Machine Learning

Exercise Sheet 8

Winter Term 2023/2024 — Available: 12.01.2024
Prof. Dr. Niels Landwehr — Hand in until: 19.01.2022 11:59am
Dr. Ujjwal — Exercise sessions: 22.01.2024/24.01.2024

Task 1 – Derivative of activation functions [10 points]

Derive a compact expression for first-order derivatives $\frac{\partial}{\partial x}\sigma_i(x)$ for the following activation functions $\sigma_i : \mathbb{R} \to \mathbb{R}$:

- $\sigma_1(x) = \frac{1}{1+\exp^{-x}}$
- $\sigma_2(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$

Specifically, express the derivative in terms of the original function, that is, in the form $\frac{\partial}{\partial x}\sigma_i(x) = h_i(\sigma_i(x))$.

Task 2 – Backpropagation for multilayer perceptron [15 points]

Consider the following simple multilayer perceptron. It has an input $x \in \mathbb{R}$ that gets mapped to an intermediate representation $a_1 = \sigma(w_1 x + b_1)$ where $w_1, b_1 \in \mathbb{R}$ are network weights and $\sigma(z) = \frac{\exp^z - \exp^{-z}}{\exp^z + \exp^{-z}}$ is the nonlinear activation function. That intermediate representation then gets mapped to an output $y = \sigma(w_2 a_1 + b_2)$. Let the overall function be denoted by $f_{\boldsymbol{\theta}}$, that is, $f_{\boldsymbol{\theta}}(x) = y$, and let the parameter vector be called $\boldsymbol{\theta} = (w_1, b_1, w_2, b_2)^\mathsf{T} \in \mathbb{R}^4$.
Let the initial weight and bias values be as follows : $w_1 = -1$, $b_1 = 1$, $w_2 = -2$ and $b_2 = 1$.
Assume a single training example $(x_1, y_1) = (-1, 1)$ and a squared loss function given by $L(\boldsymbol{\theta}) = (f_{\boldsymbol{\theta}}(x_1) - y_1)^2$.
Write down the compute graph for the function $L(\boldsymbol{\theta})$ and perform backpropagation in this graph to compute the gradient $\frac{\partial}{\partial \boldsymbol{\theta}} L(\boldsymbol{\theta})$ at the single example. Perform one update step of gradient descent, assuming a learning rate $\eta = 1$.
Hint: use the result from Task 1 and write the activation function as a single node in the compute graph.

Task 3 (Programming) – Expressivity of feedforward networks [15 points]
In this exercise, you will familiarize yourself with the deep learning API Keras, which is part of the Tensorflow deep learning toolkit. As an illustrative example, consider the function $f(x) = 1 + sin(\frac{i\pi}{4}\, x)$. In the notebook *keras.ipynb*, you find a method to generate and plot samples from this function. You will also find a method that builds a multilayer perceptron with a scalar input, single hidden layer with three neurons, and scalar output that can be used to approximate this function, and a method for learning the function from sample data. Using the provided functions, perform the following experiments:

- a) For $i = 1, 2, 4, 6, 8$, run the training and plot the predicted and desired values. Also show the average error you get in each case. Based on this, plot a curve that shows the error of the fitted model (on the training data) as a function of i.

- b) For $i = 6$, train several models in which you vary the number of hidden layer neurons from 2 to 8 in steps of 1. Based on this, plot a curve that shows the error of the fitted model (on the training data) as a function of the number of hidden layer neurons.

Hint: Colab (`https://colab.research.google.com`) is a free and reasonably performant environment for running notebooks with GPU support.