COMPUTATIONAL INTELLIGENCE

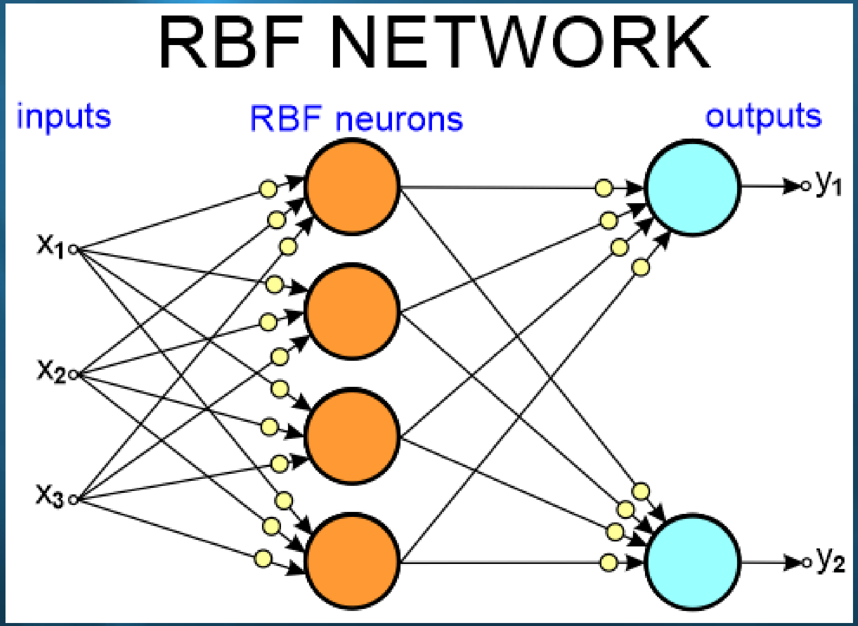
Radial Basis Function Networks

AGH

Adrian Horzyk

Preface

Radial Basis Function Networks (RBFN) are a kind of artificial neural networks that use radial basis functions (RBF) as activation functions.



Typical RBF networks have three layers:

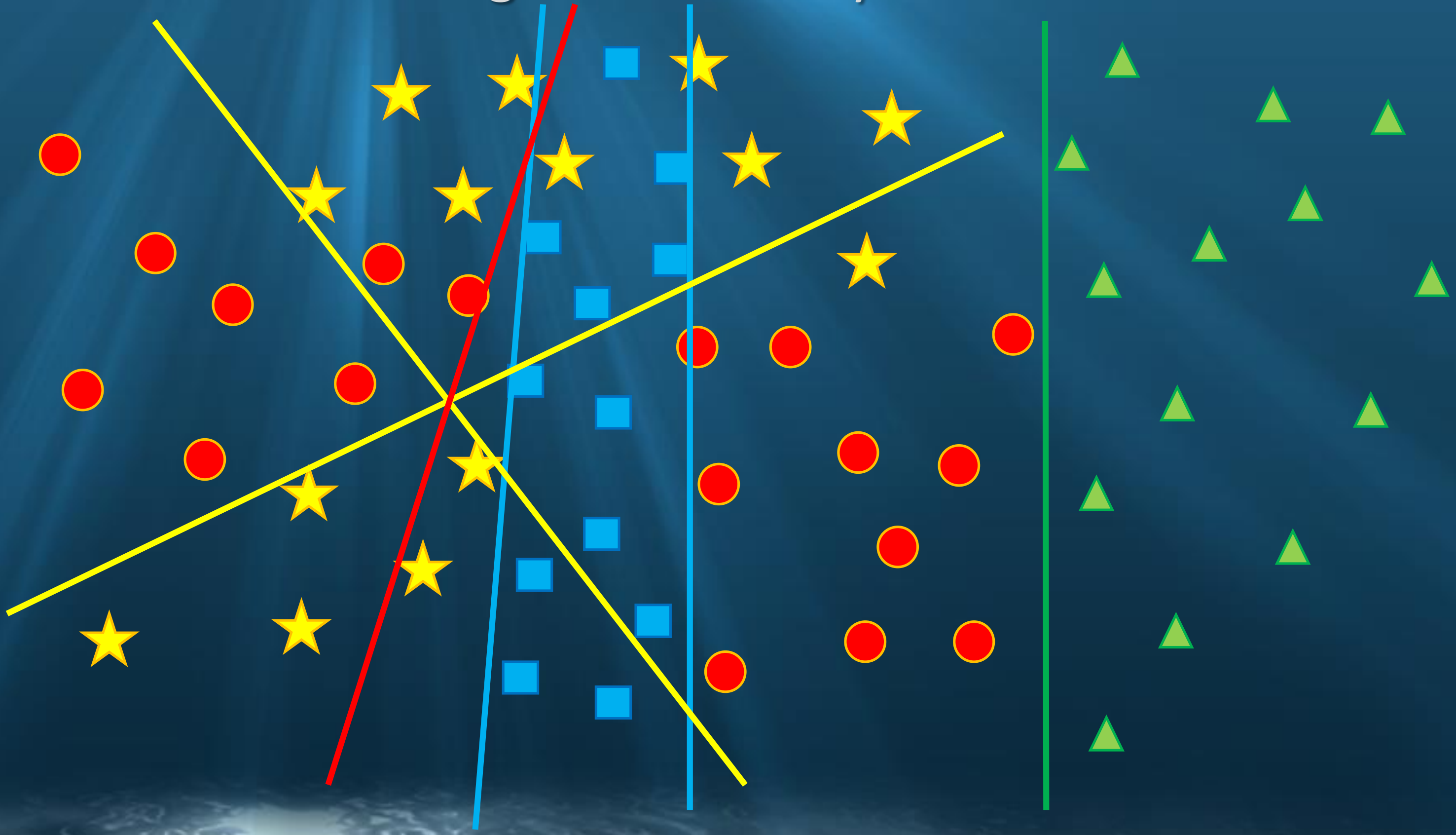
1. An input layer that forwards input signals/stimuli/data,
2. A hidden layer with a selected non-linear RBF activation function,
3. An output layer with linear activation function,

