Chapter 9
Classification
Neural Networks

Statistical Methods and Learning in Computer Vision
WS 2012/13

Claudia Nieuwenhuis
Lehrstuhl für Computer Vision and Pattern Recognition
Fakultät für Informatik
Technische Universität München

Overview

Neural Networks

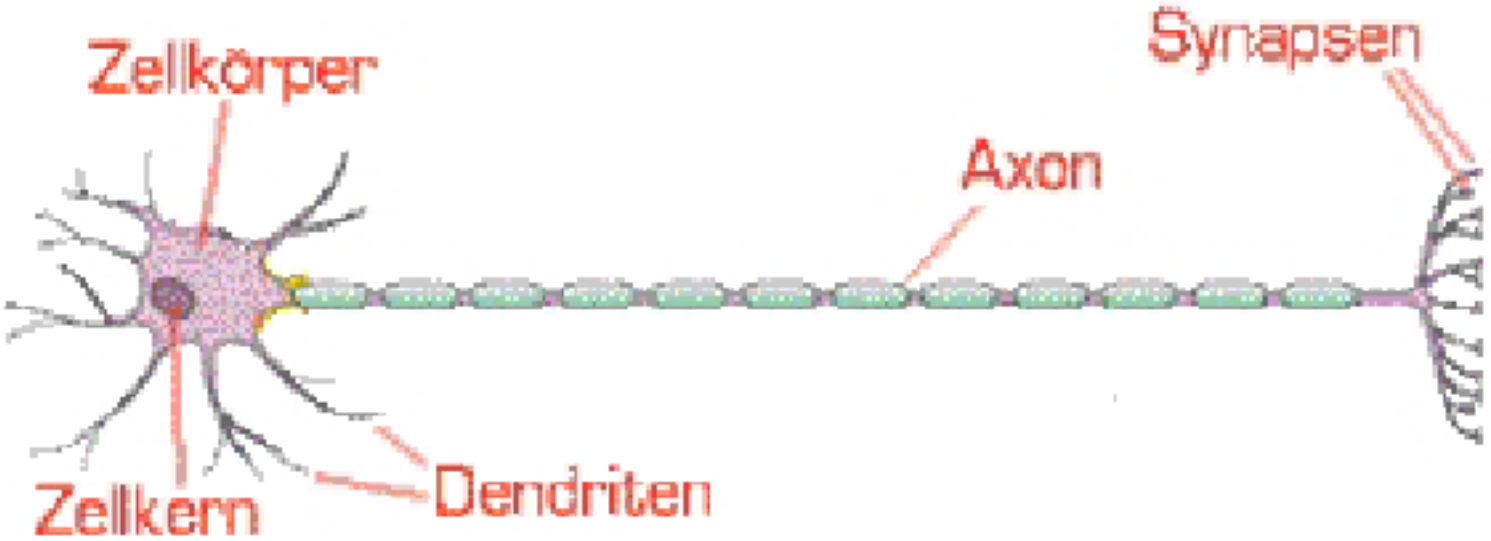
Neural Networks can be used for

- classification
- data quantization

They are typically used if the amount of input data is very large, but no formula or method is available how this input data is mapped to the correct outcome. If the complexity of the problem is overwhelming but it is easy to create a lot of sample datasets neural networks should be considered as a solution to the problem.

