What *else* can we do with more data?

Shai Shalev-Shwartz

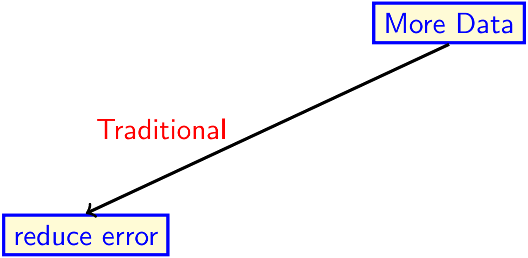
School of Computer Science and Engineering
The Hebrew University of Jerusalem

TTI,
February 2011

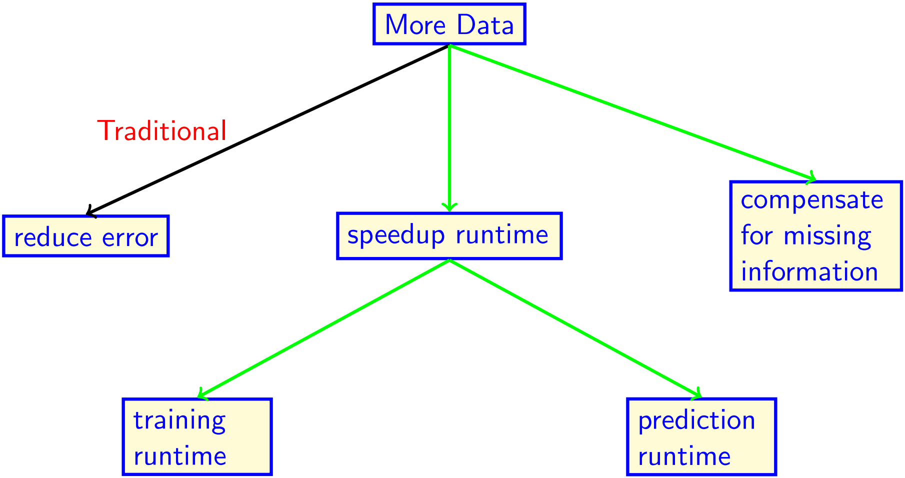
Based on joint papers with:
Ohad Shamir and Karthik Sridharan (COLT 2010)
Nicolò Cesa-Bianchi and Ohad Shamir (ICML 2010)
Shai Ben-David and Ruth Urner (Submitted)

and, of course, Nati Srebro

What *else* can we do with more data?



What *else* can we do with more data?



Outline

How can more data speedup training runtime?

- Learning using Stochastic Optimization (S. & Srebro 2008)
  Will not talk about this today
- Injecting Structure (S., Shamir, Sirdharan 2010)

How can more data speedup prediction runtime?

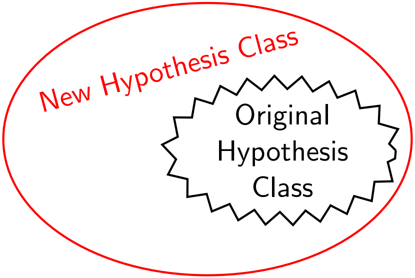
- Proper Semi-Supervised Learning (S., Ben-David, Urner 2011)

How can more data compensate for missing information?

- Attribute Efficient Learning (Cesa-Bianchi, S., Shamir 2010)
  Technique: Rely on Stochastic Optimization

Injecting Structure – Main Idea

- Replace original hypothesis class with a larger hypothesis class
- On one hand: Larger class has more structure $\Rightarrow$ easier to optimize
- On the other hand: Larger class $\Rightarrow$ larger estimation error $\Rightarrow$ need more examples



Example — Learning 3-DNF

- Goal: learn a 3-DNF Boolean function $h : \{0,1\}^d \to \{0,1\}$
- DNF is a simple way to describe a concept (e.g. "computer nerd")
- Variables are attributes. E.g.
  - $x_1 =$ can read binary code
  - $x_2 =$ runs Unix as the operating system on his home computer
  - $x_3 =$ has a girlfriend
  - $x_4 =$ blush whenever tells someone how big his hard drive is
- $h(x) = (x_1 \wedge \neg x_3) \vee (x_2 \wedge \neg x_3) \vee (x_4 \wedge \neg x_3)$