however, some modifications of this architecture are also possible, e.g. the output layer can include neurons with non-linear (e.g.) sigmoidal activation functions or there can be used a few other non-linear layers (in this case we achieve a deep ANN architecture).

Separation of Training Samples

In case of classification or clusterization (grouping), training samples have to be separated into groups which define different classes. We can distinguish a few ways how we can do it:

- Linear separation
- Non-linear separation
- Radial overlapping
- Rectangular overlapping
- Irregular overlapping



Separation of Training Samples

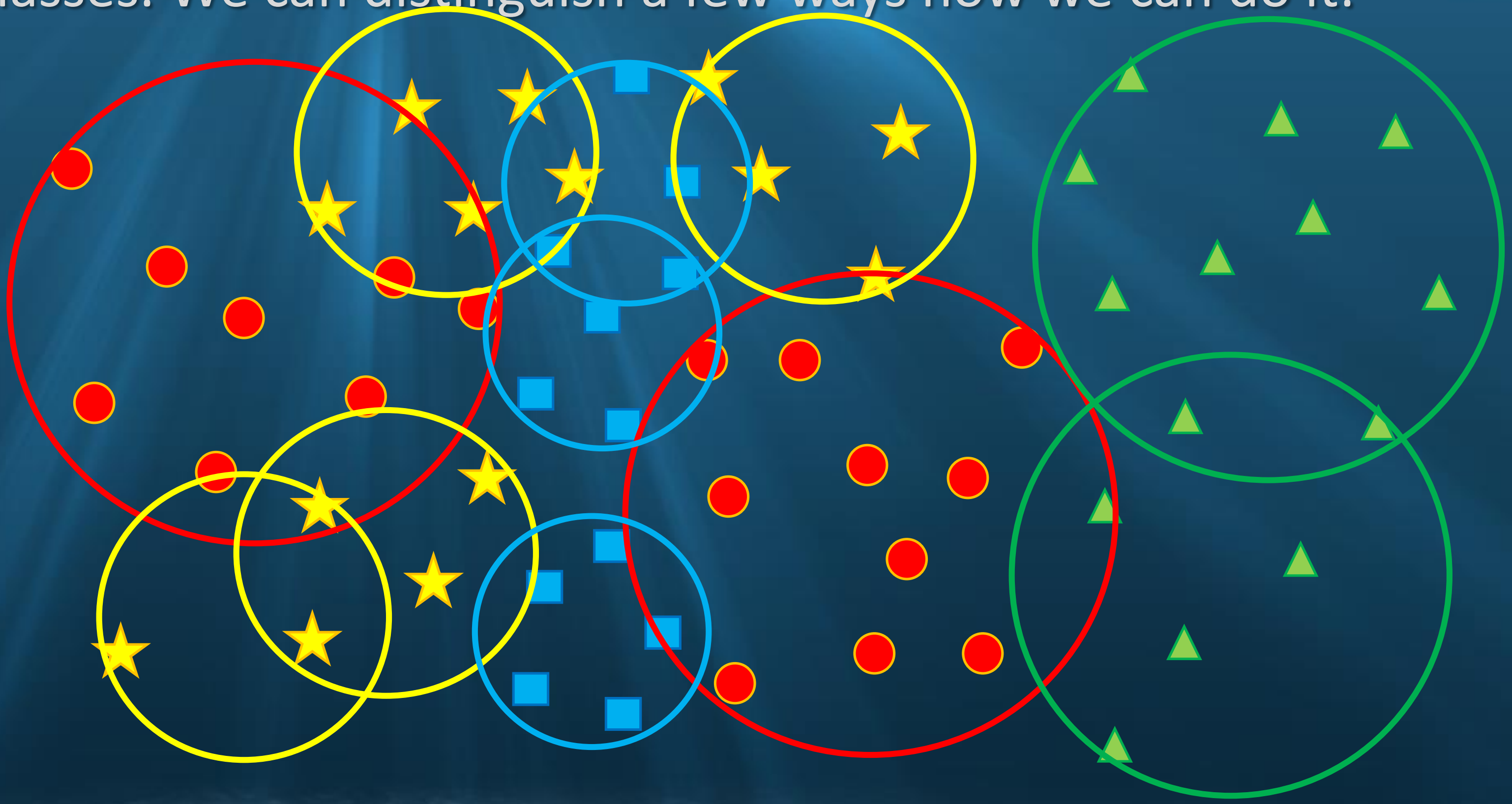