Neural Networks

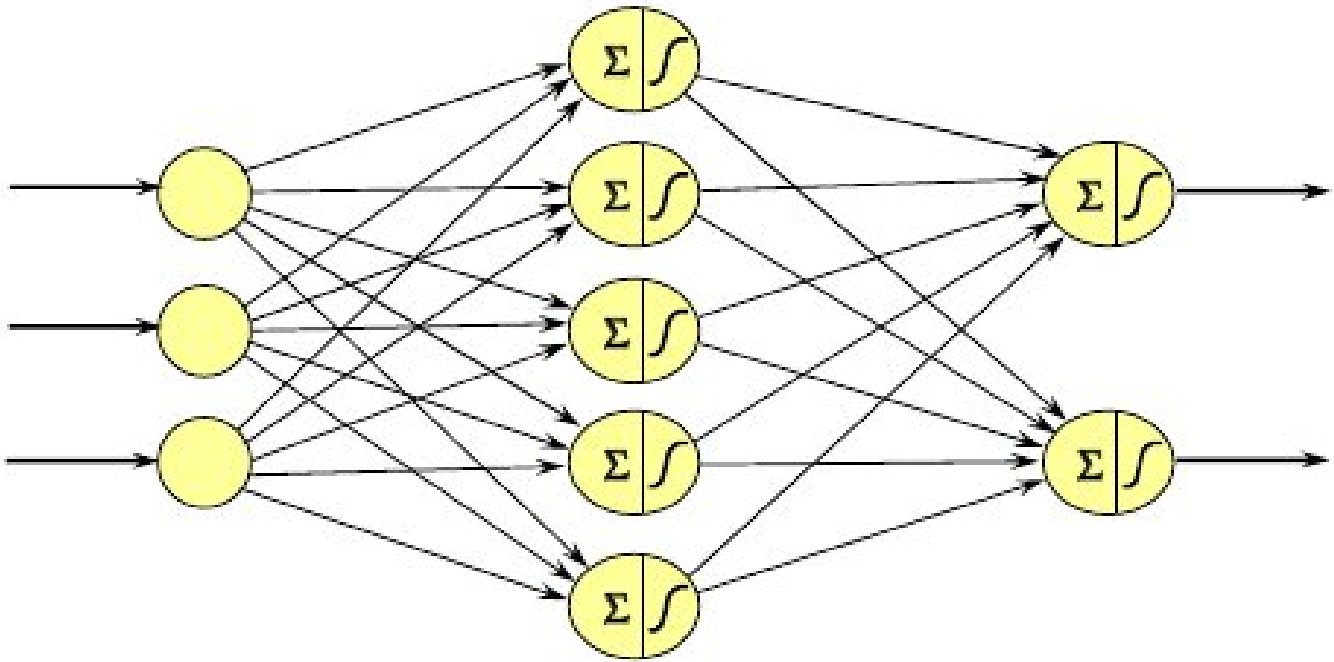
The idea of Neural Networks is based on the way information is processed in the human brain. The brain consists of 10 billions of neurons. Information is processed by transmitting electric impulses over the axons. The axon is split at the end forming several synapses. A neuron only transmits its activation to other neurons if the electric potential exceeds a certain threshold.



