Simon Wachter: Assignment 1

siwachte@ethz.ch, 19-920-198

04/12/2022 - 11:36h

Question 1:

a) Show that :

$$\nabla^2 f(x) \stackrel{!}{=} \begin{bmatrix} (\nabla(e_1^T \nabla f(x))) \\ \vdots \\ (\nabla(e_n^T \nabla f(x))) \end{bmatrix}$$

We have:

$$\nabla f(x) = \begin{bmatrix} \frac{\partial f}{\partial x_0} \\ \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix} \tag{1}$$

(2)

$$Hess(f) = \nabla^2 f(x) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \cdots & \frac{\partial^2 f}{\partial x_1 x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n x_1} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$

Each row of the hessian can be derived as follows:

$$\left[\frac{\partial^2 f}{\partial x_i x_1}, \frac{\partial^2 f}{\partial x_i x_2}, \ldots, \frac{\partial^2 f}{\partial x_i x_n}\right] = \left(\nabla\left(\frac{\partial f}{\partial x_i}\right)\right)^T \tag{3}$$

$$= \left(\nabla\left(e_i^T \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_i} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix}\right)\right)^T \tag{4}$$

$$= \left(\nabla(e_i^T \nabla f(x))\right)^T \tag{5}$$

b) In a) we have seen that by calculating $(\nabla(e_i^T \nabla f(x)))$ for all e_i for $i \in \{1, \ldots, n\}$ we can build the whole hessian. We can use the backpropagation algorithm from the lecture to calculate the gradient of f. This whole calculation can also be modeled as a computation graph G. The graph then is a combination of two graphs, first the computation graph of f and second a mirrored graph of f, where each node represents the partial derivative of f with respect to the variable represented by this node. In G there are some dependencies between the mirrored and non-mirrored graph, where $\frac{\partial f}{\partial z_i}$ depends on z_i. From G, which resembles the calculation of $\nabla f(x)$, it is easy to slightly change the calculation to $e_i^T \nabla f(x)$ and call it G_i. We can then apply backpropagation overall G_i to calculate the Hessian. For the naive approach, we would have to calculate all pairs of second derivatives separately. This can be done by backpropagation to obtain each entry of the matrix individually. There are n^2 entries in the hessian, hence the runtime of the naive algorithm would be $\mathcal{O}(n^2 m)$. The m term comes from the backpropagation because each second derivative can be calculated in $\mathcal{O}(m)$, because we know that the calculation of the derivative of f can be done in $\mathcal{O}(m)$ and because the computation graph for G is at most 2 times as big, we can calculate the second derivative in $\mathcal{O}(m)$.

In our hessian calculation, we need to do backpropagation over all G_i and each G_i can be calculated in at most $\mathcal{O}(m)$. Hence, we can calculate the hessian in $\mathcal{O}(nm)$.

c) The same approach as in b) can be used for any k-th order derivative. For the 3rd-order derivatives, we would combine G_i with a mirrored version of G_i and do backpropagation over this combined graph. The runtime is $\mathcal{O}(n^2 m)$. To generalize this approach to the k-th order derivative, we would always combine the graph of the $(k-1)$-th order derivatives with a mirrored version of themselves and do backpropagation over these combined graphs. The runtime would be:

$$\mathcal{O}\left(\sum_{i=1}^{k} n^{i-1} \cdot m\right) = \mathcal{O}(n^{k-1} \cdot m)$$

Then we sum over all derivatives up until the k-th. The n^{i-} term denotes the number of times that we need to do backpropagation.

Question 2:

Link to colab here