Using more data to speed-up training time

Shai Shalev-Shwartz

School of Computer Science and Engineering
The Hebrew University of Jerusalem

COST Workshop,
NIPS 2011

Based on joint work with

- Nati Srebro
- Ohad Shamir and Eran Tromer

Outline

- Time-Sample Complexity
- General Techniques:
  1. A larger hypothesis class
     - Formal Derivation of Gaps
  2. A different loss function
  3. Approximate optimization

Agnostic PAC Learning

- Domain Z (e.g. $Z = \mathcal{X} \times \mathcal{Y}$)
- Hypothesis class $\mathcal{H}$ (our "inductive bias")
- Loss function: $\ell : \mathcal{H} \times Z \to \mathbb{R}$
- $\mathcal{D}$ - unknown distribution over Z
- True risk: $L_{\mathcal{D}}(h) = \mathbb{E}_{z \sim \mathcal{D}}[\ell(h, z)]$
- Training set: $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m) \overset{\text{i.i.d.}}{\sim} \mathcal{D}^m$
- Goal: use S to find h_S s.t.

$$\mathop{\mathbb{E}}_{S} L_{\mathcal{D}}(h_S) \leq \min_{h \in \mathcal{H}} L_{\mathcal{D}}(h) + \epsilon$$

Joint Time-Sample Complexity

Goal:

$$\mathop{\mathbb{E}}_{S \sim \mathcal{D}^m} [L_{\mathcal{D}}(h_S)] \;\leq\; \min_{h \in \mathcal{H}} L_{\mathcal{D}}(h) \;+\; \epsilon$$

Joint Time-Sample Complexity

Goal:

$$\mathop{\mathbb{E}}_{S \sim \mathcal{D}^m} [L_{\mathcal{D}}(h_S)] \ \le \ \min_{h \in \mathcal{H}} L_{\mathcal{D}}(h) \ + \ \epsilon$$

- Sample complexity: How many examples are needed ?
- Time complexity: How much time is needed ?

Joint Time-Sample Complexity

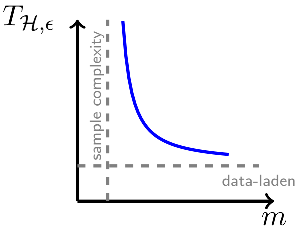
Goal:

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}(h_S)] \;\le\; \min_{h\in\mathcal{H}} L_{\mathcal{D}}(h) \;+\; \epsilon$$

- Sample complexity: How many examples are needed ?
- Time complexity: How much time is needed ?

Time-sample complexity

$T_{\mathcal{H},\epsilon}(m) =$ how much time is needed when $|S| = m$?



Joint Time-Sample Complexity

- Decatur, Goldreich, Ron 1998: "Computational Sample Complexity"
  - Only distinguishes polynomial vs. non-polynomial
  - Only binary classification in the realizable case
  - Very few results on "real-world" problems, e.g. Rocco Servedio showed gaps for 1-decision lists
- Bottou & Bousquet 2008: "The Tradeoffs of Large Scale Learning"
  - Study the effect of *optimization error* in generalized linear problems based on upper bounds

How Can More Data Reduce Runtime?

1. A larger hypothesis class
2. A different loss function
3. Approximate optimization

Example: Agnostic learning Preferences

The Learning Problem:

- $\mathcal{X} = [d] \times [d]$, $\mathcal{Y} = \{0, 1\}$, $Z = \mathcal{X} \times \mathcal{Y}$
- Given $(i, j) \in \mathcal{X}$ predict if i is preferable over j
- $\mathcal{H}$ is all permutations over $[d]$
- Loss function $=$ zero-one loss

Example: Agnostic learning Preferences

The Learning Problem:

- $\mathcal{X} = [d] \times [d]$, $\mathcal{Y} = \{0, 1\}$, $Z = \mathcal{X} \times \mathcal{Y}$
- Given $(i, j) \in \mathcal{X}$ predict if i is preferable over j
- $\mathcal{H}$ is all permutations over $[d]$
- Loss function = zero-one loss

Method I:

- $\text{ERM}_{\mathcal{H}}$
- Sample complexity is $\frac{d \log(d)}{\epsilon^2}$

Example: Agnostic learning Preferences

The Learning Problem:

- $\mathcal{X} = [d] \times [d]$, $\mathcal{Y} = \{0, 1\}$, $Z = \mathcal{X} \times \mathcal{Y}$
- Given $(i, j) \in \mathcal{X}$ predict if i is preferable over j
- $\mathcal{H}$ is all permutations over $[d]$
- Loss function $=$ zero-one loss

Method I:

- $\mathrm{ERM}_{\mathcal{H}}$
- Sample complexity is $\frac{d \log(d)}{\epsilon^2}$
- Varun Kanade and Thomas Steinke (2011): If RP$\neq$NP, it is not possible to efficiently find an ϵ-accurate permutation

Example: Agnostic learning Preferences

The Learning Problem:

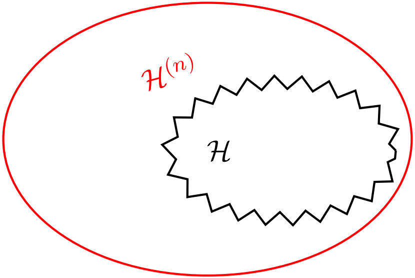
- $\mathcal{X} = [d] \times [d]$, $\mathcal{Y} = \{0, 1\}$, $Z = \mathcal{X} \times \mathcal{Y}$
- Given $(i, j) \in \mathcal{X}$ predict if i is preferable over j
- $\mathcal{H}$ is all permutations over $[d]$
- Loss function = zero-one loss

Method I:

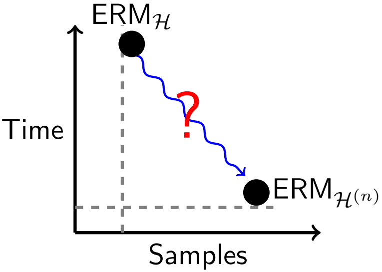
- $\mathrm{ERM}_{\mathcal{H}}$
- Sample complexity is $\frac{d \log(d)}{\epsilon^2}$
- Varun Kanade and Thomas Steinke (2011): If RP≠NP, it is not possible to efficiently find an ϵ-accurate permutation
- Claim: If $m \geq d^2/\epsilon^2$ it is possible to find a predictor with error $\leq \epsilon$ in polynomial time

Example: Agnostic learning Preferences

- Let $\mathcal{H}^{(n)}$ be the set of all functions from $\mathcal{X}$ to $\mathcal{Y}$
- $\text{ERM}_{\mathcal{H}^{(n)}}$ can be computed efficiently
- Sample complexity: $VC(\mathcal{H}^{(n)})/\epsilon^2 = d^2/\epsilon^2$
- Improper learning



More Data Less Work



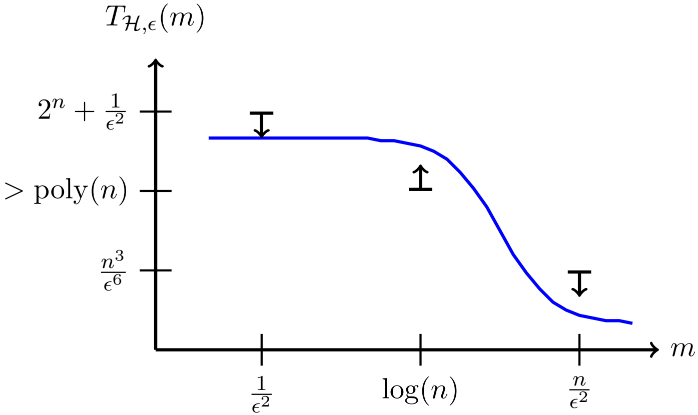
	Samples	Time
$\text{ERM}_{\mathcal{H}}$	$d\log(d)$	$d!$
$\text{ERM}_{\mathcal{H}^{(n)}}$	d^2	d^2

Lower bounds ?

- Analysis is based on upper bounds
- Is it possible to (improperly) learn efficiently with $d \log(d)$ examples ? (Posed as an open problem by Jake Abernathy)
- Main open problem: establish gaps by deriving lower bounds (for improper learning!)

