Exercises for the course
Machine Learning 2
Summer semester 2016

Abteilung Maschinelles Lernen
Institut für Softwaretechnik und theoretische Informatik
Fakultät IV, Technische Universität Berlin
Prof. Dr. Klaus-Robert Müller
Email: klaus-robert.mueller@tu-berlin.de

Exercise Sheet 8

Exercise 1: Convolution Kernel (10+20 P)

Let x, x' be two univariate real-valued discrete time series. We define the convolution kernel

$$k(x, x') = \|x * x'\|^2$$

that measures the similarity between them.

(a) *Write* a test in Python that verifies empirically that the kernel is positive semi-definite and *run* it.

(b) *Show* that the convolution kernel is positive semi-definite and *find* an explicit feature map for this kernel.

Exercise 2: Backprop in the Convolution (10+10 P)

In the slides, the forward computation of a 1D convolutional layer is defined by the cross-correlation: $y = w \star x$, and the corresponding error gradients with respect to its input x and weights w have been computed. We now assume that the forward computation is defined by the convolution $y = w * x$, and that E is an error function that depends on y.

(a) *Express* the gradient $\frac{\partial E}{\partial x}$ as a function of $\frac{\partial E}{\partial y}$ and w.

(b) *Express* the gradient $\frac{\partial E}{\partial w}$ as a function of $\frac{\partial E}{\partial y}$ and x.

Exercise 3: Recurrent Neural Networks (10+10+10+10 P)

We would like to learn a nonlinear dynamical system with state $\boldsymbol{x} \in \mathbb{R}^d$ and modeled by the set of differential equations:

$$\forall_{j=1}^d : \ \dot{x}_j = 0.1\big(\tanh\big(\textstyle\sum_{i=1}^d x_i w_{ij} + b_j\big) - x_j\big)$$

where w_{ij}, b_j are the parameters of the system.

(a) Applying Euler discretization with time step 1, *create* a recurrent neural network associated to this dynamical system, and *write* its transition function (the function mapping the current state to the state at the next time step).

(b) *Draw* a graph representing the recurrent neural network unfolded in time, and *annotate* it with the relevant variables: ($x_i^{(t)}$ for all dimensions i and time steps t, and the parameters of the model w_{ij}, b_j).

(c) *Compute* the derivative of the activation $x_i^{(t)}$ with respect to the activations at previous time steps.

(d) *Compute* the derivative of the activation $x_i^{(t)}$ with respect to the parameters of the model.