Computational and Statistical Learning Theory

TTIC 31120

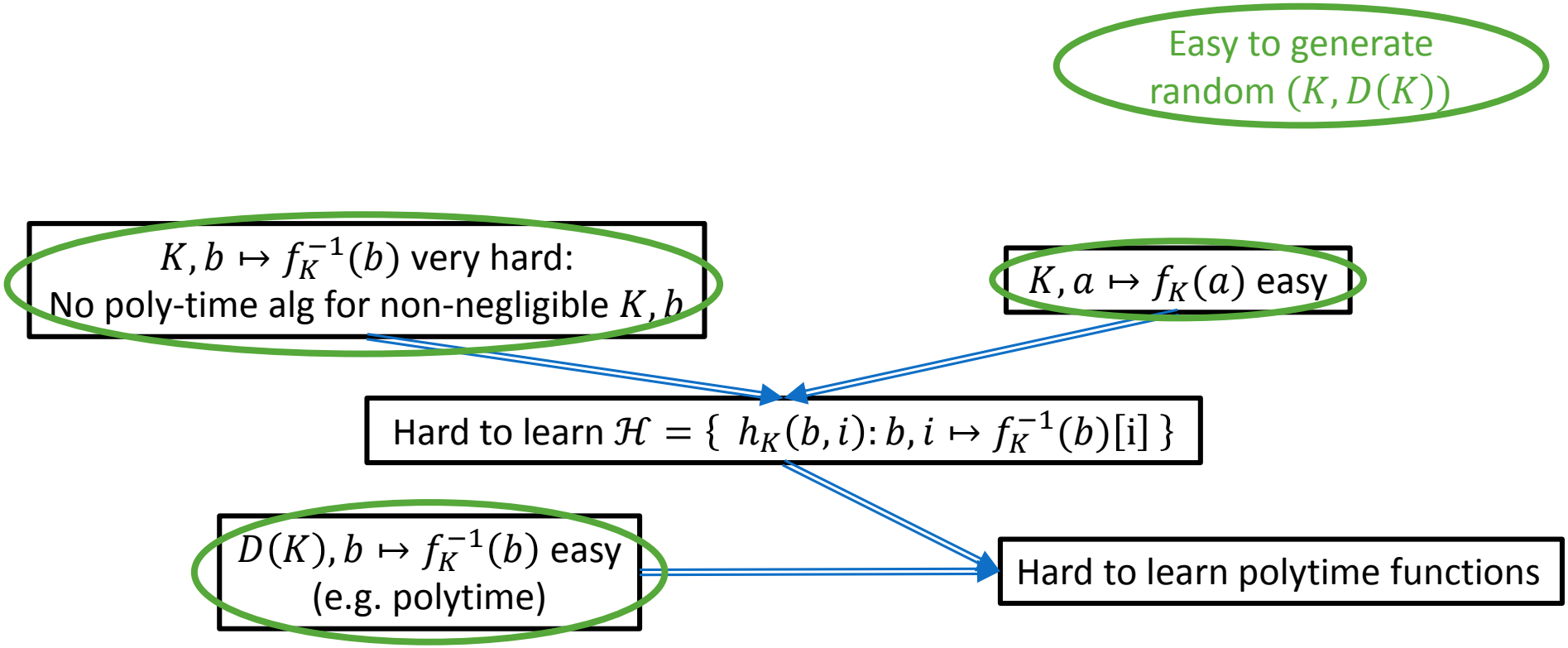
Prof. Nati Srebro

Lecture 7:

