000	120	$10^{-3} \rightarrow \ldots \rightarrow 10^{-6}$	85.5556	14.4444
100	10000	200	$10^{-3} \rightarrow \ldots \rightarrow 10^{-7}$	88.8889	11.1111
150	7500	250	$10^{-3} \rightarrow \ldots \rightarrow 10^{-7}$	92.6667	7.3333
200	3000	300	$10^{-3} \rightarrow \ldots \rightarrow 10^{-7}$	94.9444	5.0556

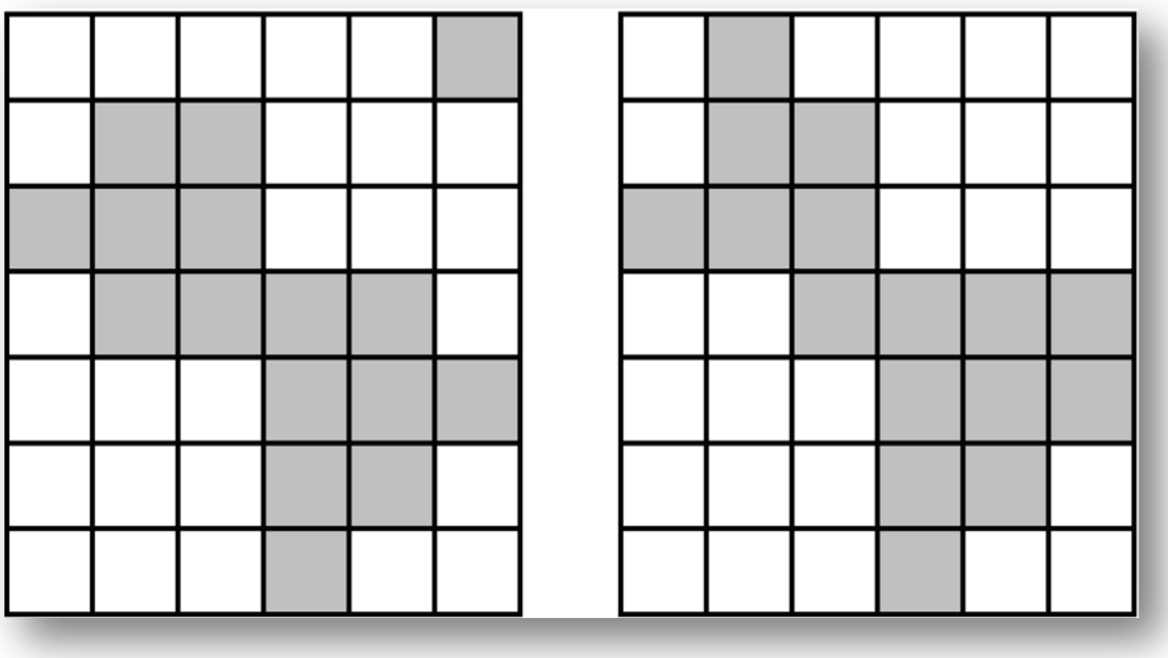
1. Connectedness and convexity

- Classification error depending on the size :



