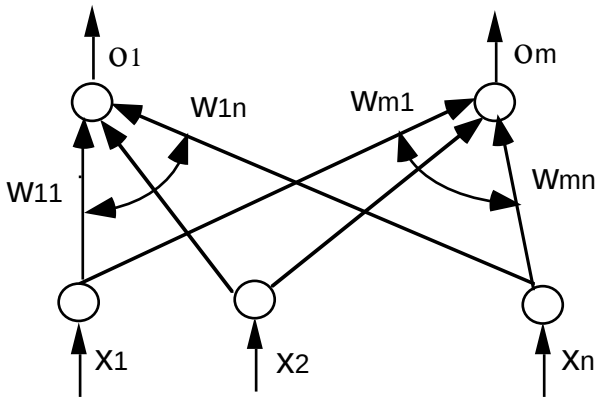
Neural networks IV: The winner-take-all learning

Unsupervised classification learning is based on clustering of input data. No *a priori* knowledge is assumed to be available regarding an input's memebership in a particular class. Rather, gradually detected characteristics and a history of training will be used to assist the network in defining classes and possible boundaries between them.

Clustering is understood to be the grouping of similar objects and separating of dissimilar ones.

We discuss Kohonen's network, which classifies input vectors into one of the specified number of m categories, according to the clusters detected in the training set

$$\{x^1, \ldots, x^K\}.$$



A winner-take-all learning network.

The learning algorithm treats the set of m weight vectors as variable vectors that need to be learned. Prior to the learning, the normalization of all (randomly chosen) weight vectors is required.

The weight adjustment criterion for this mode of training is the selection of w_r such that

$$\|x - w_r\| = \min_{i=1,\ldots,m} \|x - w_i\|$$

The index r denotes the *winning* neuron number corresponding to the vector w_r, which is the clos-

est approximation of the current input x.

Using the equality

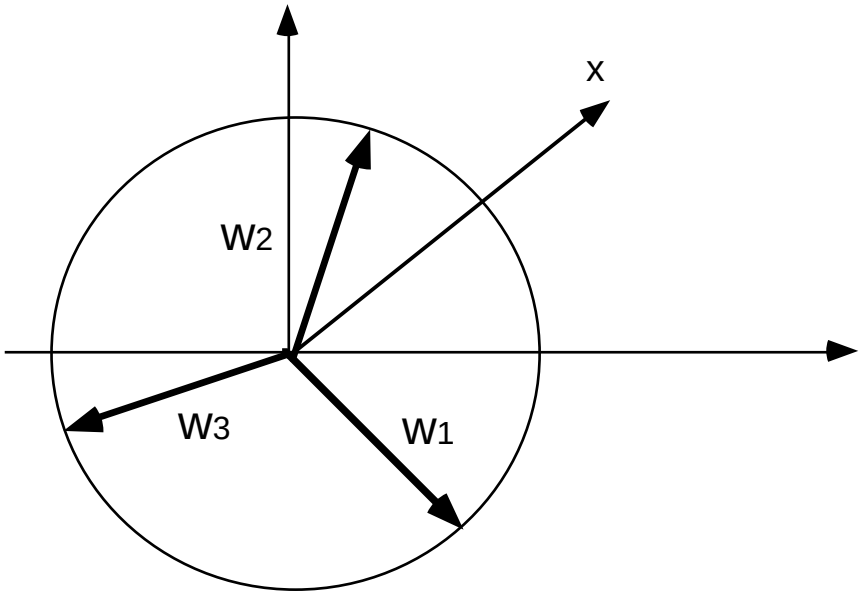
$$\|x - w_i\|^2 =< x - w_i, x - w_i >$$

$$=< x, x > -2 < w_i, x > + < w_i, w_i >=$$

$$\|x\|^2 - 2 < w, x > + \|w_i\|^2 = \|x\|^2 - 2 < w_i, x > + 1$$

we can infer that searching for the minimum of m distances corresponds to finding the maximum among the m scalar products

$$< w_r, x >= \max_{i=1,\ldots,m} < w_i, x >$$



The winner weight is w_2.

