

Week 2-1 Optimization Algo: Mini-batch GD

笔记本: DL 2 - Deep NN Hyperparameter Tuning, Regularization & Optimization

创建时间: 2021/1/9 11:06

更新时间: 2021/1/9 11:15

Mini-batch GD

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, m) $X^{\{1\}}$ $X^{\{2\}}$ $X^{\{5,000\}}$

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

$(1, m)$ $Y^{\{1\}}$ $Y^{\{2\}}$ $Y^{\{5,000\}}$

What if $m = 5,000,000$?

5,000 mini-batches of 1,000 each

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

$$\begin{bmatrix} x^{(i)} \\ z^{[l]} \\ \vdots \end{bmatrix}$$

Andrew Ng

Mini-batch gradient descent

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

Forward prop on $X^{\{t\}}$.

$$z^{[2]} = W^{[1]} X^{\{t\}} + b^{[2]}$$

$$A^{[2]} = \sigma^{[2]}(z^{[2]})$$

\vdots

$$A^{[L]} = \sigma^{[L]}(z^{[L]})$$

Vertical implementation
(1000 examples)

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

$$\text{Compute cost } J^{\{t\}} = \frac{1}{1000} \sum_{i=1}^L \mathcal{L}(\hat{y}^{(i)}, y^{(i)}) + \frac{\lambda}{2 \cdot 1000} \sum_{l=2}^L \|W^{[l]}\|_F^2$$

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

$$W^{[1]} = W^{[1]} - \alpha \Delta W^{[1]}, \quad b^{[2]} = b^{[2]} - \alpha \Delta b^{[2]}$$

}

"1 epoch"

pass through training set.

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

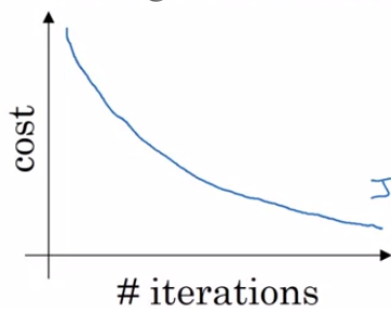
$$X, Y$$

Andrew Ng

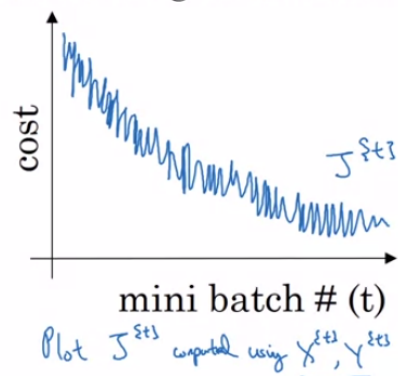
epoch

Training with mini batch gradient descent

Batch gradient descent



Mini-batch gradient descent

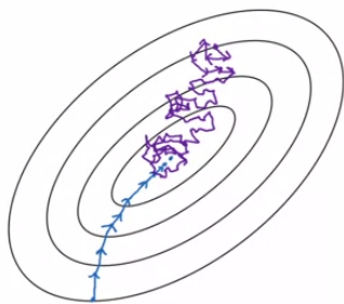


Choosing your mini-batch size

→ If mini-batch size = m : Batch gradient descent. $(X^{t+1}, Y^{t+1}) = (X, Y)$

→ If mini-batch size = 1 : Stochastic gradient descent. Every example is its own $(X^{t+1}, Y^{t+1}) = (x^{(1)}, y^{(1)}) \dots (x^{(m)}, y^{(m)})$ mini-batch.

In practice: Search in-between 1 and m



Stochastic
gradient
descent
{
Use speaker
for visualization

In-between
(mini-batch size
not too big/small)
↓
Fastest learning.
• Vectorization.
($n > 1000$)
• Make passes without
processing entire training set.

Batch
gradient descent
(mini-batch size = m)
↓
Too long
per iteration

Andrew Ng

typical mini-batch size

Choosing your mini-batch size

If small training set : Use batch gradient descent.
($m \leq 2000$)

Typical mini-batch sizes:

64 , 128 , 256 , 512 1024
 2^6 2^7 2^8 2^9 2^{10}

Make sure mini-batch fits in CPU/GPU memory.
 X^{t+1}, Y^{t+1} .

