Homework #8 (NEURL-GA 3042, Fall 2022)

Backpropagation algorithm of supervised learning
Due date: Sunday December 11

Consider a very simple network with three neurons. Denote the activity (loosely speaking, a firing rate) of each neuron as x_1, x_2, and x_3, which can be any positive or negative number. The neurons are connected in a chain as depicted in the figure below by linear synapses with weights w_1 and w_2. That is, the firing rate of x_3 is given by:

$$x_3 = w_2 x_2 = w_2 w_1 x_1$$

We will construct a learning rule that modifies w_1 and w_2 so that the firing rate of neuron x_3 matches a target firing rate, which we denote as y. To achieve this, we need a way to measure the distance between x_3 and y, which is called a loss function. While there are many loss functions we could choose, for simplicity we willl assume a quadratic loss function.

$$l(y, x_3) = 0.5(y - x_3)^2$$

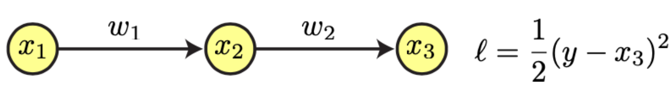

Figure 1: Three-layer feedforward network

Ultimately, we want to find the set of weights for which the loss function is minimized.

a) Warm-up. If $x_1 > 0$, $w_1 < 0$, and $x_3 < y$, should w_2 increase or decrease to improve the loss? Should w_1 increase or decrease?

To be more precise than the above procedure, we use gradient descent, which computes derivatives of the loss function with respect to the synaptic weights, and then uses these derivatives to iteratively update the weights. Backpropagation is simply a procedure for computing these derivatives by the chain rule taught in any standard calculus course.

b) Compute the derivative of l with respect to x_3. Then, using the chain rule compute the derivatives with respect to w_2, x_2 and w_1.

c) Using your results above, write down an update rule that is guaranteed to decrease the loss function for a sufficiently small step size (learning rate).

Feedforward backpropagation (nonlinear case)

Real biological networks are nonlinear, as are artificial networks used in practice in machine learning. Consider a slight modification to the above network where x_2 has a nonlinear activation function (usually, the final layer of the network is linearly read out in machine learning applications). Specifically, consider the model:

$$x_3 = w_2\sigma(w_1x_1)$$

where, σ is a nonlinear function. Unless otherwise specified, simply assume that σ is an arbitrary monotonically increasing nonlinear function with derivative σ'

d) Recompute your answers to question (b) with the nonlinearity. How do your answers change after the introduction of this nonlinearity?

e) A typical choice for σ in machine learning applications is the rectified linear unit (ReLU), $\sigma(x) = max(0, x)$. Compute the gradient of the weights for this choice of σ. Can you foresee any problems for the backpropagation learning rule in this scenario? (Hint: think about what happens when x_1 and w_1 have opposite signs.)