Taking into consideration that

$$\|w_i\| = 1,\ \forall i \in \{1, \ldots, m\}$$

the scalar product $< w_i, x >$ is nothing else but the projection of x on the direction of w_i. It is clear that the closer the vector w_i to x the bigger the projection

of x on w_i.

Note that $< w_r, x >$ is the activation value of the *winning* neuron which has the largest value net$_i$, $i = 1, \ldots, m$.

When using the scalar product metric of similarity, the synaptic weight vectors should be modified accordingly so that they become more similar to the current input vector.

With the similarity criterion being $\cos(w_i, x)$, the weight vector lengths should be identical for this training approach. However, their directions should be modified.

Intuitively, it is clear that a very long weight vector could lead to a very large output ofits neuron even if there were a large angle between the weight vector and the pattern. This explains the need for weight

normalization.

After the winning neuron has been identified and declared a winner, its weight must be adjusted so that the distance $\|x - w_r\|$ is reduced in the current training step.

Thus, $\|x - w_r\|$ must be reduced, preferebly along the gradient direction in the weight space $w_{r1}, \ldots, w_{rn}$

$$\frac{d\|x - w\|^2}{dw}$$

$$= \frac{d < x - w, x - w >}{dw}$$

$$= \frac{d(< x, x > -2 < w, x > + < w, w >)}{dw}$$

$$= \frac{d < x, x >}{dw} - 2 \times \frac{d < w, x >}{dw} + \frac{d < w, w >}{dw}$$

$$= 0 - 2 \times \frac{d(w_1 x_1 + \cdots + w_n x_n)}{dw} + \frac{d(w_1^2 + \cdots + w_n^2)}{dw}$$

$$= -2 \times \left(\frac{\partial}{\partial w_1}[w_1 x_1 + \cdots + w_n x_n], \ldots, \right.$$
$$\left. \frac{\partial}{\partial w_n}[w_1 x_1 + \cdots + w_n x_n] \right)^T$$
$$+ \left(\frac{\partial}{\partial w_1}[w_1^2 + \cdots + w_n^2], \ldots, \frac{\partial}{\partial w_n}[w_1^2 + \cdots + w_n^2)] \right)^T$$

$$= -2(x_1, \ldots, x_n)^T + 2(w_1, \ldots, w_n)^T$$

$$= -2(x - w).$$

It seems reasonable to reward the weights of the winning neuron with an increment of weight in the negative gradient direction, thus in the direction $x - w_r$.

We thus have

$$w_r := w_r + \eta(x - w_r)$$

where η is a small lerning constant selected heuristically, usually between 0.1 and 0.7.

The remaining weight vectors are left unaffected.

Summary 1. *Kohonen's learning algorithm can be summarized in the following three steps*

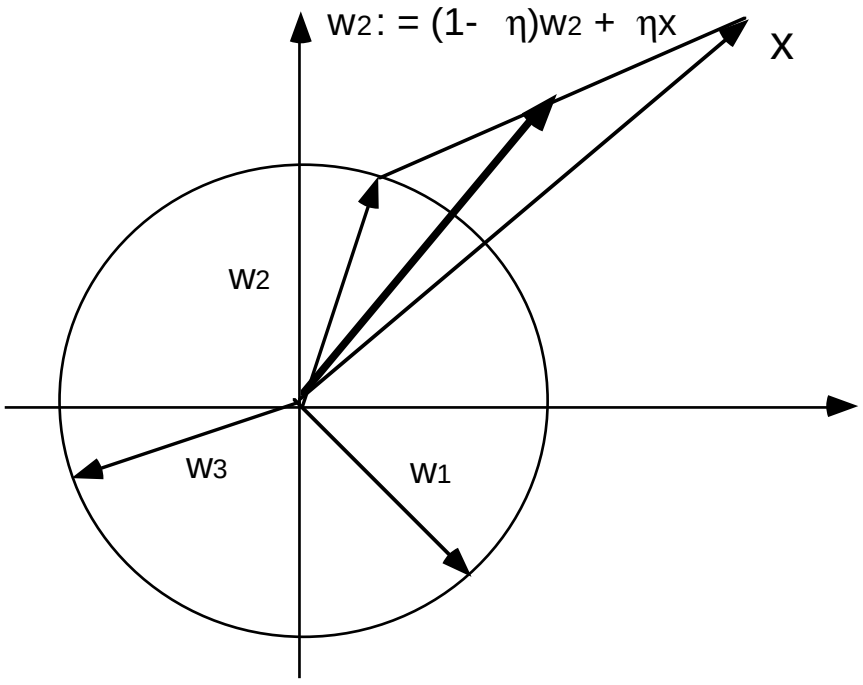
- **Step 1** $w_r := w_r + \eta(x - w_r),\ o_r := 1,$ (r is the winner neuron)
- **Step 2** $w_r := w_r / \|w_r\|$ (normalization)

