# Neuromorphic Electronics

## Selected Neural Learning Algorithms

*Philipp Häfliger* *hafliger@ifi.uio.no*

# A definition

Learning is the search of a parameter space in order to optimize performance.

# Terms (1/3)

$\vec{w}$ The vector of parameters that defines the multi-dimensional search space of the learning algorithm. In neural networks these parameters are the synaptic *weights* and they are sometimes also organized in a matrix $\mathbf{W}$, instead of in a vector.

$\vec{x}$ The (sensory) input to the system. Also organized as a matrix sometimes, e.g. when the input is an image.

$\vec{y}(\vec{x}, \vec{w})$ The (behavioural) output of the system that changes with learning.

# Terms (2/3)

$\vec{d}(\vec{x})$ The 'desired' output of the system, a teacher or expert opinion. It is normally not defined for the whole input space spanned by $\vec{x}$. (Thus, the system needs to *generalize* what we learn from a teacher and apply it to unknown situations $\vec{x}$)

$P(\vec{y})$ A performance evaluation function to judge the quality of the learning state of the system. In general this function can be stochastic. It can also have multiple extrema.

# Terms (3/3)

$E(\vec{y}, \vec{d})$ An error function, a special case of a performance evaluation function, that evaluates system performance as compared to the expert/teacher.

$\mu$ The learning rate. A parameter that is used by many learning algorithms influencing the learning speed and quality.

# Characterisation of Learning Rules

- supervised
  - supervised by expert (learning a target function)
  - reinforcement, supervised by critic (optimizing performance)
- unsupervised (optimizing statistics, data reduction/compression)
  - Correlation, Association, Hebbian Learning, Spike based learning
  - Competitive Learning, LVQ, Competitive Hebbian Learning
  - Optimizing data reduction, PCA, ICA

# Characterisation of Learning Supervised by an Expert (1/2)

- continuous target functions
  - gradient descent, Error Backpropagation (for space continuous target functions, interpolation)
  - temporal difference learning (TD$\lambda$, for time continuous target functions)
  - statistical methods, interpolation
  - weight perturbation (can also be used in reinforcement)

# Characterisation of Learning Supervised by an Expert (2/2)

- supervised classification
  - supervised LVQ
  - support vector machines

# Characterisation of Learning Supervised by a Critic, reinforcement Learning

- associative reward-penalty
- evolutionary algorithms
- deduction, induction (predicate logic learning)