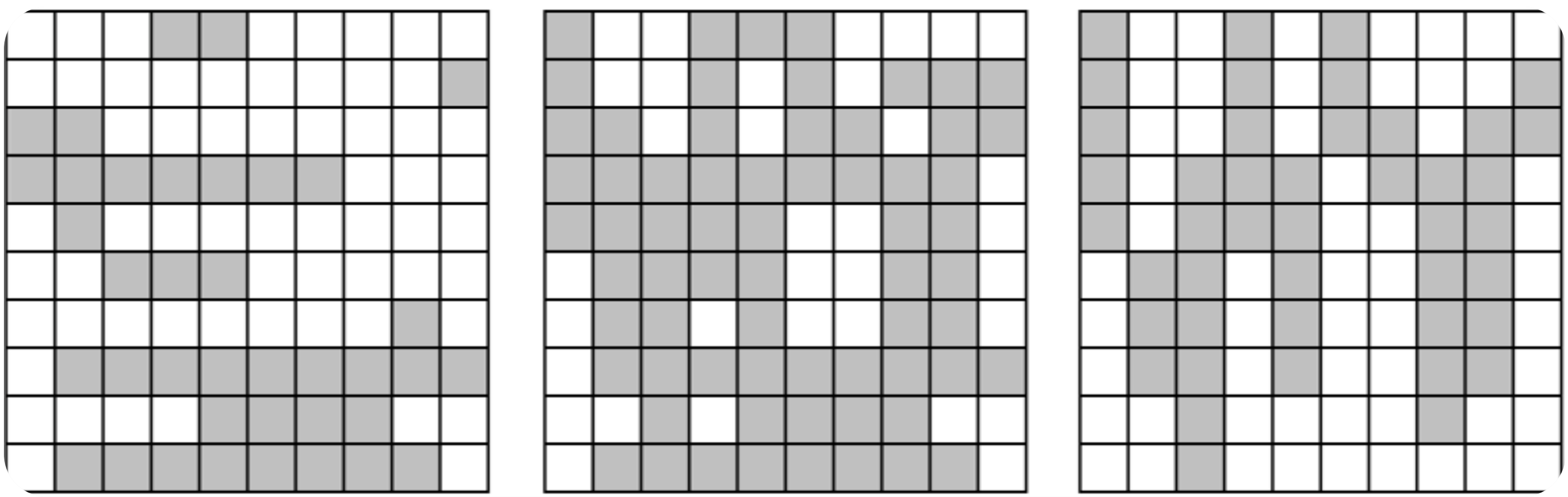
2. Estimation of perimeter

- Recently algorithms have been developed to *reconstruct* discrete sets with minimal or predefined perimeter
- Uniqueness is not guaranteed



2. Estimation of perimeter

- Generated datasets:
  - *h*-convex discrete sets
  - “*random*” discrete sets created by merging *h*-convex és *v*-convex sets



2. Estimation of perimeter

- Goal: to determine the perimeter of the discrete set in case certain degree of uncertainty is allowed
- perimeter of *h*-convex sets
  - 1500-300 samples (no validation set)
  - $\alpha = 0.001$
  - $\beta = 0.3$
  - number of hidden neurons grows with the size, from 20 (10×10) up to 80 (100×100)

2. Estimation of perimeter

- perimeter of *h*-convex sets – error rates

Uncertainty	20×20	40×40	60×60	80×80	100×100
1%	89.67	87.07	84.40	83.93	82.87
2%	78.40	75.33	68.60	67.27	64.87
3%	67.27	62.00	55.33	52.20	50.00
4%	58.27	50.60	43.53	39.07	37.13
5%	48.33	40.53	32.93	29.80	26.47
6%	39.47	33.20	25.20	20.33	18.93
7%	32.13	26.27	18.00	13.60	10.60
8%	25.27	19.87	12.73	9.73	7.07
9%	19.47	15.00	8.60	6.27	4.13
10%	15.20	10.80	5.53	4.07	3.00
20%	0.40	0.47	0.00	0.00	0.00

2. Estimation of perimeter

- perimeter of "*random*" sets
  - 1500-300 samples (no validation set)
  - $\alpha = 0.001$
  - $\beta = 0.3$
  - number of hidden neurons grows with the size, from 10 (10×10) up to 60 (100×100)

2. Estimation of perimeter

- perimeter of "*random*" sets – error rates

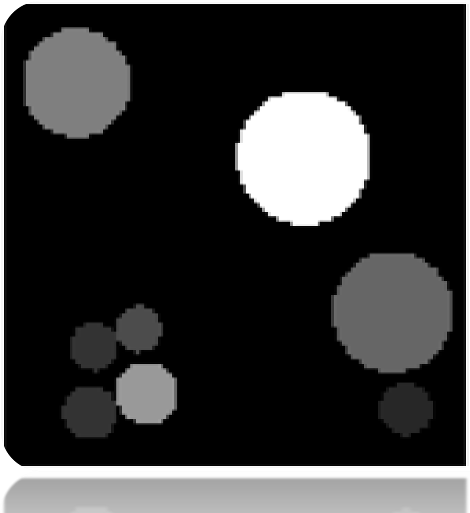
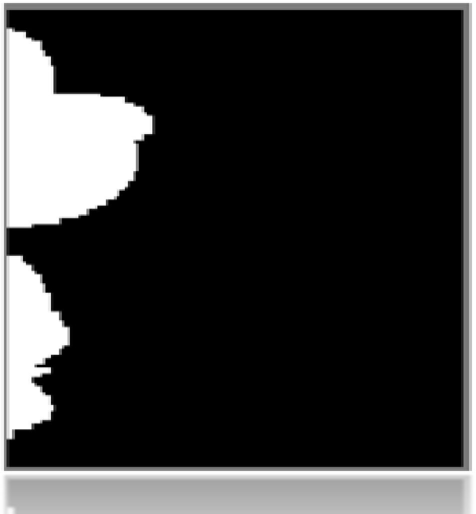
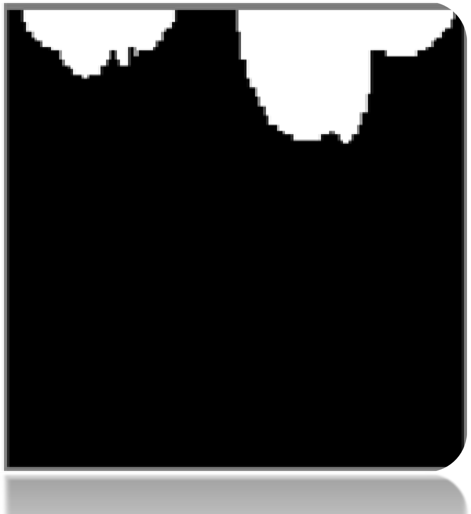
Uncertainty	20×20	40×40	60×60	80×80	100×100
1%	86.33	85.07	81.80	81.20	76.20
2%	73.40	68.87	62.13	61.47	56.13
3%	62.40	53.73	47.40	42.80	39.53
4%	51.07	41.93	33.80	30.80	24.27
5%	41.60	30.87	23.73	19.67	14.60
6%	31.33	21.67	14.13	11.33	7.33
7%	24.00	14.93	9.33	6.20	3.87
8%	19.13	10.33	5.33	3.53	1.73
9%	13.73	7.07	3.87	2.27	0.60
10%	10.93	5.13	2.07	1.13	0.20
20%	0.47	0.07	0.00	0.00	0.00

