



deeplearning.ai

Optimization Algorithms

Mini-batch
gradient descent

Batch vs. mini-batch gradient descent

Vectorization allows you to efficiently compute on m examples.

$$X = \begin{bmatrix} x^{(1)} & x^{(2)} & x^{(3)} & \dots & x^{(1000)} & | & x^{(1001)} & \dots & x^{(2000)} & | & \dots & | & \dots & x^{(m)} \end{bmatrix}$$

(n_x, m) $\underbrace{\hspace{10em}}_{X^{\{1\}} \quad (n_x, 1000)}$ $\underbrace{\hspace{10em}}_{X^{\{2\}} \quad (n_x, 1000)}$ \dots $\underbrace{\hspace{10em}}_{X^{\{5,000\}} \quad (n_x, 1000)}$

$$Y = \begin{bmatrix} y^{(1)} & y^{(2)} & y^{(3)} & \dots & y^{(1000)} & | & y^{(1001)} & \dots & y^{(2000)} & | & \dots & | & \dots & y^{(m)} \end{bmatrix}$$

$(1, m)$ $\underbrace{\hspace{10em}}_{Y^{\{1\}} \quad (1, 1000)}$ $\underbrace{\hspace{10em}}_{Y^{\{2\}} \quad (1, 1000)}$ \dots $\underbrace{\hspace{10em}}_{Y^{\{5,000\}} \quad (1, 1000)}$

What if $m = \underline{5,000,000}$?

5,000 mini-batches of 1,000 each

Mini-batch t : $X^{\{t\}}, Y^{\{t\}}$

$$\left| \begin{array}{l} x^{(i)} \\ z^{[l]} \\ X^{\{t\}}, Y^{\{t\}} \end{array} \right.$$

Mini-batch gradient descent

repeat {
for $t = 1, \dots, 5000$ {

Forward prop on X^{t+1} .

$$Z^{(i)} = W^{(i)} X^{t+1} + b^{(i)}$$

$$A^{(i)} = g^{(i)}(Z^{(i)})$$

...

$$A^{(L)} = g^{(L)}(Z^{(L)})$$

} Vectorized implementation
(1000 examples)

Compute cost $J^{t+1} = \frac{1}{1000} \sum_{i=1}^L \ell(\hat{y}^{(i)}, y^{(i)}) + \frac{\lambda}{2 \cdot 1000} \sum_i \|W^{(i)}\|_F^2$.

↙ ↘ for X^{t+1}, Y^{t+1}

Backprop to compute gradients w.r.t J^{t+1} (using X^{t+1}, Y^{t+1})

$$W^{(i)} := W^{(i)} - \alpha dW^{(i)}, \quad b^{(i)} := b^{(i)} - \alpha db^{(i)}$$

}

"1 epoch"

└ pass through training set.

1 step of grad desc
using X^{t+1}, Y^{t+1} .
(as if $m=1000$)

X, Y