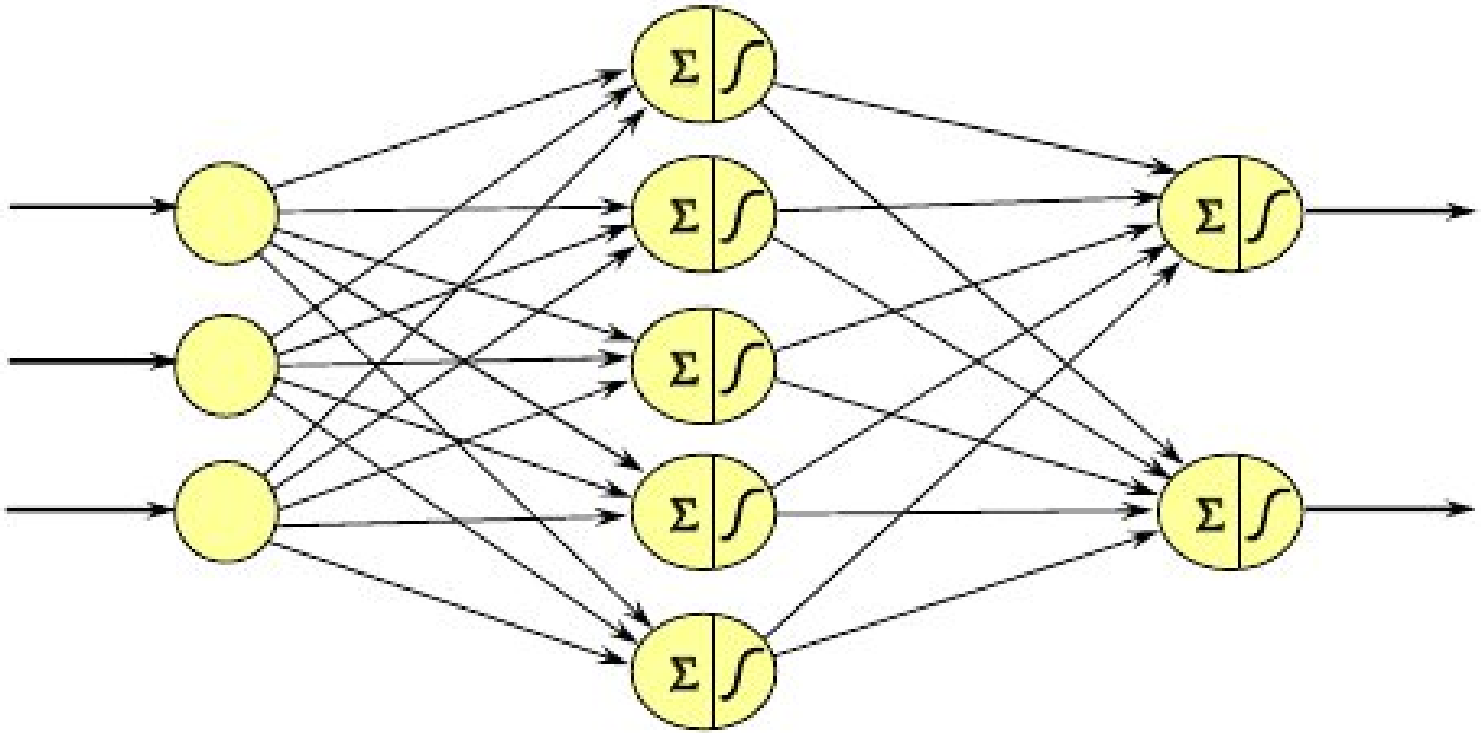
Neural Networks

Computers are not yet able to process information completely parallely as in the human brain. Therefore, neural networks are organized in layers.

- input layer: as many neurons as input variables to the network
- hidden layers: one or several layers with an arbitrary number of neurons
- output layer: returns the output of the network



