# 3D Spatially Embedded Networks: Verifying Coordinate Backpropagation on a Single Neuron

Pascal Tohouri

14 July 2025

**Abstract**

The purpose of this research note is to verify that one can apply backpropagation to a spatially embedded artificial neuron. The author applies the chain rule to obtain loss-coordinate derivatives for a single neuron embedded in 3D Euclidean Space. Mean-squared error loss is used.

## 1 Setup

An artificial neuron takes inputs $x_1, \ldots, x_N$ ($b$ - the bias) and produces a weighted linear combination [1]. Often a non-linear activation yields an output whose weighted value propagates via outgoing edges. One obtains loss-parameter derivatives by applying the chain rule to backpropagated partial derivatives [2]. In a spatially embedded neural network, backpropagating by one additional step through the spatial parameter condition can yield loss-coordinate derivatives.

## 2 Example: Single Neuron

Consider a single neuron:

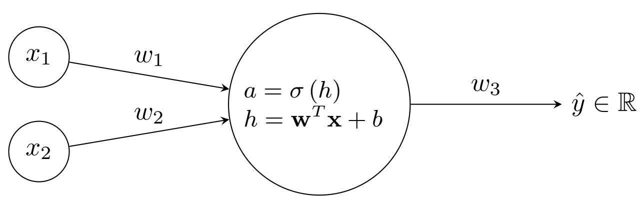


Figure 1: Sigmoid-activated Neuron

The neuron weights and sums the input vector, applying a sigmoid activation and a final weight $w_3$. Consider spatially embedding the neuron:

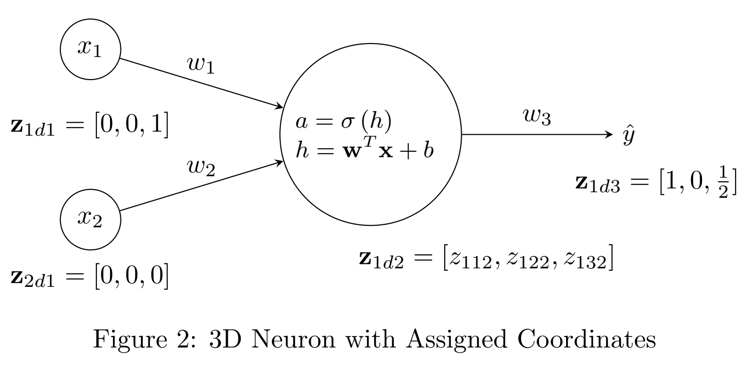


Figure 2: 3D Neuron with Assigned Coordinates

## 2.1 Parameter conditioning

For simplicity, the input and outputs pass through fixed coordinate nodes. Thus, the parameters $\theta = [w_1, w_2, w_3, b]^T$ depend only on the internal node's coordinates $\theta(\mathbf{z}_{1d2})$. Suppose each weight is the Euclidean distance between its connecting nodes:

$$w_{ij} = ||\mathbf{z}_j - \mathbf{z}_i|| \tag{1}$$

Fix the bias value to 0:

$$\theta = \begin{bmatrix} ||\mathbf{z}_{1d2} - \mathbf{z}_{1d1}|| \\ ||\mathbf{z}_{1d2} - \mathbf{z}_{2d1}|| \\ ||\mathbf{z}_{1d3} - \mathbf{z}_{1d2}|| \\ 0 \end{bmatrix} \tag{2}$$

## 2.2 Training

For simplicity, re-label $\mathbf{z}_{1d2} = [z_1, z_2, z_3]$ and optimise these coordinates to minimise the loss via gradient descent [3]:

$$\mathcal{L} = \frac{1}{2}(y^{(n)} - \hat{y})^2 \tag{3}$$

The loss has a gradient wrt $\mathbf{z}_{1d2}$, calculable via the chain rule:

$$\nabla\mathcal{L} = \begin{bmatrix} \frac{\partial\mathcal{L}}{\partial z_1} \\ \frac{\partial\mathcal{L}}{\partial z_2} \\ \frac{\partial\mathcal{L}}{\partial z_3} \end{bmatrix} \tag{4}$$

Each internal coordinate $z_d$ affects the loss via three channels:

$$\mathcal{L} = \mathcal{L}(w_1(z_d), w_2(z_d), w_3(z_d)) \tag{5}$$

The gradients are:

$$\frac{\partial\mathcal{L}}{\partial z_d} = \left(\frac{\partial\mathcal{L}}{\partial w_1} \cdot \frac{\partial w_1}{\partial z_d}\right) + \left(\frac{\partial\mathcal{L}}{\partial w_2} \cdot \frac{\partial w_2}{\partial z_d}\right) + \left(\frac{\partial L}{\partial w_3} \cdot \frac{\partial w_3}{\partial z_d}\right) \tag{6}$$

