Testing Assumptions of Learning Algorithms

Arsen Vasilyan

MIT

Talk based on joint works with

Aravind Gollakota
Apple

Adam R. Klivans
UT Austin

Ronitt Rubinfeld
MIT

Konstantinos Stavropoulos
UT Austin

Part I: Testable Agnostic Learning Framework

Distribution Testing + Agnostic Learning

Standard (aka Realizable) PAC Learning

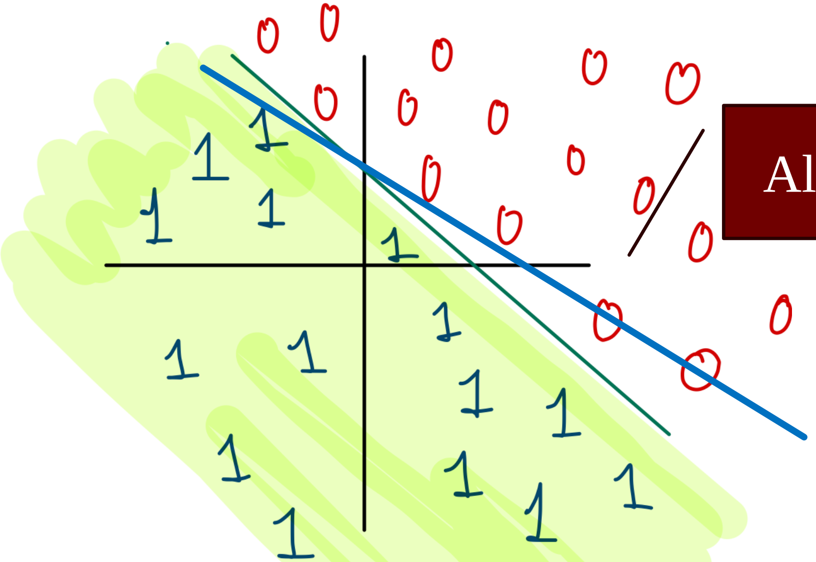
$\mathcal{F}$ = Halfspaces, Low-depth formulas, Monotone functions, etc…

Learning algo

Dataset $S \sim_{i.i.d.} D$,

labeled by g in $\mathcal{F}$

Give classifier f



Also ok

$$\mathrm{err}(f) \coloneqq \Pr_{x \sim D} [f(x) \neq g(x)],$$

Want w.h.p. $\mathrm{err}(f) \leq \epsilon$

Agnostic Learning

Adversarial label noise? Concept class doesn't quite fit?

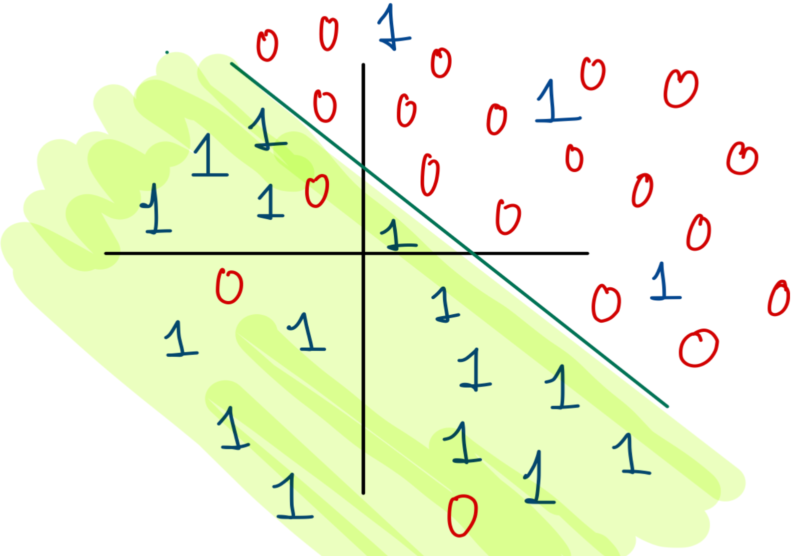
$OPT_{\mathcal{F}}$ = error of best classifier in $\mathcal{F}$

$\mathcal{F}$ = Halfspaces, Low-depth formulas, Monotone functions, etc…

Learning algo

Dataset $S \sim_{i.i.d.} D$,

labeled by **arbitrary** g



$\rightarrow$

$\rightarrow$ Give classifier f

$$\mathrm{err}(f) := \Pr_{x\sim D}[f(x) \neq g(x)], \quad \mathrm{OPT}_{\mathcal{F}} := \min_{f'\in\mathcal{F}} \mathrm{err}(f')$$

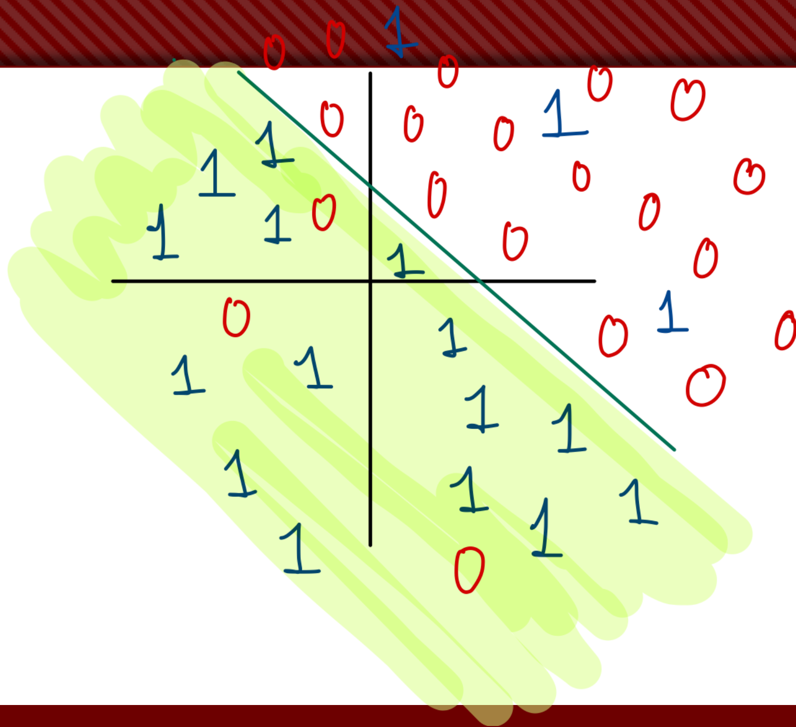
Want: w.h.p. $\mathrm{err}(f) \leq \mathrm{OPT}_{\mathcal{F}} + \epsilon$

Why not always do agnostic learning?

class $\mathcal{F}$ = halfspaces in $\mathbb{R}^d$, i.e.

- 1 on one side of hyperplane,
- 0 on other.