In case of classification or clusterization (grouping), training samples have to be separated into groups which define different classes. We can distinguish a few ways how we can do it:

- Linear separation
- Non-linear separation
- Radial overlapping
- Rectangular overlapping
- Irregular overlapping

Separation of Training Samples

In case of classification or clusterization (grouping), training samples have to be separated into groups which define different classes. We can distinguish a few ways how we can do it:

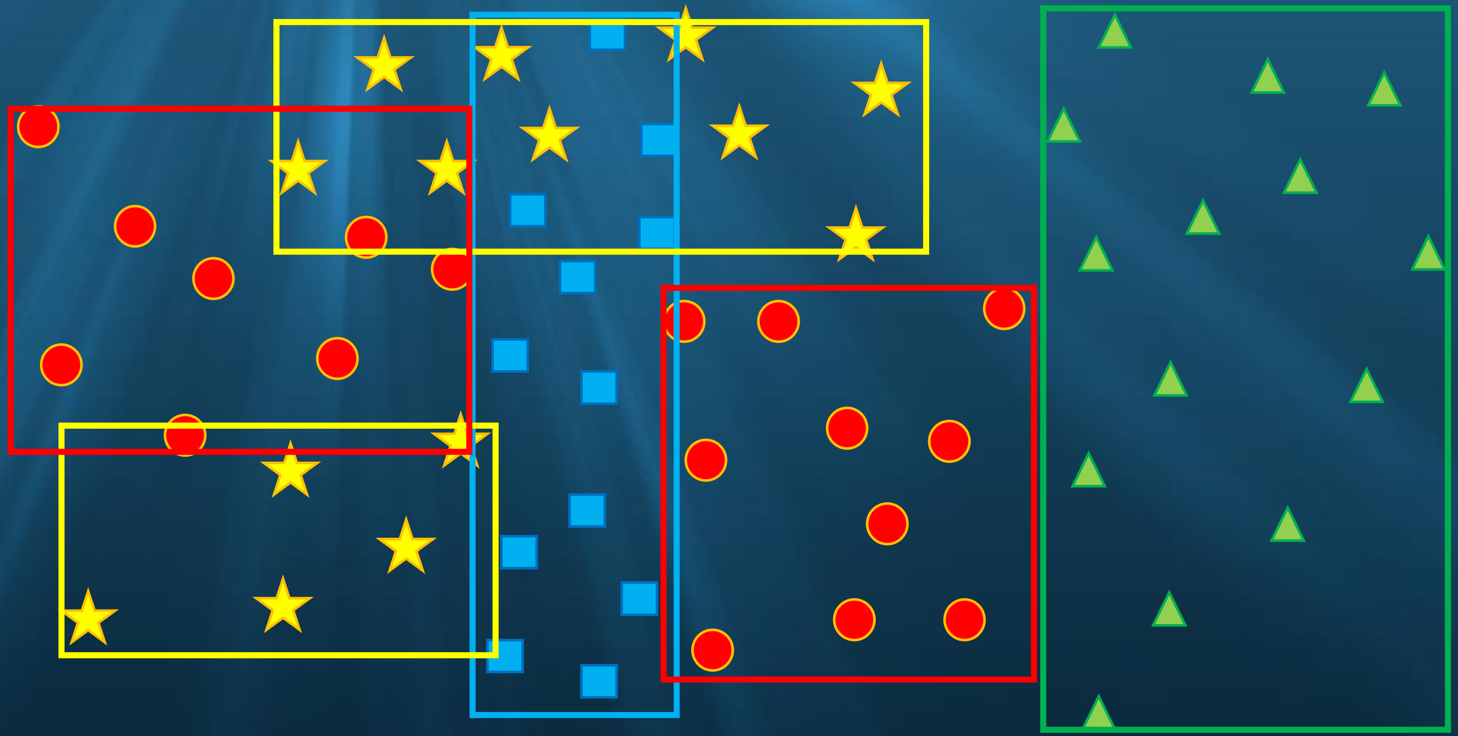
- Linear separation
- Non-linear separation
- Radial overlapping
- Rectangular overlapping
- Irregular overlapping



