Learning *Very* Large Data Sets

Apprentissage pour *Très* Grands Echantillons

Léon Bottou & Yann Le Cun
NEC Research Institute,
4 Independence Way,
Princeton NJ 08540, USA
Email: {leonb,yann}@research.nj.nec.com

Large Learning System

Example:

- Computer monitors radio broadcasts for a few months and learns how to recognize speech.

Obstacles

- *Statement:* statistics, statistic learning theory,. . .
- *Engineering:* `http://lush.sourceforge.net`.
- *Algorithms:* learning algorithms do not scale well enough!

Learning algorithms do not scale well enough

Comparing computers in 1992 and 2002:
- Speed multiplied by 100
- Disk storage multiplied by 500+

Comparing large learning systems in 1992 and 2002:
- From 10^5 to 10^6 examples,
- From 10^5 to 10^6 parameters,

Very computer intensive attempts to improve upon these numbers (Bengio & Ducharme, 2001)

SVMs are not running in this race.
Boosting does (Drucker, 1993)

Online algorithms

Large data sets are best handled by online algorithms.

1994 Bottou, Cortes & al –
MNIST experiments.

1998 LeCun, Bottou, Bengio, Haffner –
Gradient-based Learning for Document Recognition.

1998 LeCun, Bottou, Müller, Orr –
Efficient learning.

How fast can online algorithms be?

Previous work

Comparing online and batch learning algorithms:

1970 Tsypkin & others –
Optimal (online) learning systems.

1997 Saad, Solla, Caticha –
Optimal online algorithms in statistical physics
(teacher network / student network).

1997 Murata, Amari –
Natural Gradient achieves Cramer-Rao bound
(maximum likelihood)

Our contribution

- A simple and general statement on the relative speed of online and batch learning algorithms.

- Answers about the scaling laws of learning algorithms.

Cost functions

- Many cost functions are sums/averages of many terms.

$$C_L(\theta) = \frac{1}{L}\sum_{i=1}^{L} L(z_i, \theta)$$

- There are typically as many terms as examples z_i.

- $L(z, \theta)$ is known as the Loss function.

  $J(z, \theta) = \frac{\partial}{\partial\theta} L(z, \theta)$ is known as the Jacobian.

Batch Learning

- Generic form of a batch learning algorithm:

$$\theta(t) = \theta(t-1) - \Phi_t \frac{1}{L} \sum_{i=1}^{L} J(z_i, \theta(t-1))$$

Each iteration involves a loop over all the terms.
All examples must be stored in memory beforehand.

- Superlinear convergence is achieved when the *rescaling matrix* Φ_t is well chosen.

$$(\theta(t) - \theta^*_L)^2 = \mathcal{O}\left(\frac{1}{e^{e^t}}\right)$$

Online Learning (1)

- Idea: Only use one random example Z_t per iteration.

- Generic form of an online learning algorithm:

$$\theta(t) = \theta(t-1) - \Phi_t \frac{1}{t} J(Z_t, \theta(t-1))$$

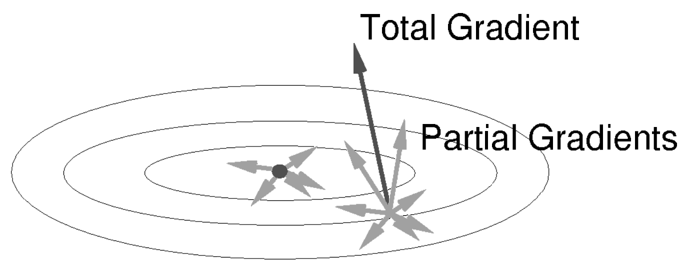
- No need to store examples beforehand.

- Converges *almost surely* to a local minimum.

- Note the *learning rate* $1/t$.

Online Learning (2)

- Residual noise depends on learning rate.



  Learning rate cannot decrease too fast.
  Optimal schedule is $1/t$.

- Consequence: $(\theta(t) - \theta^*)^2 = \mathcal{O}\left(\frac{1}{t}\right)$ at best!.