No $2^{o(d)}$ run-time algorithm known.



Computational hardness!!!

e.g. [Guruswami and Raghavendra 06], [Feldman, Gopalan, Khot, and Ponnuswami 06], [Daniely 16]) …

Way around computational hardness: distribution-specific agnostic learning.

Efficient agnostic learning with distributional assumption!

e.g. "data is uniform on $\{0,1\}^d$"

"data comes from Gaussian distribution"

Distributional assumptions for agnostic learning are popular!

e.g., [Kalai, Klivans, Mansour, and Servedio 05], [O'Donnell and Servedio 06], [Blais, O'Donnell, and Wimmer 08], [Klivans, O'Donnell, and Servedio 08], [Gopalan and Servedio 10], [Kane 10], [Wimmer 10], [Harsha, Klivans, and Meka 10], [Diakonikolas, Harsha, Klivans, Meka, Raghavendra, Servedio, and Tan 10], [Cheraghchi, Klivans, Kothari, and Lee], [Awasthi, Balcan, and Long 14], [Dachman-Soled, Feldman, Tan, Wan, and Wimmer 14], [Feldman and Vondrak 15], [Feldman and Kothari 15], [Blais, Canonne, Oliveira, Servedio, and Tan 15], [Canonne, Grigorescu, Guo, Kumar, and Wimmer 17], [Feldman, Kothari, and Vondrak 17], [Diakonikolas, Kane, Kontonis, Tzamos, and Zarifis 21] …

Agnostic Learning Summary

Agnostic learning goal (roughly):

Get classifier f that's ϵ-optimal compared to all classifiers in $\mathcal{F}$

- Fits a nearly-optimal classifier to data with arbitrary labels.
- Fundamental primitive in learning theory.
- Sidestep hardness results by making distributional assumptions.

But how you use this, actually?

You run algorithm on some data.
Real guarantee is:

- Either D=assumption, and therefore $\mathrm{err}(f) \leq OPT_{\mathcal{F}} + \epsilon$
- Or D ≠assumption and all bets are off.

Good, you can rely on predictor f.

Not clear how to proceed!

Bad, you probably want to throw f away and do something else.

Validation doesn't help

Attempt:

1. Run algorithm, get hypothesis f
2. Estimate $\mathrm{err}(f)$.
3. Check $\mathrm{err}(f) \leq OPT_{\mathcal{F}} + \epsilon$

We don't know what $OPT_{\mathcal{F}}(f)$ is!

Use traditional distribution testing?

Standard distribution testing:

Given: $S \sim D$ over $\{0,1\}^d$

Want (w.h.p.):

- D = uniform on $\{0,1\}^d$ → Accept
- D is ϵ-far from uniform on $\{0,1\}^d$ in TV distance→ Reject

Other distributions, earthmover distance: Still $2^{\Omega(d)}$ samples

Need $\Theta\left(\sqrt{\text{domain size}}\right) = \Theta(2^{d/2})$ samples.

Run-times in learning theory:

- "Efficient": $\mathrm{poly}(d/\epsilon)$
- "Dimension-efficient": $d^{O_\epsilon(1)}$

$2^{d^{1-\Omega(1)}}$ occasionally acceptable

Use traditional distribution testing?

Traditional distribution testing too expensive for us.

(See text [Cannone '22] for more info on the subject.)

Need to do something else.

However, ideas coming from distribution testing will be crucial for us.

More achievable goal:
Is data “good enough” for algorithm?

[Rubinfeld Vasilyan STOC'23]: Testable agnostic learning

$\mathcal{F}$ = Linear Classifiers, Low-depth formulas, Monotone functions, etc…

Testable agnostic learning algo

Dataset $S \sim D$,

labeled by arbitrary g →

→ **Accept,** Give classifier f

OR

Output "**Reject**, assumption is wrong!"

$\mathrm{err}(f) \coloneqq \Pr_{x \sim D}[f(x) \neq g(x)]$, $\quad \mathrm{OPT}_{\mathcal{F}} := \min_{f' \in \mathcal{F}} \mathrm{err}(f')$

Want 1) **Completeness**: $D =$ uniform on $\{0,1\}^d \rightarrow$ w.h.p. will accept and $\mathrm{err}(f) \leq \mathrm{OPT}_{\mathcal{F}} + \epsilon$

2) **Soundness** for **any** D: w.h.p. if algo accepts $\rightarrow \mathrm{err}(f) \leq \mathrm{OPT}_{\mathcal{F}} + \epsilon$.

Part II: Testable Agnostic Learning via Moment-Matching.

[Rubinfeld, Vasilyan '23]: testable agnostic learners exist for:

Run-time of the same order as optimal.

Need $d^{\tilde{\Theta}(1/\epsilon^2)}$ just for standard agnostic learning

$\mathcal{F}$ = **Halfspaces** on inputs from

- Uniform distribution on $\{0,1\}^d$
  - Testable agnostic learner with run-time $d^{\tilde{O}(1/\epsilon^4)}$
- Gaussian distribution over $\mathbb{R}^d$
  - Testable agnostic learner with run-time $d^{\tilde{O}(1/\epsilon^4)}$

First version of [RV'23] only had Gaussian. Uniform added in version 2, is simultaneous work with [Gollakota Kothari Klivans 23], who use different approach and get better ϵ-dependence of $d^{\tilde{O}(1/\epsilon^2)}$

Our tester in [RV23]: moment-matching test.

For D = Uniform, this is the k-wise independence tester. (e.g. [Alon Goldreich Mansour '03] [Alon Andoni Kaufmann Matulef Rubinfeld Xie '07])

- Set $k \leftarrow \tilde{O}(1/\epsilon^4)$.
- Draw $d^{O(k)}$ examples.
- For every monomial $m(x) = x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_d^{\alpha_d}$ of degree at most k:
  - Check

$$\left| E_{x \sim D_{\text{assumption}}}[m(x)] - E_{x \sim \{\text{Examples given to us}\}}[m(x)] \right| \leq \frac{\epsilon}{d^{O(k)}}$$

  - Check fails → **Reject**
- All checks pass → **Accept**

Useful ingredient: framework of [KKMS'05]

Agnostic learning framework via low-degree poly regression algorithm [Kalai Klivans Mansour Servedio FOCS '05]

Very general tool for distribution-specific agnostic learning.

Approximation → Learning.

You need to prove:
Halfspaces ϵ-approximated by degree-k polys relative to distribution D in L_1-norm

[KKMS '05] →

You get:
Agnostic learning algorithm for Halfspaces under distribution D in time $d^{O(k)}$ with error ϵ

$$\mathbb{E}_{x\sim D}|h(x) - P(x)| \le \epsilon$$

Testable agnostic learner for halfspaces

Ingredients:

- Tester: check that degree-k moments are right.
- Agnostic learning framework via low degree poly regression algorithm [Kalai Klivans Mansour Servedio FOCS '05]

How do we analyze this?