3. Estimation of the number of intensities

- Task: determining the number of different *intensities* present in the discrete image
- Solutions:
  - histogram based techniques based on continuous reconstruction
  - semi-automatic methods
  - …
- Proposal: apply neural networks, let the input be the **projections** themselves
- Initially let us investigate images with certain a “configuration”

3. Estimation of the number of intensities

- *configuration*: discrete images containing *n* circles that possess
  - fix position and
  - fix size
- circles differ only in their intensity
- each circle is a homogeneous object

3. Estimation of the number of intensities

- every image contain *8 circles*
- *10* diff. configurations were created
- for each configuration 3600-1200 training-, and test images were generated ⇨ obtain 2 projections
- images corresponding to a certain configuration *differ* in their circles' **intensities** only
- background intensity: *0.0*

	equidistant							
3:	0.1	0.2	0.3					
4:	0.1	0.2	0.3	0.4				
5:	0.1	0.2	0.3	0.4	0.5			
6:	0.1	0.2	0.3	0.4	0.5	0.6		
7:	0.1	0.2	0.3	0.4	0.5	0.6	0.7	
8:	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8

3. Estimation of the number of intensities

- Average parameters of the neural networks used:

Noiseless

#intensities	Learning rate	Momentum	Training time	Hidden neurons
3	0.2	0.8	100	10.5
4	0.24	0.78	190	16
5	0.27	0.75	370	41
6	0.238	0.8275	530	55.5

5% Noise

#intensities	Learning rate	Momentum	Training time	Hidden neurons
3	0.2	0.8	100	10
4	0.3	0.8	200	20
5	0.27	0.75	740	41
6	0.2218	0.8275	133	54

3. Estimation of the number of intensities

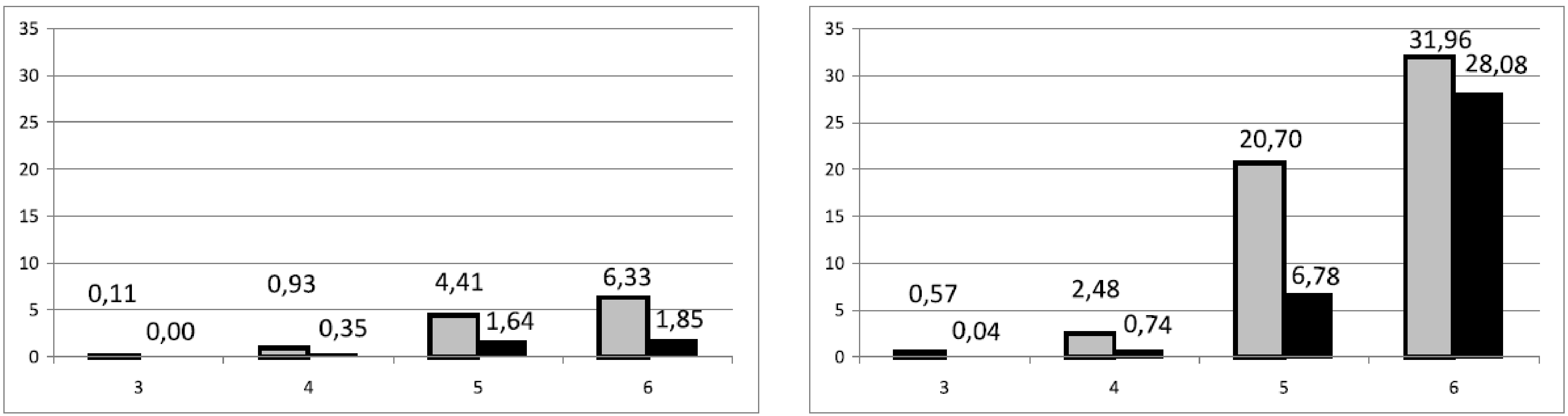
- Average confusion matrix
  - 10 different configurations and
  - 6 different intensity levels have been investigated

