deeplearning.ai

Optimization Algorithms

Mini-batch gradient descent

Batch vs. mini-batch gradient descent

X, Y $X^{\{t\}}, Y^{\{t\}}$

Vectorization allows you to efficiently compute on *m* examples.

$$\underset{(n_x, m)}{X} = [\underbrace{x^{(1)}\ x^{(2)}\ x^{(3)}\ \dots\ x^{(1000)}}_{X^{\{1\}}\ (n_x, 1000)} \mid \underbrace{x^{(1001)}\ \dots\ x^{(2000)}}_{X^{\{2\}}\ (n_x, 1000)} \mid \dots \mid \underbrace{\dots\ x^{(m)}}_{X^{\{5,000\}}\ (n_x, 1000)}]$$

$$\underset{(1, m)}{Y} = [\underbrace{y^{(1)}\ y^{(2)}\ y^{(3)}\ \dots\ y^{(1000)}}_{Y^{\{1\}}\ (1, 1000)} \mid \underbrace{y^{(1001)}\ \dots\ y^{(2000)}}_{Y^{\{2\}}\ (1, 1000)} \mid \dots \mid \underbrace{\dots\ y^{(m)}}_{Y^{\{5,000\}}\ (1, 1000)}]$$

What if $m = 5{,}000{,}000$?
5,000 mini-batches of 1,000 each
Mini-batch t: $X^{\{t\}}, Y^{\{t\}}$

$x^{(i)}$
$z^{[l]}$
$X^{\{t\}}, Y^{\{t\}}$.

Andrew Ng

Mini-batch gradient descent

```
repeat {
for $t = 1, \dots, 5000$ {
    Forward prop on $X^{\{t\}}$.
        $Z^{[1]} = W^{[1]} X^{\{t\}} + b^{[1]}$
        $A^{[1]} = g^{[1]}(Z^{[1]})$
        ⋮
        $A^{[L]} = g^{[L]}(Z^{[L]})$
    Compute cost $J^{\{t\}} = \frac{1}{1000} \sum_{i=1}^{l} \mathcal{L}(\hat{y}^{(i)}, y^{(i)}) + \frac{\lambda}{2 \cdot 1000} \sum_{l} \|W^{[l]}\|_F^2$.
    Backprop to compute gradients wrt $J^{\{t\}}$ (using $(X^{\{t\}}, Y^{\{t\}})$)
    $W^{[l]} := W^{[l]} - \alpha dW^{[l]}$, $b^{[l]} := b^{[l]} - \alpha db^{[l]}$
}
}
```

Vectorized implementation (1000 examples)

from $X^{\{t\}}, Y^{\{t\}}$.

1 step of gradient descent using $\underline{X^{\{t\}}, Y^{\{t\}}}$. (as if $m = 1000$)

$\underline{X, Y}$

"1 epoch"
pass through training set.

Andrew Ng