Known:
Every halfspace ϵ-approximated by degree-$\tilde{O}(1/\epsilon^4)$ poly relative to uniform distribution

[RV'23] shows stronger statement:
Every halfspace ϵ-approximated by degree-$\tilde{O}(1/\epsilon^4)$ polynomial relative to any $\tilde{O}(1/\epsilon^4)$-wise independent distribution

This novel statement crucial for analysis

Proof in [RV'23] uses Chebychev polynomials and critical index machinery of [Diakonikolas, Gopalan, Jaiswal, Servedio and Viola 2010]

Sandwiching polynomials

This approach introduced by [Gollakota, Klivans, Kothari STOC '23].

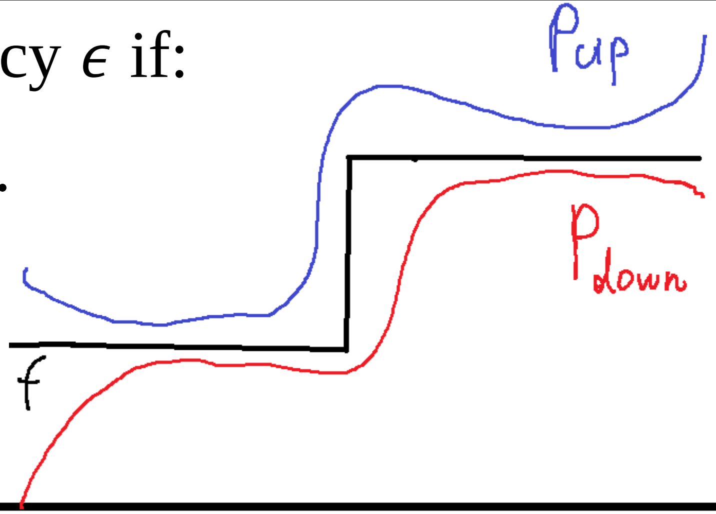
([RV '23] used different proof)

Definition: class $\mathcal{F}$ has **sandwiching** degree of k with accuracy ϵ if:

For any f in $\mathcal{F}$ there exist P_{up} and P_{down} of degree $\leq k$ s.t.

i) For every x in $\mathbb{R}^d$, $P_{\text{down}}(x) \leq f(x) \leq P_{\text{up}}(x)$

ii) $\mathbb{E}_{x\sim\{0,1\}^d}\left[P_{\text{up}}(x) - P_{\text{down}}(x)\right] \leq \epsilon$



Lemma[GKK '23]: class $\mathcal{F}$ has sandwiching degree of $\leq k$ with accuracy ϵ .

Every f in $\mathcal{F}$ is ϵ-approximated by degree-k polynomial w.r.t any k-wise independent D

Proof: $\mathbb{E}_{x\sim D}|f(x) - P_{\text{down}}(x)| \leq \mathbb{E}_{x\sim D}\left(P_{\text{up}}(x) - P_{\text{down}}(x)\right) =$

$$= \mathbb{E}_{x\sim\{0,1\}^d}(P_{\text{up}}(x) - P_{\text{down}}(x)) \leq \epsilon$$

How to bound sandwiching degree?

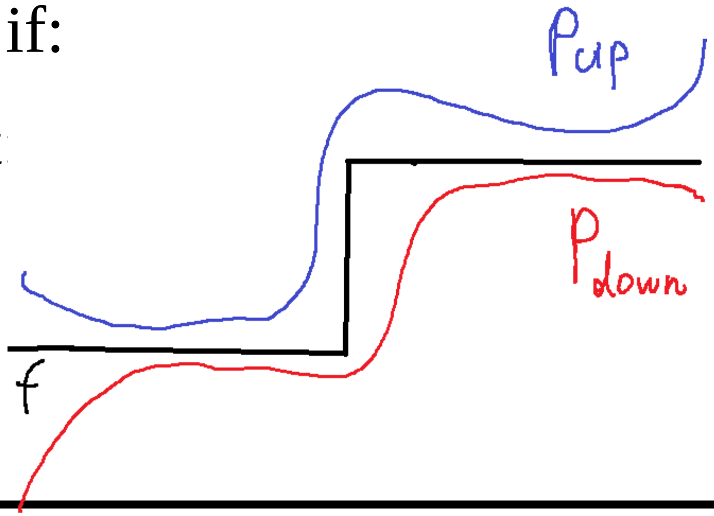
Fooling well-studied in pseudorandomness because lets you approx $\mathbb{E}_{x\sim\{0,1\}^d}[f(x)]$ deterministically.

Definition: f has **sandwiching** degree of $\leq k$ with accuracy ϵ if:

For any f in $\mathcal{F}$ there exist P_{up} and P_{down} of degree $\leq k$ s.t

i) For every x in $\mathbb{R}^d$, $P_{\text{down}}(x) \leq f(x) \leq P_{\text{up}}(x)$

ii) $\mathbb{E}_{x\sim N(0,I_d)}\left[P_{\text{up}}(x) - P_{\text{down}}(x)\right] \leq \epsilon$



Definition: function $f\colon\{0,1\}^d \to \{0,1\}$ is ϵ-fooled by k-wise independent distributions if

for all k-wise independent D over $\{0,1\}^d$ have $\left|\mathbb{E}_{x\sim\{0,1\}^d}[f(x)] - \mathbb{E}_{x\sim D}[f(x)]\right| \leq \epsilon$

[L. Bazzi FOCS '07]: f is ϵ-fooled by k-wise independent distributions $\rightleftarrows$

$\rightleftarrows$ f has sandwiching degree $\leq k$ with accuracy ϵ

Overall recap

Works for other classes $\mathcal{F}$ too (as long as fooled by k-wise independent)

[Diakonikolas, Gopalan, Jaiswal, Servedio, Viola '09] $\tilde{O}(1/\epsilon^2)$-wise independent distributions fool halfspaces

[Bazzi '07]

Any halfspace can be ϵ-sandwiched by a pair of degree-$\tilde{O}(1/\epsilon^2)$ polynomials

[GKK'23]

Every halfspace ϵ-approximated by $\tilde{O}(1/\epsilon^2)$-degree polynomial relative to any $\tilde{O}(1/\epsilon^2)$-wise independent distribution

[KKMS'05]

$d^{\tilde{O}(1/\epsilon^2)}$-time agnostic learning for halfspaces under $\tilde{O}(1/\epsilon^2)$-wise independent distributions

Moment-matching test

$d^{\tilde{O}(1/\epsilon^2)}$-time testable agnostic learning algorithm for halfspaces under uniform distribution over $\{0,1\}^d$

Price of assumption-testing

Weird:
Agnostic learning algorithm is similar – low degree poly regression

- Learn monotone functions over Uniform on $\{0,1\}^d$
  - $2^{\tilde{O}\left(\frac{\sqrt{d}}{\epsilon}\right)}$-time agnostic learning algorithm [Bshouty Tamon 96, KKMS05]
  - [RV '23] Testable agnostic learning needs $2^{\Omega(d)}$ samples
- Learn convex sets over Gaussian on $\mathbb{R}^d$
  - $2^{\tilde{O}\left(\frac{\sqrt{d}}{\epsilon^4}\right)}$-time agnostic learning algorithm [Klivans, O'Donnell, Servedio 08]
  - [RV '23] Testable agnostic learning needs $2^{\Omega(d)}$ samples

What's different between halfspaces and monotone functions?

