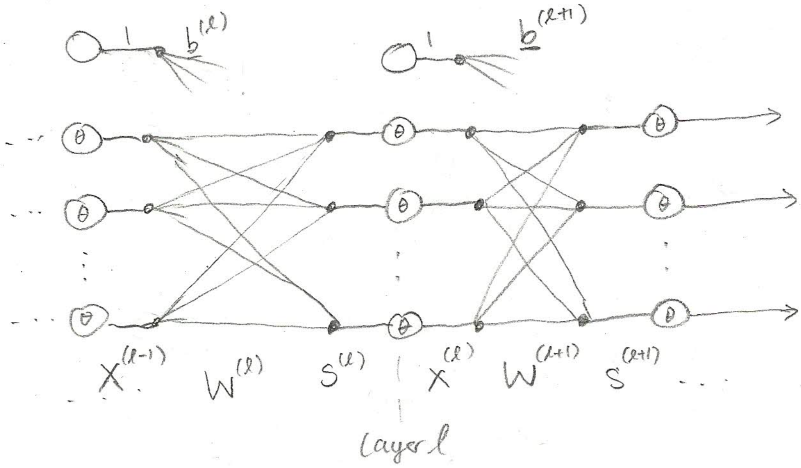
VECTORIZING the NEURAL NETWORK

Model of L-layer NN



Let $X^{(\ell)} = \begin{bmatrix} x_1^{(\ell)T} \\ \vdots \\ x_N^{(\ell)T} \end{bmatrix}$ $\quad Y = \begin{bmatrix} y_1^T \\ \vdots \\ y_N^T \end{bmatrix}$ ← one-hot encoded

$\underline{b} = [\, b_1 \cdots b_K \,]$

x_{ij} corresponds to the j^{th} feature of the i^{th} datapoint

$\underline{v}$ ← indicates broadcasting.

FORWARD PROPAGATION VECTORIZATION

$$S_{nj}^{(\ell)} = \sum_i X_{ni}^{(\ell-1)} \cdot W_{ij}^{(\ell)} + b_j^{(\ell)}$$

$$\boxed{S^{(\ell)} = X^{(\ell-1)} W^{(\ell)} + \underline{b}^{(\ell)}}$$

$$X_{nj}^{(\ell)} = \theta(S_{nj}^{(\ell)})$$

$$\boxed{X^{(\ell)} = \theta(S^{(\ell)})}$$

activation function of the ℓ^{th} layer. (ie. ReLu or Softmax)

GRADIENT VECTORIZATION

Let $\delta_{nj}^{(\ell)} = \frac{\partial E_{in}}{\partial S_{nj}^{(\ell)}}$ the sensitivity of the n^{th} datapoint at the node j layer ℓ.

$$\frac{\partial E_{in}}{\partial w_{ij}^{(\ell)}} = \sum_{n=1}^{N} \frac{\partial E_{in}}{\partial S_{nj}^{(\ell)}} \cdot \frac{\partial S_{nj}^{(\ell)}}{\partial w_{ij}^{(\ell)}}$$

(This is b/c $E_{in} = \frac{-1}{N} \sum_1^N e_n = f(S_1^{(\ell)}, S_2^{(\ell)} \cdots S_N^{(\ell)})$ which means we need to apply a branching chain rule)

$$= \sum_n \delta_{nj}^{(\ell)} \cdot X_{ni}^{(\ell-1)} = \sum_n \left(X_{in}^{(\ell-1)}\right)^T \cdot \delta_{nj}^{(\ell)} \Rightarrow \boxed{\frac{\partial E_{in}}{\partial W^{(\ell)}} = [X^{(\ell-1)}]^T \delta^{(\ell)}}$$

$$\frac{\partial E_{in}}{\partial b_j^{(\ell)}} = \sum_1^N \frac{\partial E_{in}}{\partial S_{nj}^{(\ell)}} \cdot \frac{\partial S_{nj}^{(\ell)}}{\partial b_j^{(\ell)}}$$

$$= \sum_n \delta_{nj}^{(\ell)} \cdot 1 \Rightarrow \boxed{\frac{\partial E_{in}}{\partial \underline{b}^{(\ell)}} = \underline{1}^T \delta^{(\ell)}}$$

BACKPROPAGATION VECTORIZATION

$$\delta^{(\ell)} = \frac{\partial E_{in}}{\partial s_{nj}^{(\ell)}} = \frac{\partial E_{in}}{\partial x_{nj}^{(\ell)}} \cdot \frac{\partial x_{nj}^{(\ell)}}{\partial s_{nj}^{(\ell)}}$$

