Coursera Machine Learning: Weeks 5 & 6

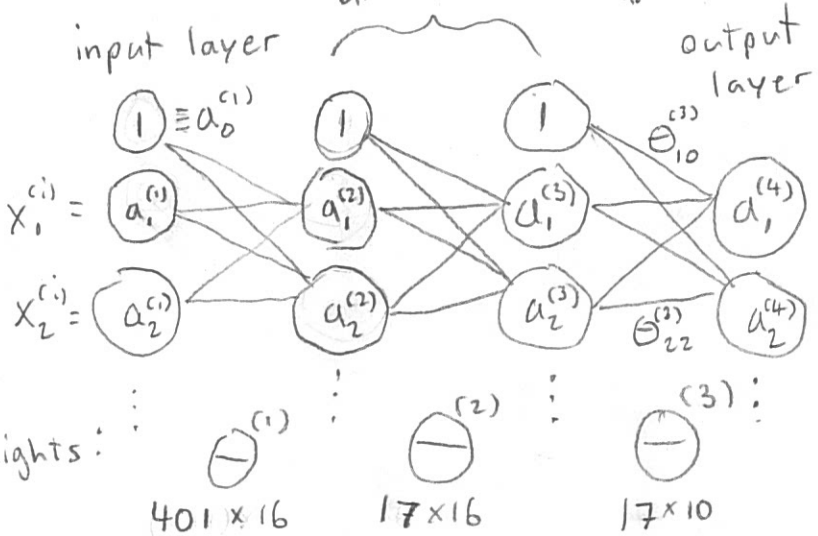
Week 5: Backpropagation

Review: Neural Networks

- Given a training set $(x^{(i)}, y^{(i)})$, adjust parameters Θ so that $p(h_\Theta(x^{(i)})) = y^{(i)}$

e.g. $x^{(i)}$ is a 20×20 pixel grayscale image: 9

$y^{(i)} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ 0 \end{bmatrix}$ ← 2, ← 9 → labeled as the digit "9"



$a^{(4)} = h_\Theta(x)$

prediction $p = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ 0 \end{bmatrix}$ ← index of $\max(h_\Theta(x))$

$x^{(i)} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \end{bmatrix}$ $a^{(1)} = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ \vdots \end{bmatrix} = [1, x^{(i)}]$

Forward Propagation: $a^{(1)} = [1; x^{(i)}]$, $a^{(2)} = [1; g(\Theta^{(1)} a^{(1)})]$,

$a^{(3)} = [1; g(\Theta^{(2)} a^{(2)})]$, $a^{(4)} = g(\Theta^{(3)} a^{(3)})$

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

Cost: $J(\Theta) = -\frac{1}{m} \sum_{i=1}^{m} \sum_{k=1} \left[y_k^{(i)} \log\left(h_\Theta(x^{(i)})\right) + \left(1 - y_k^{(i)}\right) \log\left(1 - h_\Theta(x^{(i)})\right) \right]$

$= -\frac{1}{m} \sum \sum \left[y_k \log\left(a_k^{(4)}\right) + \left(1 - y_k\right) \log\left(1 - a_k^{(4)}\right) \right]$

Backpropagation: Goal: Adjust $\Theta_{ij}^{(l)}$ to lower cost ($J(\theta)$)
by gradient descent $\Theta = \theta - \alpha \frac{\partial J}{\partial \theta}$

Problem: J seems to depend only on output layer $a^{(4)}$

Solution: Propagate cost back, layer by layer

Code:

for n in (# of iterations):

Set $\Delta_{ij}^{(l)} = 0$ for all (l,i,j)

for (x,y) in training data:

forward propagate x (compute $a^{(1)}$ then $a^{(2)}$ etc)

$\delta^{(4)} = a^{(4)} - y^{(4)}$ "error in layer 4"

$\delta^{(3)} = \Theta^{(3)T} \delta^{(4)} .* g'(\theta^{(2)} a^{(2)})$ "error in layer 3"

$\delta^{(2)} = \Theta^{(2)T} \delta^{(3)} .* g'(\theta^{(1)} a^{(1)})$ " " "