Known: halfspaces well-approximated by $poly(1/\epsilon)$ −degree polys relative to uniform distribution

We show stronger statement:
Every halfspace well-approximated by $poly(1/\epsilon)$ −degree polynomial relative to $poly(1/\epsilon)$ −wise independent distribution

Known: monotone functions well approximated by $\sqrt{d}/\epsilon$ − degree polys relative to uniform distribution

~~Are monotone functions well-approximated by $\sqrt{d}/\epsilon$ − degree polynomial relative to $\sqrt{d}/\epsilon$−wise independent distribution?~~

~~Monotone functions well-approximated by $\sqrt{d}/\epsilon$ − degree polynomial relative to $\sqrt{d}/\epsilon$−wise independent distribution~~

Part III: Poly-time Testable Agnostic Learning

Follow-up Work: Testable Agnostic Learning in Polynomial Time

Believed to be optimal due to SQ lower bound

$\mathcal{F}$ = half-spaces* D = standard Gaussian

Task	Guarantee	Run-time
Agnostic learning	$\mathrm{err}(f) \leq OPT_{\mathcal{F}} + \epsilon$	$d^{\mathrm{poly}(1/\epsilon)}$ [KKMS '05]
Semi-agnostic learning	$\mathrm{err}(f) \leq O(OPT_{\mathcal{F}}) + \epsilon$	$\mathrm{poly}(d/\epsilon)$ [Awasthi, Balcan, Long '14]

[Gollakota Klivans Stavropoulos Vasilyan ICLR '24] give testable semi-agnostic learner in this setting (see also [Diakonikolas Kane Kontonis Liu Zarifis NeurIPS '23])

* For [ABL '14], and new results above $\mathcal{F}$ = origin-centered half-spaces

Follow-up Work: Testable Agnostic Learning in Polynomial Time

$\mathcal{F}$ = half-spaces* D = standard Gaussian

Task	Guarantee	Run-time
Semi-agnostic learning	$\mathrm{err}(f) \leq O(OPT_{\mathcal{F}}) + \epsilon$	$\mathrm{poly}(d/\epsilon)$ [Awasthi, Balcan, Long '14]

[Gollakota Klivans Stavropoulos Vasilyan ICLR '24] give testable semi-agnostic learner in this setting (see also [Diakonikolas Kane Kontonis Liu Zarifis NeurIPS '23])

Testing degree-$\mathrm{poly}(1/\epsilon)$ moments now too slow.
Overcome this obstacle using new type of tester.

* For [ABL '14], and new results above $\mathcal{F}$ = origin-centered half-spaces

Our tester in in [GKSV '23a] in more detail

- Take $\hat{v} \leftarrow$ [ABL'14, DKTZ 20]
- Break $[-\sqrt{\log 1/\epsilon}, \sqrt{\log 1/\epsilon}]$ into buckets of width ϵ. Assign each example x_i to bucket containing $x_i \cdot \hat{v}$. Check that following hold:

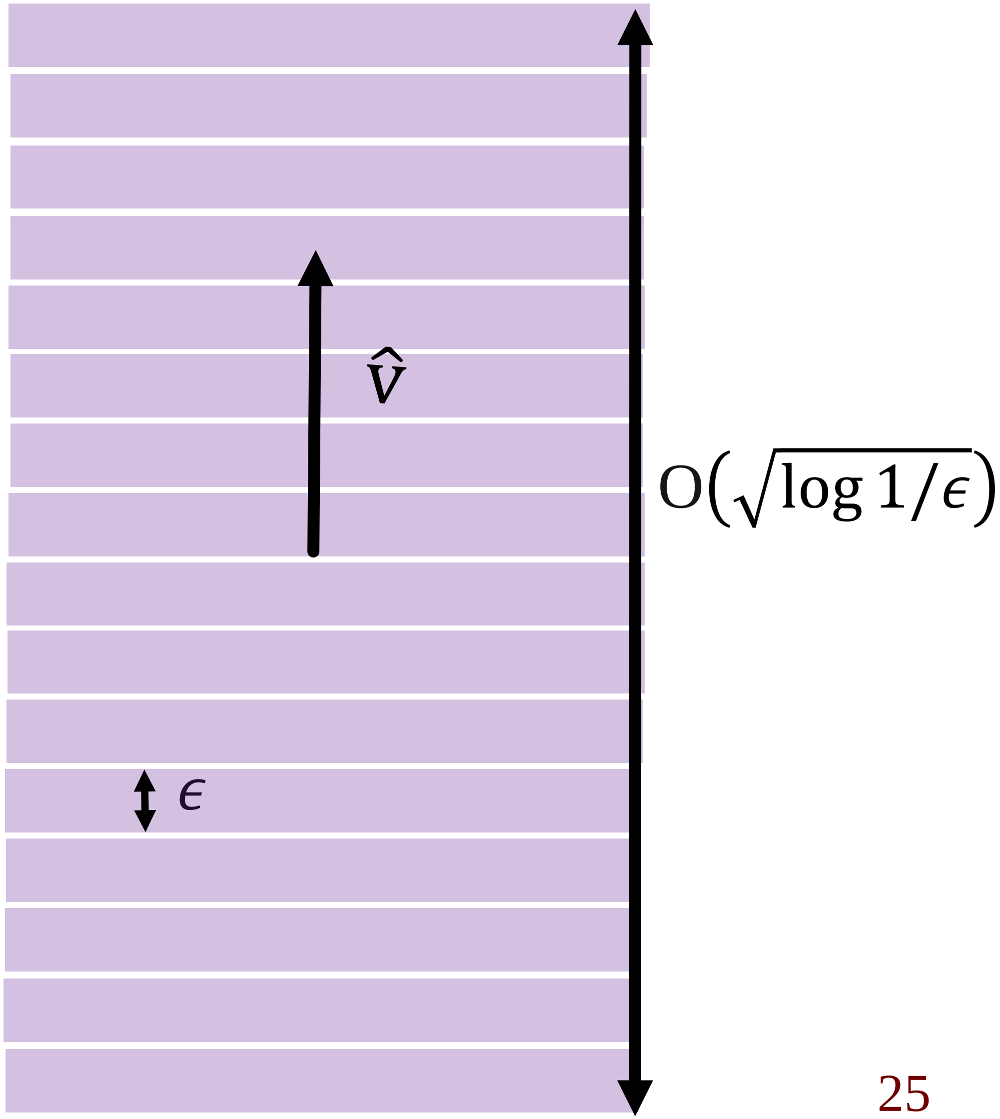
(i) $\Pr_{x \sim D}\left[x \cdot \hat{v} \in \left[-\sqrt{\log 1/\epsilon}, \sqrt{\log 1/\epsilon}\right]\right] \leq 10\epsilon$

