Learning Circuits with Few Negations

Boolean functions are not that monoton(ous).

Clément Canonne

LIAFA – 2015

Introduction

Introduction: learning

Goal: fixed, known class of Boolean functions $\mathcal{C} \subseteq 2^{\{0,1\}^n}$, and unknown $f \in \mathcal{C}$. How to learn f efficiently, i.e. output a hypothesis $\hat{f} \simeq f$?

Introduction: learning

Goal: fixed, known class of Boolean functions $\mathcal{C} \subseteq 2^{\{0,1\}^n}$, and unknown $f \in \mathcal{C}$. How to learn f efficiently, i.e. output a hypothesis $\hat{f} \simeq f$?

Many flavors:

With membership queries: Can we *approximately* learn f (in Hamming distance, with high probability) from *queries* of the form $x? \rightsquigarrow f(x)$

$$\Pr_{x \sim \{0,1\}^n}[f(x) \neq \hat{f}(x)] \leq \varepsilon \qquad \text{(w.h.p.)}$$

Introduction: learning

Goal: fixed, known class of Boolean functions $\mathcal{C} \subseteq 2^{\{0,1\}^n}$, and unknown $f \in \mathcal{C}$. How to learn f efficiently, i.e. output a hypothesis $\hat{f} \simeq f$?

Many flavors:

With membership queries: Can we *approximately* learn f (in Hamming distance, with high probability) from *queries* of the form $x? \rightsquigarrow f(x)$

$$\Pr_{x\sim\{0,1\}^n}[f(x) \neq \hat{f}(x)] \leq \varepsilon \qquad \text{(w.h.p.)}$$

PAC-learning: unknown underlying distribution D on $\{0,1\}^n$. Can we approximately learn f (with high probability) from random examples $\langle x, f(x)\rangle$ – where each x is a *sample* drawn independently from D?

$$\Pr_{x\sim D}[f(x) \neq \hat{f}(x)] \leq \varepsilon \qquad \text{(w.h.p.)}$$

Introduction: learning

Goal: fixed, known class of Boolean functions $\mathcal{C} \subseteq 2^{\{0,1\}^n}$, and unknown $f \in \mathcal{C}$. How to learn f efficiently, i.e. output a hypothesis $\hat{f} \simeq f$?

Many flavors:

With membership queries: Can we *approximately* learn f (in Hamming distance, with high probability) from *queries* of the form $x? \rightsquigarrow f(x)$

$$\Pr_{x\sim\{0,1\}^n}[f(x) \neq \hat{f}(x)] \leq \varepsilon \qquad \text{(w.h.p.)}$$

PAC-learning: unknown underlying distribution D on $\{0,1\}^n$. Can we approximately learn f (with high probability) from random examples $\langle x, f(x)\rangle$ – where each x is a *sample* drawn independently from D?

$$\Pr_{x\sim D}[f(x) \neq \hat{f}(x)] \leq \varepsilon \qquad \text{(w.h.p.)}$$

PAC-learning under the uniform distribution: PAC-learning is too hard, so assume D is uniform.

$$\Pr_{x\sim\{0,1\}^n}[f(x) \neq \hat{f}(x)] \leq \varepsilon \qquad \text{(w.h.p.)}$$

Goal: fixed, known class of Boolean functions $\mathcal{C} \subseteq 2^{\{0,1\}^n}$, and unknown $f \in \mathcal{C}$. How to learn f efficiently, i.e. output a hypothesis $\hat{f} \simeq f$?

Many flavors:

With membership queries: Can we *approximately* learn f (in Hamming distance, with high probability) from *queries* of the form $x? \rightsquigarrow f(x)$

$$\Pr_{x\sim\{0,1\}^n}[f(x) \neq \hat{f}(x)] \leq \varepsilon \qquad \text{(w.h.p.)}$$

PAC-learning: unknown underlying distribution D on $\{0,1\}^n$. Can we approximately learn f (with high probability) from random examples $\langle x, f(x)\rangle$ – where each x is a *sample* drawn independently from D?

$$\Pr_{x\sim D}[f(x) \neq \hat{f}(x)] \leq \varepsilon \qquad \text{(w.h.p.)}$$

PAC-learning under the uniform distribution: PAC-learning is too hard, so assume D is uniform.

$$\Pr_{x\sim\{0,1\}^n}[f(x) \neq \hat{f}(x)] \leq \varepsilon \qquad \text{(w.h.p.)}$$

uniform PAC learning $\preceq$ learning with queries

Monotone functions (1)

For circuit complexity theorists:

Definition. *A Boolean function* $f\colon \{0,1\}^n \to \{0,1\}$ *is* monotone *if it is computed by a Boolean circuit with no negations (only AND and OR gates).*

Monotone functions (1)

For circuit complexity theorists:

Definition. *A Boolean function* $f\colon \{0,1\}^n \to \{0,1\}$ *is* monotone *if it is computed by a Boolean circuit with no negations (only* AND *and* OR *gates).*

For analysis of Boolean functions enthusiasts:

Definition. *A Boolean function* $f\colon \{0,1\}^n \to \{0,1\}$ *is* monotone *if for any* $x \preceq y$ *in* $\{0,1\}^n$, $f(x) \leq f(y)$.

