Ludwig-Maximilians-Universitaet Muenchen 13.05.2014
Institute for Informatics
Prof. Dr. Volker Tresp
Gregor Jossé
Johannes Niedermayer

Machine Learning and Data Mining

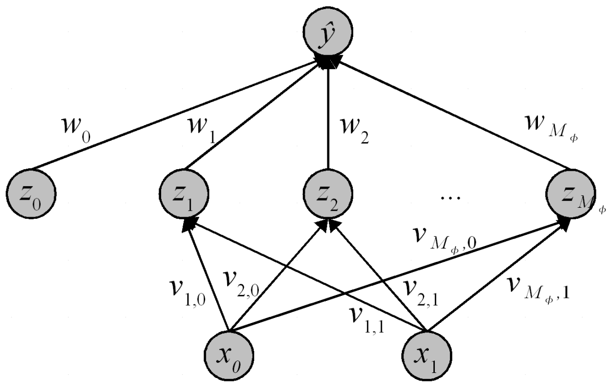
Summer 2015

Exercise Sheet 3

Presentation of Solutions to the Exercise Sheet on the 07.05.2014

Exercise 3-1 A simple Neural Network

The illustration below depicts, a two-layered neural network with inputs $x \in \mathbb{R}$ and for each input one bias $x_0 = z_0 = 1$ (i.e. $\mathbf{x}_i = (1, x_{i,1})^T$) in the input as well as the hidden layer.



As function of the hidden neurons we employ a sigmoid, i.e.

$$z_h = \phi(\mathbf{x}_i, \mathbf{v}_h) = \frac{1}{1 + \exp\left(-\sum_{j=0}^{M} v_{h,j} x_{i,j}\right)},$$

the output neuron $\hat{y}$ is, as usual, a linear combination.

a) Prove that the following holds: $\frac{\partial z_h}{\partial v_{h,j}} = x_{i,j} \cdot z_h \cdot (1 - z_h)$

b) Express the maximal value of $\hat{y}$ subject to $\mathbf{w}$, if all original weights are w_h $(h \in \{0, \dots, M_\phi\})$ positive. What's the minimal value?

c) If $v_{h,j} = 0$ for all $j \in \{0, \dots, M\}, h \in \{1, \dots, M_\phi\}$, then what is $\hat{y}$? Which functions describe $\hat{y}$ if all $v_{h,j} = c, c \neq 0$?

Exercise 3-2 An Introduction to Theano

Theano is a python library allowing the efficient evaluation of mathematical expressions on multi-dimensional arrays deeplearning.net/software/theano. Compared to numpy, it has two major advantages:

- Efficient differentiation, which is relevant for the backpropagation step in the context of artificial neural networks
- Hardware transparency: Code can be executed both on the CPU and on the GPU with minor modifications

In this exercise we aim at delving into the world of Theano. As a first step we will give a short introduction into the basic concepts of theano. Then we will exploit this knowledge to build a neural network.

(a) Install Theano deeplearning.net/software/theano/install.html#install

(b) Get a basic idea of what Theano can do deeplearning.net/software/theano/introduction.html

(c) When learning how to work with Theano, one needs to get used to the way how mathematical expressions are handled by this library. While python evaluates a mathematically expression as soon as it is executed, Theano builds an operator graph from the involved variables and operator. This graph is optimized and can then be employed to evaluated the underlying expression. Note that these steps take some time; later during execution of complex expressions, this cost will pay off.

  i. Generate two scalar variables x1 and x2. These variables will lated be filled with values.

  ii. From the two variables construct an expression e = x1**2+x1*x2+3. Print the representation of this expression

  iii. As a next step, we want to evaluate this function. Generate a theano function f that allows us to do so, it should use the variables x1 and x2 as inputs. Execute the function for x1=2 and x2=3. Print the function's representation.

  iv. Define two new variables x3 and x4, and redefine f using x3 and x4 as inputs. Hint: use the "givens" input parameter of theano.function.

  v. Another interesting property of theano functions is that parameters can be updated when executing a function. To test this define a shared variable (theano.shared), set its initial value to zero, and redefine the previous expression by replacing 3 with this shared variable. Generate a theano function from this expression that increases the output by 1 each time the function is called. Call this function several times and interpret the results. While this example is nonesense, the concept of updates allows for easily updating the weights and biases of a neural network with this technique: The theano function is defined on the network loss and updates are conducted depending on the gradient of the current input batch.

  vi. As we have already mentioned, theano also allows differentiation. Write an expression that represents the gradient of e with respect to x1, i.e. the partial derivation of e with respect to x1. Then generate a theano function that allows us to evaluate this expression. Evaluate it at x1=3, x2=1. Check the results by computing the gradient by hand.

  vii. Theano can compute several partial derivatives at the same time. Generate an expression g2 that computes the partial derivatives of e with respect to all free variables of e. How is g2 represented? Test this expression by defining an appropriate theano function.

(d) Broadcasting is an extension of matrix operations simplifying life in machine learning see

deeplearning.net/software/theano/tutorial/numpy.html#broadcasting and

docs.scipy.org/doc/numpy/user/basics.broadcasting.html

This technique is exessively used when working with artificial neural networks in the context of minibatches. In the context of artificial neural networks broadcasting is used to apply a mathematical expression on a set of input vectors, i.e. a minibatch. We will test this using numpy for the sake of simplicity. Compute A*B and B*A with A=[1,1],[2,2],[3,3],[4,4]] and B=[[2,3]]. Interpret the result.

(e) Compute A.dot(B.T). Interpret the results.

(f) Define a Perceptron with 2 inputs and one output using Theano. Use the data, labels and weights from exercise 1.4. Use the sigmoid function from theano as an activation function and set a learning rate of exercise 3. As a cost function use the squared Euclidean loss. Generate an expression for calculating the gradient of this cost function. Based on the gradient and the cost expressions, define a function receiving as input a matrix of feature vectors (a minibatch) and a label of vectors, calculating the cost of these inputs and updating the weights and biases of the neural network at the same time. Finally, train the neural network.