(ii) {fraction of examples in each bucket} $\in [\epsilon^2, \epsilon]$

(iv) For each bucket:

a) Project examples to subspace $\perp \hat{v}$

b) Run degree-4 moment test on projected points



Part IV: Universal Testable Agnostic Learning

More follow-up work: testing assumptions for families of distributions

Tester accepts **every** distribution in large family.

$\mathcal{F}$ = half-spaces* D = **Any** isotropic strongly log-concave

Task	Guarantee	Run-time
Semi-agnostic learning	$\mathrm{err}(f) \leq O(OPT_{\mathcal{F}}) + \epsilon$	$\mathrm{poly}(d, 1/\epsilon)$ [Awasthi, Balcan, Long '14]

[Gollakota Klivans Stavropoulos Vasilyan 23b NeurIPS] give testable agnostic learner in this setting.

Techniques include sum-of-squares relaxations and certifiable hypercontractivity [Kothari, Steinhardt '17]

Follow-up work: testing assumptions for families of distributions

Tester accepts **every** distribution in large family.

$\mathcal{F}$ = half-spaces* D = **Any** isotropic ~~strongly~~ log-concave

Can handle even larger class, if KLS conjecture true.

Task	Guarantee	Run-time
Semi-agnostic learning	$\mathrm{err}(f) \leq O(OPT_{\mathcal{F}}) + \epsilon$	$\mathrm{poly}(d, 1/\epsilon)$ [Awasthi, Balcan, Long '14]

[Gollakota Klivans Stavropoulos Vasilyan 23b] give testable agnostic learner in this setting.

* For [ABL '14], and new results above $\mathcal{F}$ = origin-centered half-spaces

How sum-of-squares relaxations come in

Critical property of isotropic log-concave D:

$$\max_{\mathrm{v}:\,\|\mathrm{v}\|_2=1} \mathbb{E}_{\mathrm{x}\sim D}\left((\mathrm{v}\cdot\mathrm{x})^k\right) \leq k!$$

Want to make sure dataset S has this property.

Issue: for $k > 2$ not known how to compute/approximate $\max_{\mathrm{v}:\,\|\mathrm{v}\|_2=1} \mathbb{E}_{\mathrm{x}\sim S}\left((\mathrm{v}\cdot\mathrm{x})^k\right)$ for worst-case S.

Study average-case version?

Problem:

- given S from isotropic log-concave distribution D.
- Say **yes** if $\max_{v:\, \|v\|_2=1} \mathbb{E}_{x\sim S}\left((v\cdot x)^k\right) \leq 10k!$
- Say **no** otherwise.

Algorithm that always says **yes** succeeds with very high probability.

Study average-case **certification** (aka 1-sided testing)

[Kothari, Steinhardt ’17] study certification problem:

- given S from isotropic log-concave distribution D.
- Say **yes** if $\max_{\mathrm{v}:\, \|\mathrm{v}\|_2=1} \mathbb{E}_{\mathrm{x}\sim S}\left((\mathrm{v}\cdot\mathrm{x})^k\right) \le 10k!$ **and produce certificate proving this**.
- Say **no** otherwise.

[Kothari, Steinhardt ’17] give poly(d^k) algorithm.

- Based on sum-of-squares semidefinite relaxations.
- For isotropic log-concave D, analysis conditional on KLS conjecture.
- For isotropic **strongly** log-concave D, analysis unconditional.

Open problems

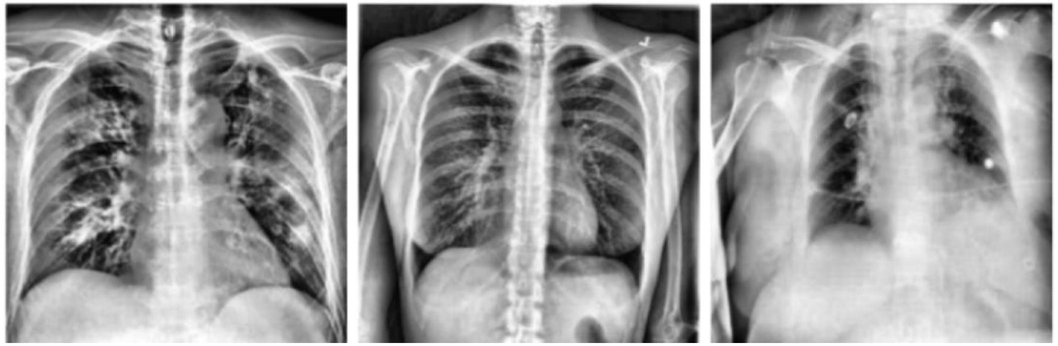
- Is sandwiching degree bound **required** for testable agnostic learning?
- Testable agnostic learners for intersections of k halfspaces under Gaussian/uniform on $\{0,1\}^d$? $d^{\tilde{O}(\mathrm{poly}(k)/\epsilon^2)}$ run-time known but maybe can match $d^{\tilde{O}(\mathrm{polylog}(k)/\epsilon^2)}$ run-time of non-testable agnostic learners?
- What other assumptions in TCS can we test?

