Exercises
Deep Learning
Fall 2018

Machine Learning Institute
Dept. of Computer Science, ETH Zürich
Fernando Pérez-Cruz
Web http://www.da.inf.ethz.ch/teaching/2018/DeepLearning/

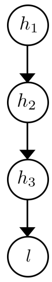
Series Monday, Oct 22, 2018
(Deep Learning, Exercise series 4)

Notations:
We adopt the following notations throughout this document which are slightly different from the lecture. For any function $y(x) : \mathbb{R}^m \to \mathbb{R}^n$, the Jacobian matrix $\boldsymbol{J}_y \in \mathbb{R}^{n \times m}$ is denoted as $\frac{\partial y}{\partial x}$. For n = 1, the same expression denotes the transposed gradient $\nabla_x^\top y = \nabla^\top y(x) \in \mathbb{R}^{1 \times m}$. Thus the chain rule for two functions $y(x) : \mathbb{R}^m \to \mathbb{R}^n$ and $z(y) : \mathbb{R}^n \to \mathbb{R}^p$ can always be written as a matrix multiplication: $\frac{\partial z}{\partial x} = \frac{\partial z}{\partial y}\frac{\partial y}{\partial x}$. This is the same as doing chain rule component wise: $\frac{\partial z_i}{\partial x_j} = \sum_{k=1}^n \frac{\partial z_i}{\partial y_k}\frac{\partial y_k}{\partial x_j}$ and putting all these values in a matrix $\frac{\partial z}{\partial x} := \left(\frac{\partial z_i}{\partial x_j}\right)_{i,j} \in \mathbb{R}^{p \times m}$.

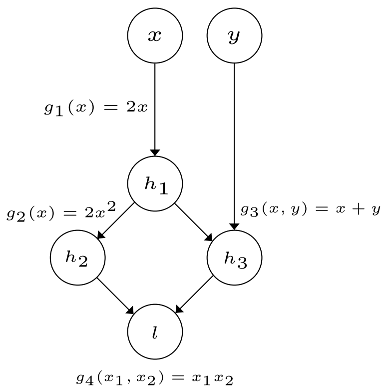
A different case we consider is one for which x is a matrix and y is a vector or a matrix. In this case, the gradient $\frac{\partial L}{\partial x}$ of a real-valued loss function L is also a matrix of the same shape as x, whereas the Jacobian $\frac{\partial y}{\partial x}$ is a 3 or 4 dimensional tensor. More about this in Problem 3 below.

Problem 1 (Backpropagation and Computational Graphs):

a) Most deep learning frameworks provide an automatic differentiation procedure to compute gradients based on the backpropagation algorithm introduced in the lecture. In these frameworks, all computations are represented as a graph and therefore also the gradient computation becomes a graph. Below you find a simple network. Use backpropagation to derive the gradient $\frac{\partial l}{\partial h_1}$ as a function of the intermediate (symbolic) gradients. Now add a node for each gradient that contributes to $\frac{\partial l}{\partial h_1}$ and connect them according to their dependencies.



b) Often you will see backpropagation applied to directed graphs that are trees. However, backpropagation can be applied to any directed acyclic graph (DAG). Below you see a simple DAG with two one-dimensional inputs $x, y \in \mathbb{R}$ and a final layer l. Derive $\frac{\partial l}{\partial x}$.



Problem 2 (Approximate Hessian for feed-forward networks):

In a general feed-forward network, each unit computes a weighted sum of its inputs of the form

$$a_j = \sum_i w_{ji} z_i, \tag{1}$$

where z_i is the activation of a unit, or input, that sends a connection to unit j, and w_{ji} is the weight associated with that connection. Let h be a nonlinear activation function, then $z_i = h(a_i)$.

In the lecture, we discussed how backpropagation can be used to obtain the first derivatives of a loss function. Here we discuss how one can evaluate the second derivatives of the loss which we denote as

$$\frac{\partial^2 L}{\partial w_{ji} \partial w_{lk}}. \tag{2}$$

Recall that the loss function decomposes over samples from the data set as $L(\cdot) = \sum_n L_n(\cdot)$.