Example — Learning 3-DNF

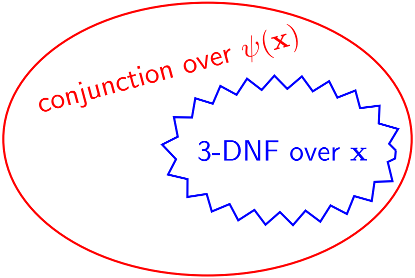
- Kearns & Vazirani: If RP$\neq$NP, it is not possible to efficiently learn an ϵ-accurate 3-DNF formula

Example — Learning 3-DNF

- Kearns & Vazirani: If RP$\neq$NP, it is not possible to efficiently learn an ϵ-accurate 3-DNF formula
- Claim: if $m \geq d^3/\epsilon$ it is possible to find a predictor with error $\leq \epsilon$ in polynomial time

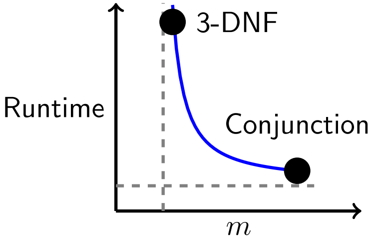
Proof

- Observation: 3-DNF formula can be rewritten as $\wedge_{u \in T_1, v \in T_2, w \in T_3}(u \vee v \vee w)$ for three sets of literals T_1, T_2, T_3
- Define: $\psi : \{0,1\}^d \to \{0,1\}^{2(2d)^3}$ s.t. for each triplet of literals u, v, w there are two variables indicating if $u \vee v \vee w$ is true or false
- Observation: Each 3-DNF can be represented as a single conjunction over $\psi(\mathbf{x})$
- Easy to learn single conjunction (greedy or LP)



Trading samples for runtime

Algorithm	samples	runtime
3-DNF over $\mathbf{x}$	$\frac{d}{\epsilon}$	2^d
Conjunction over $\psi(\mathbf{x})$	$\frac{d^3}{\epsilon}$	$\mathrm{poly}(d)$



Disclaimer

- Analysis is based on upper bounds
- Open problem: establish gaps by deriving lower bounds
- Studied by:
  "Computational Sample Complexity" (Decatur, Goldreich, Ron 1998)
- Very few (if any) results on "real-world" problems, e.g.
  Rocco Servedio showed gaps for 1-decision lists

Agnostic learning of Halfspaces with $0-1$ loss

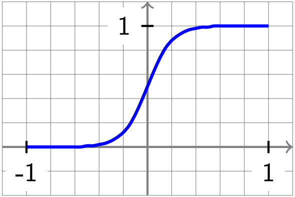
Agnostic PAC:

- $\mathcal{D}$ - arbitrary distribution over $\mathcal{X} \times \mathcal{Y}$
- Training set: $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$
- Goal: use S to find h_S s.t. w.p. $1-\delta$,

$$\mathrm{err}(h_S) \leq \min_{h \in \mathcal{H}} \mathrm{err}(h) + \epsilon$$

Hypothesis class

$$\mathcal{H} = \{\mathbf{x} \mapsto \phi(\langle \mathbf{w}, \mathbf{x} \rangle) : \|\mathbf{w}\|_2 \leq 1\}, \qquad \phi(z) = \frac{1}{1+\exp(-z/\mu)}$$



- Probabilistic classifier: $\Pr[h_{\mathbf{w}}(\mathbf{x}) = 1] = \phi(\langle \mathbf{w}, \mathbf{x} \rangle)$
- Loss function: $\mathrm{err}(\mathbf{w}; (\mathbf{x}, y)) = \Pr[h_{\mathbf{w}}(\mathbf{x}) \neq y] = \left|\phi(\langle \mathbf{w}, \mathbf{x} \rangle) - \frac{y+1}{2}\right|$
- Remark: Dimension can be infinite (kernel methods)

