CS 189/289A Introduction to Machine Learning
Fall 2023 Jennifer Listgarten, Jitendra Malik

DIS4

1 Initialization of Weights for Backpropagation

Assume a fully-connected 1-hidden-layer network. Denote the dimensionalities of the input, hidden, and output layers as $d^{(0)}$, $d^{(1)}$, and $d^{(2)}$. That is, the input (which we will denote with a superscript (0)) is a vector of the form $x_1^{(0)}, \ldots, x_{d^{(0)}}^{(0)}$. Let g denote the activation function applied at each layer. We will let $S_j^{(l)} = \sum_{i=1}^{d^{(l-1)}} w_{ij}^{(l)} x_i^{(l-1)}$ be the weighted input to node j in layer l, and let $\delta_j^{(l)} = \frac{\partial \ell}{\partial S_j^{(l)}}$ be the partial derivative of the final loss ℓ with respect to $S_j^{(l)}$.

Recall that backpropagation is an efficient method to compute the gradient of the loss function so we can use it for gradient descent. Gradient descent requires the parameters to be initialized to some value(s).

(a) To better orient yourself with the operations described in this 1-hidden-layer network, draw out a diagram of the layers, including weights, activation functions, and the outputs of each operation during the forward pass. In addition, identify where the partial derivatives $\delta_j^{(l)}$ are calculated during backpropagation.

Solution:

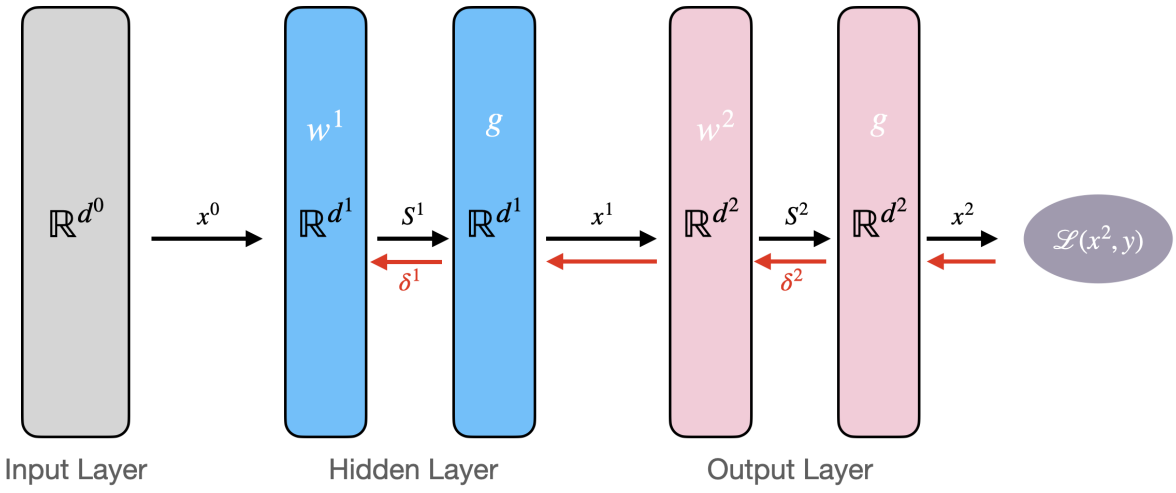

Figure 1: Simple diagram of the 1-hidden layer network

Note that the output layer does not necessarily have to include an activation function (and often in practice does not); we add one here for consistency.

(b) Imagine that we initialize every element of each weight $w^{(l)}$ to be the same constant scalar value a. After performing the forward pass, what is the value of $x_j^{(1)}$ in terms of the elements of $\{x_i^{(0)} : i = 1, \ldots, d^{(0)}\}$? What is the relationship between each $x_j^{(1)}$?

Solution: Since all of the weights are equal, we have:

$$x_j^{(1)} = g\left(\sum_{i=1}^{d^{(0)}} w_{ij}^{(1)} x_i^{(0)}\right) = g\left(a \sum_{i=1}^{d^{(0)}} x_i^{(0)}\right)$$

Note that this equation does not depend on j so all $x_j^{(1)}$ are equal.

