Machine Learning: Homework #9

Due on January 8, 2018 at 09:59am

Professor Dr. Stephan Guennemann

Marc Meissner - 03691673

Problem 1

Why do we use basis functions in neural networks? What purpose do they serve?

Solution
Usually, features can not be separated linearly, which makes it hard for logistic regression (output layers) to find a model for the data. Basis functions are nonlinear functions that transform the features into a different space. The goal is to find transformations that allow for a linear separation of the class labels.

Problem 2

Consider a neural network with hidden-unit nonlinear sigmoid activation functions. Show that there exists an equivalent network, which computes exactly the same function, but with hidden unit activation functions given by $\tanh(x)$.

Solution
The tanh - function can be transformed into a scaled sigmoid function:

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} = \frac{2e^x - (e^{-x} + e^x)}{e^x + e^{-x}} = \frac{2e^x}{e^x + e^{-x}} \frac{-(e^{-x} + e^x)}{e^x + e^{-x}} = \frac{2}{1 + e^{-2x}} - 1 = 2\sigma(2x) - 1$$

Thus:

$$\sigma(x) = \frac{\tanh(\frac{x}{2}) + 1}{2}$$

Therefore, one can replace all sigmoids with the tanh counterparts without changing the function.

Problem 3

We already know that the derivative of the sigmoid activation function can be expressed in terms of the function value itself. Show that the derivative of the tanh activation function can also be expressed in terms of the function value itself. Why is this a useful property?

Solution
The quotient rule yields:

$$\tanh'(x) = \left(\frac{e^x - e^{-x}}{e^x + e^{-x}}\right)' = \frac{(e^x + e^{-x})^2 - (e^x - e^{-x})^2}{(e^x + e^{-x})^2} = 1 - \tanh^2(x)$$

Problem 3 continued on next page...

This is useful, as it simplifies the calculation of the gradient, which is needed for solving backpropagation in neural networks.

Problem 4

The error function for binary classification problems is derived for a network having a logistic sigmoid activation function so that $0 \leq f(x_i, W) \leq 1$, and data have target values $y_i \in \{0, 1\}$. Derive the corresponding error function if we consider a network having an output $1 \leq f(x_i, W) \leq 1$ and target values $y_i \in \{-1, 1\}$. What would be the appropriate choice of activation function in this case?

Solution

We can scale f and y so that they lie in $\{-1, 1\}$ again:

$$g(x) = \frac{1}{2}(x + 1)$$

Problem 5

A simple neural network has as loss function

$$E(w) := \frac{1}{m} \sum_{i=1}^{m} l(y_i - wx_i) + \frac{\lambda}{2}||w||^2$$

Compute the gradient of $E(w)$ w.r.t. w, when optimising over all data.

Solution

Calculating the gradients separately for each summand yields:

$$\frac{\partial E(w)}{\partial w} = \frac{1}{m} \sum_{i=1}^{m} \frac{\partial f_i}{\partial w} + \lambda w$$

and

$$\frac{\partial f_i}{\partial w} = \begin{cases} -(y_i - wx_i)x_i & \text{for } |y_i - wx_i| < 1 \\ -\text{sgn}(y_i - wx_i)x_i - \frac{1}{2} & \text{else} \end{cases}$$

with $\text{sgn}(x)$ being the signum function.

Problem 6

When do you "stop training" (give the approximate number of the update/iteration) and use the corresponding neural network weights? Explain your answer. Relate your answer to the given figure in which the data is separated in a training set, a validation set, and a test set.

Solution

You stop training, when your validation error does not shrink anymore, but rather starts to rise again (global minimum of validation error). In the figure, this is the case after approximately 50 updates. You are not allowed to use the test data for anything else but reporting the final performance.

Problem 7

In machine learning you quite often come across problems which contain the following quantity

$$y = \log \sum_{i=1}^{N} e^{x_i}$$

For example if we want to calculate the log-likelihood of neural network with a softmax output we get this quantity due to the normalization constant. If you try to calculate it naively, you will quickly encounter underflows or overflows, depending on the scale of x_i . Despite working in log-space, the limited precision of computers is not enough and the result will be ∞ or $-\infty$.
To combat this issue we typically use the following identity:

$$y = \log \sum_{i=1}^{N} e^{x_i} = a + \log \sum_{i=1}^{N} e^{x_i - a}$$

for an arbitrary a. This means, you can shift the center of the exponential sum. A typical value is setting a to the maximum ($a = \max_i x_i$), which forces the greatest value to be zero and even if the other values would underflow, you get a reasonable result. Your task is to show that the identity holds.

Solution

$$a + \log \sum_{i=1}^{N} e^{x_i - a} = a + \log \left(e^{-a} \sum_{i=1}^{N} e^{x_i} \right) = a + \log(e^{-a}) + \log \sum_{i=1}^{N} e^{x_i} = \log \sum_{i=1}^{N} e^{x_i}$$

Problem 8

Similar to the previous exercise we can compute the output of the softmax function $\pi_i = \frac{e^{x_i}}{\sum_{i=1}^{N} e^{x_i}}$ in a numerically stable way by introducing an arbitrary a:

$$\frac{e^{x_i}}{\sum_{i=1}^{N} e^{x_i}} = \frac{e^{x_i - a}}{\sum_{i=1}^{N} e^{x_i - a}}$$

Problem 8 continued on next page...

often chosen $a = \max_i x_i$. Show that the above identity holds.

Solution

$$\frac{e^{x_i - a}}{\sum_{i=1}^{N} e^{x_i - a}} = \frac{e^{x_i} e^{-a}}{\sum_{i=1}^{N} e^{x_i} e^{-a}} = \frac{e^{x_i} e^{-a}}{e^{-a} \sum_{i=1}^{N} e^{x_i}} = \frac{e^{x_i}}{\sum_{i=1}^{N} e^{x_i}}$$

Problem 9

Let the logits (before applying sigmoid) of a single training example be x and the corresponding label be y. The sigmoid logistic loss (i.e. binary cross-entropy) for this example is then:

$$-(y\log(\sigma(x)) + (1-y)log(1-\sigma(x)))$$

To ensure stability and avoid overflow, usually implementations use this equivalent formulation:

$$\max(x, 0) - xy + log(1 + e^{-\text{abs}(x)})$$

Your task is to show that the equivalence holds.

Solution

$$\begin{aligned}
&-(y\log(\sigma(x)) + (1-y)log(1-\sigma(x))) \\
&= -\left(y\log\left(\frac{1}{1+e^{-x}}\right) + (1-y)log\left(1 - \frac{1}{1+e^{-x}}\right)\right) \\
&= y\log\left(1+e^{-x}\right) - (1-y)log\left(\frac{1+e^{-x}-1}{1+e^{-x}}\right) \\
&= y\log\left(1+e^{-x}\right) - (1-y)\left(-x - \log\left(1+e^{-x}\right)\right) \\
&= y\log\left(1+e^{-x}\right) + x + \log\left(1+e^{-x}\right) - yx - y\log\left(1+e^{-x}\right) \\
&= x - yx + \log\left(1+e^{-x}\right)
\end{aligned}$$

Observing the two cases $x > 0$ and $x < 0$, this can be written as:

$$\begin{aligned}
&= x - yx + \log\left(1+e^{-x}\right) \\
&= \max(x, 0) - xy + log(1 + e^{-\text{abs}(x)})
\end{aligned}$$