Monotone functions (1)

For circuit complexity theorists:

Definition. *A Boolean function* $f\colon \{0,1\}^n \to \{0,1\}$ *is* monotone *if it is computed by a Boolean circuit with no negations (only AND and OR gates).*

For analysis of Boolean functions enthusiasts:

Definition. *A Boolean function* $f\colon \{0,1\}^n \to \{0,1\}$ *is* monotone *if for any* $x \preceq y$ *in* $\{0,1\}^n$, $f(x) \leq f(y)$.

For people with a twisted mind:

Definition. *A Boolean function* $f\colon \{0,1\}^n \to \{0,1\}$ *is* monotone *if* $f(0^n) \leq f(1^n)$, *and* f *changes value* at most once *on any increasing chain from* 0^n *to* 1^n.

(These definitions are equivalent.)

Examples.

The majority function (1 iff at least half the votes are positive): more votes cannot make a candidate lose.
The s-clique function (1 iff the input graph contains a clique of size s): more edges cannot remove a clique.
The dictator function (1 iff $x_1 = 1$): more voters have no influence anyway.

Monotone functions (2)

Can we learn them?

Learning the class $\mathcal{C}^n$ of monotone Boolean functions from uniform examples (to error ε) can be done in time $2^{\tilde{O}\left(\sqrt{n}/\varepsilon\right)}$. [BT96]

Monotone functions (2)

Can we learn them?

Learning the class $\mathcal{C}^n$ of monotone Boolean functions from uniform examples (to error ε) can be done in time $2^{\tilde{O}\left(\sqrt{n}/\varepsilon\right)}$. [BT96]

Monotone functions (2)

Can we learn them?

Learning the class $\mathcal{C}^n$ of monotone Boolean functions from uniform examples (to error ε) can be done in time $2^{\tilde{O}\left(\sqrt{n}/\varepsilon\right)}$. [BT96]

Can we do better?

Monotone functions (2)

Can we learn them?

Learning the class $\mathcal{C}^n$ of monotone Boolean functions from uniform examples (to error ε) can be done in time $2^{\tilde{O}\left(\sqrt{n}/\varepsilon\right)}$. [BT96]

Can we do better?

Learning the class $\mathcal{C}^n$ *from membership queries* (to error $\frac{1}{\sqrt{n}\log n}$) requires query complexity $2^{\Omega(n)}$. [BT96]

Can we learn them?

Learning the class $\mathcal{C}^n$ of monotone Boolean functions from uniform examples (to error ε) can be done in time $2^{\tilde{O}\left(\sqrt{n}/\varepsilon\right)}$. [BT96]

Can we do better?

Learning the class $\mathcal{C}^n$ *from membership queries* (to error $\frac{1}{\sqrt{n}\log n}$) requires query complexity $2^{\Omega(n)}$. [BT96]

Are we done here?

Outline of the talk

Introduction

Generalizing monotone functions: $\mathcal{C}_t^n$.

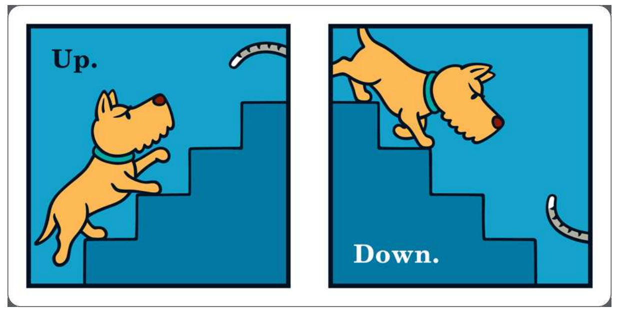
Learning $\mathcal{C}_t^n$: Upper bound.

Learning $\mathcal{C}_t^n$: Lower bound.

