April 18

Announcements

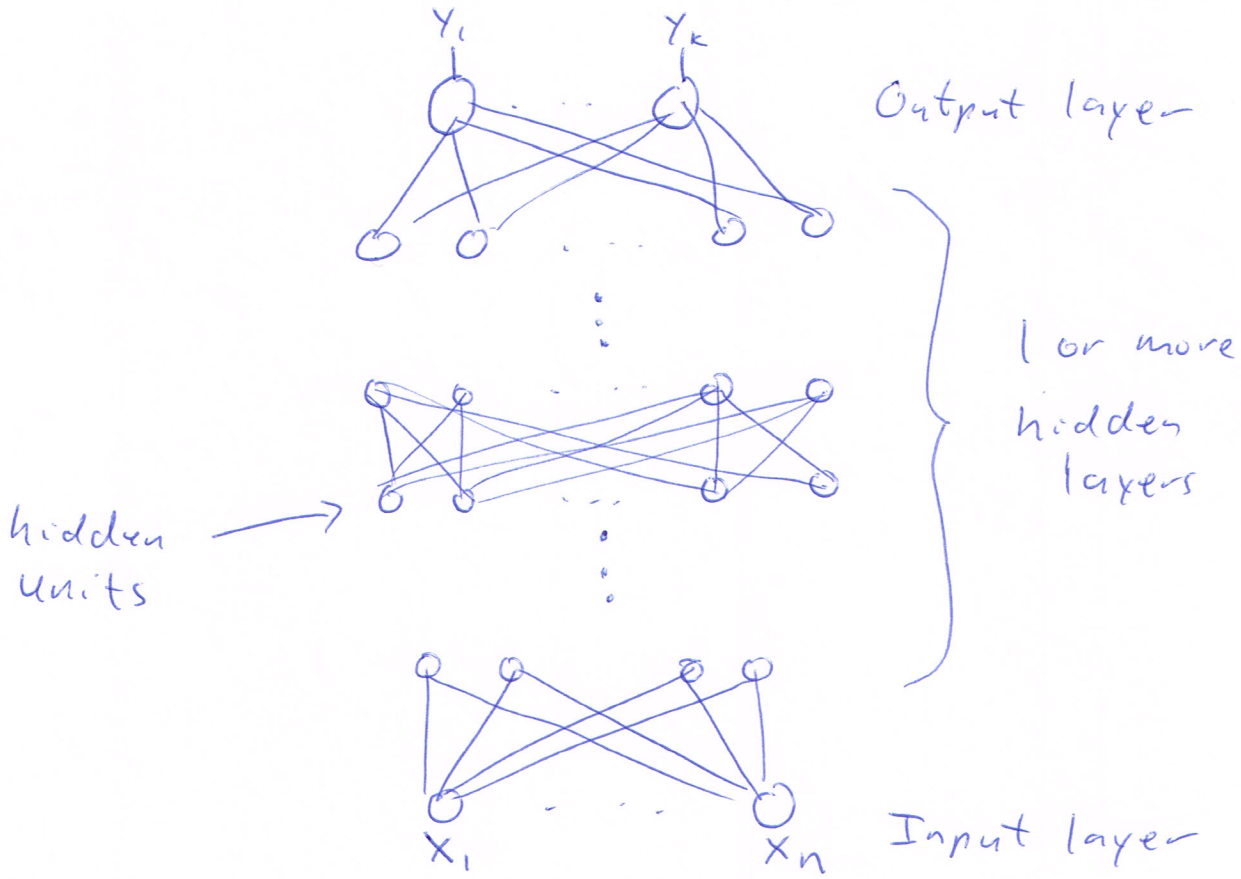
- HW3 late hw confusion
  till noon tomorrow (Wed) to (re)submit

Today: Textbook 18.6, 18.7

Multilayer neural nets

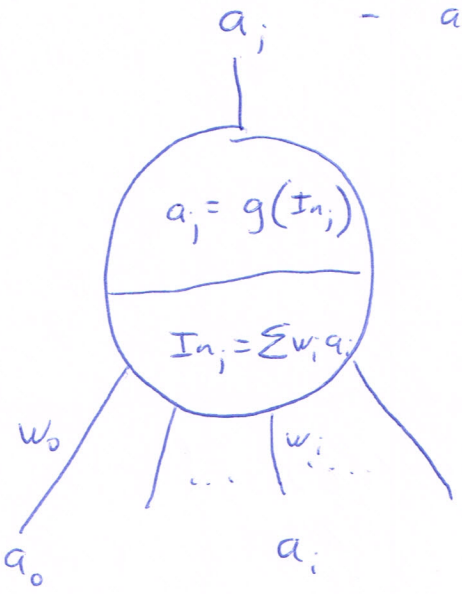
(Generalizes data to allow multiple outputs