First approach — sub-sample covering

- Claim: exists $1/(\epsilon\mu^2)$ examples from which we can efficiently learn $\mathbf{w}^\star$ up to error of ϵ
- Proof idea:
  - $S' = \{(\mathbf{x}_i, y'_i) : y'_i = y_i \text{ if } y_i\langle \mathbf{w}^\star, \mathbf{x}_i\rangle < -\mu \text{ and else } y'_i = -y_i\}$
  - Use surrogate convex loss $\frac{1}{2}\max\{0, 1 - y\langle \mathbf{w}, x\rangle/\gamma\}$
  - Minimizing surrogate loss on S' $\Rightarrow$ minimizing original loss on S
  - Sample complexity w.r.t. surrogate loss is $1/(\epsilon\mu^2)$

Analysis

- Sample complexity: $1/(\epsilon\mu)^2$
- Time complexity: $m^{1/(\epsilon\mu^2)} = \left(\frac{1}{\epsilon\mu}\right)^{1/(\epsilon\mu^2)}$

Second Approach – IDPK (S, Shamir, Sridharan)

Learning fuzzy halfspaces using **I**nfinite-**D**imensional-**P**olynomial-**K**ernel

- Original class: $\mathcal{H} = \{\mathbf{x} \mapsto \phi(\langle \mathbf{w}, \mathbf{x} \rangle)\}$

Second Approach – IDPK (S, Shamir, Sridharan)

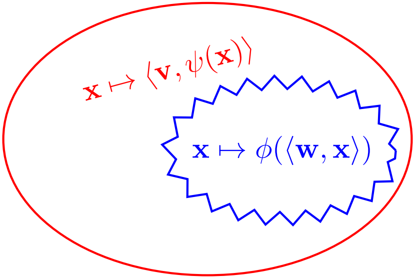
Learning fuzzy halfspaces using **I**nfinite-**D**imensional-**P**olynomial-**K**ernel

- Original class: $\mathcal{H} = \{\mathbf{x} \mapsto \phi(\langle \mathbf{w}, \mathbf{x} \rangle)\}$
- Problem: Loss is non-convex w.r.t. $\mathbf{w}$

Second Approach – IDPK (S, Shamir, Sridharan)

Learning fuzzy halfspaces using **I**nfinite-**D**imensional-**P**olynomial-**K**ernel

- Original class: $\mathcal{H} = \{\mathbf{x} \mapsto \phi(\langle \mathbf{w}, \mathbf{x} \rangle)\}$
- Problem: Loss is non-convex w.r.t. $\mathbf{w}$
- Main idea: Work with a larger hypothesis class for which the loss becomes convex



Step 2 – Learning fuzzy halfspaces with IDPK

- Original class: $\mathcal{H} = \{\mathbf{x} \mapsto \phi(\langle \mathbf{w}, \mathbf{x} \rangle) : \|\mathbf{w}\| \le 1\}$
- New class: $\mathcal{H}' = \{\mathbf{x} \mapsto \langle \mathbf{v}, \psi(\mathbf{x}) \rangle : \|\mathbf{v}\| \le B\}$ where $\psi : \mathcal{X} \to \mathbb{R}^{\mathbb{N}}$ s.t. $\forall j,\ \forall (i_1, \ldots, i_j),\ \psi(\mathbf{x})_{(i_1,\ldots,i_j)} = 2^{j/2}\, x_{i_1} \cdots x_{i_j}$

Step 2 – Learning fuzzy halfspaces with IDPK

- Original class: $\mathcal{H} = \{\mathbf{x} \mapsto \phi(\langle \mathbf{w}, \mathbf{x} \rangle) : \|\mathbf{w}\| \leq 1\}$
- New class: $\mathcal{H}' = \{\mathbf{x} \mapsto \langle \mathbf{v}, \psi(\mathbf{x}) \rangle : \|\mathbf{v}\| \leq B\}$ where $\psi : \mathcal{X} \to \mathbb{R}^{\mathbb{N}}$ s.t. $\forall j,\ \forall (i_1, \ldots, i_j),\ \psi(\mathbf{x})_{(i_1,\ldots,i_j)} = 2^{j/2}\, x_{i_1} \cdots x_{i_j}$

Lemma (S, Shamir, Sridharan 2009)

If $B = \exp(\tilde{O}(1/\mu))$ then for all $h \in \mathcal{H}$ exists $h' \in \mathcal{H}'$ s.t. for all $\mathbf{x}$, $h(\mathbf{x}) \approx h'(\mathbf{x})$.

