Neural Networks

Aditya Mehrotra

November 2020

1 Initial definitions and forward prop

We define inputs to our model as an (n_{in},1) matrix, where n_{in} is the number of input nodes in our neural network.

$$\mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_{n_{in}} \end{bmatrix}$$

We define outputs to our model as an (n_{out}, 1) matrix, where n_{out} is the number of nodes in the final layer of our neural network.

$$\mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_{n_{out}} \end{bmatrix}$$

Notation note: j refers to an arbitrary neuron in the l^{th} layer, whilst k refers to an arbitrary neuron in the $l-1^{th}$ layer. Additionally, we define m as the final neuron in the $l^{th} layer$ and n as the final neuron in the $l-1^{th}$ layer

We define the weight between two nodes in layer l and layer $l-1$ using the following notation.

$$w^l_{jk}$$

This is the weight from the k^{th} neuron in the $(l-1)^{th}$ layer to the j^{th} neuron on the l^{th} layer.

Using this, we end up getting a weight matrix for every layer that looks like the following:

$$\mathbf{w}^l = \begin{bmatrix} w_{1,1} & \cdots & w_{1,n} \\ \vdots & \ddots & \vdots \\ w_{m,1} & \cdots & w_{m,n} \end{bmatrix}$$

We define the bias for a neuron j in layer l as

$$b_j^l$$

Hence, we define our bias vector for layer l as:

$$\mathbf{b}^l = \begin{bmatrix} b_1^l \\ \vdots \\ b_m^l \end{bmatrix}$$

We define the activation of the j^{th} neuron in the l^{th} layer as

$$a_j^l = \sigma(\sum_k w_{jk}^l a_k^{l-1} + b_j^l)$$

Hence, in the same manner as the bias, we can define the activations matrix for layer l as:

$$\mathbf{a}^l = \begin{bmatrix} a_1^l \\ \vdots \\ a_m^l \end{bmatrix}$$

In order to implement backprop, we need to define an intermediate quantity z_j^l, as the weighted input to neuron j in layer l.

$$z_j^l = \sum_k w_{jk}^l a_k^{l-1} + b_j^l$$

We can use this to define an matrix z_l, which consists of the weighted inputs to the neurons in layer l as:

$$\mathbf{z}^l = \begin{bmatrix} z_1^l \\ \vdots \\ z_j^l \end{bmatrix}$$

Another way of expressing this is:

$$\mathbf{z}^l = \mathbf{w}^l \mathbf{a}^{l-1} + \mathbf{b}^l$$

Lastly, we define our cost function as a regular MSE cost:

$$C = \frac{1}{2N} \sum_x ||\mathbf{a}^L(\mathbf{x}) - \mathbf{y}(\mathbf{x})||^2$$

Where $\mathbf{y}(\mathbf{x})$ is the label for a training example $\mathbf{x}$ and $\mathbf{a}^L(\mathbf{x})$ is the activations of the final layer when $\mathbf{x}$ is fed into the neural network (this is denoted by L, which represents the final layer). We include the $\frac{1}{N}$ because we are averaging the cost over N training examples.

The cost for a single training example can be expressed as the following:

$$C = \frac{1}{2}||\mathbf{a}^L - \mathbf{y}||^2 = \frac{1}{2}\sum_j (a_j^L - y_j)^2$$

2 The backpropogation algorithm

We define a vector δ^l, where each element is δ_j^l, corresponding to the error in the j^{th} neuron in layer l.

$$\delta^l = \begin{bmatrix} \delta_1^l \\ \vdots \\ \delta_m^l \end{bmatrix}$$

Furthermore, we define the error of single neuron j in layer l to be the partial derivative of the cost function with respect to the weighted input to the j^{th} neuron in layer l.

$$\delta^l = \begin{bmatrix} \dfrac{\partial C}{\partial z_1^l} \\ \vdots \\ \dfrac{\partial C}{\partial z_m^l} \end{bmatrix}$$

Hence, using this concept, we can define four fundamental equations for back propagation, which we will prove in order.

2.1 Equation 1: Error in the final layer L

The equation to compute the error of the final layer L is given by:

$$\delta^L = \nabla_{\mathbf{a}^L} C \odot \sigma'(\mathbf{z}^L)$$

We can show this by writing each term as a matrix of partial derivatives, and computing the final product.

Before we do this, we need to do some groundwork. Recall that n_{out} is the number of nodes in our final layer.