# Gradient Descent

$$E = \|\vec{d} - \vec{y}\|$$

$$\frac{d\vec{w}}{dt} = -\mu \frac{dE}{d\vec{w}}$$

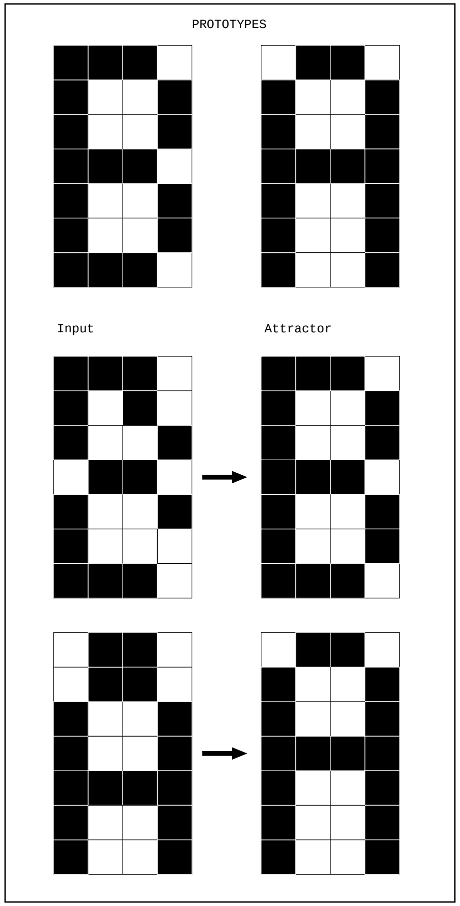
# Hebbian Learning

$$\frac{dw_i}{dt} = ...x_i y_i...$$

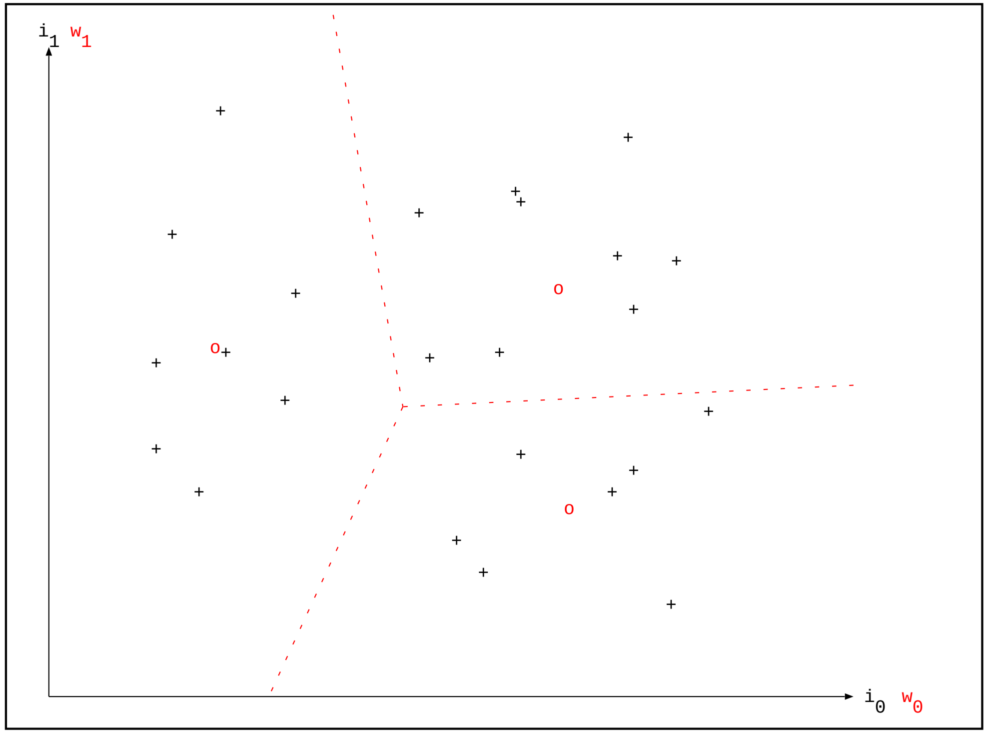
Example Associative Memory:

$$\frac{dw_{i,j}}{dt} = \frac{1}{t}(-w_{i,j} + x_j y_i)$$

# Hebbian Learning and Associative Memory

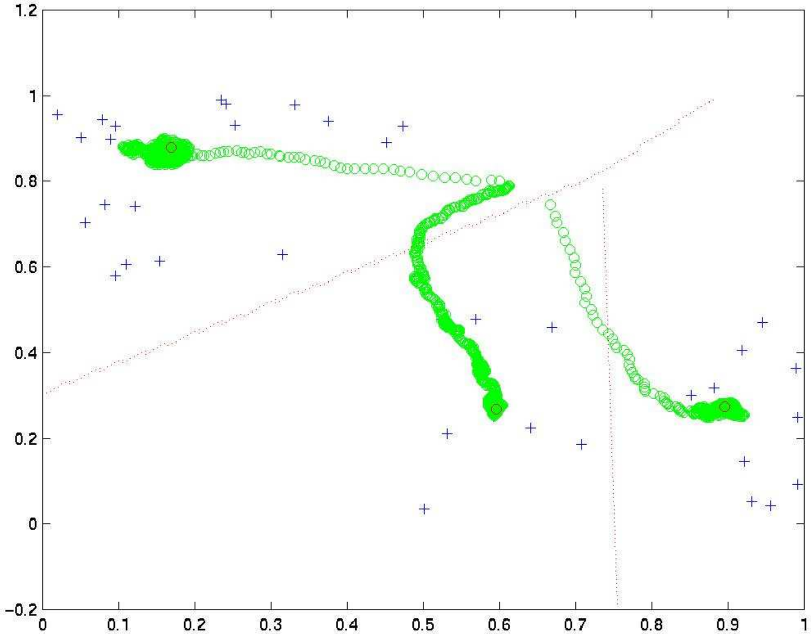


# Learning Vector Quantisation (1/2)



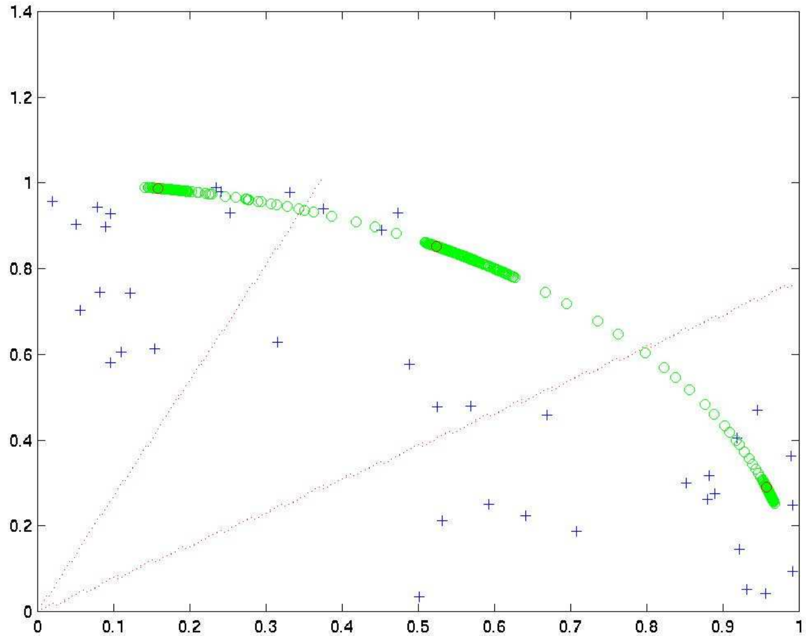

$$y = \|\vec{w} - \vec{x}\|^{-1}$$

# Learning Vector Quantisation (2/2)



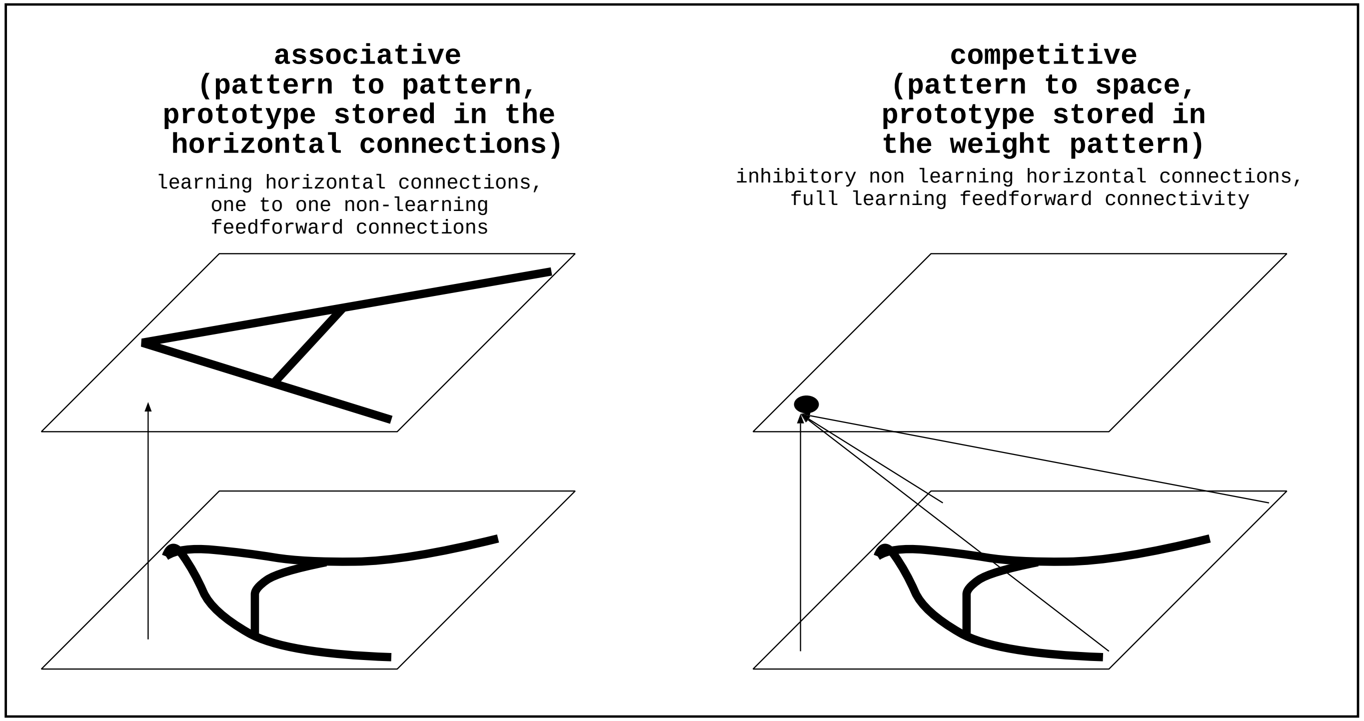

$$\frac{d\vec{w}}{dt} = \mu y(\vec{x} - \vec{w})$$

