Backpropagation Derivation for a Two-Layer Neural Network

Tong Wu

October 17, 2024

Introduction

We consider a two-layer neural network with the following structure:

- Input layer of size n_0
- Hidden layer of size n_1
- Output layer of size n_2

The goal is to compute the gradients of the loss L with respect to the weight matrices and bias vectors using backpropagation.

Forward Propagation

Let the input be $\mathbf{x} \in \mathbb{R}^{n_0}$. The forward propagation equations are:

$$\begin{aligned}
\mathbf{z}^{[1]} &= \mathbf{W}^{[1]}\mathbf{x} + \mathbf{b}^{[1]}, \\
\mathbf{a}^{[1]} &= \sigma(\mathbf{z}^{[1]}), \\
\mathbf{z}^{[2]} &= \mathbf{W}^{[2]}\mathbf{a}^{[1]} + \mathbf{b}^{[2]}, \\
\hat{\mathbf{y}} &= f(\mathbf{z}^{[2]}),
\end{aligned}$$

Here:

- $\mathbf{W}^{[1]} \in \mathbb{R}^{n_1 \times n_0}$ is the weight matrix for the hidden layer.
- $\mathbf{b}^{[1]} \in \mathbb{R}^{n_1}$ is the bias vector for the hidden layer.
- $\sigma(\cdot)$ is the activation function (e.g., sigmoid or ReLU) applied element-wise.
- $\mathbf{W}^{[2]} \in \mathbb{R}^{n_2 \times n_1}$ is the weight matrix for the output layer.
- $\mathbf{b}^{[2]} \in \mathbb{R}^{n_2}$ is the bias vector for the output layer.

- $f(\cdot)$ is the activation function of the output layer **(We take element-wise function e.g. sigmoid as an example)**.
- $\hat{\mathbf{y}} \in \mathbb{R}^{n_2}$ is the predicted output.

Loss Function

Assume we have a loss function $L(\hat{\mathbf{y}}, \mathbf{y})$ where $\mathbf{y}$ is the true label. For simplicity, we assume the loss is the mean squared error (MSE):

$$L = \frac{1}{2}\|\mathbf{y} - \hat{\mathbf{y}}\|^2,$$

Backpropagation

To update the weights, we need to calculate the gradients of the loss L with respect to the weight matrices $\mathbf{W}^{[1]}, \mathbf{W}^{[2]}$ and the biases $\mathbf{b}^{[1]}, \mathbf{b}^{[2]}$.

Step 1: Gradient at the Output Layer

First, compute the derivative of the loss with respect to the prediction $\hat{\mathbf{y}}$:

$$\frac{\partial L}{\partial \hat{\mathbf{y}}} = \hat{\mathbf{y}} - \mathbf{y},$$

Then compute the derivative of the loss with respect to the output layer pre-activation $\mathbf{z}^{[2]}$:

$$\frac{\partial L}{\partial \mathbf{z}^{[2]}} = \frac{\partial L}{\partial \hat{\mathbf{y}}} \frac{\partial \hat{\mathbf{y}}}{\partial \mathbf{z}^{[2]}} \overset{\textbf{why?}}{=} \frac{\partial L}{\partial \hat{\mathbf{y}}} \circ f'(\mathbf{z}^{[2]}) = (\hat{\mathbf{y}} - \mathbf{y}) \circ f'(\mathbf{z}^{[2]}),$$

where $\circ$ denotes element-wise multiplication and $f'(\mathbf{z}^{[2]})$ is the derivative of the output activation function. For example, if $f(\cdot)$ is the identity function (regression case), then $f'(\mathbf{z}^{[2]}) = 1$.

Now using chain rule, the gradient with respect to the weights and biases in the output layer are:

$$\frac{\partial L}{\partial \mathbf{W}^{[2]}} = \frac{\partial L}{\partial \mathbf{z}^{[2]}} \frac{\partial \mathbf{z}^{[2]}}{\partial \mathbf{W}^{[2]}} \overset{\textbf{why?}}{=} \frac{\partial L}{\partial \mathbf{z}^{[2]}} (\mathbf{a}^{[1]})^T,$$

$$\frac{\partial L}{\partial \mathbf{b}^{[2]}} = \frac{\partial L}{\partial \mathbf{z}^{[2]}} \frac{\partial \mathbf{z}^{[2]}}{\partial \mathbf{b}^{[2]}} = \frac{\partial L}{\partial \mathbf{z}^{[2]}},$$

