Backpropagation

John Purcell

June 7, 2021

1 Introduction

Back propagation adjusts the weights and biases of neural networks by repeatedly subtracting small quantities from the weights and biases that are proportional to the rates of change of the network loss with respect to those weights and biases.

Here we demonstrate how to find these rates of change for multilayer neural networks.

2 The Error: Definition and Use

We define a quantity δ^L, referred to as the *error* for layer $\mathbf{L}$. This is the vector consising of the rates of change of the loss with respect to the weighted sums of the neurons in layer $\mathbf{L}$.

The ith element of δ^L is then given by

$$\delta_i^L = \frac{\partial \lambda}{\partial z_i^L} \tag{1}$$

where z_i^L is the weighted sum of the ith neuron in layer $\mathbf{L}$ and λ is the network loss.

We will first demonstrate that, given the error for a layer, it is easy to find the rates of change of the weights and biases for that layer.

Consider the rate of change of the jth weight of the ith neuron in layer $\mathbf{L}$ with respect to the loss λ.

Applying the *chain rule* of calculus:

$$\frac{\partial \lambda}{\partial w_{ij}^L} = \frac{\partial \lambda}{\partial z_i^L} \frac{\partial z_i^L}{\partial w_{ij}^L} \tag{2}$$

$\frac{\partial \lambda}{\partial z_i^L}$ is the layer error of the ith neuron in layer **L**, by definition, i.e. δ_i^L

$\frac{\partial z_i^L}{\partial w_{ij}^L}$ is the rate of change of the weighted sum of the ith neuron with respect to the neuron's jth weight. This is simply whatever multiplies the jth weight, and that happens to be the activation (output) of the jth neuron in the previous layer, a_j^{L-1}

Therefore:

$$\frac{\partial \lambda}{\partial w_{ij}^L} = \frac{\partial \lambda}{\partial z_i^L} \frac{\partial z_i^L}{\partial w_{ij}^L} = \delta_i^L a_j^{L-1} \tag{3}$$

Given the error vector of a layer, it is therefore easy to calculate the rates of change of the network's loss with respect to each of the weights of the layer's neurons.

In the case of the biases of each of the neurons, these behave like weights, if we regard them as being multiplied by an input from the previous layer of 1.

It follows that:

$$\frac{\partial \lambda}{\partial b_i^L} = \frac{\partial \lambda}{\partial z_i^L} \frac{\partial z_i^L}{\partial b_i^L} = \delta_i^L .1 = \delta_i^L \tag{4}$$

where b_i^L is the bias of the ith neuron in layer **L**.

The elements of the error vector directly give us the rates of change of the loss with respect to the layer biases.

3 Calculating the Previous Layer's Error

We will now determine how, given the error of the **L**th layer, we can find the error of the **L-1**th layer.

Applying the chain rule again:

$$\delta_i^{L-1} = \frac{\partial \lambda}{\partial z_i^{L-1}} = \sum_j \frac{\partial \lambda}{\partial z_j^L} \frac{\partial z_j^L}{\partial z_i^{L-1}} = \sum_j \delta_j^L \frac{\partial z_j^L}{\partial z_i^{L-1}} \tag{5}$$

Where $\sum_i$ indicates summation and i ranges over all possible values for the neurons in layer **L**.

Between the z^{L-1} values and the z^L values, an activation function σ is applied and weighted sums are formed. We can therefore expand this equation further using the chain rule again.

$$\delta_i^{L-1} = \sum_j \delta_j^L \frac{\partial z_j^L}{\partial z_i^{L-1}} = \sum_j \delta_j^L \frac{\partial z_j^L}{\partial a_i^{L-1}} \frac{\partial a_i^{L-1}}{\partial z_i^{L-1}} = \sum_j \delta_j^L w_{ji}^L \sigma'(z_i^{L-1}) \tag{6}$$

Here, $\sigma'()$ is the derivative of the activation function used.

This can be written in vector form as

$$\boxed{\delta^{\mathbf{L-1}} = (\mathbf{W}^{L\top} \delta^{\mathbf{L}}) \odot \sigma'(\mathbf{z}^{\mathbf{L-1}})} \tag{7}$$

Here we are taking the product of the transpose of the weight matrix for layer **L** with the vector (or one-column matrix) consisting of the errors for layer **L**; that is, the rates of change of the network loss with respect to the weighted sums of the neurons in the layer. We then take the Hadamard product (elementwise product) with the vector consisting of the activation function σ applied to each of the weighted sums of the neurons in layer **L-1**.

4 Backpropagation

If we can now find the error for the output layer, we can use these equations to find the errors for each previous layer.

If the output layer is labelled **L** and the error vector for this layer is $\delta^{\mathbf{L}}$, we can then find the errors for $\delta^{\mathbf{L-1}}$. We can use this, in turn, to find the error vector for $\delta^{\mathbf{L-2}}$, and so on for all layers.

In order to do this we first need to find the error vector for the output layer. We need to know the output values of the network, the loss for these values, and we need to know how to differentiate the loss function with respect to each of the output values.

To then find the error vectors for earlier layers, we will need to also know the outputs for those layers, and how to calculate the derivative of the activation function used.