

CSDS 440: Machine Learning

Soumya Ray (he/him, sray@case.edu)

Olin 516

Office hours T, Th 11:15-11:45 or by appointment

Announcements

- Test 2 next week 10/17
- Start thinking about class projects with group
 - If you have a project idea, please send a note on slack to set up a meeting to discuss it
 - Details on default projects to be announced next week

Training Phase (ANN)

- As before, given a training sample and their class labels

$$D = \begin{pmatrix} x_{11} & \cdots & x_{1n} & -1 & y_1 \\ \vdots & & \vdots & \vdots & \vdots \\ x_{m1} & \cdots & x_{mn} & -1 & y_m \end{pmatrix}$$

- Find parameters \mathbf{W}

- To minimize “loss” $L(\mathbf{w})$

Activation Functions

- Sigmoid

$$h(\mathbf{x}; \mathbf{w}) = (1 + e^{-\mathbf{w} \cdot \mathbf{x}})^{-1}$$

- Radial Basis Function

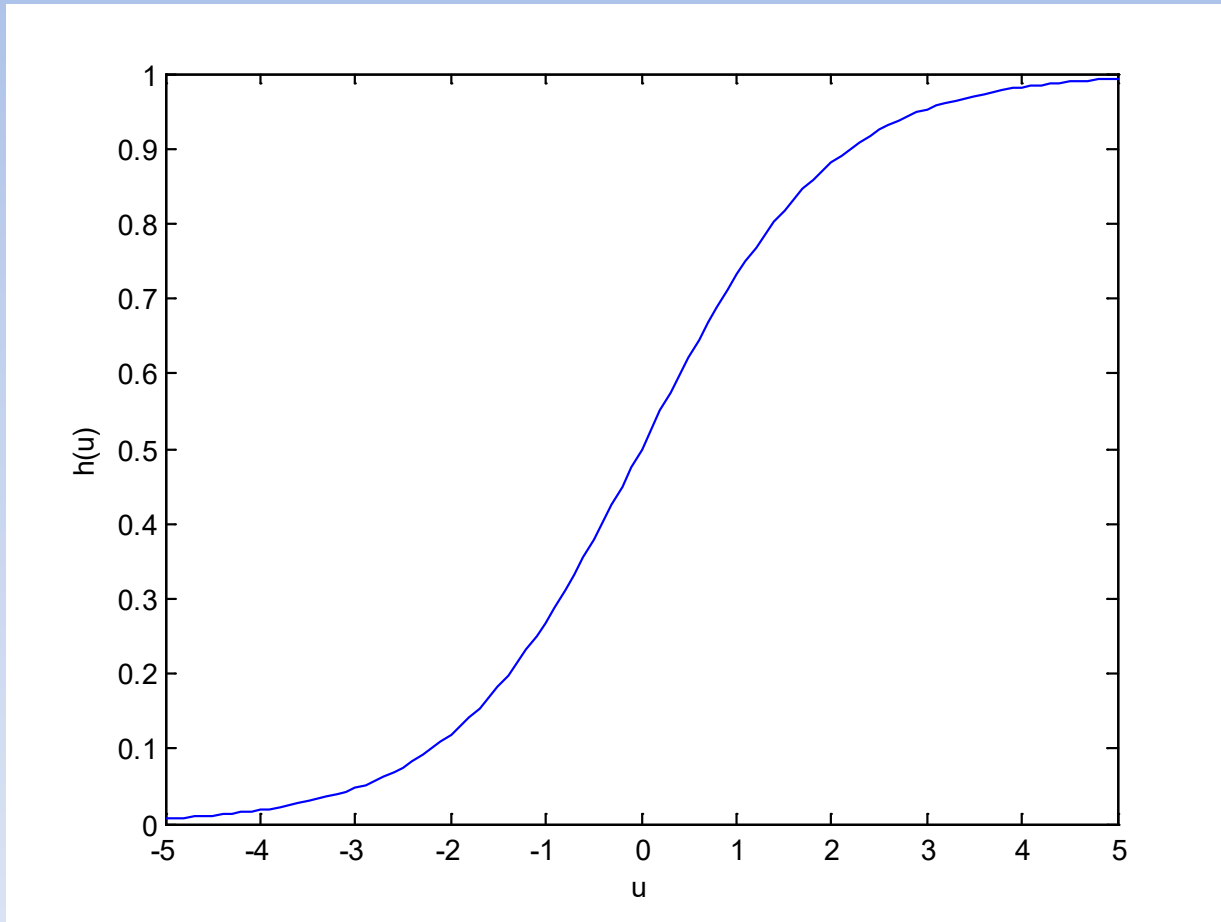
$$h(\mathbf{x}; \mathbf{w}, \mathbf{c}, \beta) = e^{\beta \|\mathbf{w} \cdot \mathbf{x} - \mathbf{c}\|^2}$$