$\Delta_{ij}^{(l)} = \Delta_{ij}^{(l)} + \delta_i^{(l+1)} a_j^{(l)}$ for all (l,i,j)

end

$\frac{\partial J}{\partial \Theta_{ij}^{(l)}} = \frac{\Delta_{ij}^{(l)}}{m}$ for all (l,i,j)

$\Theta_{ij}^{(l)} = \Theta_{ij}^{(l)} - \alpha \frac{\partial J}{\partial \Theta_{ij}^{(l)}}$ for all (l,i,j)

end

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

Derivation: (with calculus i.e. the chain rule)

$\frac{\partial J}{\partial \Theta^{(3)}} = \sum_{l=4} \frac{\partial J}{\partial a^{(4)}} \frac{\partial a^{(4)}}{\partial \Theta^{(3)}}$

$\frac{\partial J}{\partial a^{(4)}} = \frac{1}{m} \frac{a^{(4)} - y}{a^{(4)}(1-a^{(4)})}$

$\frac{\partial a^{(4)}}{\partial \Theta^{(3)}} = \frac{\partial}{\partial \Theta^{(3)}} g(\Theta^{(3)} a^{(3)})$

$= a^{(3)} g'(\Theta^{(3)} a^{(3)})$

$= a^{(3)} a^{(4)}(1-a^{(4)})$

$= \sum_{l=4} \frac{1}{m} \left(\frac{a^{(4)} - y}{a^{(4)}(1-a^{(4)})} \right) \left(a^{(3)} a^{(4)} (1-a^{(4)}) \right)$

$= \sum_{l=4} \frac{1}{m} \delta^{(4)} a^{(3)}$

$$\frac{\partial J}{\partial \theta^{(2)}} = \sum_{l=3,4} \frac{\partial J}{\partial a^{(4)}} \frac{\partial a^{(4)}}{\partial a^{(3)}} \frac{\partial a^{(3)}}{\partial \theta^{(2)}}$$

$$= \sum \frac{1}{m} \left(\frac{a^{(4)} - y}{a^{(4)}(1-a^{(4)})} \right) \left(\theta^{(3)} a^{(4)}(1-a^{(4)}) \right) \times \left(a^{(2)} g'(\theta^{(2)} a^{(2)}) \right)$$

$$= \sum \frac{1}{m} \delta^{(4)} \theta^{(3)} a^{(2)} g'(\theta^{(2)} a^{(2)})$$

$$= \sum \frac{1}{m} \delta^{(3)} a^{(2)}$$

$$\frac{\partial J}{\partial \theta^{(1)}} = \sum_{l=2,3,4} \frac{\partial J}{\partial a^{(4)}} \frac{\partial a^{(4)}}{\partial a^{(3)}} \frac{\partial a^{(3)}}{\partial a^{(2)}} \frac{\partial a^{(2)}}{\theta^{(1)}}$$

$$= \vdots$$

$$= \sum \frac{1}{m} \delta^{(2)} a^{(1)}$$

$$\frac{\partial a^{(4)}}{\partial a^{(3)}} = \frac{\partial}{\partial a^{(3)}} g(\theta^{(3)} a^{(3)}) = \theta^{(3)} g'(\theta^{(3)} a^{(3)})$$

$$= \theta^{(3)} a^{(4)}(1-a^{(4)})$$

$$\frac{\partial a^{(3)}}{\partial \theta^{(2)}} = \frac{\partial}{\partial \theta^{(2)}} g(\theta^{(2)} a^{(2)}) = a^{(2)} g'(\theta^{(2)} a^{(2)})$$

Useful Resource:

YouTube: 3Blue1Brown "Neural Networks"

$$\frac{\partial a_k^{(4)}}{\partial \theta_{ij}^{(3)}} = \frac{\partial}{\partial \theta_{ij}^{(3)}} \left(g\left(\sum_n \theta_{kn}^{(3)} a_n^{(3)}\right)\right)$$

$$= a_j^{(3)} g'\left(\sum_n \theta_{kn}^{(3)} a_n^{(3)}\right) \delta_{ik}$$

$$= a_j^{(3)} a_k^{(4)} \left(1 - a_k^{(4)}\right) \delta_{ik}$$

$$\frac{\partial J}{\partial a_k^{(4)}} = -\frac{y_k}{a_k^{(4)}} + \frac{(1-y_k)(-1)}{1-a_k}$$