Formal Derivation of Gaps

Theorem: Assume one-way permutations exist, there exists an agnostic learning problem such that:



Proof: One Way Permutations

$P : \{0,1\}^n \rightarrow \{0,1\}^n$ is one-way permutation if it's one-to-one and

- It is easy to compute $\mathbf{w} = P(\mathbf{s})$
- It is hard to compute $\mathbf{s} = P^{-1}(\mathbf{w})$

Proof: One Way Permutations

$P : \{0,1\}^n \to \{0,1\}^n$ is one-way permutation if it's one-to-one and

- It is easy to compute $\mathbf{w} = P(\mathbf{s})$
- It is hard to compute $\mathbf{s} = P^{-1}(\mathbf{w})$

Goldreich-Levin Theorem: If P is one way, then for any algorithm A,

$$\exists \mathbf{w} \text{ s.t. } \mathop{\mathbb{P}}_{\mathbf{r}}[A(\mathbf{r}, P(\mathbf{w})) = \langle \mathbf{r}, \mathbf{w} \rangle] < \frac{1}{2} + \frac{1}{\text{poly}(n)}$$

Proof: The Learning Problem

The Domain

- Let P be a one-way permutation.
- $\mathcal{X} = \{0,1\}^{2n}, \mathcal{Y} = \{0,1\}$
- Domain: $Z \subset \mathcal{X} \times \mathcal{Y}$
  - $((\mathbf{r},\mathbf{s}), b) \in Z$ iff $\langle P^{-1}(\mathbf{s}), \mathbf{r} \rangle = b$
- (Inner product over GF(2))

Proof: The Learning Problem

The Hypothesis Class

- $\mathcal{H} = \{h_{\mathbf{w}} : \mathbf{w} \in \{0,1\}^n\}$ where $h_{\mathbf{w}} : \mathcal{X} \to [0,1]$ is

$$h_{\mathbf{w}}(\mathbf{r},\mathbf{s}) = \begin{cases} \langle \mathbf{w}, \mathbf{r} \rangle & \text{if } \mathbf{s} = P(\mathbf{w}) \\ 1/2 & \text{o.w.} \end{cases}$$

The Loss Function:

- Absolute loss (= expected 0-1)

$$\ell(h, ((\mathbf{r},\mathbf{s}), b)) = |h(\mathbf{r},\mathbf{s}) - b|$$

Proof: The Learning Problem

The Hypothesis Class

- $\mathcal{H} = \{h_{\mathbf{w}} : \mathbf{w} \in \{0,1\}^n\}$ where $h_{\mathbf{w}} : \mathcal{X} \to [0,1]$ is

$$h_{\mathbf{w}}(\mathbf{r},\mathbf{s}) = \begin{cases} \langle \mathbf{w}, \mathbf{r} \rangle & \text{if } \mathbf{s} = P(\mathbf{w}) \\ 1/2 & \text{o.w.} \end{cases}$$

The Loss Function:

- Absolute loss (= expected 0-1)

$$\ell(h, ((\mathbf{r},\mathbf{s}), b)) = |h(\mathbf{r},\mathbf{s}) - b| = \begin{cases} 0 & \text{if } \mathbf{s} = P(\mathbf{w}) \\ 1/2 & \text{o.w.} \end{cases}$$

- Note: $L_{\mathcal{D}}(h_{\mathbf{w}}) = \mathbb{P}[\mathbf{s} \neq P(\mathbf{w})] \cdot \frac{1}{2}$

Proof: The Learning Problem

The Hypothesis Class

- $\mathcal{H} = \{h_{\mathbf{w}} : \mathbf{w} \in \{0,1\}^n\}$ where $h_{\mathbf{w}} : \mathcal{X} \to [0,1]$ is

$$h_{\mathbf{w}}(\mathbf{r}, \mathbf{s}) = \begin{cases} \langle \mathbf{w}, \mathbf{r} \rangle & \text{if } \mathbf{s} = P(\mathbf{w}) \\ 1/2 & \text{o.w.} \end{cases}$$

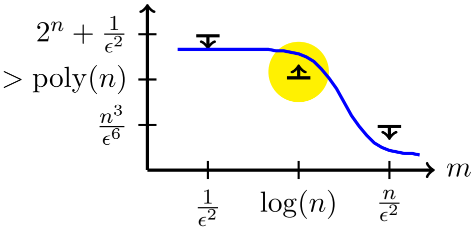
The Loss Function:

- Absolute loss (= expected 0-1)