(c) Following from the previous part, after the backward pass of backpropagation, compute the values for each member of the set $\{\delta_i^{(1)} : i = 1, \ldots, d^{(1)}\}$, assuming we have calculated $\{\delta_j^{(2)} : j = 1, \ldots, d^{(2)}\}$. What is the relationship between each $\delta_i^{(1)}$?

Solution: Since all of the weights are equal:

$$\begin{aligned}
\delta_i^{(1)} &= \sum_{j=1}^{d^{(2)}} \delta_j^{(2)} w_{ij}^{(2)} g'(S_i^{(1)}) \\
&= g'(S_i^{(1)}) a \sum_{j=1}^{d^{(2)}} \delta_j^{(2)} \\
&= g'\left(a \sum_{k=1}^{d^{(0)}} x_k^{(0)}\right) a \sum_{j=1}^{d^{(2)}} \delta_j^{(2)}
\end{aligned}$$

The sum term within the activation function is the same for all dimensions i in layer 1 (note that this is not referring to being the same across all nodes, but rather the entries within each node.) The members of the set are equal.

(d) For a reasonable loss function, are all of the $\delta_i^{(2)}$ equal to each other? Why or why not?

Solution: No. $\delta_i^{(2)}$ depends on y_i, which is different for each i. Note that *any* reasonable output loss/activation should result in $\delta_i^{(2)}$ depending on a target y_i (otherwise the output is not attempting to approach a particular target value).

(e) After the weights are updated and one iteration of gradient descent has been completed, what can we say about the weights?

Solution: Our gradient descent update looks like this, for some learning rate η:

$$\begin{aligned}
w_{ij}^{(l)} &= w_{ij}^{(l)} - \eta \delta_j^{(l)} x_i^{(l-1)} = a - \eta \delta_j^{(l)} x_i^{(l-1)} \\
\implies w_{ij}^{(1)} &= a - \eta \delta_j^{(1)} x_i^{(0)} \\
\implies w_{ij}^{(2)} &= a - \eta \delta_j^{(2)} x_i^{(1)}
\end{aligned}$$

For a given i, $w_{ij}^{(1)}$ will be the same for all j because $\delta_j^{(1)}$ is equal for all j.

For a given j, $w_{ij}^{(2)}$ will be the same for all i because $x_i^{(1)}$ is equal for all i.

(f) In the previous part, you showed that $w_{ij}^{(2)}$ is different for each j, but for a fixed j, it is the same for each i. In fact, no matter how many subsequent iterations of gradient descent you take, this property will continue to be true. Show why this is the case.

Solution: Let $w_{ij}^{(1)} = w_i^{(1)}$ for all i. Let $w_{ij}^{(2)} = w_j^{(2)}$ for all j. Then:

$$x_j^{(1)} = g\left(\sum_{i=1}^{d^{(0)}} w_{ij}^{(1)} x_i^{(0)}\right) = g\left(\sum_{i=1}^{d^{(0)}} w_i^{(1)} x_i^{(0)}\right)$$

Which does not depend on j. Also,

$$\delta_i^{(1)} = \sum_{j=1}^{d^{(2)}} \delta_j^{(2)} w_{ij}^{(2)} g'\left(\sum_{k=1}^{d^{(0)}} w_{ki}^{(1)} x_k^{(0)}\right) = \sum_{j=1}^{d^{(2)}} \delta_j^{(2)} w_j^{(2)} g'\left(\sum_{k=1}^{d^{(0)}} w_k^{(1)} x_k^{(0)}\right)$$

Which does not depend on i.

All $x_j^{(1)}$ being the same and all $\delta_i^{(1)}$ being the same will continue no matter how many iterations you perform. Intuitively, $x_j^{(1)}$ will always be the same for all j, due to the "outgoing" symmetry in the weights in layer 1, and the $\delta_i^{(1)}$ will always be the same due to the "incoming" symmetry in the weights in layer 2.

(g) To solve this problem, we randomly initialize our weights. This is called symmetry breaking. Note that for logistic regression, we don't run into this issue; that is, gradient descent will find the optimal values of the weights even if we initialize them at 0. Explain why this discrepancy exists between our 1-hidden-layer neural network and logistic regression.