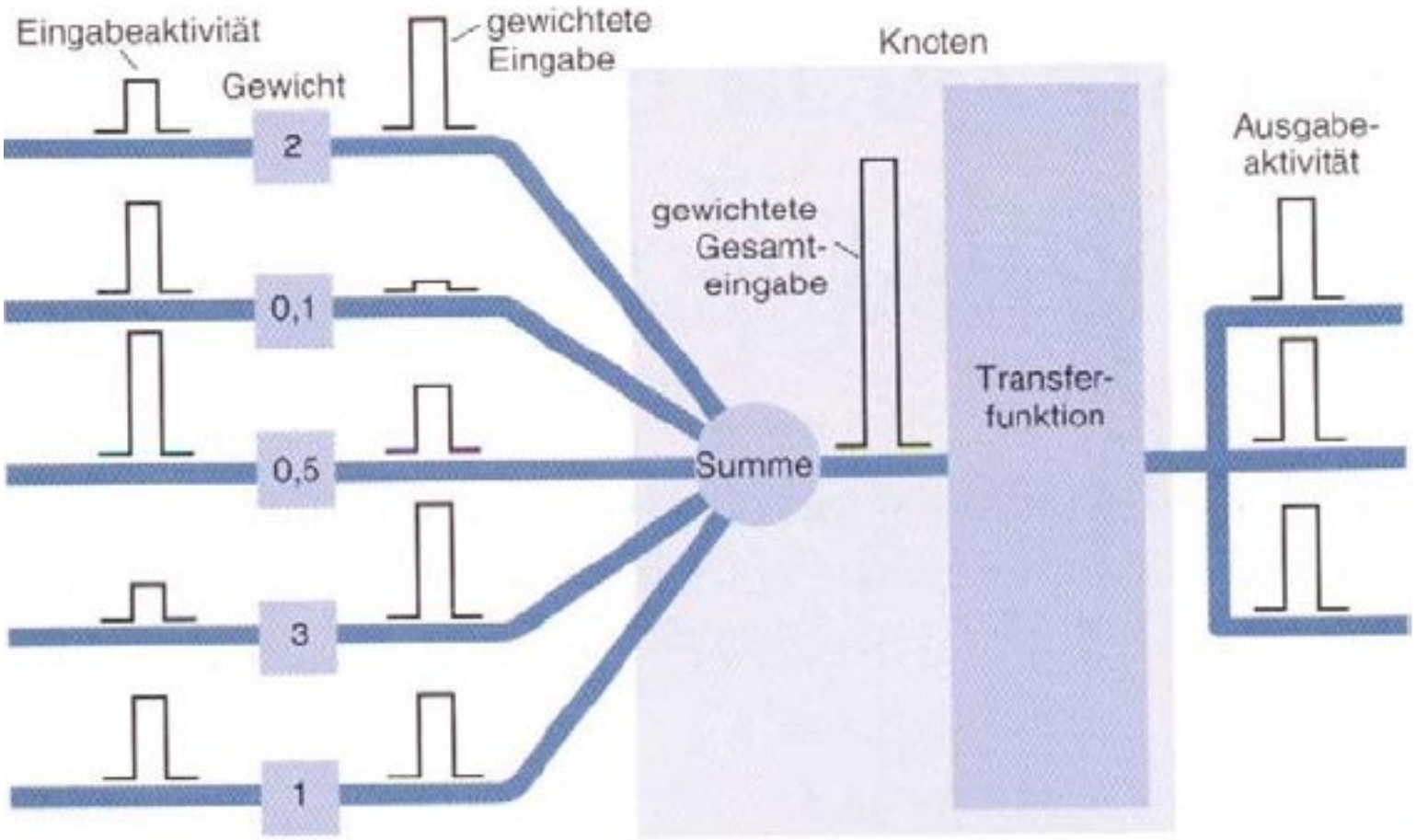
Neurons



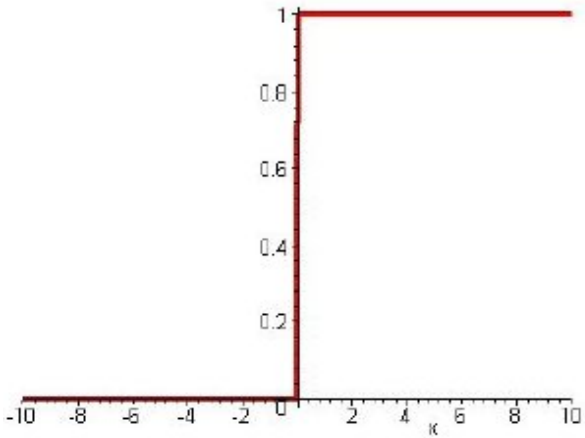
Neurons

A neuron can be thought of as an information processing unit. The activation of the neuron is computed by the **activation function**, usually the scalar product of the incoming signals and the weight vector of the neuron. The weights are important for the specialization of the neuron to a given task, since they assign more or less importance to an incoming impulse. Weights can also be negative.
The resulting activation of the neuron is passed to the **transfer function**, which determines the output value which is transmitted to the next layer or returned by the output layer. Common transfer functions are the Heaviside function or the sigmoid function.

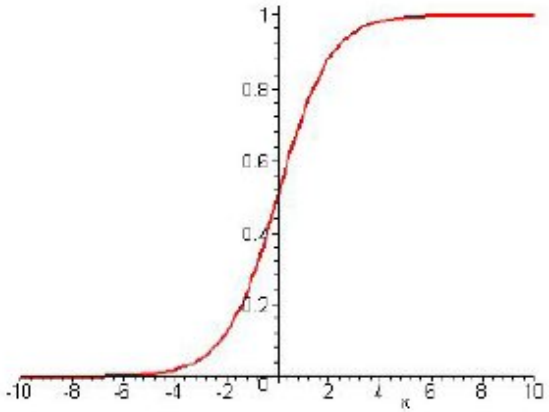
Activation Function



Transfer Functions



$$O(x) = \begin{cases} 1, & x \geq \theta \\ 0, & x < \theta \end{cases}$$



$$O(x) = \frac{1}{1+\exp^{-x}}$$

Learning

The objective of learning is to adapt the weights of the neurons to the data presented to the network according to some learning strategy. The weights are adapted so that the output corresponds to the training data. Neural Networks are then able to generalize the learned information to unknown data samples.