Separation of Training Samples

In case of classification or clusterization (grouping), training samples have to be separated into groups which define different classes. We can distinguish a few ways how we can do it:

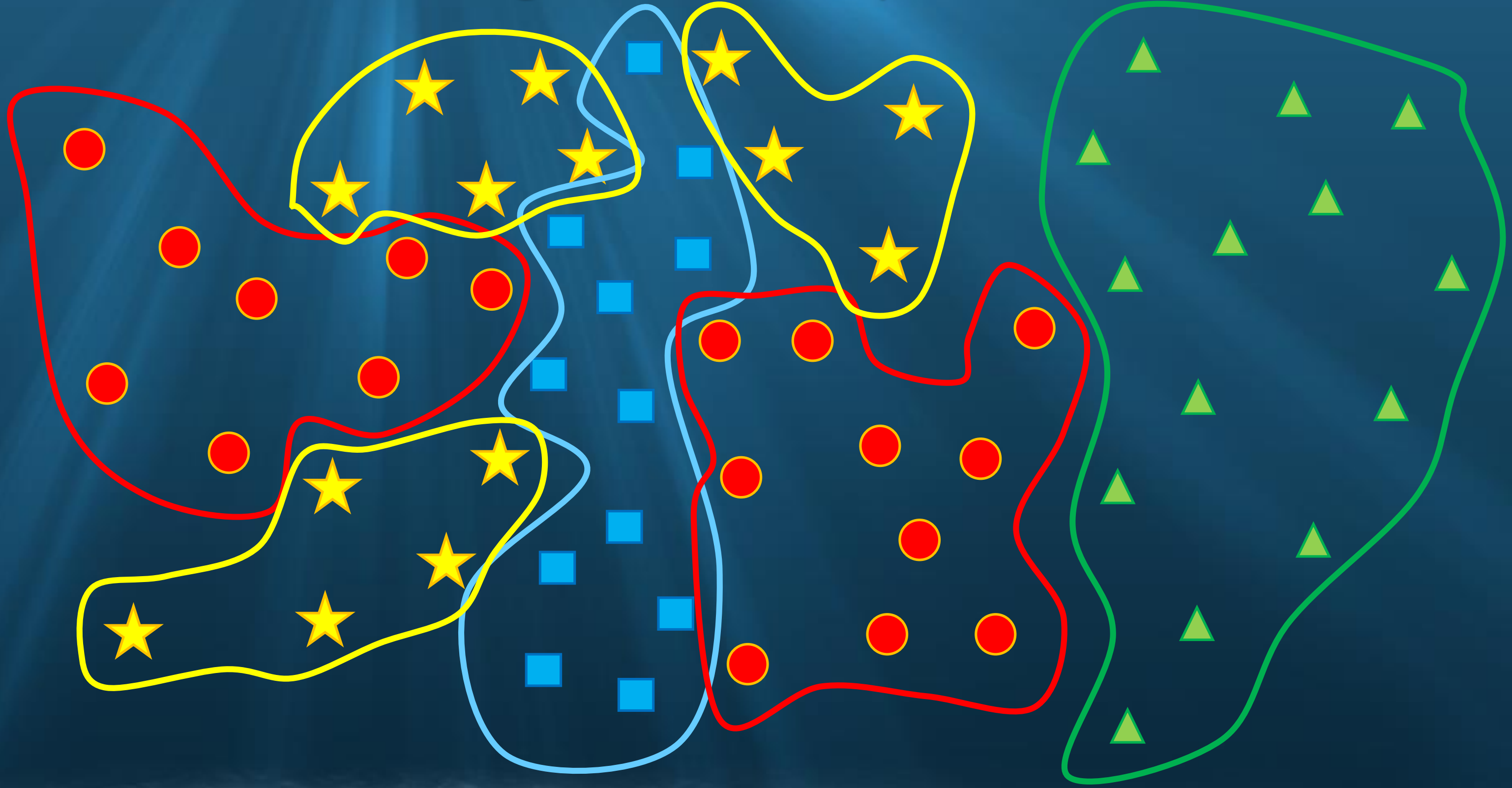
- Linear separation
- Non-linear separation
- Radial overlapping
- Rectangular overlapping
- Irregular overlapping