Additionally, recall that the cost of a single training example can be expressed as

$$C = \frac{1}{2}\sum_j (a_j^L - y_j)^2$$

Hence, we do the following.

$$\frac{\partial C}{\partial a_j^L} = (a_j^L - y_j)$$

However, if we want to compute $\nabla_{\mathbf{a}^L} C$, we just have to do the following,

$$\nabla_{\mathbf{a}^L} C = \mathbf{a}^L - \mathbf{y}$$

Putting this all together, we can show the validity of this equation using the following,

$$\delta^L = \nabla_{\mathbf{a}^L} C \odot \sigma'(\mathbf{z}^L)$$

$$= \begin{bmatrix} \frac{\partial C}{\partial a_1^L} \\ \vdots \\ \frac{\partial C}{\partial a_{n_{out}}^L} \end{bmatrix} \odot \begin{bmatrix} \frac{\partial a_1^L}{\partial z_1^L} \\ \vdots \\ \frac{\partial a_{n_{out}}^L}{\partial z_{n_{out}}^L} \end{bmatrix}$$

2.2 Equation 2: Error in a hidden layer l

The equation to compute the error in a hidden layer l is given by:

$$\delta^l = ((\mathbf{w}^{l+1})^T \delta^{l+1}) \odot \sigma'(\mathbf{z}^l)$$

We can show this by writing each term as a matrix of partial derivatives and compute the final product.

$$\delta^l = ((\mathbf{w}^{l+1})^T \delta^{l+1}) \odot \sigma'(\mathbf{z}^l)$$

$$= \begin{bmatrix} w_{1,1}^{l+1} & \cdots & w_{1,n}^{l+1} \\ \vdots & \ddots & \vdots \\ w_{m,1}^{l+1} & \cdots & w_{m,n}^{l+1} \end{bmatrix}^T \begin{bmatrix} \frac{\partial C}{\partial z_1^{l+1}} \\ \vdots \\ \frac{\partial C}{\partial z_m^{l+1}} \end{bmatrix} \odot \begin{bmatrix} \frac{\partial a_1^l}{\partial z_1^l} \\ \vdots \\ \frac{\partial a_k^l}{\partial z_n^l} \end{bmatrix}$$

$$= \begin{bmatrix} w_{1,1}^{l+1} & \cdots & w_{m,1}^{l+1} \\ \vdots & \ddots & \vdots \\ w_{1,n}^{l+1} & \cdots & w_{m,n}^{l+1} \end{bmatrix} \begin{bmatrix} \frac{\partial C}{\partial z_1^{l+1}} \\ \vdots \\ \frac{\partial C}{\partial z_m^{l+1}} \end{bmatrix} \odot \begin{bmatrix} \frac{\partial a_1^l}{\partial z_1^l} \\ \vdots \\ \frac{\partial a_k^l}{\partial z_n^l} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{\partial z_1^{l+1}}{\partial a_1^l} & \cdots & \frac{\partial z_m^{l+1}}{\partial a_1^l} \\ \vdots & \ddots & \vdots \\ \frac{\partial z_1^{l+1}}{\partial a_n^l} & \cdots & \frac{\partial z_m^{l+1}}{\partial a_n^l} \end{bmatrix} \begin{bmatrix} \frac{\partial C}{\partial z_1^{l+1}} \\ \vdots \\ \frac{\partial C}{\partial z_m^{l+1}} \end{bmatrix} \odot \begin{bmatrix} \frac{\partial a_1^l}{\partial z_1^l} \\ \vdots \\ \frac{\partial a_n^l}{\partial z_n^l} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{\partial C}{\partial z_1^{l+1}} \frac{\partial z_1^{l+1}}{\partial a_1^l} & + \cdots & \frac{\partial C}{\partial z_m^{l+1}} \frac{\partial z_m^{l+1}}{\partial a_1^l} \\ \vdots & \ddots & \vdots \\ \frac{\partial C}{\partial z_1^{l+1}} \frac{\partial z_1^{l+1}}{\partial a_n^l} & + \cdots & \frac{\partial C}{\partial z_m^{l+1}} \frac{\partial z_m^{l+1}}{\partial a_n^l} \end{bmatrix} \odot \begin{bmatrix} \frac{\partial a_1^l}{\partial z_1^l} \\ \vdots \\ \frac{\partial a_n^l}{\partial z_n^l} \end{bmatrix}$$

2.3 Equation 3: derivative of cost function w.r.t a bias b_j^l for neuron j in layer l

The way to compute the $\frac{\partial C}{\partial b_m^l}$ is given by:

$$\frac{\partial C}{\partial b_j^l} = \delta_j^l$$

We get this through the following derivation:
Recall that we define

$$\mathbf{z}^l = \mathbf{w}^l \mathbf{a}^{l-1} + \mathbf{b}^l$$

Using the above definition, we also defined that

$$z_j^l = \sum_k w_{jk}^l a_k^{l-1} + b_j^l$$

Hence we can compute the following,

$$\begin{aligned} \frac{\partial z_j^l}{\partial b_j^l} &= \frac{\partial}{\partial b_j^l}(\sum_k w_{jk}^l a_k^{l-1} + b_j^l) \\ &= 0 + \frac{\partial}{\partial b_j^l}(b_j^l) \\ &= 1 \end{aligned}$$