- Hyperbolic Tangent

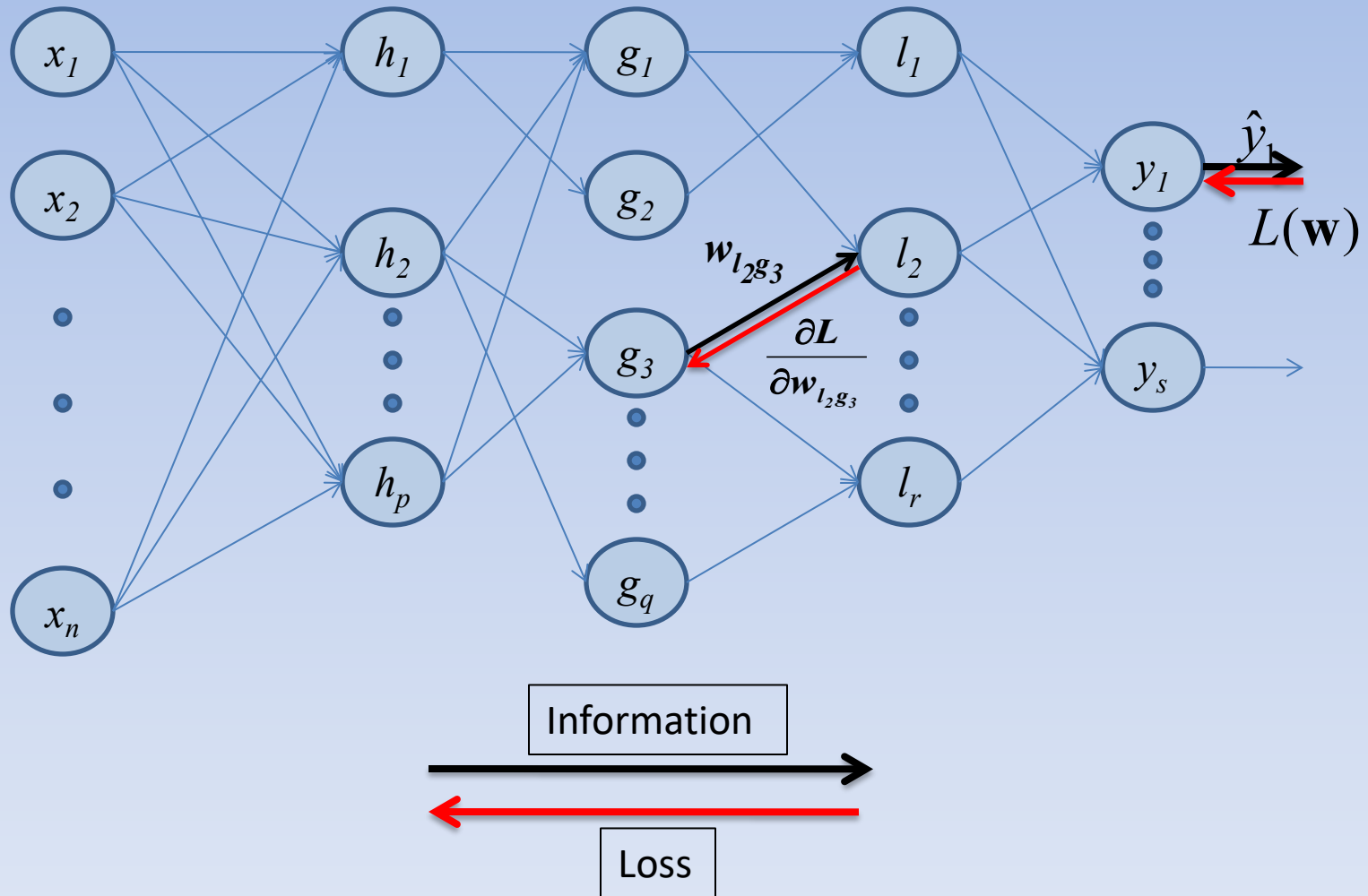
$$h(\mathbf{x}; \mathbf{w}, a, b) = a \frac{(e^{b\mathbf{w} \cdot \mathbf{x}} - e^{-b\mathbf{w} \cdot \mathbf{x}})}{(e^{b\mathbf{w} \cdot \mathbf{x}} + e^{-b\mathbf{w} \cdot \mathbf{x}})}$$

