EMOTICONS RECOGNITION USING A NEURAL NETWORK

Rafał Suszka
Informatyka, sem V
rok akademicki 2017/2018

PRESENTATION PLAN

1. Description of the problem.
2. Description of the data.
3. Implementation.
4. Results.

DESCRIPTION OF THE PROBLEM

DESCRIPTION OF THE PROBLEM

- Write a program that recognizes emoticons.
- Assumptions implemented in the program:

  - windows application written in C#,

  - implementing own neural network,

  - recognition of 5 different emoticons,

  - input data: bitmaps any sizes,

  - output data: information about the recognized emoticon.

DESCRIPTION OF THE DATA

DESCRIPTION OF THE DATA (1)

- Input data:

  - bitmap with emoticon.

- Output data:

  - information about the object's class.

DESCRIPTION OF THE DATA (2)

- Input learning data:

  - a set of images in the bmp format,

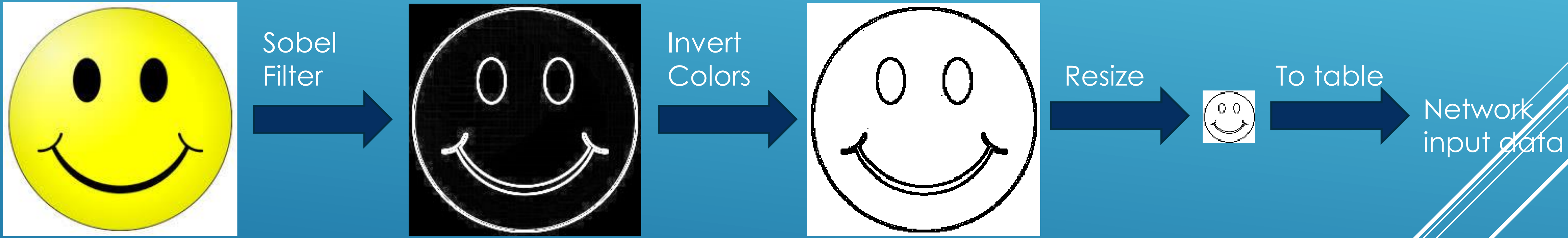
  - a text file with the correct state of the network outputs for each image.

- Output learning data:

  - information about the success of learning.

DESCRIPTION OF THE DATA (3)

- Image processing:



Sobel Filter

Invert Colors

Resize

To table

Network input data

DESCRIPTION OF THE DATA (4)

- Exaple imput data:

```
00000000000000000000000000000000000000000000000000
00000000000000000000011111111100000000000000000000
00000000000000000111110000000111110000000000000000
00000000000000011100000000000000011100000000000000
00000000000001100000000000000000000011000000000000
00000000000110000000000000000000000000110000000000
00000000001110000000000000000000000000011000000000
00000000011000000000000000000000000000001100000000
00000000110000000000000000000000000000000110000000
00000001100000000000000000000000000000000011000000
00000011000000000000000000000000000000000001100000
00000110000000000000000000000000000000000000110000
00000100000000000111000000000001110000000000010000
```

IMPLEMENTATION

IMPLEMENTATION (1)

- Network structure:

  - input layer: 2500 neurons

  - first hidden layer: 100 neurons

  - second hidden layer: 25 neurons

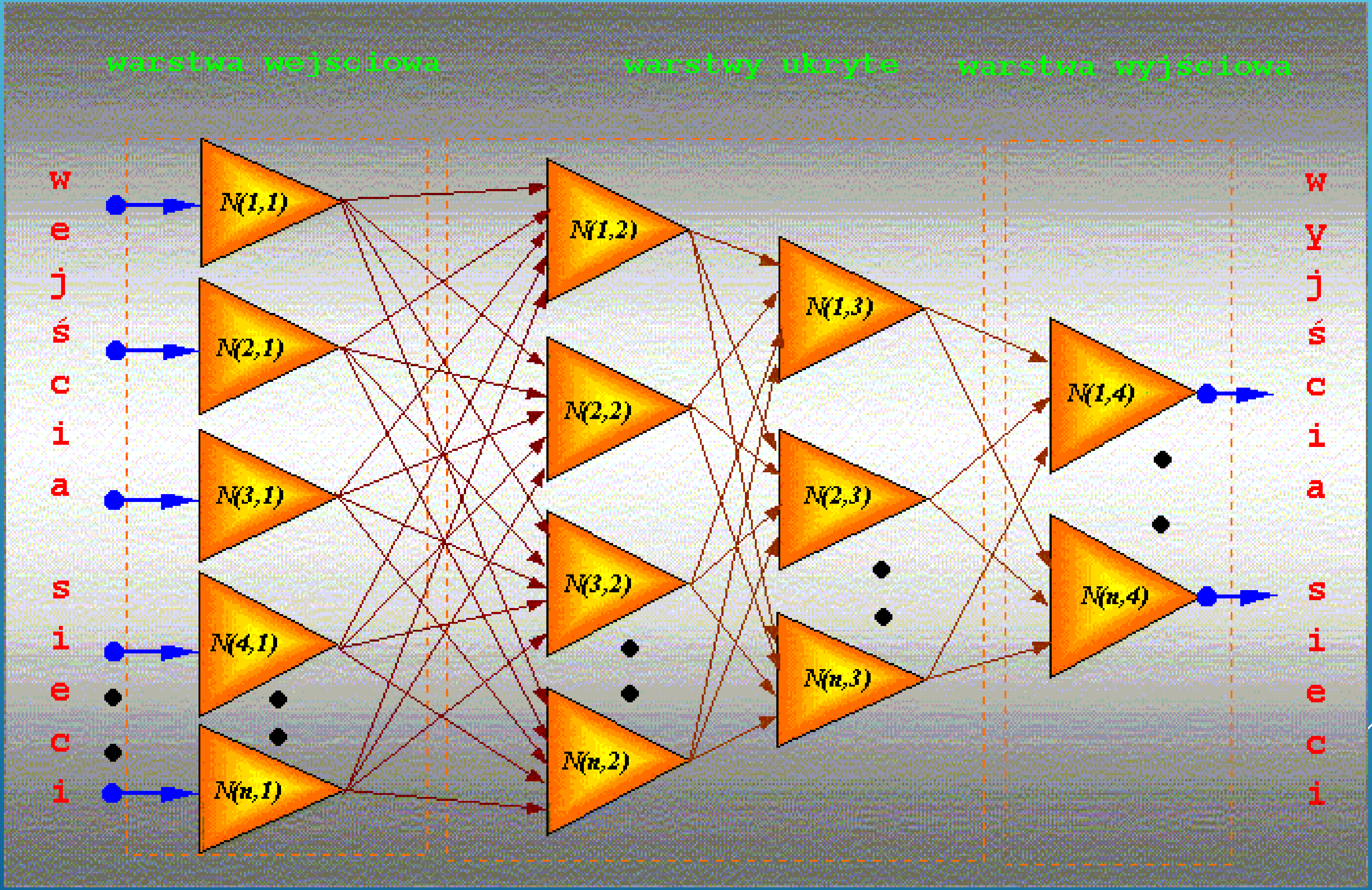
  - output layer: 5 neurons

- Activation function – sigmoid:

$$f(x) = \frac{1}{1 + e^{-(beta*x)}}$$

- Learning - reverse propagation algorithm

IMPLEMENTATION (2)



IMPLEMENTATION (3)

- Neuron class fields:

```
private double beta;
private List<double> inputsWeights;
private double value;
private double delta;
private List<double> oldInputsWeights;
```

IMPLEMENTATION (4)

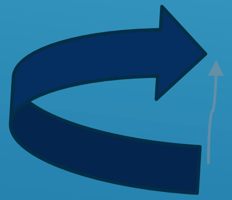
- NeuronNeutwork class fields:

```
private List<int> inputLayer;
private List<Neuron> hiddenLayerFirst;
private List<Neuron> hiddenLayerSecond;
private List<Neuron> outputLayer;
private int numberOfNeuronsInInputLayer = 0;
private int numberOfNeuronsInHiddenLayerFirst = 100;
private int numberOfNeuronsInHiddenLayerSecond = 25;
private int numberOfNeuronsInOutputLayer = 5;
private double eta = 0.3;
private double alfa = 0.9;
```

IMPLEMENTATION (5)

- Learning peogram structure:

1. Reading the directory with bitmaps.
2. Reading correct outputs file.
3. Mixing data.
4. Convert bitmap to table.
5. Learn network.



IMPLEMENTATION (6)

- Program structure:

  1. Reading bitmap.
  2. Convert bitmap to table.
  3. Run network with data.
  4. Show results.

RESULTS

RESULTS

- The program is not yet fully ready

QUESTIONS?

THANK YOU FOR YOUR ATTENTION