Homework 7 Solutions

Problem 1.

We show how to modify the back-propagation training algorithm to handle the following extended forms of neural nets. In each case, we assume that we have the value of $\frac{\partial E}{\partial z}$, where z is the output of a given neuron. We then give the partial of E with respect to all the inputs to the neuron, and with respect to all the corresponding weights; in each case, we apply the chain rule:

$$\frac{\partial E}{\partial y} = \frac{\partial E}{\partial z}\frac{\partial z}{\partial y}.$$

Once we show how to calculate these values, we can do back-propagation as described in lecture.

(a) Let $x_1, x_2, \ldots, x_k$ denote the inputs to a particular neuron, $w_1, w_2, \ldots, w_k$ denote the corresponding weights, and λ denote the "variable-sharpness" parameter of the neuron. Then the neuron we are considering outputs $z = \tanh(\lambda y)$, where $y = \sum_i x_i w_i$. We have:

$$\begin{aligned}
\frac{\partial E}{\partial \lambda} &= \frac{\partial E}{\partial z}\text{sech}^2(\lambda y) y \\
\frac{\partial E}{\partial x_i} &= \frac{\partial E}{\partial z}\text{sech}^2(\lambda y) \lambda w_i \\
\frac{\partial E}{\partial w_i} &= \frac{\partial E}{\partial z}\text{sech}^2(\lambda y) \lambda x_i
\end{aligned}$$

(b) Let $x_1, x_2, \ldots, x_k$ denote edges which are all associated with weight w. We can then consider each x_i to be an input to a multiplication node N_i, which computes the value $z_i = x_i w$. Then we have:

$$\frac{\partial E}{\partial w} = \sum_i \frac{\partial E}{\partial z_i} x_i$$

(c) Let $x_1, x_2, \ldots, x_k$ denote the inputs to a quadratic node, $U = (u_{ij})$ denote a $n \times n$ weight matrix, and w denote another n element weight matrix. Then the quadratic neuron computes $z = \tanh(x^T U x + wx) = \sum_i x_i^2 u_{ii} + \sum_{i \neq j} x_i x_j (u_{ij} + u_{ji}) + \sum_i w_i x_i$. Then we have:

$$\begin{aligned}
\frac{\partial E}{\partial x_i} &= \frac{\partial E}{\partial z}\text{sech}^2(x^T U x + wx)\left(2x_i u_{ii} + \sum_{i \neq j} x_j (u_{ji} + u_{ij}) + w_i\right) \\
\frac{\partial E}{\partial w_i} &= \frac{\partial E}{\partial z}\text{sech}^2(x^T U x + wx) x_i \\
\frac{\partial E}{\partial u_{ij}} &= \frac{\partial E}{\partial z}\text{sech}^2(x^T U x + wx) x_i x_j
\end{aligned}$$

Problem 2.

Consider the neural net which has only one input and two neurons, one in the hidden layer and one in the output layer. The network uses the logistic sigmoid function $\sigma(x) = 1/(1 + e^{-x})$. There are two weights in the system, w_1 from the input to the hidden layer, and w_2 from the hidden layer to the output (see Figure 1). We consider what happens when this network is trained to learn the identity function. The training sample consists of just two examples, $x = 0, y = 0$ and $x = 1, y = 1$, which are fed to the network repeatedly. We first give a high-level explanation (based on one in [1]) for showing that back-propagation finds different minima depending on the initial settings of the weights.

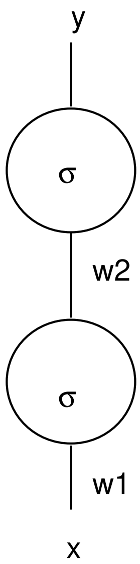

Figure 1: A 2 node neural net. Each node computes the logistic simoid function.

First consider the case when w_1 and w_2 are initially large negative numbers. Then for $x = 0$, the hidden unit receives an input of 0 and thus outputs 1/2. Thus, the network outputs $1/(1 + e^{-w_2/2})$. But since w_2 is initially a large negative number, this value is close to 0, as desired. On the other hand, for $x = 1$, since w_1 is a large negative number, the output of the hidden node is close to 0, and thus the output of the entire network is 1/2, instead of 1 as desired. Thus the mean squared error for these two examples is $1/2((1/2)^2) = 1/8$. This, in fact, is the global minimum (at $w_1 = w_2 = -\infty$).