Activation Function: Sigmoid

$$h(u) = (1 + e^{-u})^{-1}$$

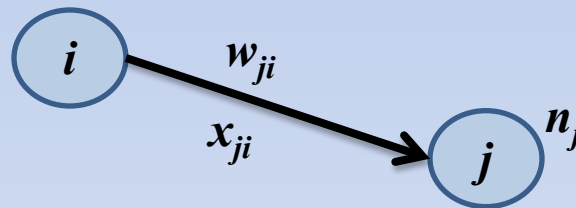


Backpropagation



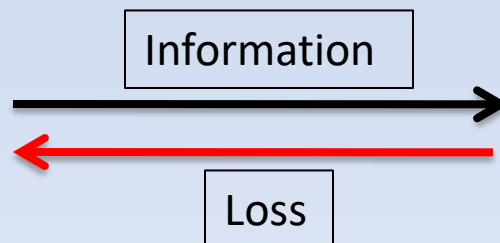
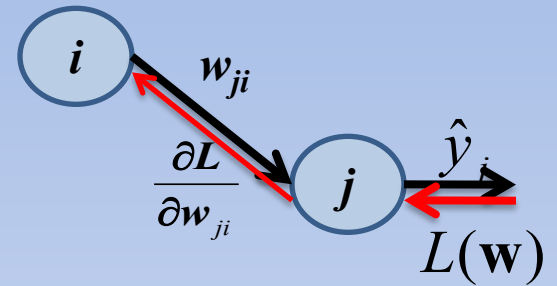
Backpropagation (SGD)

- Let x_{ji} be the i^{th} input to unit j
- Let w_{ji} be the parameter associated with x_{ji}
- Let $n_j = \sum_i w_{ji} x_{ji}$ be the “net input” to unit j



- Observe that
$$\frac{\partial L}{\partial w_{ji}} = \frac{\partial L}{\partial n_j} \frac{\partial n_j}{\partial w_{ji}} = \frac{\partial L}{\partial n_j} x_{ji}$$

Output Layer



Derivation (output layer)

$$h(u) = \frac{1}{(1 + e^{-u})}; 1 - h(u) = \frac{e^{-u}}{(1 + e^{-u})} \quad (\text{Sigmoid})$$

$$\frac{dh}{du} = \frac{e^{-u}}{(1 + e^{-u})^2} = h(u)(1 - h(u)) \quad (\text{Derivative of Sigmoid})$$

$$L(w_{ji}) = \frac{1}{2} (y_j - h(n_j))^2 \quad (\text{Squared Loss})$$

$$\frac{\partial L}{\partial n_j} = (h(n_j) - y_j) \frac{\partial h(n_j)}{\partial n_j}$$

$$\frac{\partial h(n_j)}{\partial n_j} = h(n_j)(1 - h(n_j)) \quad (\text{Using Derivative of Sigmoid})$$