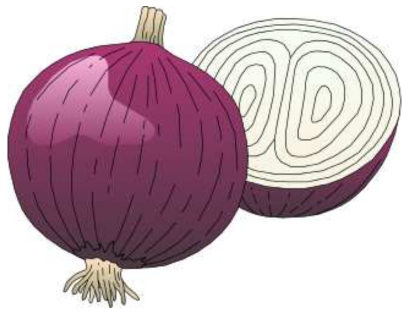
Conclusion and Open Problem(s).

Plan in more detail

- Generalizing monotone functions to “k-alternating:” two views, reconciled by Markov’s Theorem.



- A structural theorem: characterizing these new functions as combination of simpler ones $\rightsquigarrow$ upper bound on learning k-alternating functions, almost “for free.”



- Lower bound: a succession and combination thereof (from monotone… to monotone to k-alternating: hardness amplification)

Generalizing monotone functions: $\mathcal{C}_t^n$.

k-alternating functions (1)

For circuit complexity theorists:

Definition. *A Boolean function* $f\colon \{0,1\}^n \to \{0,1\}$ *has* inversion complexity t *if it can be computed by a Boolean circuit with* t *negations (besides* AND *and* OR *gates), but no less.*

k-alternating functions (1)

For circuit complexity theorists:

Definition. *A Boolean function $f\colon \{0,1\}^n \to \{0,1\}$ has* inversion complexity t *if it can be computed by a Boolean circuit with t negations (besides AND and OR gates), but no less.*

For people with a twisted mind:

Definition. *A Boolean function $f\colon \{0,1\}^n \to \{0,1\}$ is k-*alternating *if f changes value* at most k times *on any increasing chain from 0^n to 1^n.*

(Analysis of Boolean functions enthusiasts, stay with us?)

Examples.

The "not-too-many" function (1 iff between 40% and 60% of the votes are positive): more votes can harm a candidate.
The s-clique-but-no-Hamiltonian function (1 iff the input graph contains a clique of size s, but no Hamiltonian cycle): more edges can make things worse.
The Highlander function (1 iff exactly one of the x_i's is 1): there shall be only one.

k-alternating functions (2)

But are these definitions the same? Related?

k-alternating functions (2)

But are these definitions the same? Related?

Theorem 4 (Markov's Theorem [Mar57])**.** *Let $f\colon \{0,1\}^n \to \{0,1\}$ be a function which is not identically 0. Then f is k-alternating if and only if it has inversion complexity $O(\log k)$.*

k-alternating functions (2)

But are these definitions the same? Related?

Theorem 7 (Markov's Theorem [Mar57])**.** *Let $f\colon \{0,1\}^n \to \{0,1\}$ be a function which is not identically 0. Then f is k-alternating if and only if it has inversion complexity $O(\log k)$.*

A refinement of this characterization:

Theorem 8. *If f is k-alternating, then it can be written $f(x) = h(m_1(x), \dots, m_k(x))$, where each $m_i(x)$ is monotone and h is either the parity function or its negation. Conversely, any function of this form is k-alternating.*

Corollary 9. *Every $f \in \mathcal{C}_t^n$ can be expressed as $f = h(m_1, \dots, m_T)$ where h is either Parity_T or its negation, each $m_i\colon \{0,1\}^n \to \{0,1\}$ is monotone, and $T = O(2^t)$.*

k-alternating functions (2)

But are these definitions the same? Related?

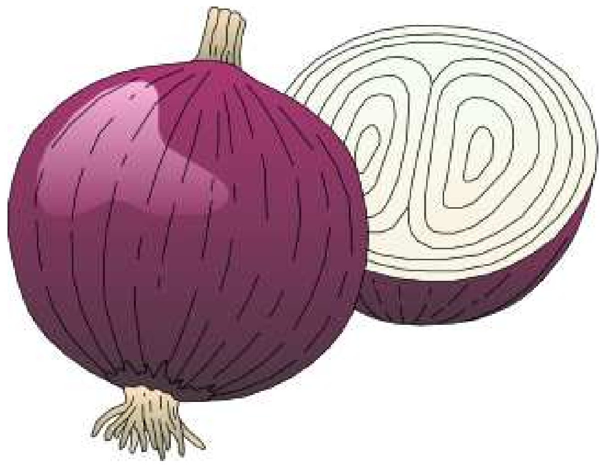
Theorem 10 (Markov's Theorem [Mar57]). *Let $f\colon \{0,1\}^n \to \{0,1\}$ be a function which is not identically 0. Then f is k-alternating if and only if it has inversion complexity $O(\log k)$.*

A refinement of this characterization:

Theorem 11. *If f is k-alternating, then it can be written $f(x) = h(m_1(x), \dots, m_k(x))$, where each $m_i(x)$ is monotone and h is either the parity function or its negation. Conversely, any function of this form is k-alternating.*

Corollary 12. *Every $f \in \mathcal{C}_t^n$ can be expressed as $f = h(m_1, \dots, m_T)$ where h is either Parity_T or its negation, each $m_i\colon \{0,1\}^n \to \{0,1\}$ is monotone, and $T = O(2^t)$.*

Proof (and interpretation). the m_i's are successive nested layers:



□

Learning $\mathcal{C}_t^n$: Upper bound.