Part V: Testing Distribution Shift, the Framework

[Stavropoulos, Klivans, Vasilyan COLT ‘24]

Supervised learning revisited

Labelled dataset S_{train}



+1 -1 +1

Unlabelled dataset S_{test}

Learning Algo

Label $f(x)$ for all x in S_{test}

-1 +1

PAC learning: the standard theoretical framework

D = uniform on $\{0,1\}^d$, Standard Gaussian in R^d, etc…

$\mathcal{F}$ = Linear Classifiers, Low-depth formulas, Functions of Linear classifiers, Monotone functions, etc…

Learning Algo

Dataset $S_{\text{train}} \sim D$, labeled by $g \in \mathcal{F}$

Unlabeled dataset $S_{\text{test}} \sim D$

Label $f(x)$ for all x in S_{test}

$$\text{err}_{\text{test}}(f) := \Pr_{x \sim S_{\text{test}}}[f(x) \neq g(x)]$$

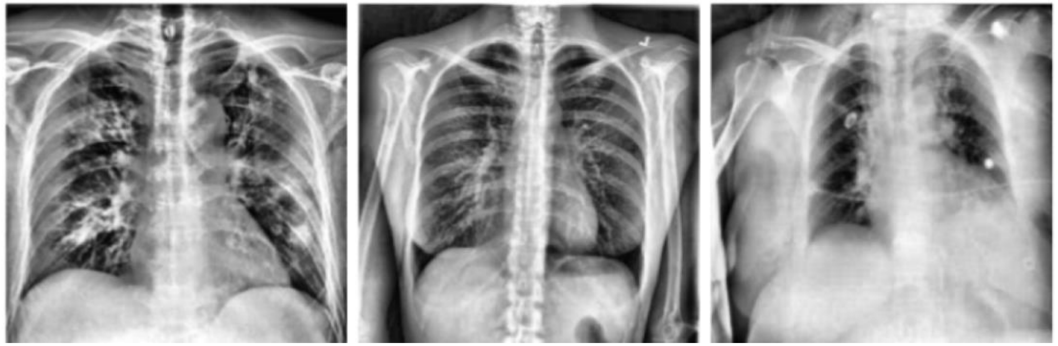
Want: w.h.p. $\text{err}_{\text{test}}(f) \leq \epsilon$

Critical assumption:
S_{train} and S_{test} come from same D

Supervised learning

What if different hospitals/ X-ray machines?

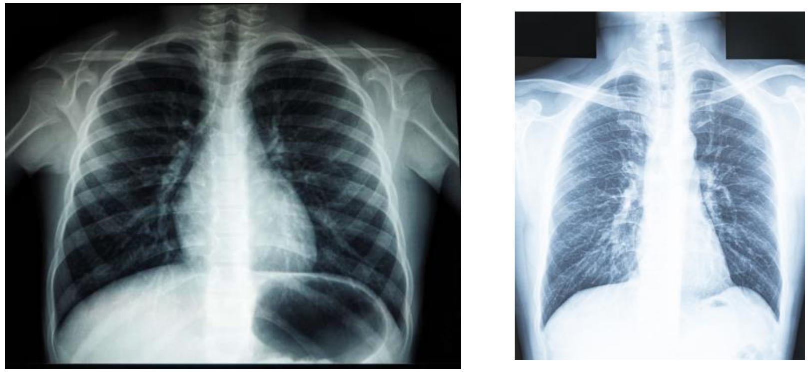
Labelled dataset S_{train}



+1 -1 +1

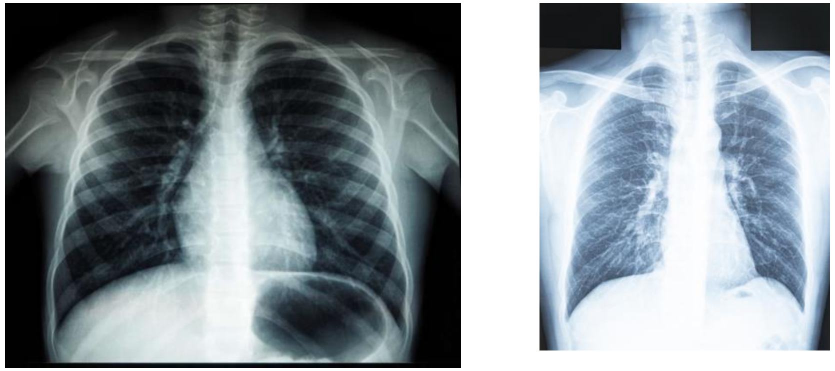
Learning Algo

Label $f(x)$ for all x in S_{test}



-1 +1

Unlabelled dataset S_{test}



Distribution shift can lead to bad predictions

D_{train} = Standard Gaussian in $\mathbb{R}^d$, uniform on $\{0,1\}^d$, etc…

$\mathcal{F}$ = Linear Classifiers, Low-depth Formulas, Intersections of Linear classifiers, Monotone functions, etc…

Learning Algo

Dataset $S_{\text{train}} \sim D_{\text{train}}$, labeled by $g \in \mathcal{F}$

Unlabeled dataset $S_{\text{test}} \sim D_{\text{test}}$

Label $f(x)$ for all x in S_{test}

Concern: $\Pr_{x \sim D_{\text{train}}}[f(x) \neq g(x)] \leq \epsilon$

but $\Pr_{x \sim S_{\text{test}}}[f(x) \neq g(x)] \gg \epsilon$

Can't trust the labeling!

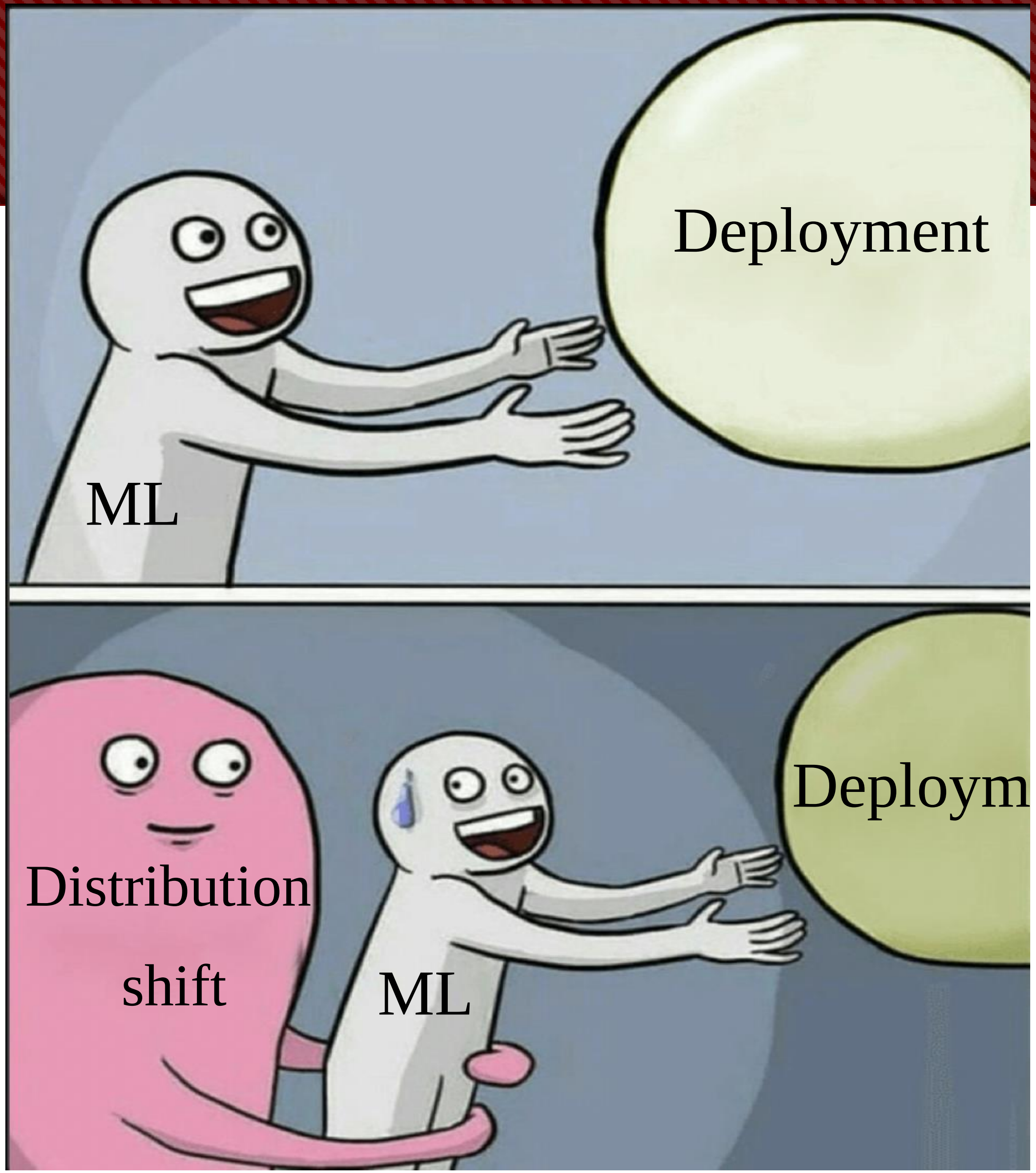
Happens all the time in practice.

