Neural Networks (2011/12)
Second Exam, April 2012

Four problems are to be solved within 3 hours. **The use of supporting material (books, notes, calculators) is not allowed.** You can achieve up 9 points, in total. The exam grade will be "1.0 + your number of points".
Important hints: never just answer a question with "Yes" or "No", always give arguments for your conclusion. Be as precise as possible and use math where it makes sense.

---

1) Perceptron storage problem (2 points)

Consider a set of data $I\!D = \{\xi^\mu, S^\mu\}_{\mu=1}^P$ where $\xi^\mu \in I\!R^N$ and $S^\mu \in \{+1, -1\}$.

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

b) Define and explain the *Rosenblatt* perceptron algorithm for a given set of examples $I\!D$. Be precise, for instance by writing it in a few lines of pseudocode. Also define a possible stopping criterion. What is known about the convergence of the algorithm?

c) Explain the basic idea of the *Pocket algorithm* as an attempt to minimize the number of errors when the data set is not linearly separable. Pseudocode is not required here.

2) Learning a linearly separable rule (2 points)

Consider a linearly separable set of data $I\!D = \{\xi^\mu, S^\mu\}_{\mu=1}^P$ where $\xi^\mu \in I\!R^N$ and $S^\mu \in \{+1, -1\}$ are correct, noise–free labels.

a) Define precisely the following terms:
(I) the stability κ^μ of an example $\{\xi^\mu, S^\mu\}$.
(II) the stability κ of a perceptron weight vector $\mathbf{w}$.
Provide a graphical illustration of (I) and (II) based on the geometrical interpretation of linearly separable functions.

b) Define and explain the term *version space* precisely in this context, provide a mathematical definition as a set of vectors and also a simple graphical illustration.

c) Explain in words, referring to the graphical illustration in (b), how the consideration of a new (correctly labeled) example $\{\xi^\nu, S^\nu\}$ that is added to the data set can decrease the size of the version space. Does every new example have this effect?

3) Multilayered Neural Networks for classification (2 points)

a) Explain the so-called committee machine with inputs $\boldsymbol{\xi} \in \mathbb{R}^N$, K hidden units $\sigma_k \in \{-1, +1\}, (k = 1, 2, \ldots K)$, and corresponding weight vectors $\mathbf{w}_k \in \mathbb{R}^N$. Define precisely the output $S(\boldsymbol{\xi})$ as a function of the input in terms of an equation and explain it in words.

b) Illustrate a committee machine with $K = 3$ hidden units in terms of the geometric interpretation in input space.

c) In class we discussed the basic strategy of *tiling* algorithms, where units are added to the network until a given labelled data set is implemented without errors. (You do not have to explain the algorithm here.) Explain in your own words, why such a strategy may yield poor generalization performance when learning a rule from examples.

4) Learning by gradient descent and over-fitting (3 points)

a) Discuss qualitatively (in words) the role of the learning rate rate η in training by batch gradient descent. Assume η is constant. What can be the consequences of chosing η too small or too large, respectively, for the success of training? To what extent is the role of η different in stochastic gradient descent?

b) Explain the effect of *overfitting* in terms of the training of a neural network with one hidden layer from a given set of example data. Provide a schematic sketch of the training and generalization error as functions of the number K of hidden units in the presence of overfitting.

c) Explain the method of *weight decay* and explain how it can help to avoid overfitting in the training of neural networks with sigmoidal activation functions. Consider the update of a single weight vector $\mathbf{w} \in \mathbb{R}^N$ by gradient descent with respect to a cost function E. How does weight decay modify the update?