1. Compute $\frac{\partial^2 L_n}{\partial w_{ji}^2}$ as a function of z_i^2.
2. Show that
$$\frac{\partial^2 L_n}{\partial a_j^2} = h'(a_j)^2 \sum_k \sum_{k'} w_{kj} w_{k'j} \frac{\partial^2 L_n}{\partial a_k \partial a_{k'}} + h''(a_j) \sum_k w_{kj} \frac{\partial L_n}{\partial a_k} \tag{3}$$
3. Assume we can neglect off-diagonal elements in the second-derivative terms. What expression do we get? What's the computational complexity compared to computing the exact Hessian matrix?

Problem 3 (Chain-rule and Jacobian matrices in more than 2 Dimensions):

We will analyze a generalization of Jacobian matrices and (matrix-form) chain rule for high dimensional (i.e. more than 2) objects. We analyze how to represent the gradient $\frac{\partial y}{\partial W}$ for matrix-vector functions $y(W) : \mathbb{R}^{m \times n} \to \mathbb{R}^p$ or matrix-matrix functions $y(W) : \mathbb{R}^{m \times n} \to \mathbb{R}^{p \times q}$. This operation is frequently used when dealing with neural networks, e.g. computing gradients of the weights W in a feed-forward layer of type $x \in \mathbb{R}^n \mapsto y := Wx + b \in \mathbb{R}^m$, where $W \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m$.

We will use the notion of a k-dimensional tensor $T \in \mathbb{R}^{d_1 \times d_2 \times \ldots \times d_k}$, which is a generalization of a 2-dimensional matrix. We use the multiplication $\times_{j_1,\ldots j_b}$ of two tensors $P \in \mathbb{R}^{d_1 \times \ldots \times d_a \times s_1 \times \ldots \times s_b}$ and $Q \in \mathbb{R}^{s_1 \times \ldots \times s_b \times t_1 \ldots \times t_c}$ which is a generalization of the matrix product case :

$$P \times_{j_1,\ldots j_b} Q := T \in \mathbb{R}^{d_1 \times \ldots \times d_a \times t_1 \ldots \times t_c}$$

$$T_{i_1,\ldots i_a,k_1,\ldots,k_c} := \sum_{j_1,\ldots j_b} P_{i_1,\ldots i_a,j_1,\ldots j_b} Q_{j_1,\ldots j_b,k_1,\ldots,k_c}$$

1. Let $y(W) : \mathbb{R}^{d_1 \times d_2} \to \mathbb{R}^{d_3}$ and $L(y) : \mathbb{R}^{d_3} \to \mathbb{R}$. The gradient $\frac{\partial y}{\partial W}$ is then a 3-dimensional tensor $T \in \mathbb{R}^{d_3 \times d_1 \times d_2}$ such that $T_{i,j,k} = \frac{\partial y_i}{\partial W_{jk}}$. Show that, in this case, the chain rule can be compactly written as the following tensor multiplication:
$$\frac{\partial L}{\partial W} = \frac{\partial L}{\partial y} \times_{d_3} \frac{\partial y}{\partial W}$$
2. Use the above to compute $\frac{\partial L}{\partial W}$ for the loss function $L = \|y\|_2^2, \quad y = Wx$, where $x \in \mathbb{R}^{d_2}$ and $W \in \mathbb{R}^{d_1 \times d_2}$.
3. Let $y(W) : \mathbb{R}^{d_1 \times d_2} \to \mathbb{R}^{d_3 \times d_4}$ and $L(y) : \mathbb{R}^{d_3 \times d_4} \to \mathbb{R}$. The gradient $\frac{\partial y}{\partial W}$ is then a 4-dimensional tensor $T \in \mathbb{R}^{d_3 \times d_4 \times d_1 \times d_2}$ such that $T_{i,j,k,l} = \frac{\partial y_{i,j}}{\partial W_{kl}}$. Show that, in this case, the chain rule can be compactly written as the following tensor multiplication:
$$\frac{\partial L}{\partial W} = \frac{\partial L}{\partial y} \times_{d_3,d_4} \frac{\partial y}{\partial W}$$
4. Use the previous result to compute $\frac{\partial tr(BA)}{\partial A}$, where $A \in \mathbb{R}^{d_1 \times d_2}$ and $B \in \mathbb{R}^{d_2 \times d_1}$.