Machine Learning 2 – Group ESHG
Assignment 05

Willi Gierke, Arik Elimelech, Mehmed Halilovic, Leon Sixt

May 31, 2017

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.

Task 1

a

Positive Semi-Definite Kernel Proof

May 31, 2017

```
In [2]: import argparse
        import numpy as np
        from tqdm import tqdm

        def run(n, d, iterations):
            """Apply the specified kernel on an certain amount of time series
            of a certain length for a certain amount of iterations
            n - Amount of time series
            d - Length of time series
            iterations - Iterations that should be performed
            """

            for _ in tqdm(range(iterations)):
                X = np.random.rand(d, n)
                c = np.random.rand(n)

                sum_ = 0
                for i in range(n):
                    for j in range(n):
                        convolution = np.convolve(X[i, :], X[j, :], 'same')
                        squared_norm = np.square(np.linalg.norm(convolution))
                        squared_norm *= c[i] * c[j]
                        sum_ += squared_norm
                if sum_ < 0:
                    print("Kernel value is non-positive")
                    return

            print("Could not find non-positive kernel value")

        run(100, 100, 100)

100%|| 100/100 [00:22<00:00,  4.45it/s]

Could not find non-positive kernel value
```

b

The kernel is given by:

$$k(x, y) = \|x * y\|^2 \tag{1}$$

To be positive semi-definite, the kernel has to fulfill:

$$\sum_{i=1}^{n} \sum_{j=1}^{n} c_i c_j k(x_i, x_j) \tag{2}$$

for $n \in \mathbb{N}$, $\{x_1, \ldots, x_n\} \in \mathbb{R}^{d \times n}$ and $c \in \mathbb{R}^n$.

For the given kernel:

$$\sum_{i=1}^{n}\sum_{j=1}^{n} c_i c_j \|x_i * x_j\|^2 \tag{3}$$

$$= \sum_{i=1}^{n}\sum_{j=1}^{n} c_i c_j \sum_{t=0}^{T} (x_i * x_j)_t^2 \tag{4}$$

$$\tag{5}$$

We can rewrite the convolution as fourier transformation. Here, $\hat{x}_i$ denotes the fourier transform of x_i and F^{-1} the inverse fourier transformation.

$$\|x_i * x_j\|^2 = \tag{6}$$

$$= \|F^{-1}(\hat{x}_i \cdot \hat{x}_j)\|^2 \tag{7}$$

$$\tag{8}$$

We assume that the feature space, ask for in the exercise, is connected to the fourier transformation of x. However, the inverse fourier transform obstruct us to separate the kernel into two feature maps.

2

Using the new definition

$$y_t = [w * x]_t = \sum_s w_s x_{t-s} \quad ,$$

we get for the derivatives:

$$\begin{aligned}
\frac{\partial E}{\partial x_u} &= \sum_t \frac{\partial E}{\partial y_t}\frac{\partial y_t}{\partial x_u} = \sum_t \frac{\partial E}{\partial y_t} \sum_s w_s \frac{\partial x_{t-s}}{\partial x_u} \\
&= \sum_{t,s} \frac{\partial E}{\partial y_t} w_s 1_{u=t-s} = \sum_t \frac{\partial E}{\partial y_t} w_{t-u} = \left[\frac{\partial E}{\partial y} \star w\right]_{-u} \\
\frac{\partial E}{\partial w_u} &= \sum_t \frac{\partial E}{\partial y_t}\frac{\partial y_t}{\partial w_u} = \sum_t \frac{\partial E}{\partial y_t} \sum_s \frac{\partial w_s}{\partial w_u} x_{t-s} \\
&= \sum_{t,s} \frac{\partial E}{\partial y_t} x_{t-s} 1_{s=u} = \sum_t \frac{\partial E}{\partial y_t} x_{t-u} = \left[\frac{\partial E}{\partial y} \star x\right]_{-u}
\end{aligned}$$

Task 3

a

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

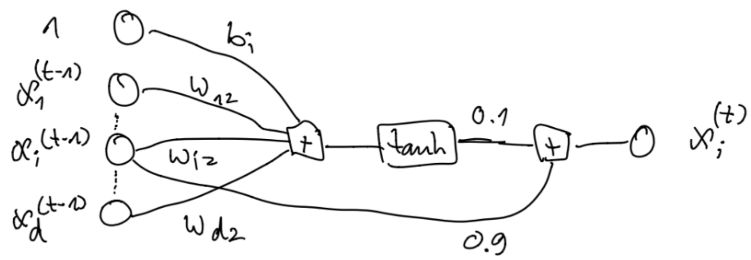
Eulers discretization of $y'(t) = f(t, y(t))$ is:

$$y^{(t)} = y^{(t-1)} + hf(t-1, y^{(t-1)})$$

We have $h = 1$:

$$x_j^{(t)} = x_j^{(t-1)} + 0.1(tanh(\sum_{i=1}^{d} x_i^{(t-1)} w_{ij} + b_j) - x_j^{(t-1)})$$
$$= 0.9x_j^{(t-1)} + 0.1(tanh(\sum_{i=1}^{d} x_i^{(t-1)} w_{ij} + b_j))$$

b



c

$$\frac{\partial x_j^{(t)}}{\partial x_j^{(t-1)}} = 0.9 + 0.1(1 - tanh^2(\sum_{i=1}^{d} x_i^{(t-1)} w_{ij} + b_j) w_{jj})$$
$$= 0.9 + 0.1 - 0.1tanh^2(\sum_{i=1}^{d} x_i^{(t-1)} w_{ij} + b_j) w_{jj}$$
$$= 1 - 0.1tanh^2(\sum_{i=1}^{d} x_i^{(t-1)} w_{ij} + b_j) w_{jj}$$

For $k \neq j$:

$$\frac{\partial x_k^{(t)}}{\partial x_j^{(t-1)}} = 0.1(1 - tanh^2(\sum_{i=1}^{d} x_i^{(t-1)} w_{ik} + b_k) w_{jk})$$
$$= 0.1 - 0.1w_{jk}tanh^2(\sum_{i=1}^{d} x_i^{(t-1)} w_{ik} + b_k)$$

d

$$\frac{\partial x_j^{(t)}}{\partial w_{kj}} = 0.1(1 - tanh^2(\sum_{i=1}^{d} x_i^{(t-1)} w_{ij} + b_j) x_k^{(t-1)})$$

$$= 0.1 - 0.1 x_k^{(t-1)} tanh^2(\sum_{i=1}^{d} x_i^{(t-1)} w_{ij} + b_j)$$

$$\frac{\partial x_j^{(t)}}{\partial w_{kl}} = 0 \ for \ l \neq j$$

$$\frac{\partial x_j^{(t)}}{\partial b_j} = 0.1 + 0.1 tanh^2(\sum_{i=1}^{d} x_i^{(t-1)} w_{ij} + b_j)$$

$$\frac{\partial x_j^{(t)}}{\partial b_k} = 0 \ for \ k \neq j$$