Now consider the case when w_1 and w_2 are both initially large positive numbers. When $x = 0$ then the output of the network is $1/(1+e^{-w_2/2})$. Since the network should output 0 in this case, there is pressure for w_2 to be smaller. On the other hand, when $x = 1$, the network outputs $1/(1 + e^{-\frac{w_2}{1+e^{-w_1}}})$. Since the output should be 1, there is pressure for $e^{-\frac{w_2}{1+e^{-w_1}}} \to 0$. In other words, there is pressure for both w_1 and w_2 to increase. However, when $x = 0$ there is pressure for w_2 to decrease. These pressures balance off at $w_2 \approx .73$, giving a mean squared error of $\approx .210$. Thus, there is a local minimum at $w_1 = \infty$ and $w_2 \approx .73$.

To solve this problem exactly, we consider the gradient of the mean squared error E. Let E_0 denote the squared error the network makes on input $x = 0$, and E_1 denote the squared error the network makes on input $x = 1$. Then $\nabla E = 1/2(\nabla E_0 + \nabla E_1)$. We have the following equalities:

$$\begin{aligned}
E_0 &= \frac{1}{1 + 2e^{-\frac{w_2}{2}} + e^{-w_2}} \\
E_1 &= \frac{e^{-\frac{2w_2}{1+e^{-w_1}}}}{\left(1 + e^{-\frac{w_2}{1+e^{-w_1}}}\right)^2} \\
\frac{\partial E_0}{\partial w_1} &= 0 \\
\frac{\partial E_0}{\partial w_2} &= \frac{e^{-\frac{w_2}{2}}}{\left(1 + e^{-\frac{w_2}{2}}\right)^3}
\end{aligned}$$

$$\frac{\partial E_1}{\partial w_1} = -2w_2 \frac{e^{-w_1} e^{-\frac{2w_2}{1+e^{-w_1}}}}{(1+e^{-w_1})^2 \left(1+e^{-\frac{w_2}{1+e^{-w_1}}}\right)^3}$$

$$\frac{\partial E_1}{\partial w_2} = \frac{-2e^{-\frac{2w_2}{1+e^{-w_1}}}}{(1+e^{-w_1}) \left(1+e^{-\frac{w_2}{1+e^{-w_1}}}\right)^3}$$

Since $\frac{\partial E_0}{\partial w_1} = 0$, weight w_1 is updated by $-\frac{1}{2}\frac{\partial E_1}{\partial w_1}$. Notice that this value is negative if w_2 is negative and positive if w_2 is positive. Now to figure out how w_2 is updated, we must consider when $\frac{\partial E_0}{\partial w_2} + \frac{\partial E_1}{\partial w_2} > 0$. We consider two cases. First consider when w_1 is a large negative number. In this case, $\frac{\partial E_0}{\partial w_2} + \frac{\partial E_1}{\partial w_2} \approx \frac{\partial E_0}{\partial w_2}$, which is always positive. So, when w_1 is a large negative number, weight w_2 is decreased by back-propagation. Now, when w_1 is a large positive number,

$$\frac{\partial E_0}{\partial w_2} + \frac{\partial E_1}{\partial w_2} \approx \frac{e^{-\frac{w_2}{2}}}{\left(1+e^{-w_2/2}\right)^3} - \frac{2e^{-2w_2}}{(1+e^{-w_2})^3}.$$

Setting this equal to 0 and solving (by simplifying, substituting $x = e^{-w^2/2}$, and using the quadratic formula) gives $w_2 \approx .73$. When $w_2 < .73$, then this quantity is negative, and thus w_2 is increased by back-propagation, and when $w_2 > .73$, this quantity is positive, and thus w_2 is decreased by back-propagation.

Thus, if w_1 and w_2 are large negative numbers, then we have shown that w_1 and w_2 will both decrease, and approach $-\infty$. On the other hand, if w_1 and w_2 are large positive numbers then w_1 approaches $+\infty$ and w_2 converges to $\approx .73$. Thus we have shown that back-propagation can find different minima depending on the initial settings of the weights.

References

[1] James L. McClelland and David E. Rumelhart. *Explorations in Parallel Distributed Processing.* MIT Press, 1988.