Theorem 13. *There is a uniform-distribution learning algorithm which learns any unknown $f \in \mathcal{C}_t^n$ from random examples to error ε in time $n^{O(2^t\sqrt{n}/\varepsilon)}$.*

Theorem 15. *There is a uniform-distribution learning algorithm which learns any unknown $f \in \mathcal{C}_t^n$ from random examples to error ε in time $n^{O(2^t\sqrt{n}/\varepsilon)}$. (Recall the $n^{O(\sqrt{n}/\varepsilon)}$ for monotone functions, i.e. $t = 0$.)*

Theorem 17. *There is a uniform-distribution learning algorithm which learns any unknown $f \in \mathcal{C}_t^n$ from random examples to error ε in time $n^{O(2^t\sqrt{n}/\varepsilon)}$. (Recall the $n^{O(\sqrt{n}/\varepsilon)}$ for monotone functions, i.e. $t = 0$.)*

Proof. Recall the *influence* of a Boolean functions is defined as

$$\mathbf{Inf}[f] = \sum_{i=1}^{n} \mathbf{Inf}_i[f], \quad \text{where} \quad \mathbf{Inf}_i[f] = \Pr_{x\in\{0,1\}^n}[f(x) \neq f(x^{\oplus i})]$$

and that monotone functions each have total influence at most $\sqrt{n}$. Moreover, *we can learn functions with good Fourier concentration*:

Theorem 18 (Low-Degree Algorithm ([LMN93])). *Let $\mathcal{C}$ be a class of Boolean functions such that for $\varepsilon > 0$ and $\tau = \tau(\varepsilon, n)$,*

$$\sum_{|S|>\tau} \hat{f}(S)^2 \leq \varepsilon$$

for any $f \in \mathcal{C}$. Then $\mathcal{C}$ can be learned from uniform random examples in time $\mathrm{poly}(n^\tau, 1/\varepsilon)$*.*

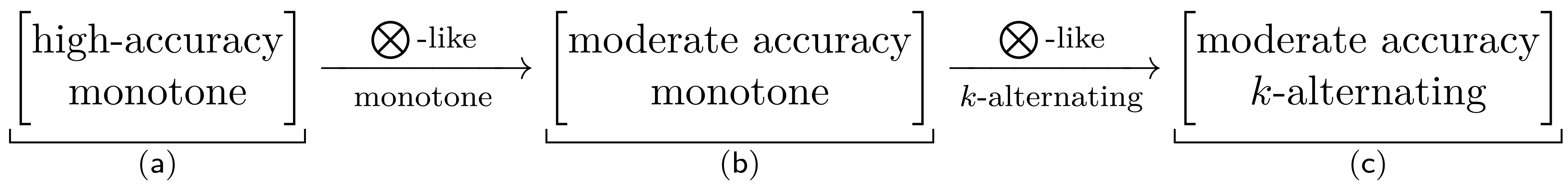
Combining the decomposition theorem, a union bound, some massaging, and the above, k-alternating functions have total influence at most $k\sqrt{n}$, and we get the theorem. □

Learning $\mathcal{C}_t^n$: Lower bound.

Three-step program

$$\underbrace{\begin{bmatrix}\text{high-accuracy}\\ \text{monotone}\end{bmatrix}}_{\text{(a)}} \xrightarrow[\text{monotone}]{\bigotimes\text{-like}} \underbrace{\begin{bmatrix}\text{moderate accuracy}\\ \text{monotone}\end{bmatrix}}_{\text{(b)}} \xrightarrow[k\text{-alternating}]{\bigotimes\text{-like}} \underbrace{\begin{bmatrix}\text{moderate accuracy}\\ k\text{-alternating}\end{bmatrix}}_{\text{(c)}}$$

Three-step program



(a) Monotone functions are hard to learn *well*. (A simple extension of [BT96].)

Learning *monotone* functions to (very small) error $\frac{.1}{\sqrt{n}}$ requires 2^{Cn} queries, for some absolute $C > 0$.

(b) Monotone functions are hard to learn, *period*. (Hardness amplification and the previous result.)

Learning *monotone* functions to (almost any) error ε requires $2^{\Omega(\sqrt{n}/\varepsilon)}$ queries.

