Artificial neural networks FFR135
Examples sheet 4

Classification problem. Download the data set from the course home page. It contains 2000 rows of data. The classification (class -1 or 1) is given by the first element of each row. Your task is to use a neural network to classify this data set (without the first column). Solve this classification task by using a combination of simple competitive learning and supervised simple-perceptron learning as described in the lectures.

- For the competitive learning part, use Gaussian nodes to classify the data set (without the first column). [Hint: no need to normalise the data. Why?] The winning neurone for pattern $\boldsymbol{x}$ is obtained by maximising the Gaussian activation function, $g_j(\boldsymbol{x})$:

$$g_j(\boldsymbol{x}) = \frac{\exp\Big[-\|\boldsymbol{x} - \boldsymbol{w}_j\|^2/2\Big]}{\sum_i \exp[-\|\boldsymbol{x} - \boldsymbol{w}_i\|^2/2]} \;. \tag{1}$$

  Here $\| \cdots \|$ denotes the Euclidean distance, and j is one of the Gaussian nodes. Suggested parameter values for competitive learning: learning rate $\eta = 0.02$, number of updates 10^5. Initialise all weights to random numbers uniformly distributed between -1 and 1.

- For the simple-perceptron learning use the outputs $g_j(\boldsymbol{x})$ (that you obtained after the competitive learning is completed) as inputs to one simple perceptron. Thus, the number of input units in your perceptron network is equal to the number of Gaussian nodes that you used in the competitive learning part. The number of output units in the simple-perceptron network is one. Using the gradient-descent algorithm train the perceptron to recognise the classes in the data set. Use *tanh* activation function with the parameter $\beta = 1/2$. Randomly divide the data set in two parts: one for training (70% of the data points) and the other for validation (the remaining 30% of the points). Suggested parameter values for supervised learning: learning rate $\eta = 0.1$, take at least 3000 training steps. Initialise weights and thresholds to random numbers uniformly distributed between -1 and 1.

Tasks

1. Set the number of Gaussian nodes for the competitive learning to 5. Perform the above algorithm 20 times.

1a. Plot the training and the validation energy that you obtain during training in each of the runs made. Compute the average energy at the end of training. Discuss your results. **(1p)**

1b. Plot the classification error in the training and validation set during training in the runs you made. Compute the average classification error at the end of training. Discuss the results. **(1p)**

1c. From the runs you made, choose the one with the best performance (smallest classification error). Numerically determine the decision boundary in input space learned by your network: find the line in the input plane where the output of the network equals zero. Plot the decision boundary together with the input data (use different symbols and/or colours for the two classes). Also plot the positions of the Gaussian nodes obtained at the end of competitive learning in your network (that is, display the weights of the Gaussian nodes that you obtained after the simple competitive learning). Discuss your results. **(1p)**

2. Repeat the task **1.** but with 20 Gaussian nodes.

2a. Plot the classification error in the training and classification set during the trainings you made. Compute the average classification error at the end of training. Discuss differences to the results you obtained in task **1b. (1p)**

2b. Among the runs you made, choose one with the best performance. Numerically determine the decision boundary in input space learned by your network. Plot the decision boundary together with the input data. Also plot the positions of the Gaussian nodes obtained at the end of competitive learning in your network. Discuss your results. **(1p)**

3. How does the performance of the network depend on the number of Gaussian nodes? Test the performance of the network with $1, 2, \ldots, 20$ Gaussian nodes (the results for 5 and 20 nodes are readily available from tasks 1 and 2). For each value of the number of Gaussian nodes, perform 20 independent trainings, and compute the classification error in the training and in the validation set that you obtain at the end of training. Plot the average classification error obtained at the end of training as a function of the number of Gaussian nodes. Discuss your results. **(1p)**