Solution: Logistic regression is a fully-connected neural network with one output node and zero hidden layers (remember that the $\delta_j^{(2)}$'s are different).

Another reason, following the above logic: in logistic regression, the loss function is convex. Any starting point should lead us to the global optimum.

2 Backpropagation Practice

(a) Chain rule of multiple variables: Assume that you have a function given by $f(x_1, x_2, \ldots, x_n)$, and that $g_i(w) = x_i$ for a scalar variable w. What is its computation graph? Sketch out a diagram of what the computation graph would look like. How would you compute

$$\frac{\mathrm{d}}{\mathrm{d}w} f(g_1(w), g_2(w), \ldots, g_n(w))$$

?

Solution: This is the chain rule for multiple variables. In general, we have

$$\frac{\mathrm{d}f}{\mathrm{d}w} = \sum_{i=1}^{n} \frac{\partial f}{\partial x_i}\frac{\partial x_i}{\partial w} = \frac{\partial f}{\partial x}\frac{\partial x}{\partial w}.$$

The function graph of this computation is given in Figure 2.

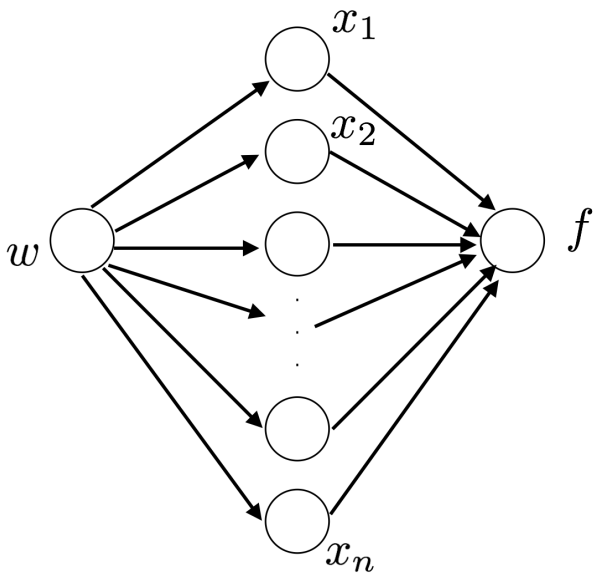

Figure 2: Example function computation graph

(b) Let $w_1, w_2, \ldots, w_n \in \mathbb{R}^d$, and we refer to these weights together as $W \in \mathbb{R}^{n \times d}$. We also have $x \in \mathbb{R}^d$ and $y \in \mathbb{R}$. Consider the function

$$f(W, x, y) = \left(y - \sum_{i=1}^{n} \phi(w_i^\top x + b_i) \right)^2 .$$

Write out the function computation graph (also sometimes referred to as a pictorial representation of the network). This is a directed graph of decomposed function computations, with the output of the function at one end, and the input to the function, x at the other end, where b are the bias terms corresponding to each weight vector, i.e. $b = [b_1, \cdots, b_n]$.

Solution:

See Figure 3.

(c) Suppose $\phi(x)$ (from the previous part) is the sigmoid function, $\sigma(x)$. Compute the partial derivatives $\frac{\partial f}{\partial w_i}$ and $\frac{\partial f}{\partial b_i}$. Use the computational graph you drew in the previous part to guide you.

Solution: Denote $r = y - \sum_{i=1}^{n} \sigma(w_i^\top x + b_i)$ and $z_i = w_i^\top x + b_i$.

To remind ourselves, this is the 'forward' computation:

