Backpropagation

Dr. Alireza Aghamohammadi

April 28, 2024

Problem Statement

- Given a deep neural network with K hidden layers, the model parameters ϕ include biases β_i and weights Ω_i.
- Our objective is to compute the gradient of the loss function L with respect to these parameters, denoted as $\frac{\partial L}{\partial \phi}$.
- This gradient is essential for optimizing the network using gradient-based algorithms like stochastic gradient descent (SGD).

$$
\begin{aligned}
h_1 &= \texttt{ReLU}(\beta_0 + \Omega_0 \cdot x) \\
h_2 &= \texttt{ReLU}(\beta_1 + \Omega_1 \cdot h_1) \\
h_3 &= \texttt{ReLU}(\beta_2 + \Omega_2 \cdot h_2) \\
&\vdots \\
h_K &= \texttt{ReLU}(\beta_{K-1} + \Omega_{K-1} \cdot h_{K-1}) \\
\hat{y} &= \beta_K + \Omega_K \cdot h_K
\end{aligned}
$$

Calculus Chain Rule Recap

Chain Rule:

The chain rule allows us to find the derivative of a composite function. For a function $f(g(x))$, we have:

$$\frac{\mathrm{d}}{\mathrm{d}x}f(g(x)) = \frac{\mathrm{d}f}{\mathrm{d}g} \cdot \frac{\mathrm{d}g}{\mathrm{d}x}$$

Multivariate Chain Rule:

The multivariate chain rule extends this concept to functions with multiple variables. For a function $f(g_1(x), g_2(x), ..., g_M(x))$, we get:

$$\frac{\mathrm{d}}{\mathrm{d}x}f(g_1(x), g_2(x), ..., g_M(x)) = \sum_{i=1}^{M} \frac{\partial f}{\partial g_i} \cdot \frac{\mathrm{d}g_i}{\mathrm{d}x}$$

Backpropagation Algorithm: Toy Example

Forward Pass:

(1) $f_0 = \beta_0 + \Omega_0 \cdot x$

(2) $h_1 = \texttt{ReLU}(f_0)$

(3) $L = -y \log h_1 - (1-y)\log(1-h_1)$

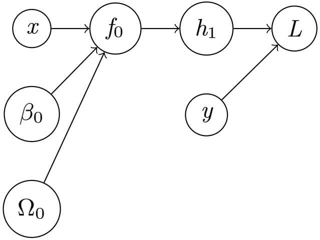
Backward Pass:

(3) $\frac{\partial L}{\partial h_1} = \frac{h_1 - y}{h_1(1-h_1)}$

(2) $\frac{\partial L}{\partial f_0} = \frac{\partial L}{\partial h_1}\frac{\partial h_1}{\partial f_0} = \mathbb{I}[f_0 > 0] \odot \frac{h_1 - y}{h_1(1-h_1)}$

(1) $\frac{\partial L}{\partial \beta_0} = \frac{\partial L}{\partial f_0}\frac{\partial f_0}{\partial \beta_0} = \frac{\partial L}{\partial f_0}$

(1) $\frac{\partial L}{\partial \Omega_0} = \frac{\partial L}{\partial f_0}\frac{\partial f_0}{\partial \Omega_0} = \frac{\partial L}{\partial f_0} \cdot x^T$



Backpropagation Algorithm

To compute $\frac{\partial L}{\partial \beta_i}$ when $f_i = \beta_i + \Omega_i \cdot h_i$

$$
\begin{aligned}
\frac{\partial L}{\partial \beta_i} &= \frac{\partial L}{\partial f_i}\frac{\partial f_i}{\partial \beta_i} \\
&= \frac{\partial}{\partial \beta_i}(\beta_i + \Omega_i \cdot h_i)\frac{\partial L}{\partial f_i} \\
&= \frac{\partial L}{\partial f_i}
\end{aligned}
$$

To compute $\frac{\partial L}{\partial \Omega_i}$ when $f_i = \beta_i + \Omega_i \cdot h_i$

$$
\begin{aligned}
\frac{\partial L}{\partial \Omega_i} &= \frac{\partial L}{\partial f_i}\frac{\partial f_i}{\partial \Omega_i} \\
&= \frac{\partial}{\partial \Omega_i}(\beta_i + \Omega_i \cdot h_i)\frac{\partial L}{\partial f_i} \\
&= \frac{\partial L}{\partial f_i} h_i^T
\end{aligned}
$$

Backpropagation Algorithm

To compute $\frac{\partial L}{\partial f_i}$ given $h_i = \texttt{ReLU}(f_{i-1})$ and $f_i = \beta_i + \Omega_i \cdot h_i$

$$\begin{aligned}
\frac{\partial L}{\partial f_{i-1}} &= \frac{\partial L}{\partial h_i}\frac{\partial h_i}{\partial f_{i-1}} \\
&= (\frac{\partial L}{\partial f_i}\frac{\partial f_i}{\partial h_i})\frac{\partial h_i}{\partial f_{i-1}} \\
&= \mathbb{I}[f_{i-1} > 0] \odot (\Omega_i^T \frac{\partial L}{\partial f_i})
\end{aligned}$$