Step 2 – Learning fuzzy halfspaces with IDPK

- Original class: $\mathcal{H} = \{\mathbf{x} \mapsto \phi(\langle \mathbf{w}, \mathbf{x} \rangle) : \|\mathbf{w}\| \le 1\}$
- New class: $\mathcal{H}' = \{\mathbf{x} \mapsto \langle \mathbf{v}, \psi(\mathbf{x}) \rangle : \|\mathbf{v}\| \le B\}$ where $\psi : \mathcal{X} \to \mathbb{R}^{\mathbb{N}}$ s.t. $\forall j,\ \forall (i_1, \ldots, i_j),\ \psi(\mathbf{x})_{(i_1,\ldots,i_j)} = 2^{j/2}\, x_{i_1} \cdots x_{i_j}$

Lemma (S, Shamir, Sridharan 2009)

If $B = \exp(\tilde{O}(1/\mu))$ then for all $h \in \mathcal{H}$ exists $h' \in \mathcal{H}'$ s.t. for all $\mathbf{x}$, $h(\mathbf{x}) \approx h'(\mathbf{x})$.

Remark: The above is a pessimistic choice of B. In practice, smaller B suffices. Is it tight? Even if it is, are there natural assumptions under which a better bound holds ?
(e.g. Kalai, Klivans, Mansour, Servedio 2005)

Proof idea

- Polynomial approximation: $\phi(z) \approx \sum_{j=0}^{\infty} \beta_j z^j$

Proof idea

- Polynomial approximation: $\phi(z) \approx \sum_{j=0}^{\infty} \beta_j z^j$
- Therefore:

$$\begin{aligned}
\phi(\langle \mathbf{w}, \mathbf{x} \rangle) &\approx \sum_{j=0}^{\infty} \beta_j (\langle \mathbf{w}, \mathbf{x} \rangle)^j \\
&= \sum_{j=0}^{\infty} \sum_{k_1,\ldots,k_j} 2^{-j/2} \beta_j 2^{j/2} w_{k_1} \cdots w_{k_j} x_{k_1} \cdot \cdots \cdot x_{k_j} \\
&= \langle \mathbf{v}_{\mathbf{w}}, \psi(\mathbf{x}) \rangle
\end{aligned}$$

Proof idea

- Polynomial approximation: $\phi(z) \approx \sum_{j=0}^{\infty} \beta_j z^j$
- Therefore:

$$\begin{aligned}
\phi(\langle \mathbf{w}, \mathbf{x} \rangle) &\approx \sum_{j=0}^{\infty} \beta_j (\langle \mathbf{w}, \mathbf{x} \rangle)^j \\
&= \sum_{j=0}^{\infty} \sum_{k_1,\ldots,k_j} 2^{-j/2} \beta_j 2^{j/2} w_{k_1} \cdots w_{k_j} x_{k_1} \cdots \cdot x_{k_j} \\
&= \langle \mathbf{v_w}, \psi(\mathbf{x}) \rangle
\end{aligned}$$

- To obtain a concrete bound we use Chebyshev approximation technique: Family of orthogonal polynomials w.r.t. inner product:

$$\langle f, g \rangle = \int_{x=-1}^{1} \frac{f(x) g(x)}{\sqrt{1 - x^2}} dx$$

Infinite-Dimensional-Polynomial-Kernel

- Although the dimension is infinite, can be solved using the kernel trick
- The corresponding kernel (a.k.a. Vovk's infinite polynomial):

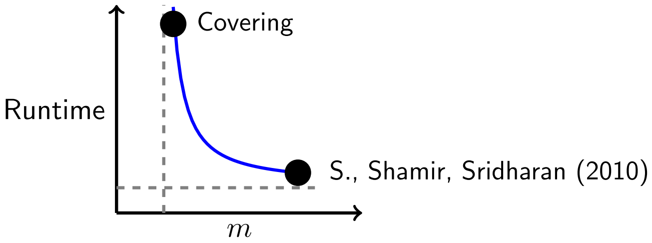
$$\langle \psi(\mathbf{x}), \psi(\mathbf{x}') \rangle = K(\mathbf{x}, \mathbf{x}') = \frac{1}{1 - \frac{1}{2}\langle \mathbf{x}, \mathbf{x}' \rangle}$$