Hence, using the chain rule, we get that

$$\frac{\partial C}{\partial b_j^l} = \frac{\partial C}{\partial z_j^l} \frac{\partial z_j^l}{\partial b_j^l}$$

Using what we previously computed

$$\begin{aligned} \frac{\partial C}{\partial b_j^l} &= \frac{\partial C}{\partial z_j^l} \\ &= \delta_j^l \end{aligned}$$

Therefore, we can say that

$$\nabla_{\mathbf{b}^l} C = \delta^l$$

2.4 Equation 4: derivative of cost function w.r.t a weight w_j^l for neuron j in layer l

The way to compute the $\dfrac{\partial C}{\partial w_{jk}^l}$ is given by:

$$\frac{\partial C}{\partial w_{jk}^l} = a_k^{l-1}\delta_j^l$$

We prove that these two statements are equivalent using the following:

Recall that

$$z_j^l = \sum_k w_{jk}^l a_k^{l-1} + b_j^l$$

We compute

$$\frac{\partial z_j^l}{\partial w_{jk}^l} = a_k^{l-1}$$

Hence, we have that

$$\frac{\partial C}{\partial w_{jk}^l} = \frac{\partial C}{\partial z_j^l}\frac{\partial z_j^l}{\partial w_{jk}^l}$$

Substituting $\dfrac{\partial C}{\partial z_j^l}$ as δ_j^l and $\dfrac{\partial z_j^l}{\partial w_{jk}^l}$ as a_k^{l-1}, as we calculated previously. We get the following,

$$\frac{\partial C}{\partial w_{jk}^l} = a_k^{l-1}\delta_j^l$$

When programming a neural network, we usually want to compute the a matrix of the same dimension as the weights matrix, with each element being the partial derivative of the weight at that position with respect to the cost. This matrix looks like the following:

$$\begin{bmatrix} \frac{\partial C}{\partial z_1^l}\frac{\partial z_1^l}{\partial w_{1,1}^l} & \cdots & \frac{\partial C}{\partial z_1^l}\frac{\partial z_1^l}{\partial w_{1,n}^l} \\ \vdots & \ddots & \vdots \\ \frac{\partial C}{\partial z_m^l}\frac{\partial z_m^l}{\partial w_{m,1}^l} & \cdots & \frac{\partial C}{\partial z_m^l}\frac{\partial z_m^l}{\partial w_{m,n}^l} \end{bmatrix}$$

We can compute this using the following formula:

$$\delta^l(a^{l-1})^T$$

Writing each term as a matrix, we get the following

$$\delta^l (a^{l-1})^T = \begin{bmatrix} \dfrac{\partial C}{\partial z_1^l} \\ \vdots \\ \dfrac{\partial C}{\partial z_m^l} \end{bmatrix} \begin{bmatrix} a_1^l & \dots & a_n^{l-1} \end{bmatrix}$$

$$= \begin{bmatrix} \dfrac{\partial C}{\partial z_1^l} a_1^{l-1} & \dots & \dfrac{\partial C}{\partial z_1^l} a_n^{l-1} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial C}{\partial z_m^l} a_1^{l-1} & \dots & \dfrac{\partial C}{\partial z_m^l} a_n^{l-1} \end{bmatrix}$$

We previously found that

$$\frac{\partial z_j^l}{\partial w_{jk}^l} = a_k^{l-1}$$

This statement means that the $\dfrac{\partial z_j^l}{\partial w_{jk}^l}$ does not rely on any terms indexed with neurons in layer $l-1$. Hence, we have that for all $j \in \{1, \dots, m\}$, $\dfrac{\partial z_j^l}{\partial w_{jk}^l} = a_k^{l-1}$.

Hence, we get the following,

$$= \begin{bmatrix} \dfrac{\partial C}{\partial z_1^l} \dfrac{\partial z_1^l}{\partial w_{1,1}^l} & \dots & \dfrac{\partial C}{\partial z_1^l} \dfrac{\partial z_1^l}{\partial w_{1,n}^l} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial C}{\partial z_m^l} \dfrac{\partial z_m^l}{\partial w_{m,1}^l} & \dots & \dfrac{\partial C}{\partial z_m^l} \dfrac{\partial z_m^l}{\partial w_{m,n}^l} \end{bmatrix}$$