$$= \frac{-y_k(1-a_k) + (1-y_k)a_k}{a_k(1-a_k)}$$

$$= \frac{a_k - y_k}{a_k(1-a_k)}$$

$$\frac{\partial J}{\partial \theta_{ij}^{(3)}} = \sum_k \frac{\partial J}{\partial a_k^{(4)}} \frac{\partial a_k^{(4)}}{\partial \theta_{ij}^{(3)}}$$

$$= \sum_k \frac{(a_k - y_k)}{a_k(1-a_k)} a_k(1-a_k)\delta_{ik} a_j^{(3)}$$

$$= (a_i^{(4)} - y_i^{(4)}) a_j^{(3)}$$

$$= \delta_i^{(4)} a_j^{(3)}$$

$$\frac{\partial a_k^{(4)}}{\partial a_p^{(3)}} = \frac{\partial}{\partial a_p^{(3)}} g\left(\sum_n \theta_{kn}^{(3)} a_n^{(3)}\right)$$

$$= \theta_{kp}^{(3)} g'\left(\sum_n \theta_{kn}^{(3)} a_n^{(3)}\right)$$

$$\frac{\partial a_p^{(3)}}{\partial \theta_{ij}^{(2)}} = a_j^{(2)} a_p^{(3)} \left(1 - a_p^{(3)}\right) \delta_{ip}$$

$$\frac{\partial J}{\partial \theta_{ij}^{(2)}} = \sum_k \sum_p \frac{\partial J}{\partial a_k^{(4)}} \frac{\partial a_k^{(4)}}{\partial a_p^{(3)}} \frac{\partial a_p^{(3)}}{\partial \theta_{ij}^{(2)}}$$

$$= \sum_k \sum_p \frac{(a_k^{(4)} - y_k^{(4)})}{a_k^{(4)}(1-a_k^{(4)})} \theta_{kp}^{(3)} a_k^{(4)}(1-a_k^{(4)}) a_j^{(2)} \underbrace{a_p^{(3)}\left(1-a_p^{(3)}\right)}_{g'(\Sigma \theta^{(2)} a^{(2)})} \delta_{ip}$$

$$= \sum_k \delta_k^{(4)} \theta_{ki}^{(3)} g'\left(\sum_n \theta_{in}^{(2)} a_n^{(2)}\right) a_j^{(2)}$$

$$= a_j^{(2)} \delta_i^{(3)}$$

$$\delta_i^{(3)} = \sum_k \theta_{ki}^{(3)} \delta_k^{(4)} g'\left(\sum_n \theta_{in}^{(2)} a_n^{(2)}\right)$$

$$\vec{\delta}^{(3)} = \Theta^{(3)T} \vec{\delta}^{(4)} .* g'\left(\Theta^{(2)} \vec{a}^{(2)}\right)$$

Week 6: Advice for Applying Machine Learning

Options to Improve an Algorithm

- increase # of training examples (m)
- reduce # of features $\begin{matrix} x_1 \\ \vdots \\ x_{100} \end{matrix} \rightarrow \begin{matrix} x_1 \\ \vdots \\ x_{50} \end{matrix}$
- increase # of features
- add polynomial features (increase d) $x_1, x_1^2, \dots x_1^d$
- decrease λ (regularization parameter) $\theta := \theta - \alpha \frac{\partial J}{\partial \theta} - \lambda\theta$
- increase λ
- add more layers

Fixes:	Underfitting (high bias)	Overfitting (high variance)
- increase # of training examples (m)		✓
- reduce # of features		✓
- increase # of features	✓	
- add polynomial features	✓	
- decrease λ	✓	
- increase λ		✓
- add more layers	✓	

Model Selection and Testing

Split data set:

e.g.

$x^{(1)}, y^{(1)}$ … $x^{(6)}$ } training set ~60%
- use to train algorithm
- algorithm will usually be most successful on this set

$x^{(7)}$ $x^{(8)}$ } cross validation set ~20%
- use to choose model parameters (d, λ, etc)

$x^{(9)}$ $x^{(10)}$ } test set ~20%
- check if model generalizes

Learning Curves