(c) k-alternating functions are hard to learn, *too*! (Hardness amplification again – and a truncated parity.)

Learning *k-alternating* functions to (almost any) error ε requires $2^{\Omega(k\sqrt{n}/\varepsilon)}$ queries.

In more detail: tools for (b) and (c) – bear with me

Definition (Composition). *For* $f\colon \{0,1\}^m \to \{0,1\}$ *and* $g\colon \{0,1\}^r \to \{0,1\}$, $g \otimes f$ *is the function on* $n = mr$ *variables defined by*

$$(g \otimes f)(x) \stackrel{\text{def}}{=} g(f(x_1, \dots, x_m), \dots, f(x_{(r-1)m+1}, \dots, x_{rm}))$$

For any $g\colon \{0,1\}^r \to \{0,1\}$ *and* $\mathcal{F}_m \subseteq 2^{\{0,1\}^m}$, $g \otimes \mathcal{F}_m = \{\, g \otimes f \,:\, f \in \mathcal{F}_m \,\}$ *and* $g \otimes \mathcal{F} = \{g \otimes \mathcal{F}_m\}_{m \geq 1}$.

Definition (Noise stability). *For* $f\colon \{0,1\}^n \to \{0,1\}$, *the* noise stability *of* f *at* $\eta \in [-1,1]$ *is*

$$\operatorname{Stab}_\eta(f) \stackrel{\text{def}}{=} 1 - 2\Pr[\, f(x) \neq f(y)\,]$$

where $x \sim \{0,1\}^n$, *and* x *and* y *are* η-correlated.

In more detail: tools for (b) and (c) – bear with me

Definition (Composition). *For $f\colon\{0,1\}^m\to\{0,1\}$ and $g\colon\{0,1\}^r\to\{0,1\}$, $g\otimes f$ is the function on $n=mr$ variables defined by*

$$(g\otimes f)(x)\stackrel{\text{def}}{=}g(f(x_1,\dots,x_m),\dots,f(x_{(r-1)m+1},\dots,x_{rm}))$$

For any $g\colon\{0,1\}^r\to\{0,1\}$ and $\mathcal{F}_m\subseteq 2^{\{0,1\}^m}$, $g\otimes\mathcal{F}_m=\{\ g\otimes f\ :\ f\in\mathcal{F}_m\ \}$ and $g\otimes\mathcal{F}=\{g\otimes\mathcal{F}_m\}_{m\geq 1}$.

Definition (Noise stability). *For $f\colon\{0,1\}^n\to\{0,1\}$, the* noise stability *of f at $\eta\in[-1,1]$ is*

$$\mathrm{Stab}_\eta(f)\stackrel{\text{def}}{=}1-2\Pr[\,f(x)\neq f(y)\,]$$

where $x\sim\{0,1\}^n$, and x and y are η-correlated.

Definition (Bias and expected bias). *The* bias *of a Boolean function $h\colon\{0,1\}^n\to\{0,1\}$ is* $\mathrm{bias}(h)\stackrel{\text{def}}{=}\max(\Pr[\,h=1\,],\Pr[\,h=0\,])$ *while the* expected bias of h at δ *is defined as*

$$\mathrm{ExpBias}_\delta(h)\stackrel{\text{def}}{=}\mathbb{E}_\rho[\mathrm{bias}(h_\rho)]$$

where ρ is a random δ-restriction on n coordinates.

In more detail: tools for (b) and (c) – bear with me

Definition (Composition). *For* $f\colon\{0,1\}^m \to \{0,1\}$ *and* $g\colon\{0,1\}^r \to \{0,1\}$, $g\otimes f$ *is the function on* $n = mr$ *variables defined by*

$$(g\otimes f)(x) \stackrel{\text{def}}{=} g(f(x_1,\dots,x_m),\dots,f(x_{(r-1)m+1},\dots,x_{rm}))$$

For any $g\colon\{0,1\}^r \to \{0,1\}$ *and* $\mathcal{F}_m \subseteq 2^{\{0,1\}^m}$, $g\otimes\mathcal{F}_m = \{\, g\otimes f \,:\, f\in\mathcal{F}_m \,\}$ *and* $g\otimes\mathcal{F} = \{g\otimes\mathcal{F}_m\}_{m\geq 1}$.

Definition (Noise stability). *For* $f\colon\{0,1\}^n \to \{0,1\}$, *the* noise stability *of* f *at* $\eta\in[-1,1]$ *is*

$$\mathrm{Stab}_\eta(f) \stackrel{\text{def}}{=} 1 - 2\Pr[\, f(x) \neq f(y)\,]$$

where $x\sim\{0,1\}^n$, *and* x *and* y *are* η-correlated.