- Supervised Learning: the correct output values which the network is supposed to return are known and can be compared to the output values of the network, e.g. in hand sign recognition. Typical representatives are the multilayer perceptron.
- Unsupervised Learning: the network approximates the structure of the data by forming clusters, which mirror the density of the input data in any point of the feature space. Typical representatives are Kohonen maps and the Neural Gas.

Supervised Learning

It can be shown that every solvable classification problem can be learned by a multilayer perceptron. If the classes are

- linearly separable, no hidden layer is necessary.
- separable by convex regions, one hidden layer is necessary.
- arbitrary, two hidden layers are necessary.

Error Backpropagation Algorithm

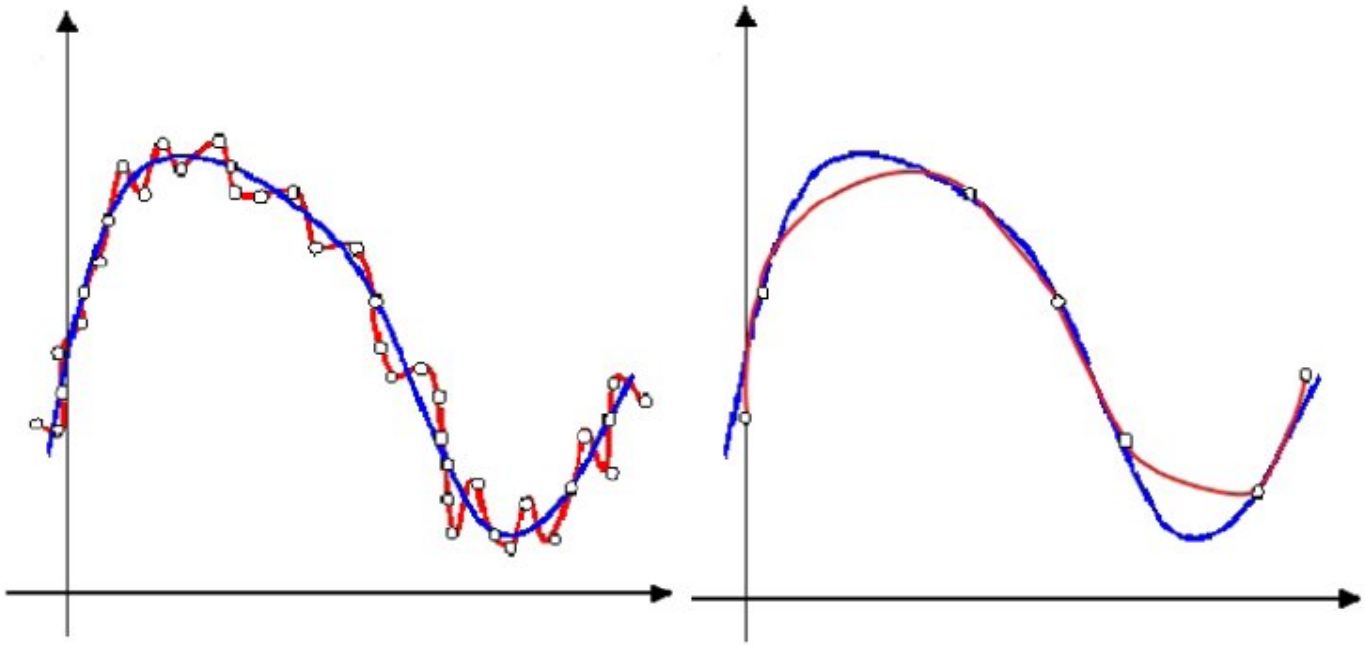
The error backpropagation algorithm minimizes the mean squared error at the n output neurons given the teaching vector t and the network output y^L based on L layers.

$$E(y,t) = \frac{1}{2}\sum_{i=1}^{n}(t_i - y_i^L)^2$$

This error is recursively propagated back to adapt the weights of the neurons in the previous layers. The adaptation depends on the output error and the derivative of the transfer function at each neuron.