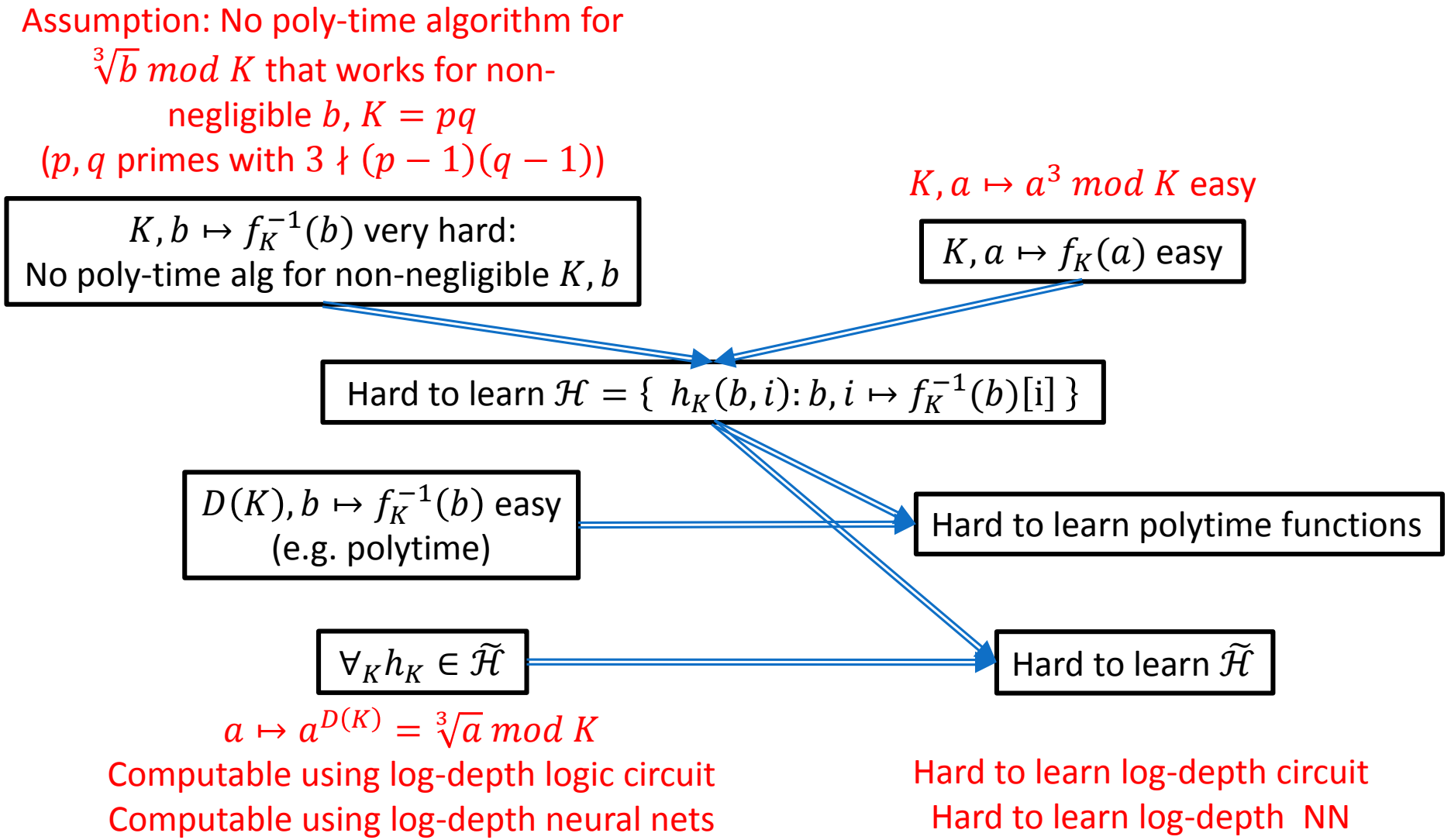
Computational Complexity of Learning—Hardness of Improper Learning (continued)

Agnostic Learning

Hardness of Learning via Crypto



Hardness of Learning via Crypto



Hardness of Learning via Crypto

- Public-key crypto is possible
  - ➔ hard to learn poly-time functions
- Hardness of Discrete Cube Root
  - ➔ hard to learn log(n)-depth logic circuits
  - ➔ hard to learn log(n)-depth poly-size neural networks
- Hardness of breaking RSA
  - ➔ hard to learn poly-length logical formulas
  - ➔ hard to learn poly-size automata
    - ➔ hard to learn push-down automata, ie regexps
  - ➔ for some depth d, hard to learn poly-size depth-d threshold circuits
    (output of unit is one iff number of input units that are one is greater than threshold)
- Hardness of lattice-shortest-vector based cryptography
  - ➔ hard to learn intersection of n^r halfspaces (for any $r > 0$)

Intersections of Halfspaces

$$\mathcal{H}_n^{k(n)} = \left\{ x \mapsto \wedge_{i=1}^{k(n)} \left[[\langle w_i, x \rangle > 0] \right] \mid w_1, \ldots, w_{k(n)} \in \mathbb{R}^n \right\}$$