Step 2: Gradient at the Hidden Layer

The derivative at the hidden layer is computed by backpropagating similarly. First we calculate the derivative of first activation $\mathbf{a}^{[1]}$:

$$\frac{\partial L}{\partial \mathbf{a}^{[1]}} = \frac{\partial L}{\partial \mathbf{z}^{[2]}} \frac{\partial \mathbf{z}^{[2]}}{\partial \mathbf{a}^{[1]}} \stackrel{\textbf{why?}}{=} (\mathbf{W}^{[2]})^T \frac{\partial L}{\partial \mathbf{z}^{[2]}},$$

Then we can calculate pre-activation easily:

$$\frac{\partial L}{\partial \mathbf{z}^{[1]}} = \frac{\partial L}{\partial \mathbf{a}^{[1]}} \frac{\partial \mathbf{a}^{[1]}}{\partial \mathbf{z}^{[1]}} = (\mathbf{W}^{[2]})^T \delta^{[2]} \circ \sigma'(\mathbf{z}^{[1]}),$$

where $\sigma'(\mathbf{z}^{[1]})$ is the derivative of the activation function σ with respect to $\mathbf{z}^{[1]}$.

Now, the gradients with respect to the weights and biases in the hidden layer are:

$$\frac{\partial L}{\partial \mathbf{W}^{[1]}} = \frac{\partial L}{\partial \mathbf{z}^{[1]}} (\mathbf{x})^T,$$

$$\frac{\partial L}{\partial \mathbf{b}^{[1]}} = \frac{\partial L}{\partial \mathbf{z}^{[1]}},$$

Summary of Gradients

The gradients for backpropagation are:

$$\frac{\partial L}{\partial \mathbf{W}^{[2]}} = \left[(\hat{\mathbf{y}} - \mathbf{y}) \circ f'(\mathbf{z}^{[2]})\right] (\mathbf{a}^{[1]})^T,$$

$$\frac{\partial L}{\partial \mathbf{b}^{[2]}} = (\hat{\mathbf{y}} - \mathbf{y}) \circ f'(\mathbf{z}^{[2]}),$$

$$\frac{\partial L}{\partial \mathbf{W}^{[1]}} = \left[(\mathbf{W}^{[2]})^T \left((\hat{\mathbf{y}} - \mathbf{y}) \circ f'(\mathbf{z}^{[2]})\right) \circ \sigma'(\mathbf{z}^{[1]})\right] (\mathbf{x})^T,$$

$$\frac{\partial L}{\partial \mathbf{b}^{[1]}} = (\mathbf{W}^{[2]})^T \left((\hat{\mathbf{y}} - \mathbf{y}) \circ f'(\mathbf{z}^{[2]})\right) \circ \sigma'(\mathbf{z}^{[1]}),$$

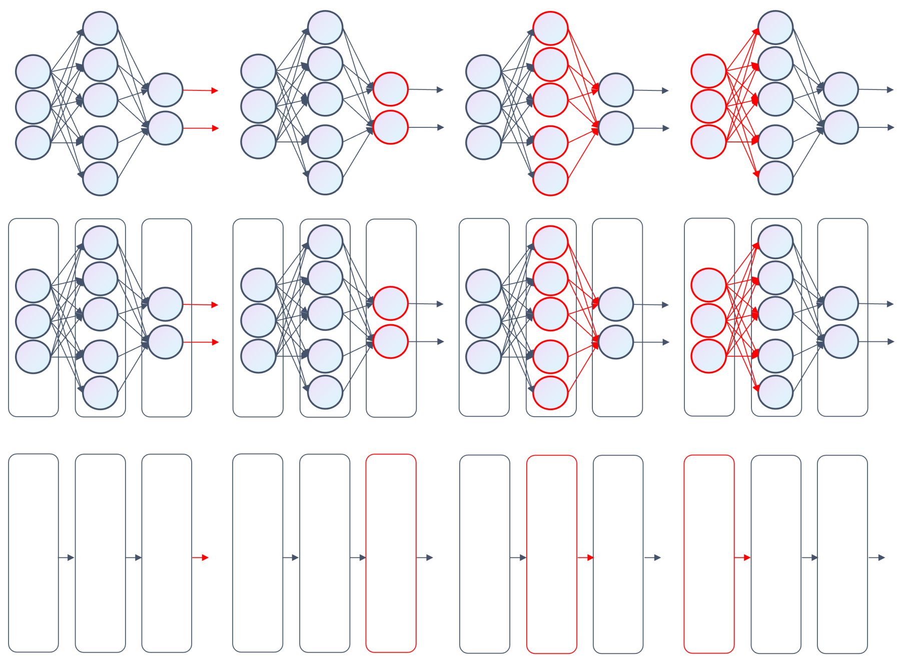

Figure 1: **Visualization of Backpropogation: From Scalars to Linear Algebra.**