Neural Networks *Hertentamen II (2007)*

The problems are to be solved within 3 hrs. The use of supporting material (books, notes, calculators) is not allowed. In each of the four problems you can achieve up to 2.5 points, with a total maximum of 10 points.

---

1. Perceptron storage problem

Consider a set of data $\mathbb{D} = (\xi^\mu, S^\mu)_{\mu=1}^P$ where $\xi^\mu \in \mathbb{R}^N$ and $S^\mu \in \{+1, -1\}$. In this problem, we assume that $\mathbb{D}$ is homogeneously linearly separable.

a) Formulate the perceptron storage problem as the search for a vector $\mathbf{w} \in \mathbb{R}^N$ which satisfies a set of equations. Re-write the problem using a set of inequalities.

b) Define the stability $\kappa(\mathbf{w})$ of a perceptron solution $\mathbf{w}$ with respect to a given set of data $\mathbb{D}$. Give a geometric interpretation (sketch an illustration) and explain (in words) why $\kappa(\mathbf{w})$ quantifies the stability of the outputs with respect to noise.

c) Assume we have found two different solutions $\mathbf{w}^{(1)}$ and $\mathbf{w}^{(2)}$ of the perceptron storage problem for $\mathbb{D}$. Assume furthermore that $\mathbf{w}^{(1)}$ can be written as a linear combination

$$\mathbf{w}^{(1)} = \sum_{\mu=1}^{P} x^\mu \, \xi^\mu \, S^\mu \quad \text{with} \quad x^\mu \in \mathbb{R}$$

whereas the difference $(\mathbf{w}^{(2)} - \mathbf{w}^{(1)})$ is orthogonal to all the ξ^μ in $\mathbb{D}$, i.e. $(\mathbf{w}^{(2)} - \mathbf{w}^{(1)}) \cdot \xi^\mu = 0$ for $\mu = 1, 2, \ldots P$.

Show that $\kappa(\mathbf{w}^{(1)}) > \kappa(\mathbf{w}^{(2)})$. What does the result imply for the perceptron of optimal stability $\mathbf{w}_{max}$?

2. Learning a linearly separable rule

Here we consider perceptron training from linearly separable data $\mathbb{D} = \{\xi^\mu, S_R^\mu\}_{\mu=1}^P$ where noise-free labels $S_R^\mu = \text{sign}[\mathbf{w}^* \cdot \xi^\mu]$ are provided by a teacher vector $\mathbf{w}^* \in \mathbb{R}^N$ with $|\mathbf{w}^*| = 1$. Assume that by some training process we have obtained a perceptron vector $\mathbf{w} \in \mathbb{R}^N$ from the data $\mathbb{D}$.

a) Define the terms *training error* and *generalization error* in the context of this situation.

b) Assume that random input vectors $\xi \in \mathbb{R}^N$ are generated with equal probability anywhere on the *hypersphere* with squared radius $\xi^2 = 1$. Given $\mathbf{w}^*$ and a vector $\mathbf{w} \in \mathbb{R}^N$, what is the probability for *disagreement*, $\text{sign}[\mathbf{w} \cdot \xi] \neq \text{sign}[\mathbf{w}^* \cdot \xi]$? You can "derive" the result from a sketch of the situation in $N = 2$ dimensions.

c) Explain Rosenblatt's perceptron algorithm for a given set of examples $\mathbb{D}$ in terms of a few lines of *pseudocode*.

3. **Classification with multilayer networks**

a) Consider the so-called *committee machine* with inputs $\boldsymbol{\xi} \in I\!R^N$, K hidden units $\left(\{\sigma_k = \pm 1\}_{k=1}^{K}\right)$, and corresponding weight vectors $\mathbf{w}_k \in I\!R^N$. Define the output $S(\boldsymbol{\xi}) \in \{-1, +1\}$ as a function of the input.

b) Now consider the *parity machine* with N-dim. input and K hidden units. Define the output $S(\boldsymbol{\xi}) \in \{-1, +1\}$ as a function of the input.

c) Illustrate the case $K = 3$ for *parity* and *committee machine* in terms of a geometric interpretation. Why would you expect that the parity machine should have a greater *storage capacity* in terms of implementing random sets $I\!D = \{\boldsymbol{\xi}^\mu, S(\boldsymbol{\xi}^\mu)\}$?

4. **Regression**

a) Explain the term *overfitting* in the context of a simple regression problem. What is the meaning of *bias* and *variance* in this context?

b) The choice of the appropriate network complexity (size, architecture) is a key problem in learning. Explain how the method of *n-fold cross validation* can be used in this context. You may discuss it in terms of the same example as in (a).

c) Consider a feed–forward continuous neural network ($N-2-1$ architecture) with output

$$\sigma(\boldsymbol{\xi}) = \sum_{j=1}^{2} v_j \, g(\mathbf{w}_j \cdot \boldsymbol{\xi}).$$

Here, $\boldsymbol{\xi} \in I\!R^N$ denotes an input vector, $\mathbf{w}_1, \mathbf{w}_2 \in I\!R^N$ are the adaptive weight vectors in the first layer and $v_1, v_2 \in I\!R$ are the adaptive hidden-to-output weights. Assume the transfer function $g(x)$ has the known derivative $g'(x)$.
Given a single training example $\{\boldsymbol{\xi}^\mu, \tau^\mu\}$ with input $\boldsymbol{\xi}^\mu$ and output $\tau^\mu \in I\!R$ consider the quadratic error measure

$$\varepsilon^\mu = \frac{1}{2}\left(\sigma(\boldsymbol{\xi}^\mu) - \tau^\mu\right)^2.$$

Write down a gradient descent step for all adaptive weight with respect to the (single example) cost function ε^μ.