$$\tilde{O}(n^{1.5}) - uSVP \notin RP$$

$$\Downarrow$$

Lattice-based cryptosystem is secure

$$\Downarrow$$

For any $r > 0$, hard to learn $H_n^{k(n)=n^r}$

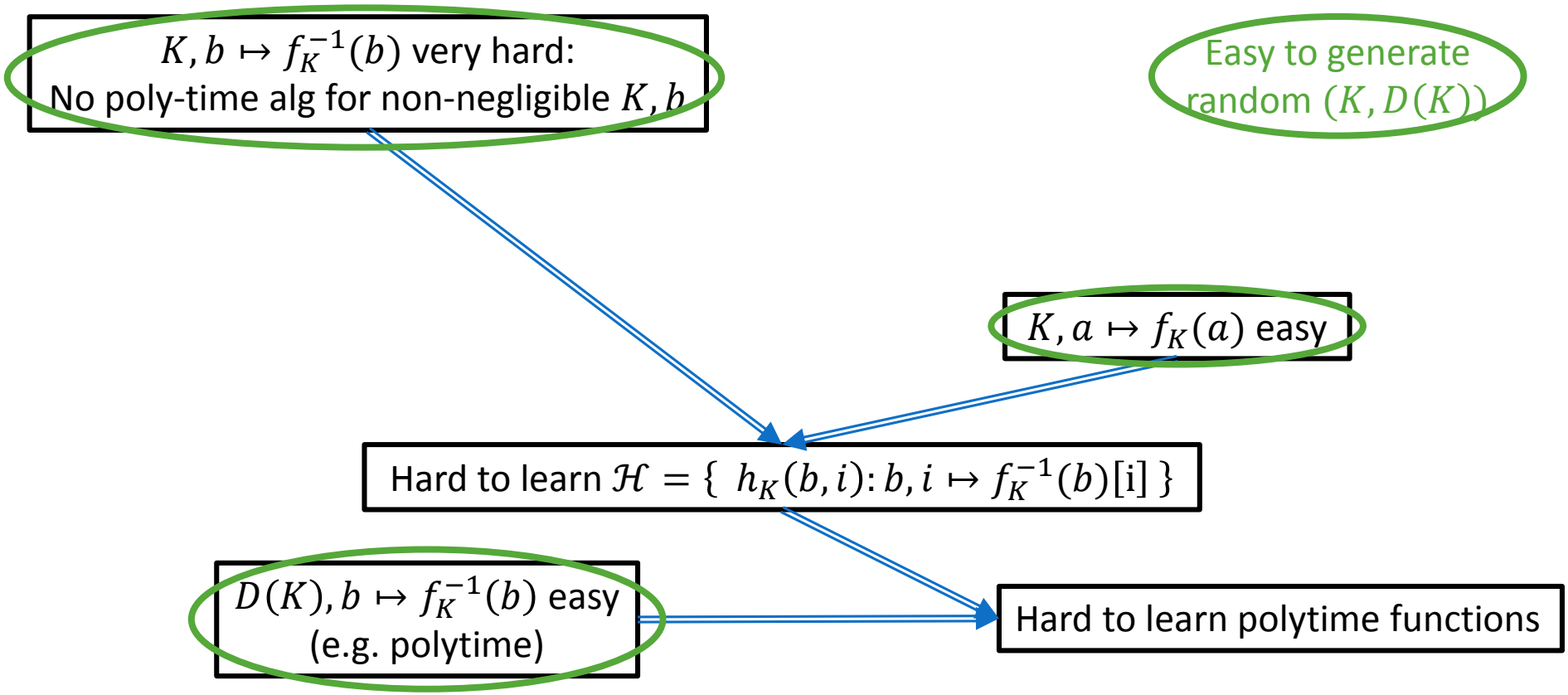
$$\Downarrow$$

Hard to learn 2-layer NN with n^r hidden units

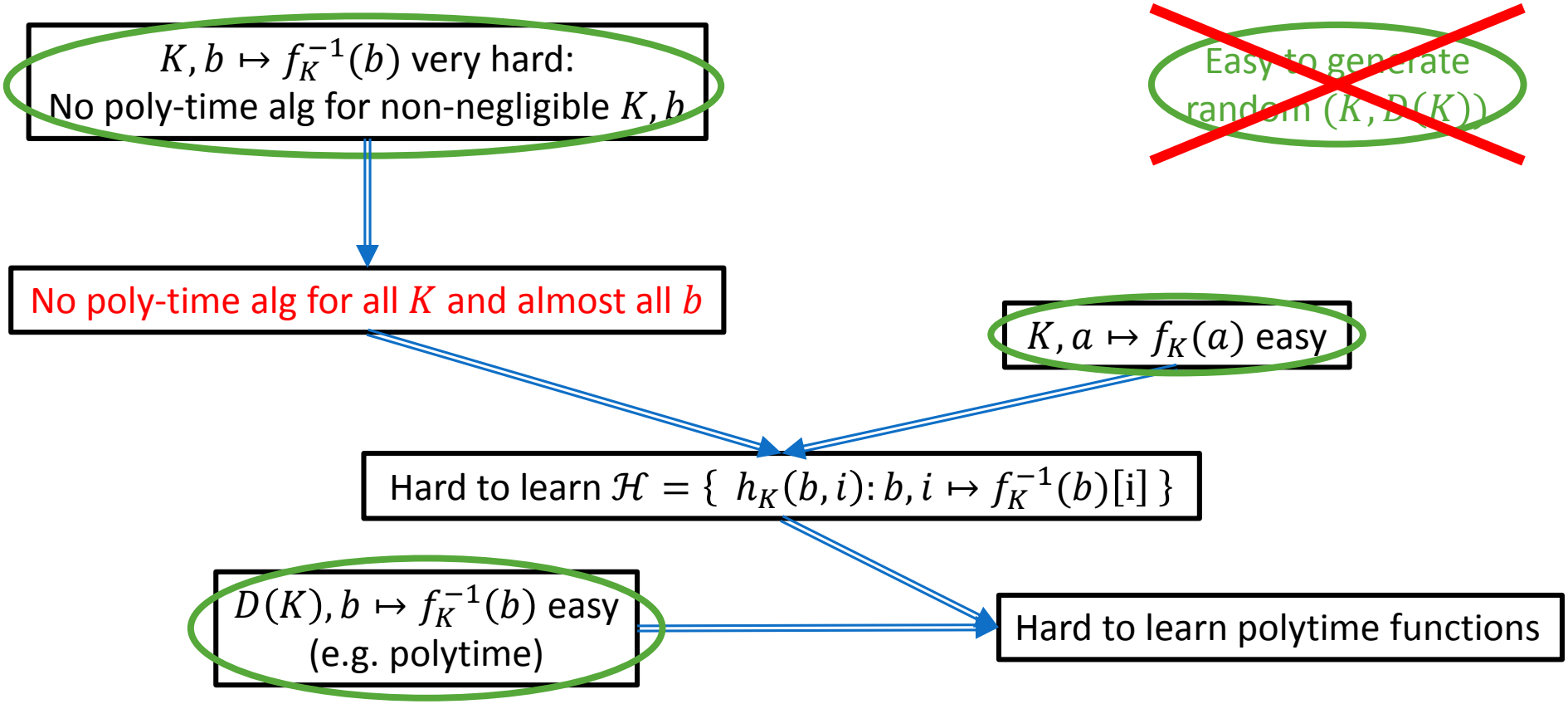
The unique shortest lattice vector problem:

- $\mathrm{SVP}(v_1, v_2, \ldots, v_n \in \mathbb{R}^n) = \arg\min_{a_1, a_2, \ldots, a_n \in \mathbb{Z}} \|a_1 v_1 + a_2 v_2 + \cdots + a_n v_n\|$
- $\tilde{O}(n^{1.5}) - uSVP$: only required to return SVP if next-shortest is $\tilde{O}(n^{1.5})$ times longer

Hardness of Learning via Crypto



Hardness of Learning via Crypto



Hardness of Learning: Take II

- Recall how we proved hardness of proper learning:
  - Reduction from deciding consistency with $\mathcal{H}$
  - If we had efficient *proper* learner, could train it and find consistent hypothesis in $\mathcal{H}$ if it exists
- Problem: if learning is not proper, might return good hypothesis not in $\mathcal{H}$, even though $\mathcal{D}$ not consistent with $\mathcal{H}$
- Instead: reduction from deciding between two possibilities:
  - Sample is consistent with $\mathcal{H}$
    - For every consistent sample, return 1 w.p. $\geq 3/4$ (over randomization in algorithm)
  - Sample comes from random "unpredictable" distribution
    - E.g. sampled such that labels y independent of x
    - For all but negligible samples $S \sim \mathcal{D}^m$, return 0 w.p. $\geq 3/4$

Hardness Relative to RSAT

- RSAT assumption: For some $f(K) = \omega(1)$, there is no poly-time randomized algorithm that gets as input a K-SAT formula with $n^{f(K)}$ constraints, and:
  - If the input is satisfiable, then w.p. $\geq 3/4$ (over the randomization in the algorithm), it outputs 1
  - If each constraint is generated independently and uniformly at random, then with probability approaching 1 (as $n \to \infty$) over the formula, w.p. $\geq 3/4$ (over the randomization in the algorithm), it outputs 0

- Theorem: Under the RSAT assumption,
  - Poly-length DNFs are not efficiently PAC learnable

    e.g. $h(x) = \left(x[1] \wedge \overline{x[7]} \wedge x[15] \wedge x[17]\right) \vee (x[2] \wedge x[24]) \vee \cdots$
  - Intersection of $\omega(\log n)$ halfspaces are not efficiently PAC learnable

    ➔ 2-layer Neural Networks with $O(\log^{1.1} n)$ hidden layers are not efficiently PAC learnable

Hardness of Learning

- Axis-aligned rectangles in n dimensions
- Halfspaces in n dimensions
- Conjunctions on n variables

Efficiently Properly Learnable

- 3-term DNF's

Efficiently Learnable, but not Properly

- DNF formulas of size poly(n)
- Generic logical formulas of size poly(n)
- Neural nets with at most poly(n) units
- Functions computable in poly(n) time

Not Efficiently Learnable

Realizable vs Agnostic

- **Definition**: A family $\mathcal{H}_n$ of hypothesis classes is **efficiently properly** PAC-Learnable if there exists a learning rule A such that $\forall_n \forall \epsilon, \delta > 0$, $\exists m(n, \epsilon, \delta)$, $\forall \mathcal{D}$ s.t. $L_{\mathcal{D}}(h) = 0$ for some $h \in \mathcal{H}$, $\forall^{\delta}_{S \sim \mathcal{D}^{m(n,\epsilon,\delta)}}$,

$$L_{\mathcal{D}}\big(A(S)\big) \leq \epsilon$$

and $A(S)(x)$ can be computed in time $poly(n, {}^{1}/_{\epsilon}, log\, {}^{1}/_{\delta})$
and A always outputs a predictor in $\mathcal{H}_n$