Separation of Training Samples

In case of classification or clusterization (grouping), training samples have to be separated into groups which define different classes. We can distinguish a few ways how we can do it:

- Linear separation
- Non-linear separation
- Radial overlapping
- Rectangular overlapping
- Irregular overlapping

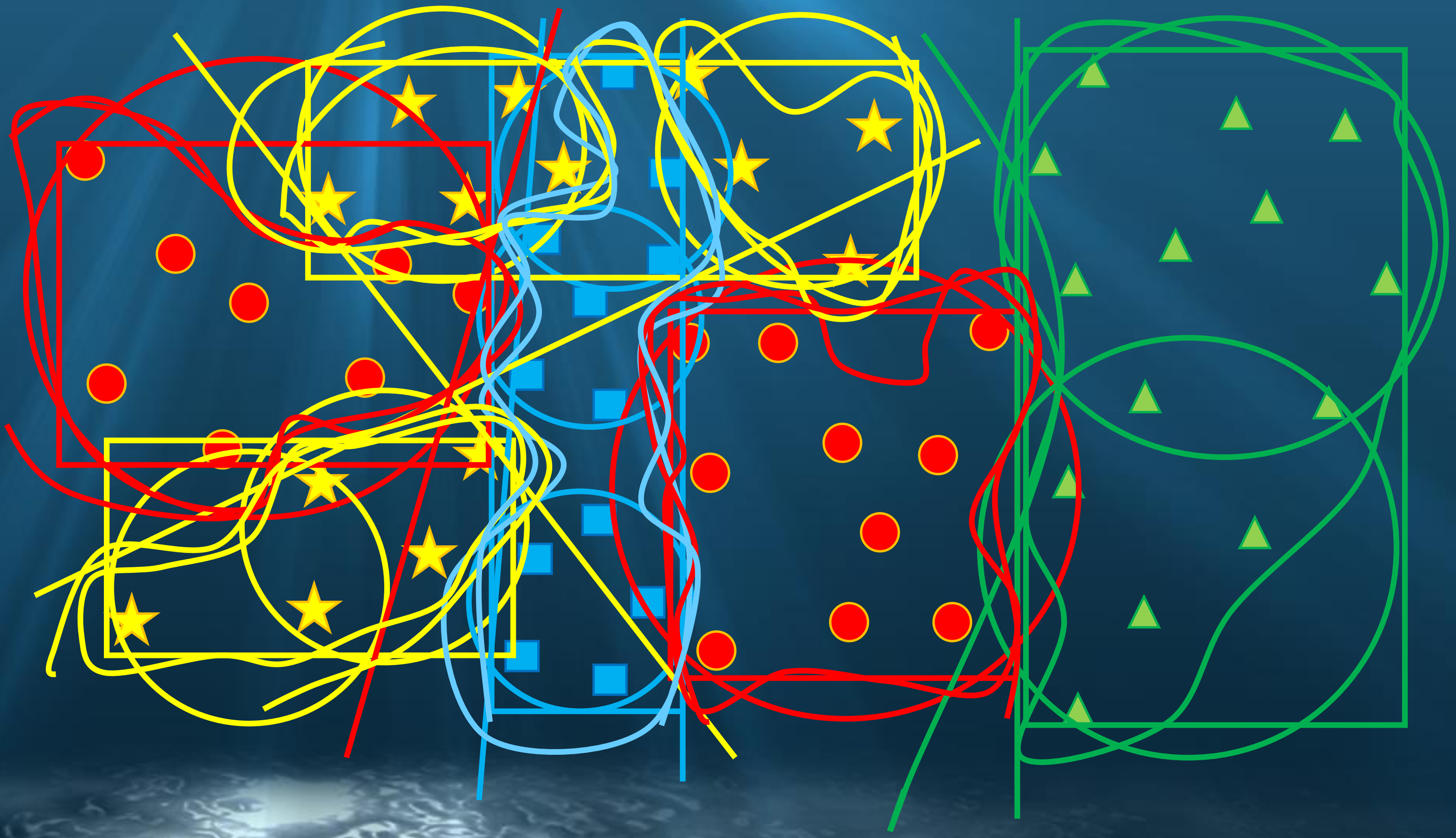