Definition (Bias and expected bias). *The* bias *of a Boolean function* $h\colon\{0,1\}^n \to \{0,1\}$ *is* $\mathrm{bias}(h) \stackrel{\text{def}}{=} \max(\Pr[\,h=1\,], \Pr[\,h=0\,])$ *while the* expected bias of h at δ *is defined as*

$$\mathrm{ExpBias}_\delta(h) \stackrel{\text{def}}{=} \mathbb{E}_\rho[\mathrm{bias}(h_\rho)]$$

where ρ *is a random* δ*-restriction on* n *coordinates.*

Theorem 21 (Theorem 12 of [FLS11]). *Fix* $g\colon\{0,1\}^r \to \{0,1\}$, *and let* $\mathcal{F}$ *be a class of* m*-variable functions with "very small bias." Let* A *be a membership query algorithm that learns* $g\otimes\mathcal{F}$ *to accuracy* $\mathrm{ExpBias}_\gamma(g)+\epsilon$ *using* $T(m,r,1/\epsilon,1/\gamma)$ *queries. Then there is a membership query algorithm to learn* $\mathcal{F}$ *to accuracy* $1-\gamma$, *using* $O(T\cdot\mathrm{poly}(m,r,1/\epsilon,1/\gamma))$ *membership queries.*

In more detail: step (b)

Theorem 22. *There exists a class $\mathcal{H}_n$ of balanced n-variable monotone Boolean functions such that for any $\varepsilon \in [1/n^{1/6}, .49]$, learning $\mathcal{H}_n$ to error ε requires $2^{\Omega\left(\sqrt{n}/\varepsilon\right)}$ membership queries.*

In more detail: step (b)

Theorem 23. *There exists a class $\mathcal{H}_n$ of balanced n-variable monotone Boolean functions such that for any $\varepsilon \in [1/n^{1/6}, .49]$, learning $\mathcal{H}_n$ to error ε requires $2^{\Omega\left(\sqrt{n}/\varepsilon\right)}$ membership queries.*

Sketch.

- Choose suitable $m, r = \omega(1)$ such that $mr = n$.

In more detail: step (b)

Theorem 24. *There exists a class $\mathcal{H}_n$ of balanced n-variable monotone Boolean functions such that for any $\varepsilon \in [1/n^{1/6}, .49]$, learning $\mathcal{H}_n$ to error ε requires $2^{\Omega\left(\sqrt{n}/\varepsilon\right)}$ membership queries.*

Sketch.

- Choose suitable $m, r = \omega(1)$ such that $mr = n$.
- Take the "Mossel–O'Donnell function" g_r [MO03] (a balanced monotone function minimally stable under very small noise)
  (Why? We want $\mathrm{ExpBias}_\gamma(g_r) + \epsilon' \leq 1 - \varepsilon$*, and less stable means smaller expected bias)*

In more detail: step (b)

Theorem 25. *There exists a class $\mathcal{H}_n$ of balanced n-variable monotone Boolean functions such that for any $\varepsilon \in [1/n^{1/6}, .49]$, learning $\mathcal{H}_n$ to error ε requires $2^{\Omega\left(\sqrt{n}/\varepsilon\right)}$ membership queries.*

Sketch.

- Choose suitable $m, r = \omega(1)$ such that $mr = n$.
- Take the "Mossel–O'Donnell function" g_r [MO03] (a balanced monotone function minimally stable under very small noise)
  (Why? We want $\text{ExpBias}_\gamma(g_r) + \epsilon' \leq 1 - \varepsilon$, *and less stable means smaller expected bias)*
- Apply the hardness amplification theorem on $g_r \otimes \mathcal{G}_m$, $\mathcal{G}_m$ being the "hard class of monotone functions" from Step (a).

In more detail: step (b)

Theorem 26. *There exists a class $\mathcal{H}_n$ of balanced n-variable monotone Boolean functions such that for any $\varepsilon \in [1/n^{1/6}, .49]$, learning $\mathcal{H}_n$ to error ε requires $2^{\Omega\left(\sqrt{n}/\varepsilon\right)}$ membership queries.*

Sketch.

- Choose suitable $m, r = \omega(1)$ such that $mr = n$.
- Take the "Mossel–O'Donnell function" g_r [MO03] (a balanced monotone function minimally stable under very small noise)
  (Why? We want $\mathrm{ExpBias}_\gamma(g_r) + \epsilon' \leq 1 - \varepsilon$*, and less stable means smaller expected bias)*
- Apply the hardness amplification theorem on $g_r \otimes \mathcal{G}_m$, $\mathcal{G}_m$ being the "hard class of monotone functions" from Step (a).
- Hope all the constants and parameters work out.