- **Definition**: A family $\mathcal{H}_n$ of hypothesis classes is **efficiently properly agnostically** PAC-Learnable if there exists a learning rule A such that $\forall_n \forall \epsilon, \delta > 0$, $\exists m(n, \epsilon, \delta)$, $\forall \mathcal{D}$ $\forall^{\delta}_{S \sim \mathcal{D}^{m(n,\epsilon,\delta)}}$,

$$L_{\mathcal{D}}\big(A(S)\big) \leq \inf_{h \in \mathcal{H}_n} L_{\mathcal{D}}(h) + \epsilon$$

and $A(S)(x)$ can be computed in time $poly(n, {}^{1}/_{\epsilon}, log\, {}^{1}/_{\delta})$
and A always outputs a predictor in $\mathcal{H}_n$

Conditions for Efficient Agnostic Learning

$$ERM_{\mathcal{H}}(S) = \arg\min_{h \in \mathcal{H}} L_S(h)$$

- Claim: If
  - $\text{VCdim}(\mathcal{H}_n) \leq poly(n)$, and
  - Each $h \in \mathcal{H}_n$ is computable in time poly(n)
  - There is a poly-time (in size of input) algorithm for $ERM_{\mathcal{H}}$ (i.e. that returns any ERM)

  then $\mathcal{H}_n$ is efficiently agnostically properly PAC learnable.

$$AGREEMENT_{\mathcal{H}}(S, k) = 1 \; iff \; \exists_{h \in \mathcal{H}} L_S(h) \leq (1 - k|S|)$$

- Claim: If $\mathcal{H}_n$ is efficiently properly agnostically PAC learnable, then $AGREEMENT_{\mathcal{H}} \in RP$

What is Properly Agnostically Learnable?

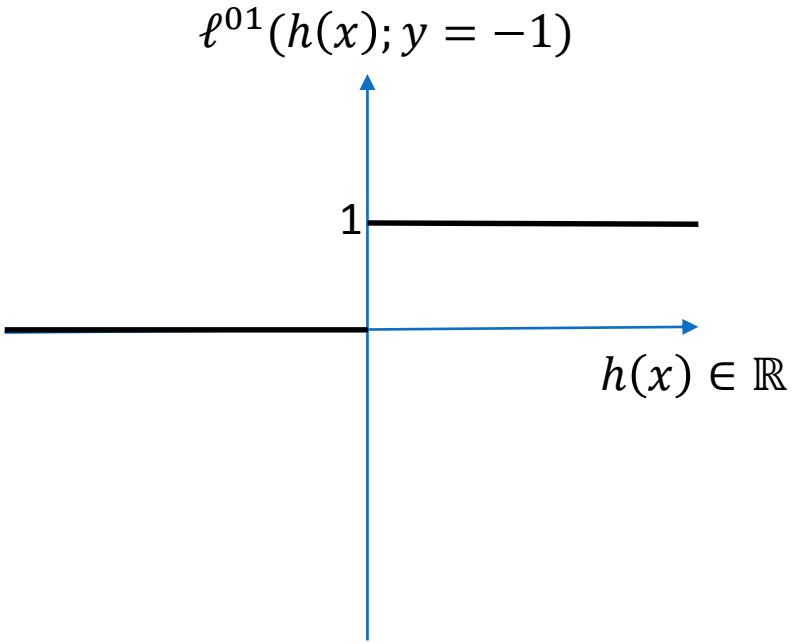
- Poly-time functions? **No!** (not even in realizable case)
- Poly-length logical formulas? **No!** (not even in realizable case)
- Poly-size depth-2 neural networks? **No!** (not even in realizable case)
- Halfspaces (linear predictors)? **No!**
  - $\mathcal{X}_n = \{0,1\}^n, \mathcal{H}_n = \left\{ x \mapsto \left[\left[\langle w, x\rangle > 0\right]\right] \mid w \in \mathbb{R}^n \right\}$
  - Claim: $AGREEMENT_{\mathcal{H}}$ is NP-Hard (optional HW problem)
  - Conclusion: If $NP \neq RP$, halfspaces are not efficiently properly agnostically learnable
- Conjunctions? **No!**
  - Also NP-hard!
- Unions of segments on the line **Yes!**
  - $\mathcal{X}_n = [0,1], \mathcal{H}_n = \left\{ x \mapsto \bigvee_{i=1}^{n} \left[\left[a_i \leq x \leq b_i\right]\right] \mid a_i, b_i \in [0,1] \right\}$
  - Efficiently Properly Agnostically PAC Learnable!