# Competitive Hebbian Learning

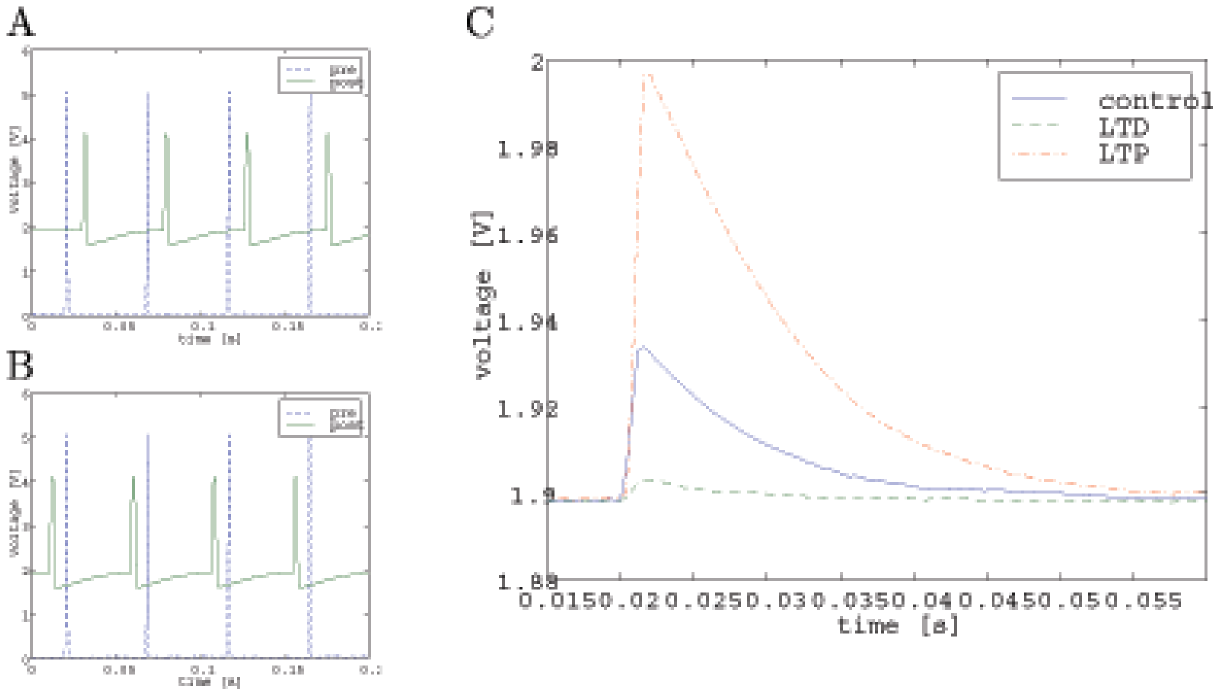



$$\frac{d\vec{w}}{dt} = y(\mu_0\vec{x} - \mu_1 y\vec{w})$$

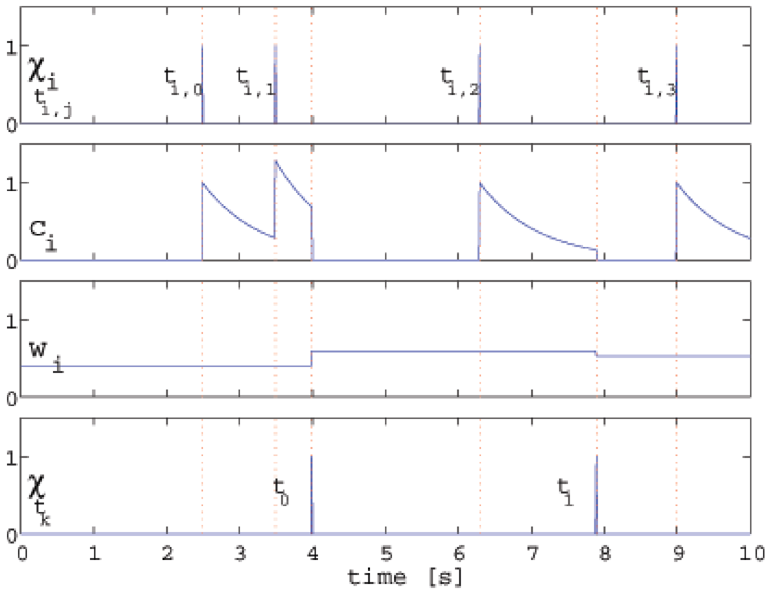
# Comparison Associative / Comp. Hebbian



# Spike Based Learning (1/2)



# Spike Based Learning (2/2)




$y = \sum_s \delta(t - t_s)$

$\frac{d\vec{w}}{dt} = (\mu_0 \vec{c} - \mu_1 \vec{w}) y$