Separation of Training Samples

In case of classification or clusterization (grouping), training samples have to be separated into groups which define different classes. We can distinguish a few ways how we can do it:

- Linear separation
- Non-linear separation
- Radial overlapping
- Rectangular overlapping
- Irregular overlapping

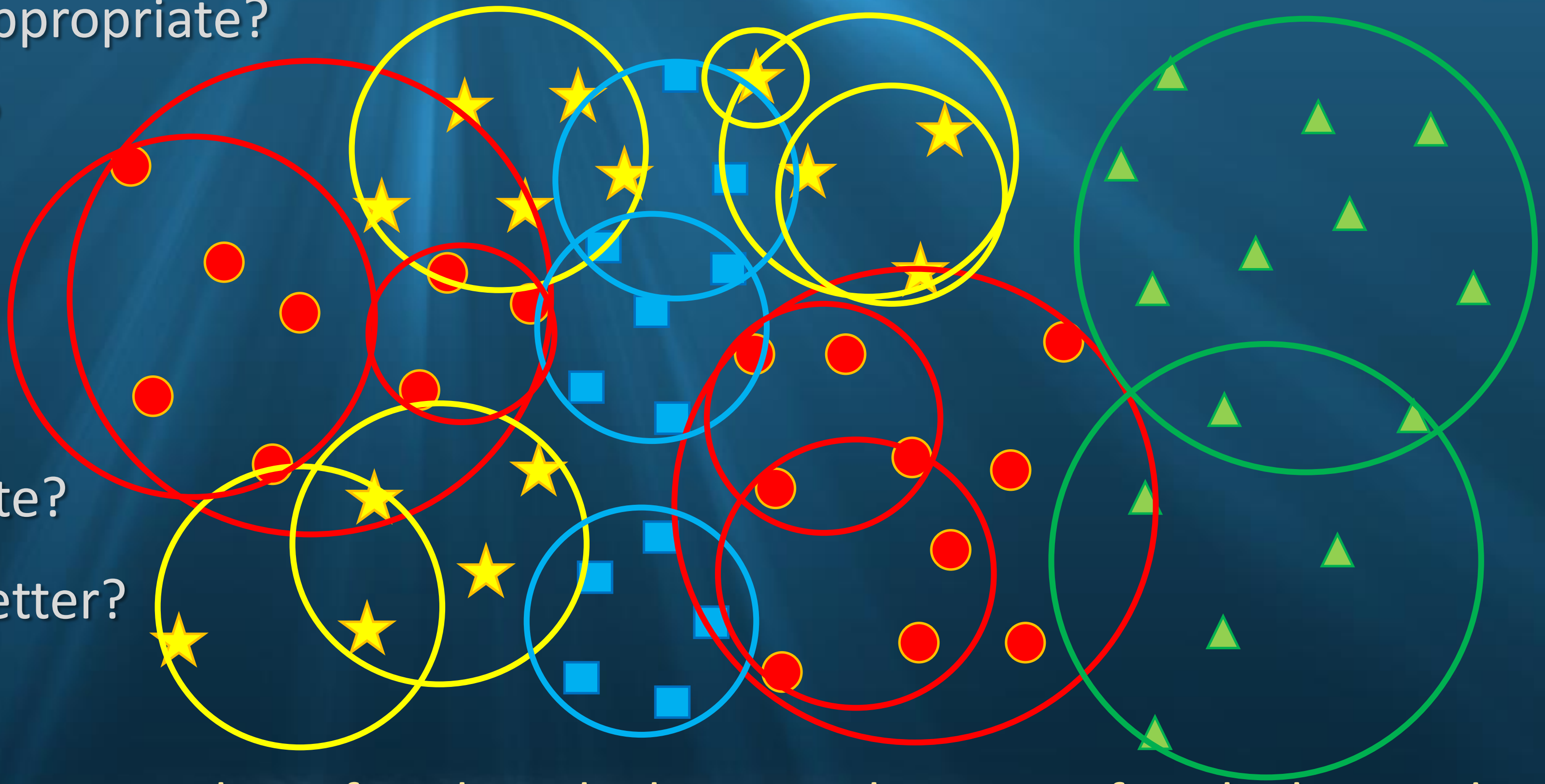
Which method is the best?