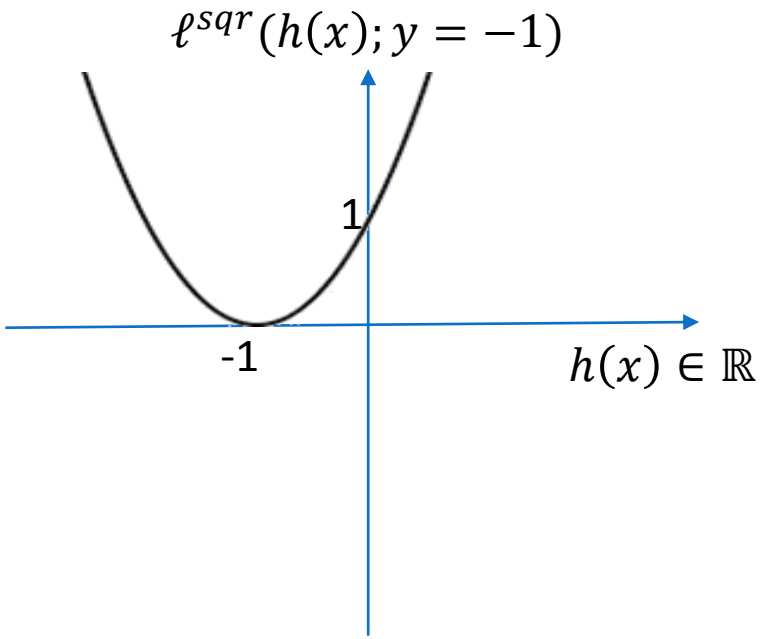
Source of the Hardness

$$\min_{h \in \mathcal{H}} \sum_i \ell(h_w(x_i); y_i)$$

$h_w(x) = \langle w, x \rangle$

$\ell^{01}(h(x); y) = [\![y h(x) \leq 0]\!]$

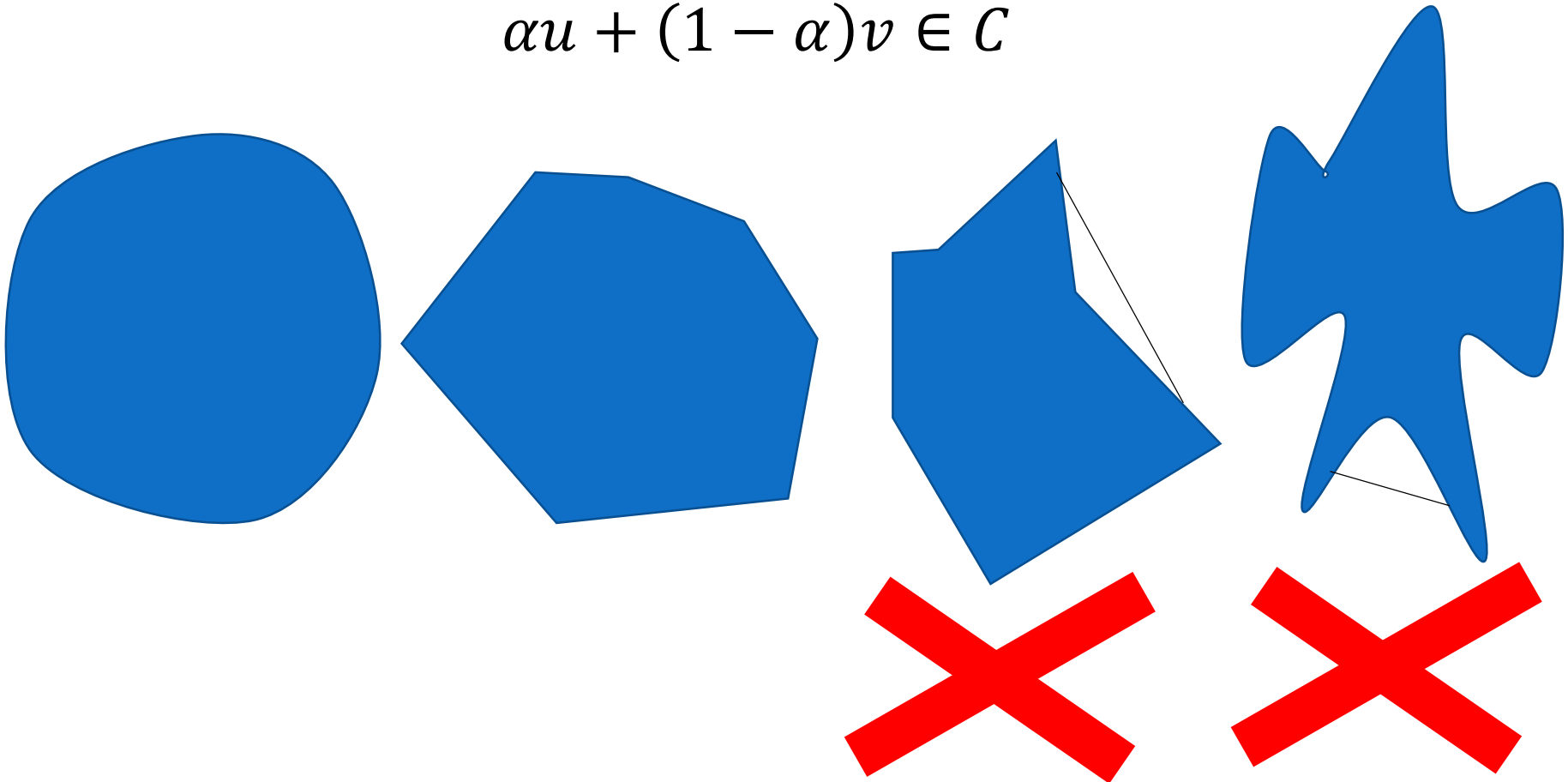




Convexity

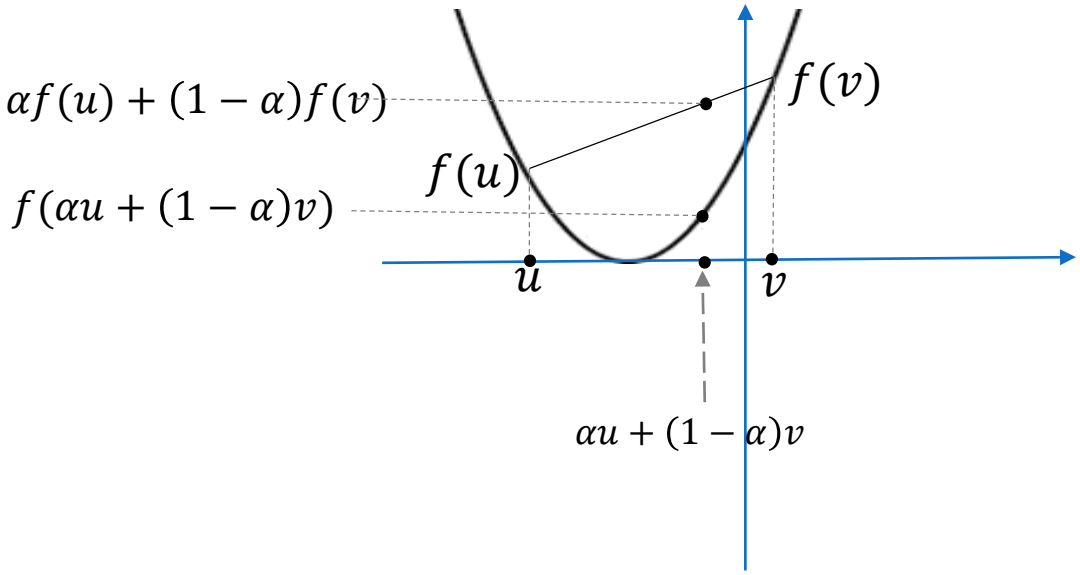
- Definition (convex set):
  A set C in a vector space is convex if $\forall u, v \in C$ and for all $\alpha \in [0,1]$:

$$\alpha u + (1 - \alpha)v \in C$$



Convexity

- Definition (convex function):
  A function $f: C \mapsto \mathbb{R}$ is convex if $\forall u, v \in C$:
  $$f(\alpha u + (1-\alpha)v) \leq \alpha f(u) + (1-\alpha) f(v)$$



Using a surrogate loss

$$\min_{h \in \mathcal{H}} \sum_i \ell(h_w(x_i); y_i)$$

- Instead of $\ell^{01}(z; y)$, use a surrogate $\ell(z; y)$ s.t.:
  - $\forall_y \ell(z; y)$ is convex in z