Leads to unexpected wrong predictions!

Distribution shift

Mitigating distribution shift remains one of the major challenges of machine learning.

For example, classifiers trained on data from one hospital often fail to generalize to other hospitals [ZBL+18, WOD+21, TCK+22].



Common goal in ML: mitigate distribution shift

Our work [Stavropoulos, Klivans, Vasilyan COLT ‘24]:
Theory framework for mitigating distribution shift.

Want to be confident in labeling given by learning algo.

Goal: raise alarm instead of assigning bad labels

Concern: $\Pr_{x \sim D_{\text{train}}}[f(x) \neq g(x)] \leq \epsilon$

but $\Pr_{x \sim S_{\text{test}}}[f(x) \neq g(x)] \gg \epsilon$

Can’t trust the labeling!

New framework:
Testable Learning with Distribution Shift (TDS Learning)

D_{train} = Standard Gaussian in R^d, uniform on $\{0,1\}^d$, etc…

$\mathcal{F}$ = Linear Classifiers, Low-depth formulas, Functions of Linear classifiers, Monotone functions, etc…

Learning Algo

Dataset $S_{\text{train}} \sim D_{\text{train}}$, labeled by $g \in \mathcal{F}$

Unlabeled dataset $S_{\text{test}} \sim D_{\text{test}}$

Accept and Label $f(x)$ for all x in S_{test}

OR

Output "**Reject**, there is distribution shift"

$$\text{err}(f) \coloneqq \Pr_{x \sim S_{\text{test}}} [f(x) \neq g(x)]$$

Want w.h.p. 1) **Completeness**: $D_{\text{test}} = D_{\text{train}} \rightarrow$ will accept and $\text{err}(f) \leq \epsilon$

2) **Soundness** for **any** D_{test}: if algo accepts $\rightarrow \text{err}(f) \leq \epsilon$.

Goal [SKV ’24]: Develop techniques for TDS learning for various D_{train} and $\mathcal{F}$.

Part VI: Previous work on distribution shift

a) Domain adaptation
b) PQ learning

Work on domain-adaptation

Work [S Ben-David, Blitzer, Crammer Pereira NeurIPS '06], [Blitzer, Crammer, Kulesza, Pereira, Wortma NeurIPS '07] and [Mansour, Mohri, and Rostamizadeh COLT '09] give bounds on $\Pr_{x\sim S_2}[f(x) \neq g(x)]$ in terms of

$$\Delta_{\mathcal{F}}(S_{\text{test}}) := \max_{f_1, f_2 \in \mathcal{F}} \left(\left| \Pr_{x \in S_{\text{test}}}[f_1(x) \neq f_2(x)] - \Pr_{x \in S_{\text{train}}}[f_1(x) \neq f_2(x)] \right| \right)$$

And similar quantities, involving enumeration over $\mathcal{F}$.

Not known how to compute in time $2^{o(d)}$.

Work on PQ learning

Framework studied in [Goldwasser, A. Kalai, Y. Kalai, Montasser NeurIPS ‘20], [A. Kalai, Kanade ALT ’21].

PQ learning → TDS learning

(roughly) PQ learning requires to reject/reject individual elements in S_{test}

[GKKM ’20] studies sample complexity, not run-time.

[KK ‘21] gives poly(d/ϵ) algorithm for parities over $\{0,1\}^d$

Part VII: the moment-matching method and L_2-sandwiching polynomials

Moment matching method

- Dataset $S_{\text{train}} \sim_{iid} \{0,1\}^d$, labeled by $f \in \mathcal{F}$
- Unlabeled dataset $S_{\text{test}} \sim D_{\text{test}}$

Can also consider for e.g. D_{train} = standard Gaussian.

Parameter: k

TDS-learning via moment-matching:

1. For every monomial m over $\mathbb{R}^d$ with $\deg(m) \leq 2k$:

   If $\left|\mathbb{E}_{x \sim S_{\text{test}}}[m(x)] - \mathbb{E}_{x \sim \{0,1\}^d}[m(x)]\right| > \frac{\epsilon}{d^{2k}}$, then **Reject**.

2. $P^* \leftarrow \operatorname{argmin}_{P \text{ of degree } k} \left(\mathrm{E}_{x \sim S_{\text{train}}}\left[\left(g(x) - P(x)\right)^2\right]\right)$
3. **Accept** and label each x in S_{test} as $\operatorname{sign}(P^*(x))$

Run-time: $d^{O(k)}$.

For which k will this work?