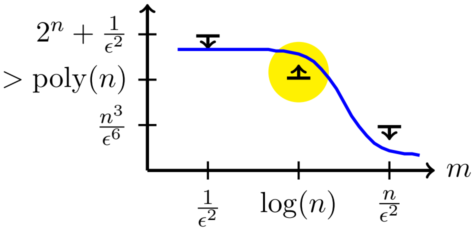
$$\ell(h, ((\mathbf{r}, \mathbf{s}), b)) = |h(\mathbf{r}, \mathbf{s}) - b| = \begin{cases} 0 & \text{if } \mathbf{s} = P(\mathbf{w}) \\ 1/2 & \text{o.w.} \end{cases}$$

- Note: $L_{\mathcal{D}}(h_{\mathbf{w}}) = \mathbb{P}[\mathbf{s} \neq P(\mathbf{w})] \cdot \frac{1}{2}$
- Agnostic: $L_{\mathcal{D}}(h_{\mathbf{w}}) = 0$ only if $\mathbb{P}[\mathbf{s} = P(\mathbf{w})] = 1$

Proof of Second Claim

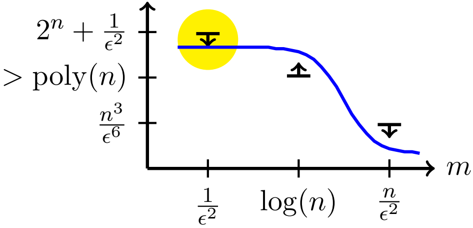


Proof of Second Claim



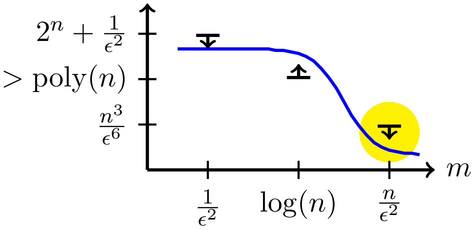
- Suppose we can learn with $m = O(\log(n))$ examples
- $\forall \mathbf{w}$, define $\mathcal{D}_{\mathbf{w}}$ s.t. $\mathbf{r}$ is uniform, $\mathbf{s} = P(\mathbf{w})$, and $b = \langle \mathbf{r}, \mathbf{w} \rangle$
- To generate an i.i.d. training set from $\mathcal{D}_{\mathbf{w}}$:
  - Pick $\mathbf{r}_1, \ldots, \mathbf{r}_m$ and $b_1, \ldots, b_m$ at random
  - If $b_i = \langle \mathbf{r}_i, \mathbf{w} \rangle$ for all i we're done
  - This happens w.p. $1/2^m = 1/\text{poly}(n)$
- Feed the training set to the learner, to get $h_{\mathbf{w}'}(\mathbf{r}, P(\mathbf{w})) \approx \langle \mathbf{r}, \mathbf{w} \rangle$
- Goldreich-Levin theorem $\Rightarrow$ contradiction

Proof of First Claim

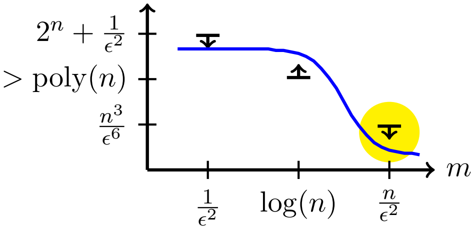


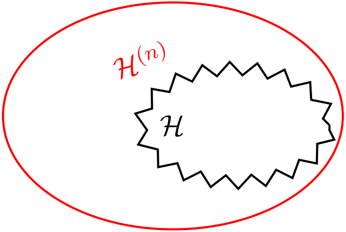
- Recall: $L_{\mathcal{D}}(h_{\mathbf{w}}) = \mathbb{P}[\mathbf{s} \neq P(\mathbf{w})] \cdot \frac{1}{2} = \mathbb{P}[P^{-1}(\mathbf{s}) \neq \mathbf{w}] \cdot \frac{1}{2}$
- Problem reduces to *multiclass* prediction with hypothesis class of constant predictors
- Sample complexity is $1/\epsilon^2$

Proof of Third Claim

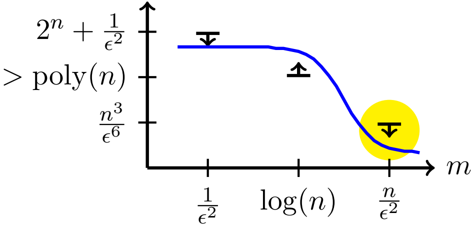


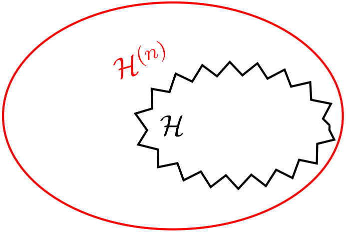
Proof of Third Claim





Proof of Third Claim





- Original class:

$$h_{\mathbf{w}}(\mathbf{r},\mathbf{s}) = \begin{cases} \langle \mathbf{w}, \mathbf{r} \rangle & \text{if } \mathbf{s} = P(\mathbf{w}) \\ 1/2 & \text{o.w.} \end{cases}$$