In more detail: step (b)

Theorem 27. *There exists a class $\mathcal{H}_n$ of balanced n-variable monotone Boolean functions such that for any $\varepsilon \in [1/n^{1/6}, .49]$, learning $\mathcal{H}_n$ to error ε requires $2^{\Omega\left(\sqrt{n}/\varepsilon\right)}$ membership queries.*

Sketch.

- Choose suitable $m, r = \omega(1)$ such that $mr = n$.
- Take the "Mossel–O'Donnell function" g_r [MO03] (a balanced monotone function minimally stable under very small noise)
  (Why? We want $\mathrm{ExpBias}_\gamma(g_r) + \epsilon' \leq 1 - \varepsilon$, *and less stable means smaller expected bias)*
- Apply the hardness amplification theorem on $g_r \otimes \mathcal{G}_m$, $\mathcal{G}_m$ being the "hard class of monotone functions" from Step (a).
- Hope all the constants and parameters work out.

□

In more detail: step (c)

Theorem 28. *For any function* $k\colon \mathbb{N} \to \mathbb{N}$*, there exists a class* $\mathcal{H}^{(k)}$ *of balanced* $k(n)$*-alternating Boolean functions (on* n *variables) such that, for any* n *sufficiently large and* $\varepsilon > 0$ *such that (i)* $2 \leq k < n^{1/14}$*, and (ii)* $k^{7/3}/n^{1/6} \leq \varepsilon \leq .49$*, learning* $\mathcal{H}^{(k)}$ *to accuracy* $1 - \varepsilon$ *requires* $2^{\Omega\left(k\sqrt{n}/\varepsilon\right)}$ *membership queries.*

In more detail: step (c)

Theorem 29. *For any function $k\colon \mathbb{N} \to \mathbb{N}$, there exists a class $\mathcal{H}^{(k)}$ of balanced $k(n)$-alternating Boolean functions (on n variables) such that, for any n sufficiently large and $\varepsilon > 0$ such that (i) $2 \leq k < n^{1/14}$, and (ii) $k^{7/3}/n^{1/6} \leq \varepsilon \leq .49$, learning $\mathcal{H}^{(k)}$ to accuracy $1 - \varepsilon$ requires $2^{\Omega\left(k\sqrt{n}/\varepsilon\right)}$ membership queries.*

Sketch.

- Choose suitable $m, r = \omega(1)$ such that $mr = n$ and $r \approx k^2$.

In more detail: step (c)

Theorem 30. *For any function $k\colon \mathbb{N} \to \mathbb{N}$, there exists a class $\mathcal{H}^{(k)}$ of balanced $k(n)$-alternating Boolean functions (on n variables) such that, for any n sufficiently large and $\varepsilon > 0$ such that (i) $2 \leq k < n^{1/14}$, and (ii) $k^{7/3}/n^{1/6} \leq \varepsilon \leq .49$, learning $\mathcal{H}^{(k)}$ to accuracy $1-\varepsilon$ requires $2^{\Omega\left(k\sqrt{n}/\varepsilon\right)}$ membership queries.*

Sketch.

- Choose suitable $m, r = \omega(1)$ such that $mr = n$ and $r \approx k^2$.
- Take $\mathsf{Parity}_{k,r}$, the "k-Truncated Parity function on r variables" as combining function, *in lieu* of the previous g_r.

(Why? We want our function to be k-alternating, very little stable, and $r \approx k^2$ instead of k is a technicality)

In more detail: step (c)

Theorem 31. *For any function $k\colon \mathbb{N} \to \mathbb{N}$, there exists a class $\mathcal{H}^{(k)}$ of balanced $k(n)$-alternating Boolean functions (on n variables) such that, for any n sufficiently large and $\varepsilon > 0$ such that (i) $2 \leq k < n^{1/14}$, and (ii) $k^{7/3}/n^{1/6} \leq \varepsilon \leq .49$, learning $\mathcal{H}^{(k)}$ to accuracy $1-\varepsilon$ requires $2^{\Omega\left(k\sqrt{n}/\varepsilon\right)}$ membership queries.*

Sketch.

- Choose suitable $m, r = \omega(1)$ such that $mr = n$ and $r \approx k^2$.
- Take $\mathsf{Parity}_{k,r}$, the "k-Truncated Parity function on r variables" as combining function, *in lieu* of the previous g_r.
  (Why? We want our function to be k-alternating, very little stable, and $r \approx k^2$ instead of k is a technicality)
- Apply the hardness amplification theorem on $\mathsf{Parity}_{k,r} \otimes \mathcal{H}_m$, $\mathcal{H}_m$ being the "hard class of monotone functions" from Step (b).