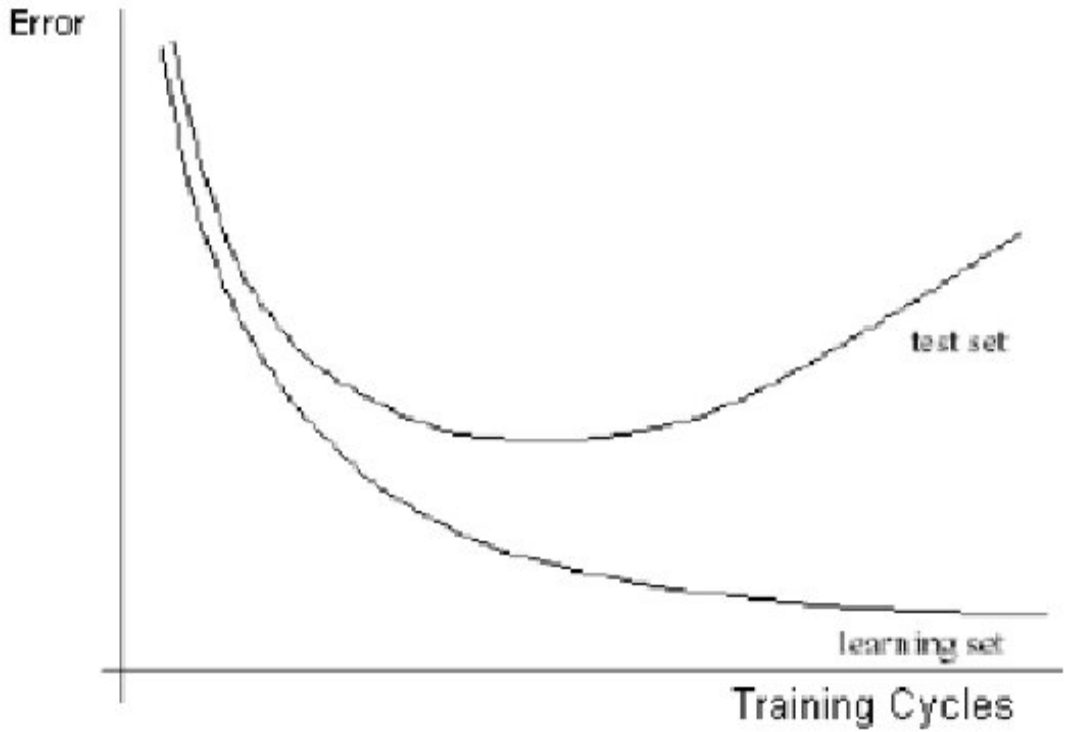
Network Size and Training

If the network is too large (i.e. too many layers or neurons) overfitting occurs ('learning by heart').
If the network is too small underfitting becomes a problem.



Validation

Solution: Separation of the training data into training and validation data. Training goes on based on the training data only until the error in the validation data increases.



Momentum Term

To improve the speed and quality of the gradient descent approach, a momentum term can be used. Each time a weight is adapted, a certain percentage of the previous adaptation is added. This has three advantages

- The speed of the optimization increases along flat plateaus.
- Oscillations in narrow valleys can be avaoided.
- Local minima can be avoided.

Unsupervised Learning

Unsupervised networks are used for finding optimal **data representations** based on feature vectors. During the adaptation process the feature vectors distribute themselves in the data space. New data points can be classified based on the closest neuron in feature space.