- New class:

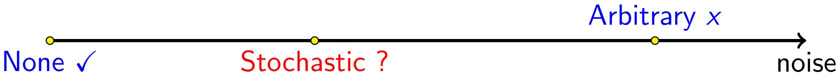
$$h_{((\mathbf{r}_1,\mathbf{s}'),b_1),\ldots,((\mathbf{r}_{m'},\mathbf{s}'),b_{m'})}(\mathbf{r},\mathbf{s}) = \begin{cases} \sum_i \alpha_i b_i & \text{if } \mathbf{r} = \sum_i \alpha_i \mathbf{r}_i \wedge \mathbf{s} = \mathbf{s}' \\ 1/2 & \text{o.w.} \end{cases}$$

- New class is efficiently learnable with $m = n/\epsilon^2$

Outline

- Time-Sample Complexity ✓
- General Techniques:
  1. A larger hypothesis class ✓
     - Formal Derivation of Gaps (for a synthetic problem) ✓
  2. A different loss function
  3. Approximate optimization

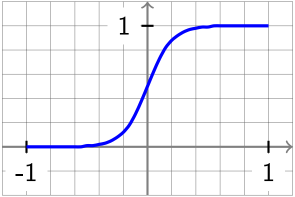
Example: Learning Margin-Based Halfspaces



- Without noise, can learn efficiently even if $m =$ sample complexity
- With arbitrary noise, cannot learn efficiently even if $m = \infty$ (S., Shamir, Sridharan 2010)
- What about stochastic noise ?

Learning Margin-Based Halfspaces with Stochastic Noise

$$\mathcal{H} = \{\mathbf{x} \mapsto \phi(\langle \mathbf{w}, \mathbf{x} \rangle) : \|\mathbf{w}\|_2 \leq 1\}, \quad \phi : \mathbb{R} \to [0,1] \text{ is } \tfrac{1}{\mu}\text{-Lipschitz}$$



- Probabilistic classifier: $\Pr[h_{\mathbf{w}}(\mathbf{x}) = 1] = \phi(\langle \mathbf{w}, \mathbf{x} \rangle)$
- Loss function: $\ell(\mathbf{w}; (\mathbf{x}, y)) = \Pr[h_{\mathbf{w}}(\mathbf{x}) \neq y] = |\phi(\langle \mathbf{w}, \mathbf{x} \rangle) - y|$
- Assumption: $\Pr[y = 1|\mathbf{x}] = \phi(\langle \mathbf{w}^\star, \mathbf{x} \rangle)$

Learning Halfspaces with Stochastic Noise

- Goal: find h s.t.

$$\mathbb{E}[|h(\mathbf{x}) - y|] - \mathbb{E}[|\phi(\langle \mathbf{w}^\star, \mathbf{x} \rangle) - y|] \le \epsilon \ .$$

Learning Halfspaces with Stochastic Noise

- Goal: find h s.t.

$$\mathbb{E}[|h(\mathbf{x}) - y|] - \mathbb{E}[|\phi(\langle \mathbf{w}^\star, \mathbf{x} \rangle) - y|] \le \epsilon \ .$$

- Idea: replace the loss function:

$$\begin{aligned} \mathbb{E}\,|h(\mathbf{x}) - y| - \mathbb{E}\,|\phi(\langle \mathbf{w}^\star, \mathbf{x} \rangle) - y| \\ &\le \mathbb{E}[|h(\mathbf{x}) - \phi(\langle \mathbf{w}^\star, \mathbf{x} \rangle)|] \\ &\le \sqrt{\mathbb{E}[(h(\mathbf{x}) - \phi(\langle \mathbf{w}^\star, \mathbf{x} \rangle))^2]} \end{aligned}$$

Learning Halfspaces with Stochastic Noise

- Goal: find h s.t.