Specifying each bracketed term for coordinate $z_d$:

$$\frac{\partial\mathcal{L}}{\partial w_1} \cdot \frac{\partial w_1}{\partial z_d} = \underbrace{(y^{(n)} - \hat{y}) \cdot w_3 \cdot \sigma(h)(1 - \sigma(h)) \cdot x_1}_{\frac{\partial\mathcal{L}}{\partial w_1}} \cdot \underbrace{\frac{(z_{1d2} - z_{1d1})}{w_1}}_{\frac{\partial w_1}{\partial z_d}} \tag{7}$$

$$\frac{\partial \mathcal{L}}{\partial w_2} \cdot \frac{\partial w_2}{\partial z_d} = \underbrace{(y^{(n)} - \hat{y}) \cdot w_3 \cdot \sigma(h)(1-\sigma(h)) \cdot x_2}_{\frac{\partial \mathcal{L}}{\partial w_2}} \cdot \underbrace{\frac{(z_{1d2} - z_{2d1})}{w_2}}_{\frac{\partial w_2}{\partial z_d}} \tag{8}$$

$$\frac{\partial \mathcal{L}}{\partial w_3} \cdot \frac{\partial w_3}{\partial z_d} = \underbrace{(y^{(n)} - \hat{y})a}_{\frac{\partial \mathcal{L}}{\partial w_3}} \cdot - \underbrace{\frac{(z_{1d3} - z_{1d2})}{w_3}}_{\frac{\partial w_3}{\partial z_d}} \tag{9}$$

Putting all the partial derivatives together:

$$\nabla \mathcal{L} = \begin{bmatrix} \frac{\partial \mathcal{L}}{\partial z_1} \\ \frac{\partial \mathcal{L}}{\partial z_2} \\ \frac{\partial \mathcal{L}}{\partial z_3} \end{bmatrix} = \begin{bmatrix} \left(\frac{\partial \mathcal{L}}{\partial w_1} \cdot \frac{\partial w_1}{\partial z_1}\right) + \left(\frac{\partial \mathcal{L}}{\partial w_2} \cdot \frac{\partial w_2}{\partial z_1}\right) + \left(\frac{\partial \mathcal{L}}{\partial w_3} \cdot \frac{\partial w_3}{\partial z_1}\right) \\ \left(\frac{\partial \mathcal{L}}{\partial w_1} \cdot \frac{\partial w_1}{\partial z_2}\right) + \left(\frac{\partial \mathcal{L}}{\partial w_2} \cdot \frac{\partial w_2}{\partial z_2}\right) + \left(\frac{\partial \mathcal{L}}{\partial w_3} \cdot \frac{\partial w_3}{\partial z_2}\right) \\ \left(\frac{\partial \mathcal{L}}{\partial w_1} \cdot \frac{\partial w_1}{\partial z_3}\right) + \left(\frac{\partial \mathcal{L}}{\partial w_2} \cdot \frac{\partial w_2}{\partial z_3}\right) + \left(\frac{\partial L}{\partial w_3} \cdot \frac{\partial w_3}{\partial z_3}\right) \end{bmatrix} \tag{10}$$

The updates are thus:

$$\mathbf{z}_{t+1}^T = \mathbf{z}_t^T - \eta \nabla \mathcal{L} \tag{11}$$

$$\mathbf{z}_{t+1}^T = \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} - \eta \begin{bmatrix} \frac{\partial \mathcal{L}}{\partial z_1} \\ \frac{\partial \mathcal{L}}{\partial z_2} \\ \frac{\partial \mathcal{L}}{\partial z_3} \end{bmatrix} \tag{12}$$

# 3 Conclusion

This research note presented the derivation of the gradient descent update condition for a single spatially embedded neuron with parameter conditioning. Key limitations were the explicit sigmoid and loss function and the bias omission. This note is an illustrative precursor to more general update derivations.

# References


[1] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bulletin of Mathematical Biophysics*, vol. 5, no. 4, pp. 115–133, 1943. DOI: 10.1007/BF02478259.

[2] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, 1986. DOI: 10.1038/323533a0.

[3] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford University Press, 1995, Chapter 4 details the mean-squared-error (least-squares) loss., ISBN: 978-0198538646.