Self-Organizing Networks

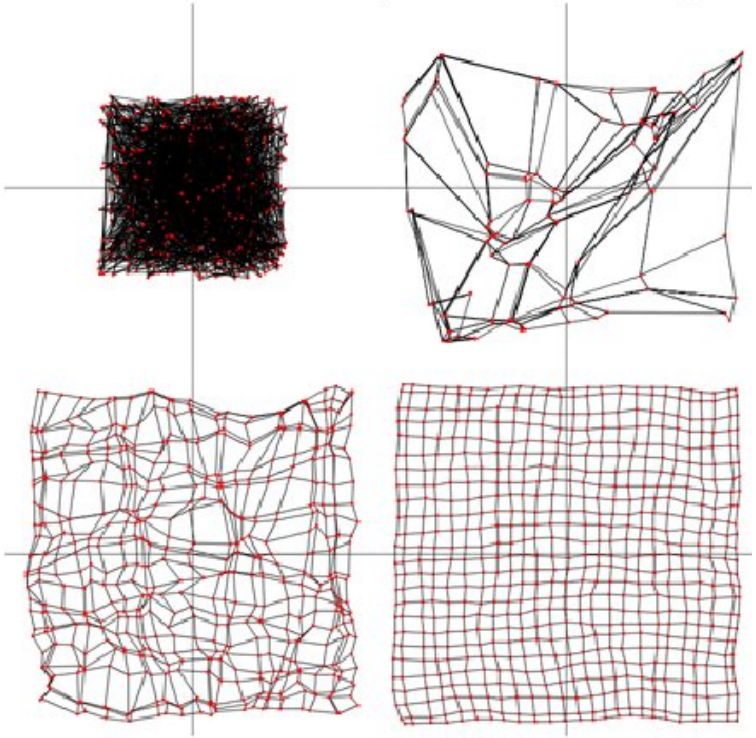
Given a probability distribution $P(x)$ of data vectors x. We want to represent the data vectors as good as possible by a fixed number of neurons. Each neuron represents a **feature vector**, which is adapted to the data.
At each time step a randomly chosen feature vector is presented to the network. The distance of each neuron's feature vector to the given data vector is evaluated and sorted leading to indices i_1 (for the closest) to i_n (for the farthest neuron). The closer the feature vector is to the presented data vector the more it is adapted to the presented data point.

$$w_{i_k}^{t+1} = w_{i_k}^{t} + \epsilon \exp^{-\frac{k}{\lambda}(x - w_{i_k}^{t})}$$

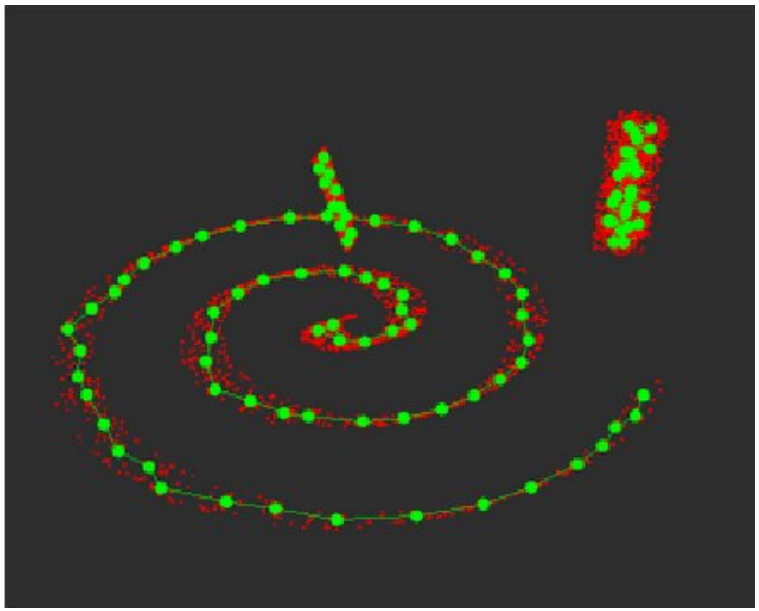
Here, ϵ is the adaptation step size and λ the neighborhood range.

Self-Organizing Maps

Two main types of self-organizing networks are known: Kohonen Maps and Neural Gas. Kohonen Maps preserve the neighborhood structure between neurons (e.g. initialization becomes less important), whereas the neural gas neurons are free to move in space allowing for arbitrary topologies.



Kohohnen Maps



Neural Gas