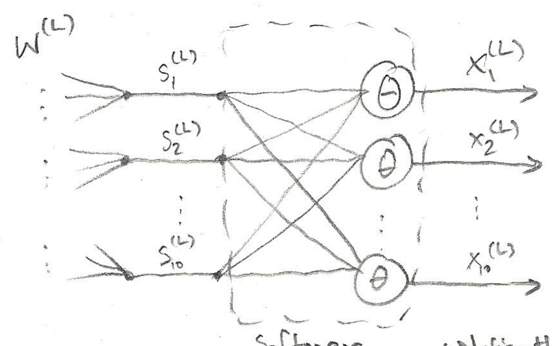
$$= \left(\sum_k \frac{\partial E_{in}}{\partial s_{nk}^{(\ell+1)}} \cdot \frac{\partial s_{nk}^{(\ell+1)}}{\partial x_{nj}^{(\ell)}} \right) \cdot \theta'(s_{nj}^{(\ell)})$$

$$= \left(\sum_k \delta_{nk}^{(\ell+1)} w_{jk}^{(\ell+1)} \right) \cdot \theta'(s_{nj}^{(\ell)}) = \left(\sum_k \delta_{nk}^{(\ell+1)} \left(w_{kj}^{(\ell+1)} \right)^T \right) \cdot \theta'(s_{nj}^{(\ell)})$$

$$\Rightarrow \boxed{\delta^{(\ell)} = \left(\delta^{(\ell+1)} \left[W^{(\ell+1)} \right]^T \right) \otimes \theta'(s^{(\ell)})}$$

in the hidden layer for ReLu.

$\theta'(s^{(\ell)}) = \text{sign}(s^{(\ell)})$



SEEDING the SENSITIVITY

What is $\delta^{(L)}$? L = output layer.

$$\delta_{nj}^{(L)} = \frac{\partial E_{in}}{\partial s_{nj}^{(L)}} = \sum_k \frac{\partial E_{in}}{\partial x_{nk}^{(L)}} \cdot \frac{\partial x_{nk}^{(L)}}{\partial s_{nj}^{(L)}}$$

(Note: unlike for hidden layer activation, the softmax depends on all $s_{nj}^{(L)}$ $\rightarrow \therefore$ we need branching chain rule)

*Notice that the softmax activation depends on all $s_j^{(L)}$

$$E_{in} = -\frac{1}{N} \sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} \ln(x_{nk}^{(L)})$$

$$\frac{\partial E_{in}}{\partial x_{nk}^{(L)}} = -\frac{1}{N} \frac{y_{nk}}{x_{nk}^{(L)}} \Rightarrow \frac{\partial E_{in}}{\partial X^{(L)}} = \overbrace{-\frac{1}{N} \frac{Y}{X^{(L)}}}^{\text{for part 1.1.5}}$$

$$\frac{\partial x_{nk}^{(L)}}{\partial s_{nj}^{(L)}} = \frac{\partial}{\partial s_{nj}^{(L)}} \left(\sigma(s_{nk}^{(L)}) \right) = x_{nk}^{(L)} \left(\delta_{kj} - x_{nj}^{(L)} \right)$$

dirac delta.

$$\delta_{nj}^{(L)} = \sum_k \frac{\partial E_{in}}{\partial x_{nk}^{(L)}} \cdot \frac{\partial x_{nk}^{(L)}}{\partial s_{nj}^{(L)}}$$

$$= -\frac{1}{N} \sum_k \frac{y_{nk}}{x_{nk}^{(L)}} \cdot x_{nk}^{(L)} \left(\delta_{kj} - x_{nj}^{(L)} \right)$$

$$= -\frac{1}{N} \sum_k \left(y_{nk} \delta_{kj} - y_{nk} x_{nj}^{(L)} \right)$$

$$= -\frac{1}{N} \left(\sum_k y_{nk} \delta_{kj} - x_{nj}^{(L)} \underbrace{\sum_k y_{nk}}_{\text{is always } = 1 \text{ b/c one-hot encoded.}} \right)$$

$$\Rightarrow \delta^{(L)} = -\frac{1}{N} \left(Y \cdot I - X^{(L)} \right)$$

$$\boxed{\delta^{(L)} = \frac{1}{N} \left(X^{(L)} - Y \right)}$$

For the specific 2-Layer Neural Network we are asked to implement...

$$\begin{cases} \dfrac{\partial L}{\partial W_o} = \dfrac{\partial E_{in}}{\partial W^{(2)}} = [X^{(1)}]^T \delta^{(2)} \\[2ex] \dfrac{\partial L}{\partial b_o} = \dfrac{\partial E_{in}}{\partial \underline{b}^{(2)}} = \underline{1}^T \delta^{(2)} \end{cases}$$

$$\begin{cases} \delta^{(2)} = \dfrac{1}{N}(X^{(2)} - Y) \\[2ex] \delta^{(1)} = (\delta^{(2)} [W^{(2)}]^T) \otimes \text{Sign}(S^{(1)}) \end{cases}$$

$$\begin{cases} \dfrac{\partial L}{\partial W_h} = \dfrac{\partial E_{in}}{\partial W^{(1)}} = [X^{(0)}]^T \delta^{(1)} \\[2ex] \dfrac{\partial L}{\partial b_h} = \dfrac{\partial E_{in}}{\partial \underline{b}^{(1)}} = \underline{1}^T \delta^{(1)} \end{cases}$$