Computing $\nabla_x a^{[L]}(x)$ in a Feedforward Fully-Connected Neural Network

Ou Chia Yu
Department of Information Management and Finance, NYCU

September 15, 2025

Abstract

This report explains how to compute the gradient of the output of a feedforward fully-connected neural network with respect to its input, i.e., $\nabla_x a^{[L]}(x)$. We describe the assumptions, the backpropagation algorithm used, and illustrate with a simple example.

1 Introduction

In this report, we focus on a feedforward fully-connected neural network, also known as a multi-layer perceptron (MLP). Given an input $x \in \mathbb{R}^{n_1}$, the output of the network is defined layer by layer as:

$$a^{[1]} = x, \quad a^{[l]} = \sigma(W^{[l]} a^{[l-1]} + b^{[l]}), \quad l = 2, \ldots, L,$$

where $\sigma(\cdot)$ is an activation function. We assume that the final layer has only one neuron ($n_L = 1$), so the network output is a scalar $a^{[L]}(x) \in \mathbb{R}$. The goal is to compute the gradient

$$\nabla_x a^{[L]}(x) \in \mathbb{R}^{n_1}.$$

2 Model Assumptions

We assume the following:

- The activation function σ is differentiable, e.g., sigmoid or ReLU.
- The parameters $(W^{[l]}, b^{[l]})$ are fixed during computation.
- The output is scalar, i.e., $n_L = 1$.

3 Algorithm: Backpropagation to Input

The computation of $\nabla_x a^{[L]}(x)$ follows the chain rule and can be efficiently done using the backpropagation algorithm:

Algorithm 1 Compute $\nabla_x a^{[L]}(x)$

1: Perform a forward pass to compute $z^{[l]} = W^{[l]}a^{[l-1]} + b^{[l]}$ and $a^{[l]} = \sigma(z^{[l]})$ for all layers.
2: Initialize $\delta^{[L]} = 1$ since $a^{[L]}$ is a scalar.
3: **for** $l = L, L-1, \ldots, 2$ **do**
4: $\quad \delta^{[l-1]} = (W^{[l]})^T \left(\sigma'(z^{[l]}) \odot \delta^{[l]}\right)$
5: **end for**
6: Output: $\nabla_x a^{[L]}(x) = \delta^{[1]}$

4 Example

Consider a 2-layer network with input dimension $n_1 = 3$, one hidden layer of size 2, and scalar output. The forward pass is:

$$a^{[2]} = \sigma(W^{[2]}x + b^{[2]}), \quad a^{[3]} = \sigma(W^{[3]}a^{[2]} + b^{[3]}).$$

The gradient with respect to the input x is obtained by applying Algorithm 1, resulting in $\nabla_x a^{[3]}(x) \in \mathbb{R}^3$.

5 Applications

The gradient $\nabla_x a^{[L]}(x)$ is useful for:

- **Feature sensitivity analysis**: identifying which input features most affect the output.
- **Adversarial examples**: finding perturbations of the input that significantly change the output.
- **Mathematical understanding**: interpreting neural networks as composite functions and analyzing their derivatives.

6 Conclusion

We used backpropagation to compute the gradient of the network output with respect to its input. This method generalizes the standard training backpropagation but focuses on the input rather than the parameters. The result provides insights into feature importance and adversarial robustness.