$$\mathbb{E}[|h(\mathbf{x}) - y|] - \mathbb{E}[|\phi(\langle \mathbf{w}^\star, \mathbf{x}\rangle) - y|] \le \epsilon \ .$$

- Idea: replace the loss function:

$$\begin{aligned} \mathbb{E}\,|h(\mathbf{x}) - y| - \mathbb{E}\,|\phi(\langle \mathbf{w}^\star, \mathbf{x}\rangle) - y| \\ \le \mathbb{E}[|h(\mathbf{x}) - \phi(\langle \mathbf{w}^\star, \mathbf{x}\rangle|] \\ \le \sqrt{\mathbb{E}[(h(\mathbf{x}) - \phi(\langle \mathbf{w}^\star, \mathbf{x}\rangle))^2]} \end{aligned}$$

- Kalai-Sastry, Kakade-Kalai-Kanade-Shamir: The GLM-Tron algorithm learns h such that

$$\mathbb{E}[(h(\mathbf{x}) - \phi(\langle \mathbf{w}, \mathbf{x}\rangle))^2] \le O\left(\sqrt{\frac{1/\mu^2}{m}}\right)$$

Learning Halfspaces with Stochastic Noise

- Goal: find h s.t.

$$\mathbb{E}[|h(\mathbf{x}) - y|] - \mathbb{E}[|\phi(\langle \mathbf{w}^\star, \mathbf{x}\rangle) - y|] \le \epsilon \ .$$

- Idea: replace the loss function:

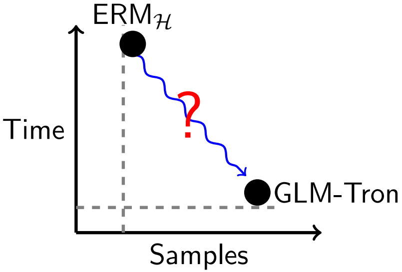
$$\begin{aligned}\mathbb{E}\,|h(\mathbf{x}) - y| - \mathbb{E}\,|\phi(\langle \mathbf{w}^\star, \mathbf{x}\rangle) - y| \\ &\le \mathbb{E}[|h(\mathbf{x}) - \phi(\langle \mathbf{w}^\star, \mathbf{x}\rangle|] \\ &\le \sqrt{\mathbb{E}[(h(\mathbf{x}) - \phi(\langle \mathbf{w}^\star, \mathbf{x}\rangle))^2]}\end{aligned}$$

- Kalai-Sastry, Kakade-Kalai-Kanade-Shamir: The GLM-Tron algorithm learns h such that

$$\mathbb{E}[(h(\mathbf{x}) - \phi(\langle \mathbf{w}, \mathbf{x}\rangle))^2] \le O\left(\sqrt{\frac{1/\mu^2}{m}}\right)$$

- Corollary: There is an efficient algorithm that learns Halfspaces with stochastic noise using $(1/(\mu\epsilon)^4)$ examples

More Data Less Work



	Samples	Time
ERM$_{\mathcal{H}}$	$\frac{1}{\mu^2\epsilon^2}$	$e^{\frac{1}{\mu\epsilon}}$
GLM-Tron	$\frac{1}{\mu^4\epsilon^4}$	$\frac{1}{\mu^4\epsilon^4}$

The General Technique

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(h_S)] - \min_{h \in \mathcal{H}} L_{\mathcal{D}}(h) \le f\left(\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}^{(n)}(h_S)] - \min_{h \in \mathcal{H}} L_{\mathcal{D}}^{(n)}(h)\right)$$

How Can More Data Reduce Runtime?

1. A larger hypothesis class ✓
2. A different loss function ✓
3. Approximate optimization

3-term error decomposition (Bottou & Bousquet)

$h^\star = \text{argmin}_{h \in \mathcal{H}} L_{\mathcal{D}}(h) \quad ; \quad h_S^\star = \text{argmin}_{h \in \mathcal{H}} L_S(h)$

3-term error decomposition (Bottou & Bousquet)