$$f = r^2$$

$$r = y - \sum_{i=1}^{n} \sigma(z_i)$$

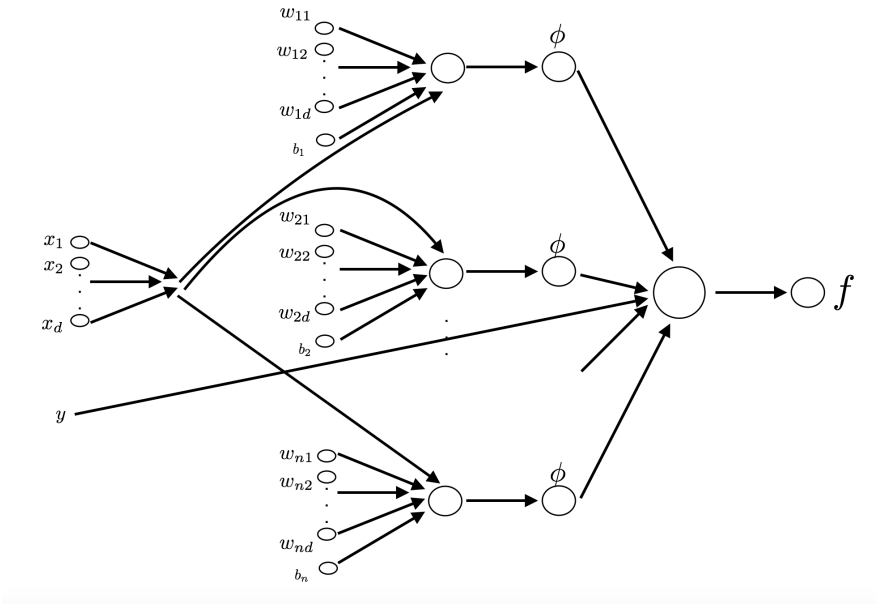

Figure 3: Example function computation graph

$$z_i = w_i^\top x + b_i$$

Now the backward pass:

$$\begin{aligned}
\frac{\partial f}{\partial r} &= 2r \\
\frac{\partial r}{\partial z_i} &= -\sigma(z_i)(1-\sigma(z_i)) \\
\frac{\partial z_i}{\partial w_i} &= x^\top \\
\frac{\partial z_i}{\partial b_i} &= 1
\end{aligned}$$

By applying chain rule

$$\begin{aligned}
\frac{\partial f}{\partial w_i} &= 2(\sum_{j=1}^{n} \sigma(w_j^\top x + b_j) - y)\sigma(w_i^\top x + b_i)(1-\sigma(w_i^\top x + b_i))x^\top \\
\frac{\partial f}{\partial b_i} &= 2(\sum_{j=1}^{n} \sigma(w_j^\top x + b_j) - y)\sigma(w_i^\top x + b_i)(1-\sigma(w_i^\top x + b_i))
\end{aligned}$$

(d) Write down a single gradient descent update for $w_i^{(t+1)}$ and $b_i^{(t+1)}$, assuming step size η. You answer should be in terms of $w_i^{(t)}$, $b_i^{(t)}$, x, and y.

Solution:

$$w_i^{(t+1)} \leftarrow w_i^{(t)} - 2\eta(\sum_{j=1}^{n} \sigma({w_j^{(t)}}^\top x + b_j^{(t)}) - y)\sigma({w_i^{(t)}}^\top x + b_i^{(t)})(1-\sigma({w_i^{(t)}}^\top x + b_i^{(t)}))x$$

$$b_i^{(t+1)} \leftarrow b_i^{(t)} - 2\eta(\sum_{j=1}^{n} \sigma({w_j^{(t)}}^\top x + b_j^{(t)}) - y)\sigma({w_i^{(t)}}^\top x + b_i^{(t)})(1 - \sigma({w_i^{(t)}}^\top x + b_i^{(t)}))$$

(e) Define the cost function

$$\ell(x) = \frac{1}{2}\|W^{(2)}\Phi\left(W^{(1)}x + b\right) - y\|_2^2, \tag{1}$$

where $W^{(1)} \in \mathbb{R}^{d\times d}$, $W^{(2)} \in \mathbb{R}^{d\times d}$, and $\Phi : \mathbb{R}^d \to \mathbb{R}^d$ is some nonlinear transformation. Compute the partial derivatives $\frac{\partial \ell}{\partial x}, \frac{\partial \ell}{\partial W^{(1)}}, \frac{\partial \ell}{\partial W^{(2)}}$, and $\frac{\partial \ell}{\partial b}$.