(b) Neural network

(1)	(2)	(3)	(4)	(5)	(6)	← classified as
100.00	0.00	0.00	0.00	0.00	0.00	(1)
0.00	98.90	1.00	0.10	0.00	0.00	(2)
0.20	2.30	91.85	4.20	0.85	0.60	(3)
0.00	0.75	3.10	75.90	12.35	7.90	(4)
0.00	0.00	0.70	3.90	95.40	0.00	(5)
0.00	0.00	0.00	0.00	0.00	100.00	(6)

3. Estimation of the number of intensities

- 3–6 different intensity levels
- added uniform noise

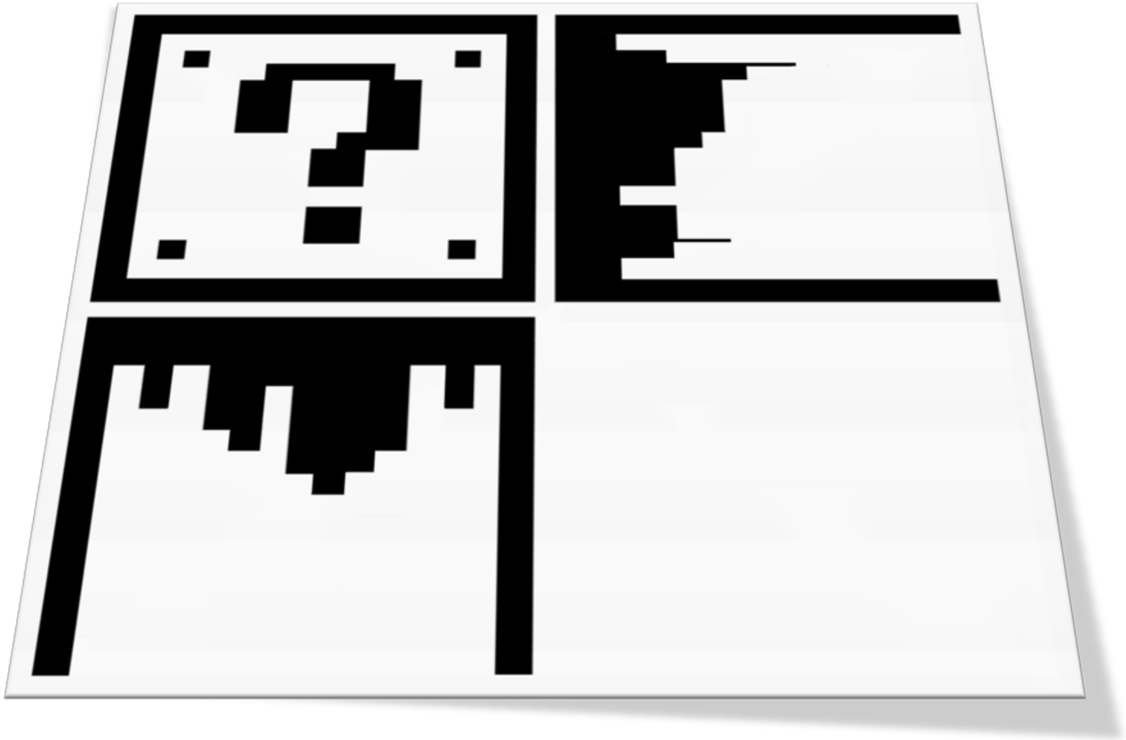


(a) Noiseless (b) With 5% noise

Remarks about neural networks

- Implementation should preferably contain:
  - momentum technique
  - advanced data partitioning methods
    (pl. "windowing", growing subset, random shuffle)
  - **automated decreasing** of learning rate (WEKA)
- The momentum is not always optimal at ~0.9
- Longer learning time is not always better!
  (e.g. the case of noisy projections)

Questions?



Acknowledgement

NEW SZÉCHENYI PLAN

HUNGARY'S RENEWAL

The project is supported by
the European Union and co-financed
by the European Social Fund.

The presentation is supported by the European Union and co-funded by the European Social Fund.
Project title: "Broadening the knowledge base and supporting the long term professional sustainability of the Research University Centre of Excellence at the University of Szeged by ensuring the rising generation of excellent scientists".

Project number: TÁMOP-4.2.2/B-10/1-2010-2012