- **Step 3** $w_i := w_i,\ o_i := 0,\ i \neq r$ (losers are unaffected)

It should be noted that from the identity

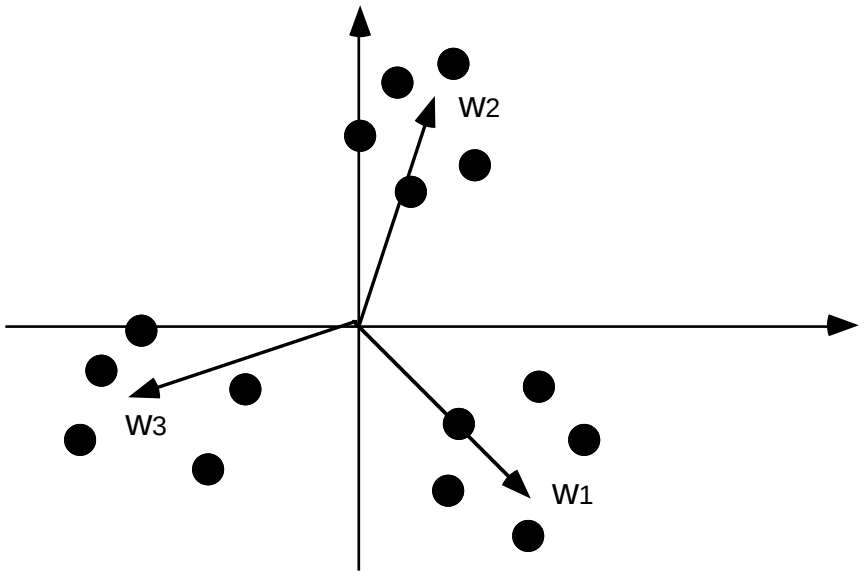
$$w_r := w_r + \eta(x - w_r) = (1 - \eta)w_r + \eta x$$

it follows that the updated weight vector is a convex linear combination of the old weight and the pattern vectors.



Updating the weight of the winner neuron.

In the end of the training process the final weight vectors point to the centers of gravity of classes.



The final weight vectors point to the center of gravity of the classes.

The network will only be trainable if classes/clusters of patterns are linearly separable from other classes by hyperplanes passing through origin.

To ensure separability of clusters with *a priori* unknown numbers of training clusters, the unsupervised training can be performed with an excessive number of neurons, which provides a certain separability safety margin.

During the training, some neurons are likely not to develop their weights, and if their weights change chaotically, they will not be considered as indicative of clusters.

Therefore such weights can be omitted during the recall phase, since their output does not provide any essential clustering information. The weights of remaining neurons should settle at values that are indicative of clusters.

In many practical cases instead of linear activation functions we use semi-linear ones. The next table shows the most-often used types of activation functions.

Linear: $f(< w, x >) = w^T x$

Piecewise linear:

$$f(< w, x >) = \begin{cases} 1 & \text{if } < w, x > > 1 \\ < w, x > & \text{if } | < w, x > | \leq 1 \\ -1 & \text{if } < w, x > < -1 \end{cases}$$

Hard : $f(< w, x >) = \text{sign}(w^T x)$

Unipolar sigmoidal:

$$f(< w, x >) = 1/(1 + \exp(-w^T x))$$

Bipolar sigmoidal (1): $f(< w, x >) = \tanh(w^T x)$

Bipolar sigmoidal (2):

$$f(< w, x >) = \frac{2}{1 + \exp(w^T x)} - 1$$

Table Activation functions.