Radial Separation of RBF Networks

Using RBF Networks we need to answer a few question:

- How many circles will be appropriate?
- Which circles are the best?
- How to compute centers?
- How to achieve the best generalization property?
- Is the set of circles adequate?
- Can we determine them better?
- Should circle overlap?



We will seek to identify the least number of circles which cover elements of each class and simultaneously well generalize them.

Basic RBF Network

The input can be modeled as a vector (or a matrix) of real numbers $\boldsymbol{x} \in \mathbb{R}^n$.

The outputs of the network is then a function of this input vector $\varphi: \mathbb{R}^n \to \mathbb{R}^m$:

$$\boldsymbol{y} = \varphi(\boldsymbol{x}) = \sum_{i=1}^{M} w_i \cdot \rho(\|\boldsymbol{x} - \boldsymbol{c_i}\|)$$

where

M is the number of neurons in the hidden layer that should be much less than the number of training samples N.

$\boldsymbol{c}_i$ is the center vector for neuron i

$\boldsymbol{w}_i$ is the weight of neuron i in the linear output neuron

$\boldsymbol{y}$ is the output vector of computed results (approximation)

$\rho: \mathbb{R} \to \mathbb{R}$ is an RBF function that depends on the distance of input vector from a center vector represented by the given neuron, usually taken to be Gaussian: $\rho(\|\boldsymbol{x} - \boldsymbol{c_i}\|) = e^{-\beta_i \cdot \|\boldsymbol{x} - \boldsymbol{c}_i\|^2}$

Basic RBF Network

In the basic form of this network, all inputs are connected to each hidden neuron.

The norm ||…|| is typically taken to be:

- Euclidean distance: $\|\boldsymbol{x} - \boldsymbol{c_i}\|_2 = \sqrt{\sum_{n=1}^{N}(x_n - c_{in})^2}$
- Mahalanobis distance: $\|\boldsymbol{x} - \boldsymbol{c_i}\|_2 = \sqrt{(\boldsymbol{x} - \boldsymbol{c_i})^T S^{-1}(\boldsymbol{x} - \boldsymbol{c_i})}$
- Manhattan distance: $\|\boldsymbol{x} - \boldsymbol{c_i}\|_1 = \sum_{n=1}^{N}|x_n - c_{in}|$

where

S is the covariance matrix that consists of elements,
which are defined as covariance between the i-th and the j-th elements of a random vector.

The parameters w_i, $\boldsymbol{c_i}$, and β_i are determined during the training process in a manner that optimizes the fit between φ and the data.

