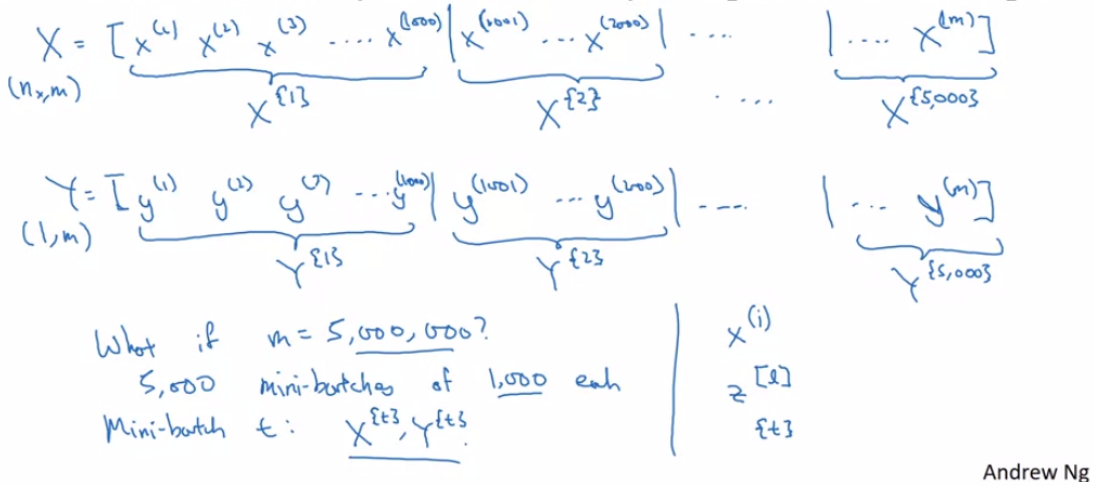
Week 2-1 Optimization Algo: Mini-batch GD

笔记本: DL 2 - Deep NN Hyperparameter Tunning, Regularization & Optimization
创建时间: 2021/1/9 11:06 **更新时间:** 2021/1/9 11:15

Mini-batch GD

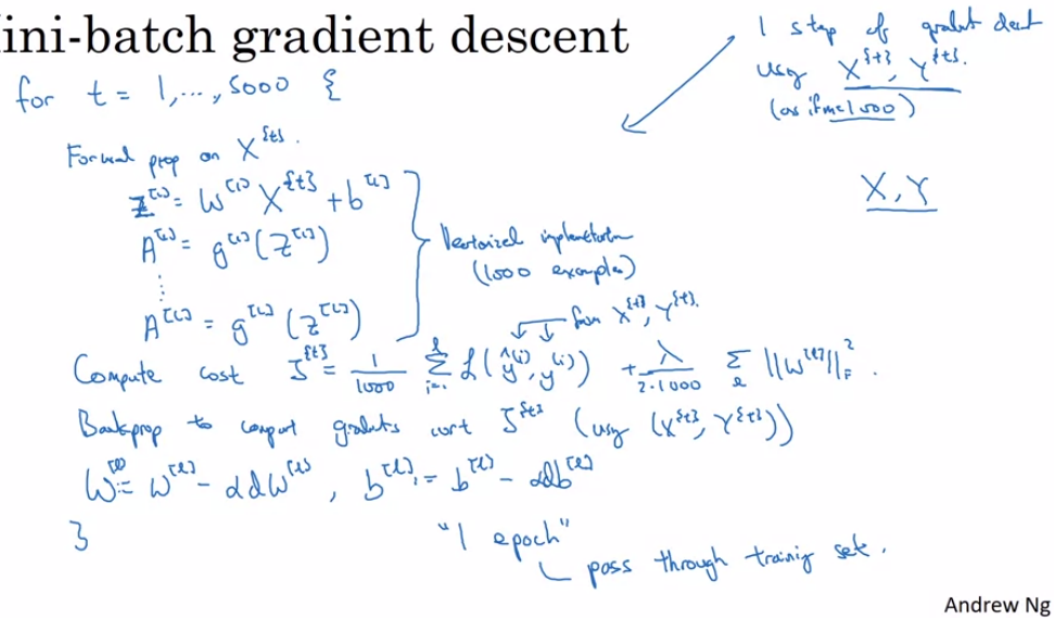
Batch vs. mini-batch gradient descent

Vectorization allows you to efficiently compute on m examples.



Andrew Ng

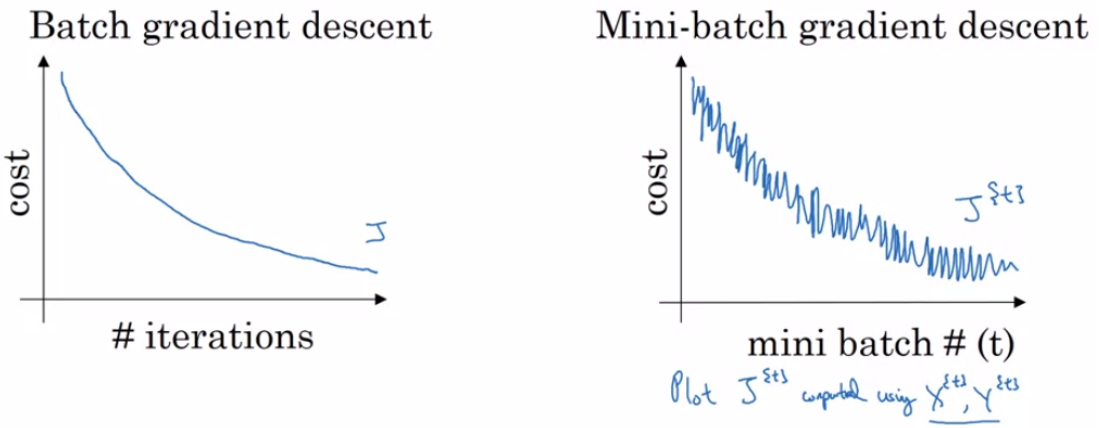
Mini-batch gradient descent



Andrew Ng

epoch

Training with mini batch gradient descent

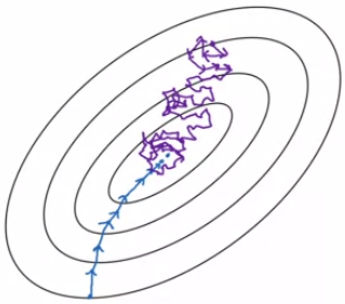


Choosing your mini-batch size

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

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

In practice: Somewhere in-between 1 and m



Stochastic gradient descent
- Lose speedup from vectorization

In-between (mini-batch size not too big/small)
- Fastest learning.
- Vectorization. (n 1000)
- Make progress 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 fit in CPU/GPU memory.
$X^{\{t\}}, Y^{\{t\}}$.