- Algorithm boils down to linear regression with the above kernel
- Convex! Can be solved efficiently
- Sample complexity: $(B/\epsilon)^2 = 2^{\tilde{O}(1/\mu)}/\epsilon^2$
- Time complexity: m^2

Trading samples for time

Algorithm	sample	time
Covering	$\frac{1}{\epsilon^2\mu^2}$	$\left(\frac{1}{\epsilon\mu}\right)^{1/(\epsilon\mu^2)}$
	$\Uparrow$	$\Downarrow$
IDPK	$\left(\frac{1}{\epsilon\mu}\right)^{1/\mu} \frac{1}{\epsilon^2}$	$\left(\frac{1}{\epsilon\mu}\right)^{2/\mu} \frac{1}{\epsilon^4}$

Agnostic learning of Halfspaces with $0-1$ loss



Outline

How can more data speedup training runtime?

- Learning using Stochastic Optimization (S. & Srebro 2008)
- Injecting Structure (S., Shamir, Sirdharan 2010) ✓

How can more data speedup prediction runtime?

- Proper Semi-Supervised Learning (S., Ben-David, Urner 2011)

How can more data compensate for missing information?

- Attribute Efficient Learning (Cesa-Bianchi, S., Shamir 2010)
  Technique: Rely on Stochastic Optimization

More data can speedup prediction time

- Semi-Supervised Learning: Many unlabeled examples, few labeled examples
- Most previous work: how unlabeled data can improve accuracy ?
- Our goal: how unlabeled data can help constructing *faster* classifiers
- Modeling: *Proper*-Semi-Supervised-Learning — we must output a classifier from a predefined class $\mathcal{H}$

Proper Semi-Supervised Learning

A simple two phase procedure:

- Use labeled examples to learn an arbitrary classifier (which is as accurate as possible)
- Apply the learned classifier to label the unlabeled examples
- Feed the now-labeled examples to a *proper* supervised learning for $\mathcal{H}$

Proper Semi-Supervised Learning

A simple two phase procedure:

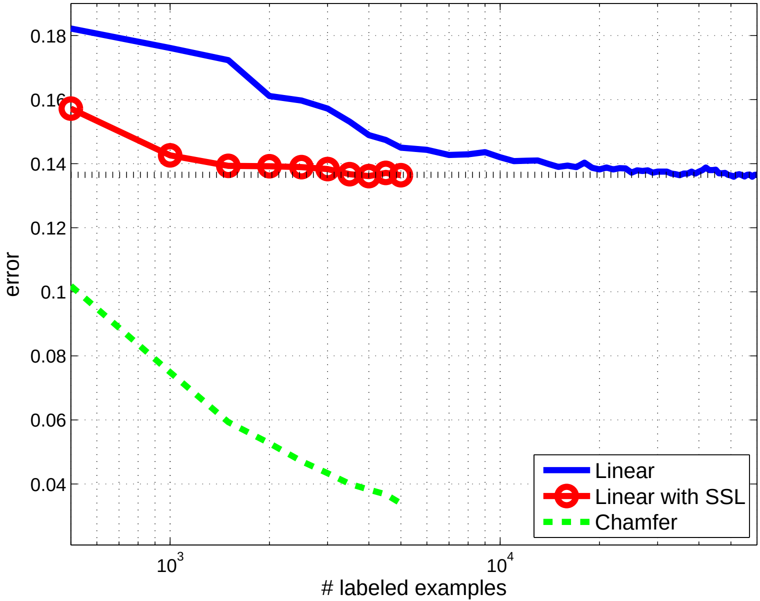
- Use labeled examples to learn an arbitrary classifier (which is as accurate as possible)
- Apply the learned classifier to label the unlabeled examples
- Feed the now-labeled examples to a *proper* supervised learning for $\mathcal{H}$