Radial Basis Functions

The most often used radial basis functions:

Gaussian: $\rho(\|\boldsymbol{x}-\boldsymbol{c_i}\|) = e^{-\beta_i \cdot \|\boldsymbol{x}-\boldsymbol{c_i}\|^2}$

Multi-Quadric: $\rho(\|\boldsymbol{x}-\boldsymbol{c_i}\|) = \sqrt{1+\beta_i \cdot \|\boldsymbol{x}-\boldsymbol{c_i}\|^2}$

Inverse Quadratic: $\rho(\|\boldsymbol{x}-\boldsymbol{c_i}\|) = \frac{1}{1+\beta_i \cdot \|x-c_i\|^2}$

Inverse Multi-Quadric : $\rho(\|\boldsymbol{x}-\boldsymbol{c_i}\|) = \frac{\mathbf{1}}{\sqrt{1+\beta_i \cdot \|\boldsymbol{x}-\boldsymbol{c_i}\|^2}}$

Polyharmonic Spline: $\rho(\|\boldsymbol{x}-\boldsymbol{c_i}\|) = \begin{cases} \|\boldsymbol{x}-\boldsymbol{c_i}\|^{\boldsymbol{k}} & \text{for } k = 1, 3, 5, 7\ldots \\ ln(\|\boldsymbol{x}-\boldsymbol{c_i}\|) \cdot \|\boldsymbol{x}-\boldsymbol{c_i}\|^{\boldsymbol{k}} & \text{for } k = 2, 4, 6, 8\ldots \end{cases}$

Normalized RBN Network

We often use a normalized architecture that usually achieves better results.

In this case, the outputs is computed using the following function:

$$\boldsymbol{y} = \varphi(\boldsymbol{x}) = \frac{\sum_{i=1}^{I} w_i \cdot \rho(\|\boldsymbol{x} - \boldsymbol{c_i}\|)}{\sum_{i=1}^{I} \rho(\|\boldsymbol{x} - \boldsymbol{c_i}\|)} = \sum_{i=1}^{I} w_i \cdot \omega(\|\boldsymbol{x} - \boldsymbol{c_i}\|)$$

Where

$$\omega(\|\boldsymbol{x} - \boldsymbol{c_i}\|) = \frac{\rho(\|\boldsymbol{x} - \boldsymbol{c_i}\|)}{\sum_{i=1}^{I} \rho(\|\boldsymbol{x} - \boldsymbol{c_i}\|)}$$

is called a normalized radial basis function.

CONCLUSIONS:

RBF networks are universal approximators that can be used to various regression or classification tasks.
An RBF networks with enough hidden neurons can approximate any continuous function with arbitrary precision.

RBF Network Adaptation and Training

The adaptation and training process for RBF networks is usually divided into two steps:

1. Unsupervised step: The center vectors c_i of the RBF functions in the hidden layer are chosen (e.g. randomly sampled from the set of training samples, obtained by Orthogonal Least Square Learning, or determined using k-means clustering) and their width determined. The choice of the centers are not obvious. The width are usually fixed to the same value which is proportional to the maximum distance between chosen centers.
2. Supervised step: The backpropagation or gradient descent algorithm is used to fine-tune weights of the output layer neurons. Thus, the weights are adjusted at each time step by moving them in a direction opposite from the gradient of the function.

Bibliography and References

http://home.agh.edu.pl/~horzyk/lectures/ahdydci.php

ACADEMIC WEBSITE - ADRIAN HORZYK, PhD, Prof.

AGH University of Science and Technology in Cracow, Poland
Faculty of Electrical Engineering, Automatics, Computer Science and Biomedical Engineering
Department of Automatics and Biomedical Engineering, Field of Biocybernetics

LECTURES

COMPUTATIONAL INTELLIGENCE

is already under construction and will be available at the spring semester in 2016/2017.

This course will include 28 lectures, 14 laboratory classes and 14 project classes.

What is this course about?

This course is intended to give students a broad overview and deep knowledge about popular solutions and efficient neural network models as well as to learn how to construct and train intelligent learning systems in order to use them in everyday life and work. During the course we will deal with the popular and most efficient models and methods of neural networks, fuzzy systems and other learning systems that enable us to find specific highly generalizing models solving difficult tasks. We will also tackle with various CI and AI problems and work with various data and try to model their structures in such a way to optimize operations on them throughout making data available without necessity to search for them. This is a unique feature of associative structures and systems. These models and methods will allow us to form and represent knowledge in a modern and very efficient way which will enable us to mine it and automatically draw conclusions. You will be also able to understand solutions associated with various tasks of motivated learning and cognitive intelligence.