- Online learning seems hopelessly slow . . . but . . .

Generalization (1)

- Empirical error (i.e. training error)

$$C_L(\theta) = \frac{1}{L} \sum_{i=1}^{L} L(z_i, \theta)$$

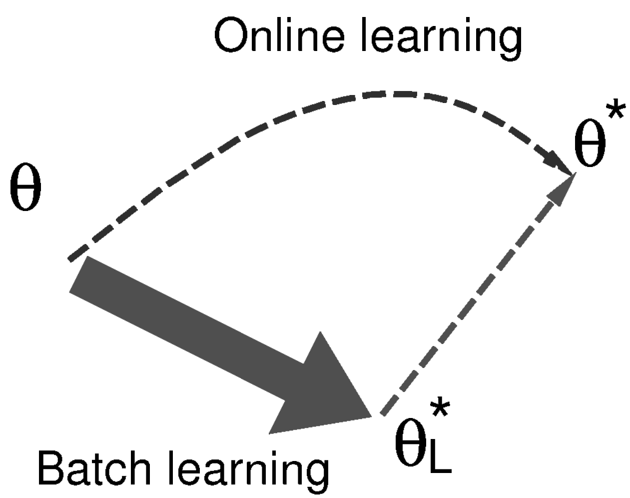
  Expected error (i.e. generalization error)

$$C_\infty(\theta) = \mathbf{E}\left(L(Z,\theta)\right) = \int L(Z,\theta)\, dp(Z)$$

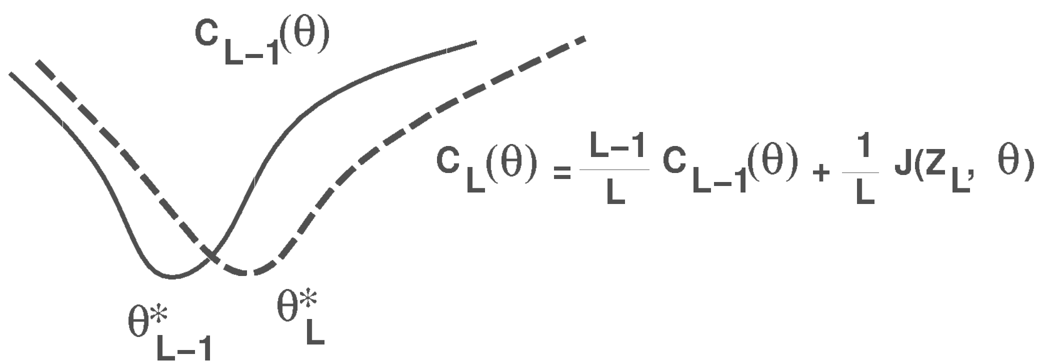
- Batch learning converges *quickly* to the optimum θ^*_L of the empirical error.

- Online learning converges to the optimum θ^* of the expected error.

Generalization (2)



Dynamics of the empirical optimum θ^*_L



- Simple expansion shows

$$\theta^*_L = \theta^*_{L-1} - \Psi_L \frac{1}{L} J(Z_L, \theta^*_{L-1}) + \mathcal{O}\left(\frac{1}{L^2}\right)$$

where Ψ_L converges to the inverse Hessian matrix

$$\Psi_L \longrightarrow H^{-1} \text{ with } H = \mathbf{E}\left(\frac{\partial^2}{\partial\theta^2} C(\theta^*)\right)$$

Compare...

- Convergence of the empirical optimum:

$$\theta^*_L = \theta^*_{L-1} - \Psi_L \frac{1}{L} J(Z_L, \theta^*_{L-1}) + \mathcal{O}\left(\frac{1}{L^2}\right)$$

- Online learning:

$$\theta(t) = \theta(t-1) - \Phi_t \frac{1}{t} J(Z_t, \theta(t-1))$$

Same thing?