Lemma

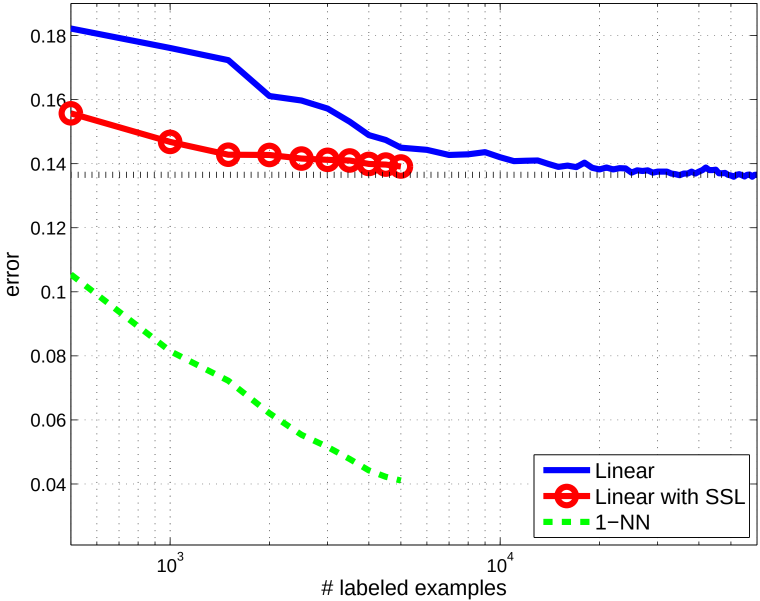
Agnostic learners are robust with respect to small changes in the input distribution:

$$P[h(x) \neq f(x)] \leq P[h(x) \neq g(x)] + P[g(x) \neq f(x)]$$

Demonstration



Demonstration



Outline

How can more data speedup training runtime?

- Learning using Stochastic Optimization (S. & Srebro 2008)
- Injecting Structure (S., Shamir, Sirdharan 2010) ✓

How can more data speedup prediction runtime?

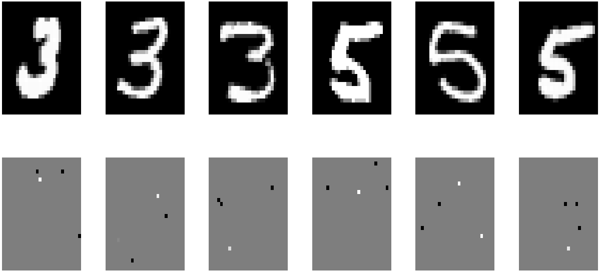
- Proper Semi-Supervised Learning (S., Ben-David, Urner 2011) ✓

How can more data compensate for missing information?

- Attribute Efficient Learning (Cesa-Bianchi, S., Shamir 2010)
  Technique: Rely on Stochastic Optimization

Attribute efficient regression

- Each training example is a pair $(\mathbf{x}, y) \in \mathbb{R}^d \times \mathbb{R}$
- Partial information: can only view $O(1)$ attributes of each individual example



How more data helps?

Three main techniques:

1. Missing information as noise
2. Active Exploration — try to “fish” the relevant information
3. Inject structure — problem hard in the original representation but becomes simple in another representation (different hypothesis class)

More data helps because:

1. It reduces variance — compensates for the noise
2. It allows more exploration
3. It compensates for larger sample complexity due to using larger hypotheses classes

Attribute efficient regression

Formal problem statement:

- Unknown distribution $\mathcal{D}$ over $\mathbb{R}^d \times \mathbb{R}$
- Goal: learn a linear predictor $\mathbf{x} \mapsto \langle \mathbf{w}, \mathbf{x} \rangle$ with low risk:
- Risk: $L_{\mathcal{D}}(\mathbf{w}) \;=\; \mathbb{E}_{\mathcal{D}}[(\langle \mathbf{w}, \mathbf{x} \rangle - y)^2]$
- Training set: $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$
- Partial information: For each $(\mathbf{x}_i, y_i)$, learner can view only k attributes of $\mathbf{x}_i$
- Active selection: learner can choose which k attributes to see