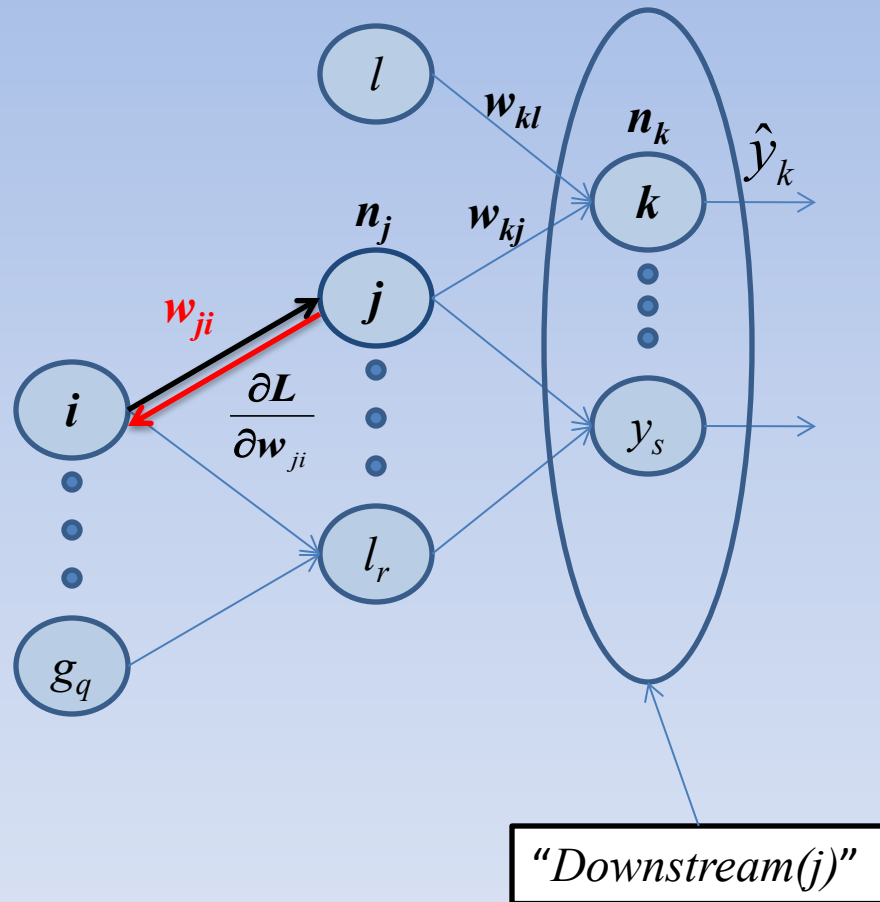
Derivation (output layer)

$$\frac{\partial L}{\partial n_j} = (h(n_j) - y_j) \frac{\partial h(n_j)}{\partial n_j}$$

$$\frac{\partial h(n_j)}{\partial n_j} = h(n_j)(1 - h(n_j))$$

$$\frac{\partial L}{\partial w_{ji}} = (h(n_j) - y_j) h(n_j)(1 - h(n_j)) x_{ji}$$

Hidden Layer




Derivation (Hidden Layer)

- Since j affects the output only through $Downstream(j)$,

$$\frac{\partial L}{\partial n_j} = \sum_{k \in Downstream(j)} \frac{\partial L}{\partial n_k} \frac{\partial n_k}{\partial n_j}$$

Already calculated,
next layer



$$n_k = \sum_l w_{kl} h(n_l); \frac{\partial n_k}{\partial n_j} = \frac{\partial (w_{kj} h(n_j))}{\partial n_j}$$

$$= w_{kj} \frac{\partial h(n_j)}{\partial n_j} = w_{kj} h(n_j)(1 - h(n_j))$$

$$\frac{\partial L}{\partial n_j} = h(n_j)(1 - h(n_j)) \sum_{k \in Downstream(j)} \frac{\partial L}{\partial n_k} w_{kj}$$

Derivation (Hidden Layer)

$$\frac{\partial L}{\partial w_{ji}} = \frac{\partial L}{\partial n_j} x_{ji}$$

$$= h(n_j)(1 - h(n_j))x_{ji} \sum_{k \in \text{Downstream}(j)} \frac{\partial L}{\partial n_k} w_{kj}$$

$$= h(n_j)(1 - h(n_j))x_{ji} \sum_{k \in \text{Downstream}(j)} \frac{\partial L}{\partial w_{kj}} \frac{w_{kj}}{x_{kj}}$$

Review

- Consider a neural network with 2 input units, 2 hidden units and 1 output unit and all weights initialized to 1, with the bias set to zero. Using squared loss, show the weights after the first backprop update with these examples.

x_1	x_2	f
0	0	0
0	1	1

Updates

$$\frac{\partial L}{\partial w_{oh}} = h(n_o)(1 - h(n_o))x_{oh}(h(n_o) - y_o)$$

$$\frac{\partial L}{\partial w_{hi}} = h(n_h)(1 - h(n_h))x_{hi} \sum_{k \in \text{Downstream}(h)} \frac{\partial L}{\partial w_{kh}} \frac{w_{kh}}{x_{kh}}$$

Example notes

- Zeros as inputs
- SGD effects
- Vanishing gradients