Theorem

We consider the process

$$\theta_t = \theta_{t-1} - \Phi_t \frac{1}{L} J(Z_t, \theta_{t-1}) + \mathcal{O}\left(\frac{1}{t^2}\right)$$

with $\mathbf{E}\left(||\Phi_t - H^{-1}||\right) \to 0$ and many mild assumptions.

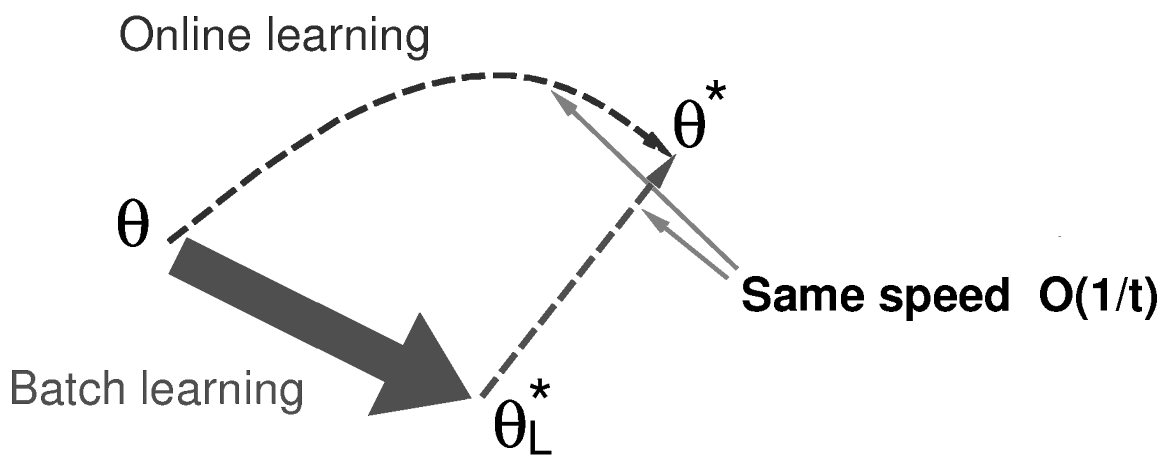
Then

$$\mathbf{E}\left((\theta_t - \theta^*)^2\right) = \frac{K}{t} + \mathrm{o}\left(\frac{1}{t}\right)$$

Remark: the constant K does not depend on the details.

$$K = \mathbf{tr}\left(H^{-1}\,\mathbf{E}\left(J(Z,\theta^*)J'(Z,\theta^*)\right)\,H^{-1}\right)$$

Corollary



Special Case: Maximum Likelihood

- Log loss $L(z,\theta) = -\log \phi_\theta(z)$

- Hessian H equals Fisher information matrix $\mathcal{I}(\theta^*)$.

We reach Cramer-Rao efficiency as soon as $\Phi_t \to \mathcal{I}^{-1}(\theta^*)$.

Example:
Natural Gradient achieves Cramer-Rao bound.
(Murata & Amari, 1998)

Conclusion (1)

A batch algorithm that optimizes the cost function faster than an efficient online algorithm. . .

. . . is just overfitting !

Conclusion (2)

We have a very large number of examples at hand.
Should we:

1. run an efficient online algorithm and process as many examples as we can?

2. run a superlinear batch algorithm on the largest set of examples we can process in the same time.

Answer: 1

Conclusion (3)

Learning N examples.
Fixed capacity.

	Memory	CPU
Efficient online learning	$\mathcal{O}(1)$	$\mathcal{O}(N)$
Superlinear batch	$\mathcal{O}(N)$	$\mathcal{O}(N \log \log N)$
SVM (!)	$\mathcal{O}\left(N^{2?}\right)$	$\mathcal{O}\left(N^{3?}\right)$

Future work

Assumption

$$\mathbf{E}\left(||\Phi_t - H^{-1}||\right) \to 0$$

means that Φ_t is a full rank matrix.

We do not want to handle this.

- Find a way to use reduced rank approximations.

- Find a way to make the Hessian H block-diagonal.