  - $\forall_{z,y} \ell^{01}(z; y) \leq \ell(z; y)$

- E.g.
  - $\ell^{sqr}(z; y) = (y - z)^2$
  - $\ell^{logistic}(z; y) = \log(1 + \exp\{-yz\})$
  - $\ell^{hinge}(\mathrm{z}; y) = [1 - yz]_+ = \max\{0, 1 - yz\}$

Minimizing a Surrogate Loss Does Not Minimize 0/1 Loss!

$$\ell^{01}(z;y) = \left[[yz \leq 0]\right] \leq [1 - yz]_+ = \ell^{hinge}(\mathrm{z};y)$$

- Realizable case:

$\exists_w L_S^{01}(x \mapsto \langle w, x\rangle) = 0$

➔ $L_S^{hinge}\left(x \mapsto \langle \frac{1}{\gamma} w, x\rangle\right) = 0, \gamma = \min_i yh_w(x) > 0$

➔ $L_S^{hinge}\left(\mathrm{ERM}^{\mathrm{hinge}}(\mathrm{S})\right) = 0$

➔ $L_S^{01}\left(ERM^{hinge}(S)\right) \leq L_S^{hinge}\left(\mathrm{ERM}^{\mathrm{hinge}}(\mathrm{S})\right) = 0$

- Non-Realizable case:

- What can we ensure by minimizing surrogate loss???