Training data $D = (\bar{x}^1, y_1^1, y_2^1, \cdots, y_k^1),$
$(\bar{x}^2, y_1^2, y_2^2, \cdots, y_k^2),$
$\vdots$
$(\bar{x}^m, y_1^m, y_2^m, \cdots, y_k^m))$



Notation:

Single Unit:

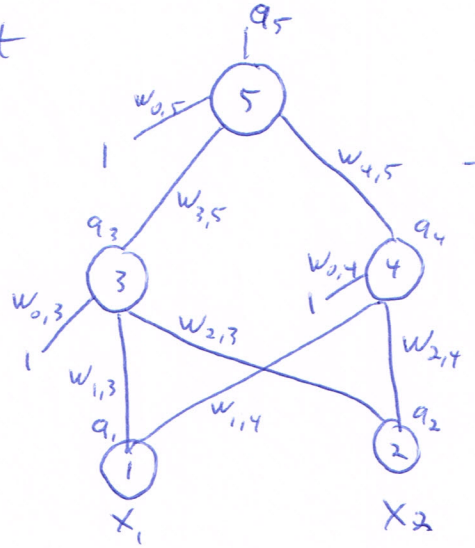


$g(x)$ = activation function

Example: Single perceptron $g(In) = \begin{cases} 1 \text{ if } In \geq 0 \\ 0 \text{ otherwise} \end{cases}$

This notation separates the weighted sum of the inputs from the precise threshold function

Multilayer net example:



— Notation: $w_{i,j}$ Weight between nodes i and j

$$In_3 = w_{0,3} \times 1 + w_{1,3} \times a_1 + w_{2,3} \times a_2$$

$$a_3 = g(In_3)$$

$$In_5 = w_{0,5} * 1 + w_{3,5} \times a_3 + w_{4,5} \times a_4$$

$a_3 = g(In_3)$ $\quad$ $a_4 = g(In_4)$

$$a_5 = g(In_5)$$

Stochastic Gradient Descent

$$w_{i,j} \leftarrow w_{i,j} - \alpha \frac{\partial Loss_{\bar{w}}}{\partial w_{i,j}}$$

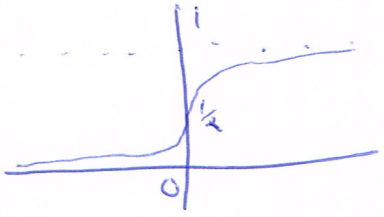
Backpropagation neural nets

With k outputs

$$Err_{\bar{w}}^{k}(\bar{x}) = (y_k - a_k)^2$$

$$Loss_{\bar{w}}(\bar{x}) = \sum_{i=1}^{k} Err_{\bar{w}}^{i}(\bar{x})^2$$

$$g(z) = \frac{1}{1+e^{-z}}$$



$$g'(z) = g(z)(1-g(z))$$

$\frac{\partial Loss_{\bar{w}}}{\partial w_{i,j}}$ depends on layer

Define $\Delta_k = Err_{\bar{w}}^{k} \times g'(In_k)$

for each output k

For all other layers, if j is connected to input of k

$$\Delta_j = g'(In_j) \sum_k w_{j,k} \Delta_k$$

Update rule:

$$w_{j,k} \leftarrow w_{j,k} + \alpha a_j \Delta_k$$

Backprop:

Set each $w_{i,j}$ to a random value

For $d=1$ to m <For each example>

FeedForward

For each node i in input layer

$a_i \leftarrow x_i^d$

For $l=2$ to # layers

For each node j in layer l

$In_j \leftarrow \sum_{i \text{ coming into } j} w_{i,j} a_i$

$a_j \leftarrow g(In_j)$

Backprop

For each node j in output layer

$\Delta_j \leftarrow g'(In_j) \times (y_j^d - a_j)$

For $l = $ # layers -1 to 1

For each node in layer l

$\Delta_i \leftarrow g'(In_i) \times \sum_{j \text{ that } i \text{ goes into}} w_{i,j} \Delta_j$

For each $w_{i,j}$ $w_{i,j} \leftarrow w_{i,j} + \alpha a_i \Delta_j$

<Repeat until stopping criterion>