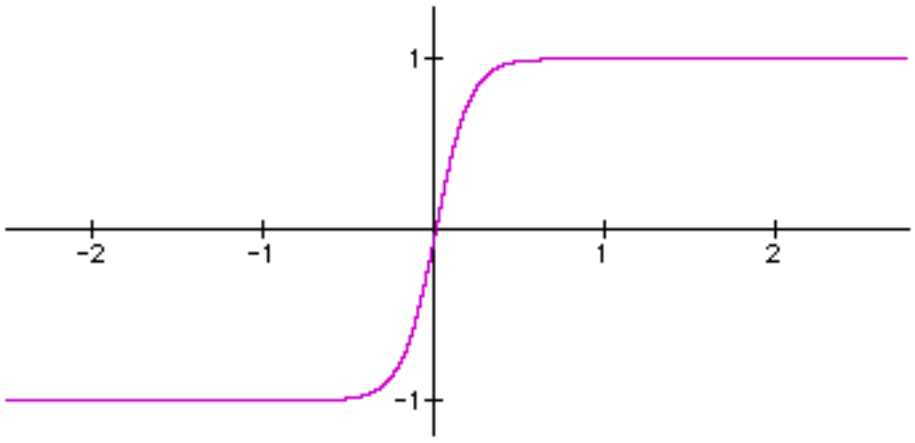
The derivatives of sigmoidal activation functions are extensively used in learning algorithms.

- If f is a bipolar sigmoidal activation function of the form

$$f(t) = \frac{2}{1 + \exp(-t)} - 1.$$

Then the following equality holds

$$f'(t) = \frac{2\exp(-t)}{(1 + \exp(-t))^2} = \frac{1}{2}(1 - f^2(t)).$$



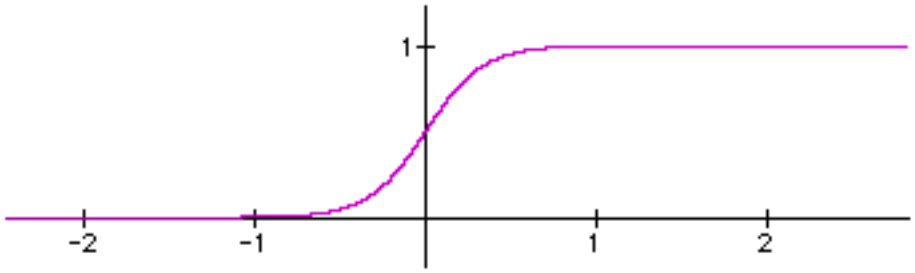
Bipolar activation function.

- If f is a unipolar sigmoidal activation function of the form

$$f(t) = \frac{1}{1 + \exp(-t)}.$$

Then f' satisfies the following equality

$$f'(t) = f(t)(1 - f(t)).$$



Unipolar activation function.

Exercise 1. *Let f be a bip olar sigmoidal activation function of the form*

$$f(t) = \frac{2}{1 + \exp(-t)} - 1.$$

Show that f satisfies the following differential equal-

ity

$$f'(t) = \frac{1}{2}(1 - f^2(t)).$$

Solution 1. *By using the chain rule for derivatives of composed functions we get*

$$f'(t) = \frac{2exp(-t)}{[1 + exp(-t)]^2}.$$

From the identity

$$\frac{1}{2}\left[1 - \left(\frac{1 - exp(-t)}{1 + exp(-t)}\right)^2\right] = \frac{2exp(-t)}{[1 + exp(-t)]^2}$$

we get

$$\frac{2exp(-t)}{[1+exp(-t)]^2} = \frac{1}{2}(1 - f^2(t)).$$

Which completes the proof.

Exercise 2. *Let f be a unipolar sigmoidal activation function of the form*

$$f(t) = \frac{1}{1+\exp(-t)}.$$

Show that f satisfies the following differential equality

$$f'(t) = f(t)(1 - f(t)).$$

Solution 2. *By using the chain rule for derivatives of composed functions we get*

$$f'(t) = \frac{exp(-t)}{[1+exp(-t)]^2}$$

and the identity

$$\frac{1}{[1+exp(-t)]^2} = \frac{exp(-t)}{1+exp(-t)}\left(1 - \frac{exp(-t)}{1+exp(-t)}\right)$$

verifies the statement of the exercise.