Improper Learning?

- Halfspaces not efficiently *properly* agnostically PAC learnable
- What about improper learning?

… we'll use boosting to reduce learning intersection of halfspaces to agnostic learning halfspaces

Why Study Hardness?

- Understand why machine learning is essentially a computational problem
- Understand why we must sometimes take a non-exact/heuristic approach, and that it cannot be exact (eg use surrogate loss)
- Understand what we can never guarantee, and not try to guarantee it (e.g. cannot learn with a large NN just because there is a small NN that completely explains the data)
- Understand and be able to argue about sample complexity gaps between the statistical limit (using any learning rule) and the computational limit (using a tractable learning rule)

"Weak" vs "Strong" Learning

- Recall definition of (realizable) PAC learning of $\mathcal{H}$ using rule $A(\cdot)$:

For any $\mathcal{D}$ s.t. $\inf_{h\in\mathcal{H}} L_{\mathcal{D}}(h) = 0$, and **any** $\epsilon, \delta > 0$, using $m(\epsilon, \delta)$ sample,

$$\forall^{\delta}_{S\sim\mathcal{D}^{m(\epsilon,\delta)}} \quad L_{\mathcal{D}}\big(A(S)\big) < \epsilon$$

- $A(\cdot)$ is a **weak learner** for $\mathcal{H}$ if:

There **exists** $\epsilon < 1/2$, $\delta < 1$, m, s.t. for any $\mathcal{D}$ with $\inf_{h\in\mathcal{H}} L_{\mathcal{D}}(h) = 0$,

$$\forall^{\delta}_{S\sim\mathcal{D}^{m}} \quad L_{\mathcal{D}}\big(A(S)\big) < \epsilon$$

(e.g. $\epsilon = 0.49$ and $1 - \delta = 0.01$)

- If $\mathcal{H}$ is weakly learnable, is it also strongly learnable?
  - Yes: $\mathcal{H}$ is weakly learnable ➔ VCdim($\mathcal{H}$)<∞ ➔ $\mathcal{H}$ is (strongly) learnable
- If $\mathcal{H}_n$ is *efficiently* weakly learnable, is it also strongly efficiently learnable?
- If we have access to an (efficient) weak learner $A(\cdot)$, can we use it to build an (efficient) strong learner?

The Boosting Problem

- Boosting the Confidence:
  If the learning algorithm works only with some very small fixed probability $1-\delta_0$ (e.g. $1-\delta_0 = 0.01$), can we construct a new algorithm that works with arbitrarily high probability $1-\delta$ (for any $\delta > 0$) ?

- Boosting the error:
  If the learning algorithm only returns a predictor that is guaranteed to be slightly better then chance, i.e. has error $\epsilon_0 = \frac{1}{2} - \gamma < \frac{1}{2}$ (for some fixed $\gamma > 0$), can we construct a new algorithm that achieves arbitrarily low error ϵ?

Boosting the Confidence

- For any δ:

1. For i=1..k: $\left(k = \frac{\log 2/\delta}{\log 1/\delta_0}\right)$
   Collect m_0 independent samples S_i
   $h_i = A(S_i)$
2. Collect $m_{\text{val}} = \frac{4\log(4k/\delta)}{\epsilon^2}$ additional independent samples S_{val}
3. Return $\hat{h} = \arg\min_{h_1,\ldots,h_k} L_{S_{val}}(h_i)$

w.p. $\geq 1-\delta$, $\inf_i L(h_i) \leq \epsilon_0$

ERM from class of size k

- Claim: w.p. $\geq 1-\delta$, $L(\hat{h}) \leq \epsilon_0 + \epsilon$
- Total samples used: $O\left(m_0(\epsilon_0) \cdot \log\frac{1}{\delta} + \frac{\log\frac{1}{\delta}}{\epsilon^2}\right)$
- Efficient algorithm for some $\delta_0 < 1$ and all $\epsilon > 0$ with runtime and sample complexity $poly(n, \epsilon_0)$
  ➔ efficient algorithm for any $\delta > 0$ with runtime $poly(n, \epsilon, \log 1/\delta)$

Boosting the Error?

- What if we can only find a predictor with relatively high excess error ϵ?
- We can always find a predictor with error $\leq \frac{1}{2}$
- What if we have an algorithm that, for any source dist $\mathcal{D}$ s.t. $\inf_h L_{\mathcal{D}}(h) = 0$, finds $L_{\mathcal{D}}\big(A(S)\big) \leq \frac{1}{2} - \gamma$.
- Can we use $A(\cdot)$ to find a predictor with arbitrarily low error?