Similar to "Learning with restricted focus of attention" (Ben-David & Dichterman 98)

Dealing with missing information

- Usually difficult — exponential ways to complete the missing information
- Popular approach — Expectation Maximization (EM)

Previous methods usually do not come with guarantees
(neither sample complexity nor computational complexity)

Partial information as noise

- Observation:

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{pmatrix} = \frac{1}{d}\begin{pmatrix} dx_1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} + \ldots + \frac{1}{d}\begin{pmatrix} 0 \\ \vdots \\ 0 \\ dx_d \end{pmatrix}$$

- Therefore, choosing i uniformly at random gives

$$\mathbb{E}_i[dx_i\mathbf{e}^i] = \mathbf{x} \ .$$

- If $\|\mathbf{x}\| \le 1$ then $\|dx_i\mathbf{e}^i\| \le d$ (i.e. variance increased)
- Reduced missing information to unbiased noise
- Many examples can compensate for the added noise

A Stochastic Optimization Approach

- Our goal: minimize over $\mathbf{w}$ the true risk $L_{\mathcal{D}}(\mathbf{w}) = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}[(\langle \mathbf{w}, \mathbf{x}\rangle - y)^2]$
- We can only obtain i.i.d. samples from $\mathcal{D}$

$\boxed{\text{Goal: } \min_{\mathbf{w}} L_{\mathcal{D}}(\mathbf{w})}$

ERM
(traditional)

$$\boxed{\min_{\mathbf{w}} \frac{1}{m}\sum_{i=1}^{m}(\langle \mathbf{w}, \mathbf{x}_i\rangle - y_i)^2}$$

A Stochastic Optimization Approach

- Our goal: minimize over $\mathbf{w}$ the true risk $L_{\mathcal{D}}(\mathbf{w}) = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}[(\langle \mathbf{w}, \mathbf{x} \rangle - y)^2]$
- We can only obtain i.i.d. samples from $\mathcal{D}$

$$\boxed{\text{Goal: } \min_{\mathbf{w}} L_{\mathcal{D}}(\mathbf{w})}$$

ERM (traditional)

$$\boxed{\min_{\mathbf{w}} \frac{1}{m} \sum_{i=1}^{m} (\langle \mathbf{w}, \mathbf{x}_i \rangle - y_i)^2}$$

SGD

$$\boxed{\mathbf{w} \leftarrow \mathbf{w} - \eta \mathbf{v} \text{ where } \mathbb{E}[\mathbf{v}] = \nabla L_{\mathcal{D}}(\mathbf{w})}$$

A Stochastic Optimization Approach

How to construct an unbiased estimate of the gradient:

- Sample $(\mathbf{x}, y) \sim \mathcal{D}$
- Sample j uniformly from $[d]$
- Sample i from $[d]$ based on $P[i] = |w_i|/\|\mathbf{w}\|_1$
- Set $\mathbf{v} = 2(\mathrm{sign}(w_i)\|\mathbf{w}\|_1 x_j - y) d x_j \mathbf{e}^j$
- Claim: $\mathbb{E}[\mathbf{v}] = \nabla L_{\mathcal{D}}(W)$

A Stochastic Optimization Approach

Theorem (Cesa-Bianchi, S, Shamir)

Let $\hat{\mathbf{w}}$ *be the output of AER and let* $w^\star$ *be a competing vector. Then, with high probability*

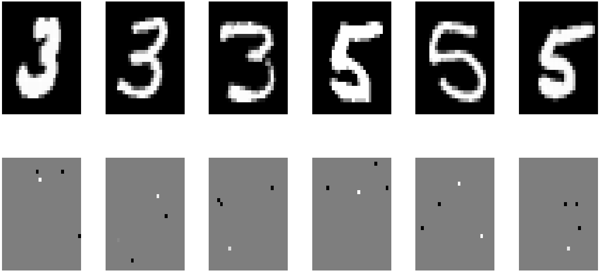
$$L_{\mathcal{D}}(\hat{\mathbf{w}}) \leq L_D(\mathbf{w}^\star) + \tilde{O}\left(\frac{d\,\|\mathbf{w}^\star\|_2\,\|\mathbf{w}^\star\|_1}{\sqrt{m}}\right) \;,$$