Sandwiching polynomials

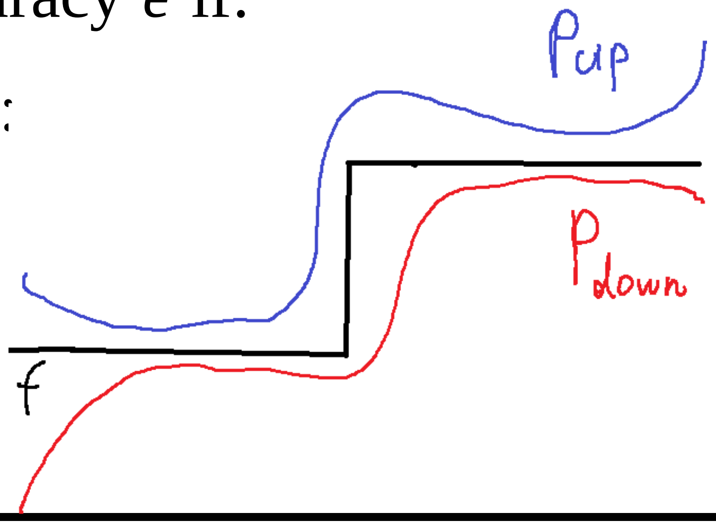
Idea for using sandwiching polynomials inspired by [Gollakota, Klivans, Kothari STOC '23].

Definition: class $\mathcal{F}$ has $\boldsymbol{L_2}$**-sandwiching** degree of k with accuracy ϵ if:

For any f in $\mathcal{F}$ there exist P_{up} and P_{down} of degree $\leq k$ s.t:

i) For every x in $\mathbb{R}^d$, $P_{\text{down}}(x) \leq f(x) \leq P_{\text{up}}(x)$

ii) $\mathbb{E}_{x\sim\{0,1\}^d}\left[\left(P_{\text{up}}(x) - P_{\text{down}}(x)\right)^2\right] \leq \epsilon$



Transfer Lemma[KSV '24]: class $\mathcal{F}$ has $\boldsymbol{L_2}$**-sandwiching** degree of k with accuracy ϵ

⇓

Moment-matching with parameter k is a TDS-learning algorithm

for $\mathcal{F}$ with error $\leq \epsilon$ under $D_{\text{train}} \sim \{0,1\}^d$.

run-time = $d^{O(k)}$

But which function classes have L_2-sandwiching polynomials?

In [KSV '24] we use techniques from **pseudorandomness** to show a number of function classes have low-degree L_2-sandwiching polynomials.

E.g. an intersection of ℓ halfspaces has

L_2-sandwiching polynomials of degree $\tilde{O}(\ell^6/\epsilon^2)$

this + Transfer Lemma → TDS learning in time $d^{\tilde{O}(\ell^6/\epsilon^2)}$

for D_{train}=Standard Gaussian OR uniform in $\{0,1\}^d$.

Why techniques from pseudorandomness useful?

Well-studied in pseudorandomness because lets you approx $\mathbb{E}_{x\sim\{0,1\}^d}[f(x)]$ deterministically.

Definition: function $f:\{0,1\}^d \to \{0,1\}$ is ϵ-**fooled** by k-wise independent distributions if

for all k-wise independent D over $\{0,1\}^d$ have $\left|\mathbb{E}_{x\sim\{0,1\}^d}[f(x)] - \mathbb{E}_{x\sim D}[f(x)]\right| \leq \epsilon$

[L. Bazzi FOCS '07]: f fooled by k-wise independent distributions $\rightleftarrows$

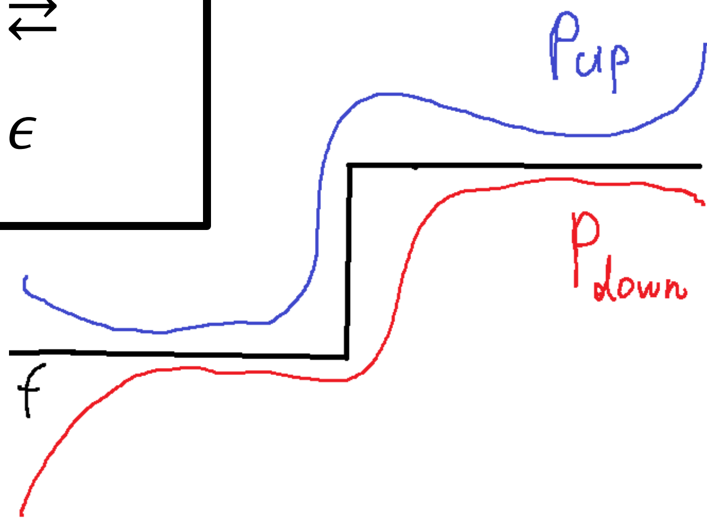
$\rightleftarrows$ f has sandwiching polynomials of degree k and accuracy ϵ

Definition: f has **sandwiching** degree of $\leq k$ with accuracy ϵ if:

For any f in $\mathcal{F}$ there exist P_{up} and P_{down} of degree $\leq k$ s.t:

i) For every x in $\mathbb{R}^d$, $P_{\text{down}}(x) \leq f(x) \leq P_{\text{up}}(x)$

ii) $\mathbb{E}_{x\sim\{0,1\}^d}\left[P_{\text{up}}(x) - P_{\text{down}}(x)\right] \leq \epsilon$



Almost what we need.

Note [Gollakota, Klivans, Kothari STOC '23] used this connection for testing assumptions of agnostic learning algorithms.

We take ideas from pseudorandamness and use them to bound the L_2-sandwiching degree.

Specifically, we build on [Gopalan, O'Donnell, Wu, Zuckerman CCC '10] and [Diakonikolas, Gopalan, Jaiswal, Servedio, Emanuele Viola STOC '07].

Is moment-matching method best?

- Dataset $S_{\text{train}} \sim_{iid} N(0, I_d)$,
  labeled by $g(x) \ = \text{sign}(\text{v} \cdot x + \theta)$
- Unlabeled dataset $S_{\text{test}} \sim D_{\text{test}}$

We show: can do better with other methods!

Let $\mathcal{F}$={general halfspaces}.

Moment-matching requires d^{1/ϵ^2} run-time.

We give algorithm with run-time $d^{\log\left(\frac{1}{\epsilon}\right)}$.

Open: How fast can one TDS-learn halfspaces for $S_{\text{train}} \sim_{iid} \{0,1\}^d$? Best we know is $d^{\tilde{O}\,(1/\epsilon^2)}$ time

Conjecture: impossible to do **distribution-free** TDS learning of halfspaces in time $d^{O_\epsilon(1)}$

We also show evidence of optimality:

Theorem No statistical query algorithm with run-time $d^{\log^{0.99}\left(\frac{1}{\epsilon}\right)}$

Surprising, because regular learning can be done in time poly(d/ϵ).

Conclusion

- We introduce novel frameworks for testing assumptions of learning algorithms.
- We give computationally efficient algorithms.
- Traditional distribution testing algorithms too slow. However, we build on ideas from distribution testing such as k-wise independence testing.

Thank you!