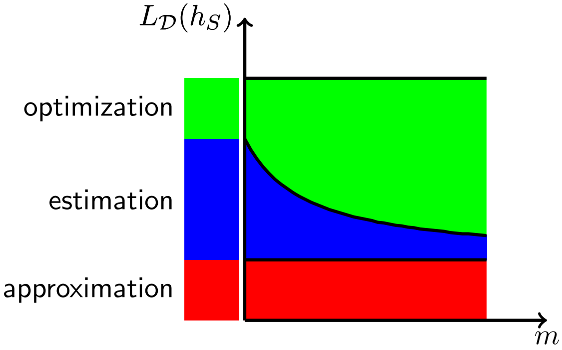
$$h^\star = \mathrm{argmin}_{h \in \mathcal{H}} L_{\mathcal{D}}(h) \quad ; \quad h_S^\star = \mathrm{argmin}_{h \in \mathcal{H}} L_S(h)$$

$$L_{\mathcal{D}}(h_S) = \underbrace{L_{\mathcal{D}}(h^\star)}_{\text{approximation}} + \underbrace{L_{\mathcal{D}}(h_S^\star) - L_{\mathcal{D}}(h^\star)}_{\text{estimation}} + \underbrace{L_{\mathcal{D}}(h_S) - L_{\mathcal{D}}(h_S^\star)}_{\text{optimization}}$$

3-term error decomposition (Bottou & Bousquet)

$$h^\star = \operatorname{argmin}_{h \in \mathcal{H}} L_{\mathcal{D}}(h) \quad ; \quad h_S^\star = \operatorname{argmin}_{h \in \mathcal{H}} L_S(h)$$

$$L_{\mathcal{D}}(h_S) = \underbrace{L_{\mathcal{D}}(h^\star)}_{\text{approximation}} + \underbrace{L_{\mathcal{D}}(h_S^\star) - L_{\mathcal{D}}(h^\star)}_{\text{estimation}} + \underbrace{L_{\mathcal{D}}(h_S) - L_{\mathcal{D}}(h_S^\star)}_{\text{optimization}}$$



Convex Learning Problems

- $\mathcal{H}$ is a convex set
- For all $\mathbf{z}$, the function $\ell(\cdot, \mathbf{z})$ is convex and Lipschitz
- Example: SVM learning (hinge-loss minimization)

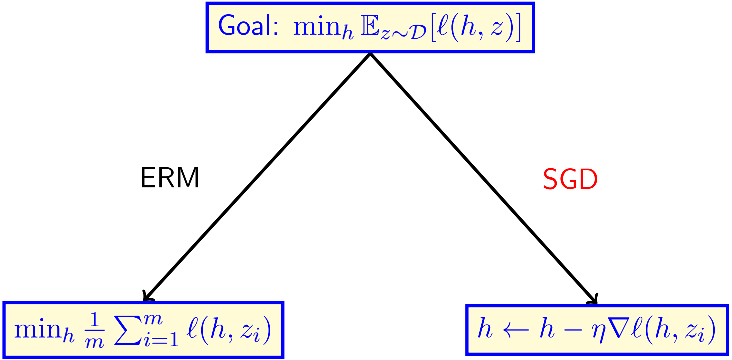
Solving Convex Learning Problems

Goal: $\min_h \mathbb{E}_{z\sim\mathcal{D}}[\ell(h,z)]$

ERM

$$\min_h \frac{1}{m}\sum_{i=1}^m \ell(h,z_i)$$

Solving Convex Learning Problems



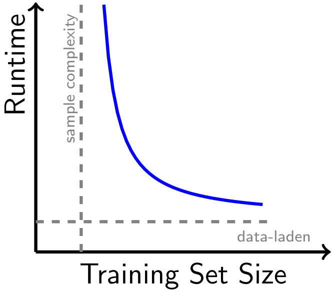
- Both methods have the same sample complexity in the worst case.
- But, ERM can be better on many distributions

Second-Order Stochastic Gradient Descent

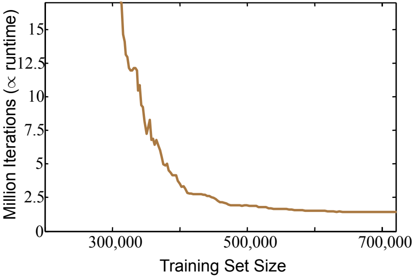
- Smaller sample complexity under some spectral assumptions, E.g Leon Bottou's talk today
- But, runtime is $\Omega(d^2)$ per iteration
- When d is large, we might prefer running SGD (for approximately solving the ERM problem)

More Data Less Work for SGD

Theoretical



Empirical (CCAT)



Summary

- A formal model for Time-Sample Complexity
- Different techniques for improving training time when more examples are available
- Formal derivation of gaps

Open Questions

- Other techniques ?
- Showing gaps for real-world problems ?