Solution: First, we write out the intermediate variable for our convenience.

$$\begin{aligned}
x^{(1)} &= W^{(1)}x + b \\
x^{(2)} &= \Phi(x^{(1)}) \\
x^{(3)} &= W^{(2)}x^{(2)} \\
x^{(4)} &= x^{(3)} - y \\
\ell &= \frac{1}{2}\|x^{(4)}\|_2^2.
\end{aligned}$$

Remember that the superscripts represents the index rather than the power operators. We have

$$\begin{aligned}
\frac{\partial \ell}{\partial x^{(4)}} &= x^{(4)\top} \\
\frac{\partial \ell}{\partial x^{(3)}} &= \frac{\partial \ell}{\partial x^{(4)}}\frac{\partial x^{(4)}}{\partial x^{(3)}} = \frac{\partial \ell}{\partial x^{(4)}} \\
\frac{\partial \ell}{\partial x^{(2)}} &= \frac{\partial \ell}{\partial x^{(3)}}\frac{\partial x^{(3)}}{\partial x^{(2)}} = \frac{\partial \ell}{\partial x^{(3)}}W^{(2)} \\
\frac{\partial \ell}{\partial W^{(2)}} &= \frac{\partial \ell}{\partial x^{(3)}}\frac{\partial x^{(3)}}{\partial W^{(2)}} = x^{(2)}\frac{\partial \ell}{\partial x^{(3)}} \\
\frac{\partial \ell}{\partial x^{(1)}} &= \frac{\partial \ell}{\partial x^{(2)}}\frac{\partial \Phi}{\partial x^{(1)}} \\
\frac{\partial \ell}{\partial x} &= \frac{\partial \ell}{\partial x^{(1)}}\frac{\partial x^{(1)}}{\partial x} = \frac{\partial \ell}{\partial x^{(1)}}W^{(1)} \\
\frac{\partial \ell}{\partial b} &= \frac{\partial \ell}{\partial x^{(1)}}\frac{\partial x^{(1)}}{\partial b} = \frac{\partial \ell}{\partial x^{(1)}} \\
\frac{\partial \ell}{\partial W^{(1)}} &= \frac{\partial \ell}{\partial x^{(1)}}\frac{\partial x^{(1)}}{\partial W^{(1)}} = x\frac{\partial \ell}{\partial x^{(1)}}.
\end{aligned}$$

The easy trick to solve the derivatives with respect to (each element of) a matrix is to "guess" the ordering of the expression so that the dimensions match up on both sides. More formally, we could express it as follows:

$$\frac{\partial \ell}{\partial x^{(3)}}\frac{\partial x^{(3)}}{\partial W^{(2)}} = \frac{\partial \ell}{\partial x^{(3)}}x^{(2)} = \mathrm{Tr}\left(\frac{\partial \ell}{\partial x^{(3)}}(\cdot)x^{(2)}\right) = \mathrm{Tr}\left(x^{(2)}\frac{\partial \ell}{\partial x^{(3)}}(\cdot)\right) = x^{(2)}\frac{\partial \ell}{\partial x^{(3)}} \tag{2}$$

(f) Suppose Φ is the identity map. Write down a single gradient descent update for $W_{t+1}^{(1)}$ and $W_{t+1}^{(2)}$ assuming step size η. Your answer should be in terms of $W_t^{(1)}, W_t^{(2)}, b_t$ and x, y.

Solution:

$$W_{t+1}^{(1)} \leftarrow W_t^{(1)} - \eta (W_t^{(2)})^\top \left(W_t^{(2)} \left(W_t^{(1)} x + b_t \right) - y \right) x^\top$$
$$W_{t+1}^{(2)} \leftarrow W_t^{(2)} - \eta \left(W_t^{(2)} \left(W_t^{(1)} x + b_t \right) - y \right) (W_t^{(1)} x + b)^\top$$

Side note: The computation complexity of computing the $\frac{\partial \ell}{\partial W}$ for Equation (1) using the analytic derivatives and numerical (finite-difference) derivatives is quite different!

For numerical differentiation, we use the following first order formula:

$$\frac{\partial \ell}{\partial W_{ij}} = \frac{\ell \left(W_{ij} + \epsilon, \cdot \right) - \ell \left(W_{ij}, \cdot \right)}{\epsilon}.$$

Which requires $O(d^4)$ operations to compute $\frac{\partial \ell}{\partial W}$. On the other hand, it only takes $O(d^2)$ operations to compute it analytically.