where d *is dimension and* m *is number of examples.*

Corollary

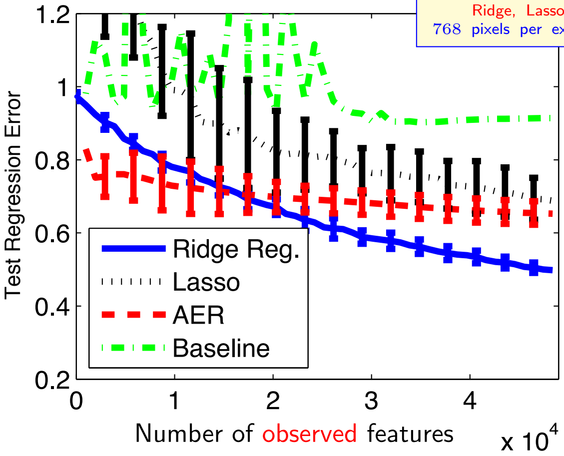
Factor of d^2 *additional examples compensates for the lack of full information on each individual example.*

Demonstration



- Full information classifiers (top line) $\Rightarrow$ error of $\sim 1.1\%$
- Our algorithm (bottom line) $\Rightarrow$ error of $\sim 3.5\%$

Demonstration



AER, Baseline:
4 pixels per example
Ridge, Lasso:
768 pixels per example

What to do with other loss functions?

- General question: Given r.v. X and function $f : \mathbb{R} \to \mathbb{R}$, how to construct an unbiased estimate of $f(\mathbb{E}[X])$?

What to do with other loss functions?

- General question: Given r.v. X and function $f : \mathbb{R} \to \mathbb{R}$, how to construct an unbiased estimate of $f(\mathbb{E}[X])$?
- Claim (Paninski 2003): In general, not possible

What to do with other loss functions?

- General question: Given r.v. X and function $f : \mathbb{R} \to \mathbb{R}$, how to construct an unbiased estimate of $f(\mathbb{E}[X])$?
- Claim (Paninski 2003): In general, not possible
- Claim (Singh 1964, The Indian Journal of Statistics): Possible if sample size is also a random number !

The key idea

- Can construct $Q_n(x) = \sum_{i=0}^{n} \gamma_{n,i} x^i \stackrel{n\to\infty}{\longrightarrow} f(x)$
- Let $Q'_n(x_1, \ldots, x_n) = \sum_{i=0}^{n} \gamma_{n,i} \prod_{j=1}^{i} x_j$
- Estimator:
  - draw a positive integer N according to $\Pr(N = n) = p_n$
  - sample i.i.d. $x_1, x_2, \ldots, x_N$
  - return $\frac{1}{p_N}\left(Q'_N(x_1, \ldots, x_N) - Q'_{N-1}(x_1, \ldots, x_{N-1})\right)$,
- Claim: This is an unbiased estimator of $f(\mathbb{E}[X])$

$$
\begin{aligned}
&\mathop{\mathbb{E}}_{N, x_1, \ldots, x_N}\left[\frac{1}{p_N}\left(Q'_N(x_1, \ldots, x_N) - Q'_{N-1}(x_1, \ldots, x_{N-1})\right)\right] \\
&= \sum_{n=1}^{\infty} \frac{p_n}{p_n} \mathop{\mathbb{E}}_{x_1, \ldots, x_n}\left[Q'_n(x_1, \ldots, x_n) - Q'_{n-1}(x_1, \ldots, x_{n-1})\right] \\
&= \sum_{n=1}^{\infty}\left(Q_n(\mathbb{E}[X]) - Q_{n-1}(\mathbb{E}[X])\right) = f(\mathbb{E}[X]).
\end{aligned}
$$

Summary

- Learning theory: Many examples $\Rightarrow$ smaller error
- This work: Many examples $\Rightarrow$
  - Speedup training time
  - Speedup prediction time
  - Compensating for missing information
- Techniques:
  1. Stochastic optimization
  2. Inject structure
  3. Missing information as noise
  4. Active Exploration