In more detail: step (c)

Theorem 32. *For any function $k\colon \mathbb{N} \to \mathbb{N}$, there exists a class $\mathcal{H}^{(k)}$ of balanced $k(n)$-alternating Boolean functions (on n variables) such that, for any n sufficiently large and $\varepsilon > 0$ such that (i) $2 \leq k < n^{1/14}$, and (ii) $k^{7/3}/n^{1/6} \leq \varepsilon \leq .49$, learning $\mathcal{H}^{(k)}$ to accuracy $1 - \varepsilon$ requires $2^{\Omega\left(k\sqrt{n}/\varepsilon\right)}$ membership queries.*

Sketch.

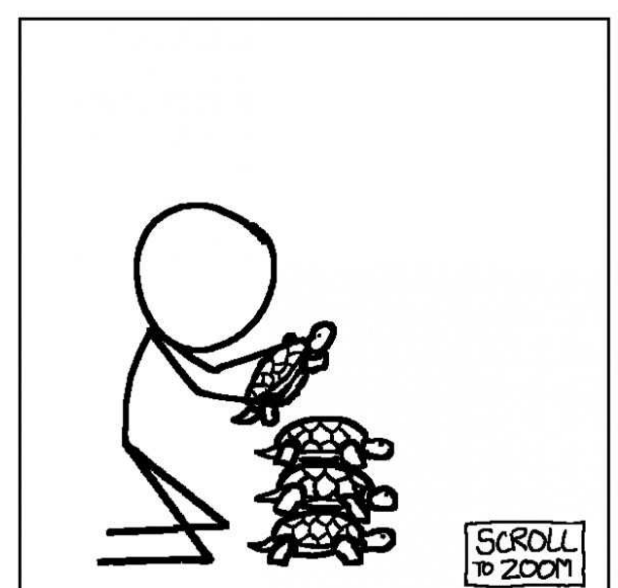
- Choose suitable $m, r = \omega(1)$ such that $mr = n$ and $r \approx k^2$.
- Take $\mathsf{Parity}_{k,r}$, the "k-Truncated Parity function on r variables" as combining function, *in lieu* of the previous g_r.
  (Why? We want our function to be k-alternating, very little stable, and $r \approx k^2$ instead of k is a technicality)
- Apply the hardness amplification theorem on $\mathsf{Parity}_{k,r} \otimes \mathcal{H}_m$, $\mathcal{H}_m$ being the "hard class of monotone functions" from Step (b).
- *Really* hope all the constants and parameters work out.

In more detail: step (c)

Theorem 33. *For any function* $k\colon \mathbb{N} \to \mathbb{N}$, *there exists a class* $\mathcal{H}^{(k)}$ *of balanced* $k(n)$*-alternating Boolean functions (on* n *variables) such that, for any* n *sufficiently large and* $\varepsilon > 0$ *such that (i)* $2 \leq k < n^{1/14}$, *and (ii)* $k^{7/3}/n^{1/6} \leq \varepsilon \leq .49$, *learning* $\mathcal{H}^{(k)}$ *to accuracy* $1-\varepsilon$ *requires* $2^{\Omega\left(k\sqrt{n}/\varepsilon\right)}$ *membership queries.*

Sketch.

- Choose suitable $m, r = \omega(1)$ such that $mr = n$ and $r \approx k^2$.
- Take $\mathsf{Parity}_{k,r}$, the "k-Truncated Parity function on r variables" as combining function, *in lieu* of the previous g_r.
  (Why? We want our function to be k-alternating, very little stable, and $r \approx k^2$ *instead of k is a technicality)*
- Apply the hardness amplification theorem on $\mathsf{Parity}_{k,r} \otimes \mathcal{H}_m$, $\mathcal{H}_m$ being the "hard class of monotone functions" from Step (b).
- *Really* hope all the constants and parameters work out.



□

Conclusion and Open Problem(s).

Open problems

Weak Learning: can one learn $\mathcal{C}_t^n$ to error $\frac{1}{2} - \frac{1}{\mathrm{poly}(n)}$ ("barely better than random") in polynomial time?
(Related) Fourier spectrum: Can we get any further understanding of the Fourier spectrum of k-alternating functions?

Concrete example:

Let f, g be monotone Boolean functions, and $h = \mathsf{Parity}(f, g)$. Can we prove

$$\sum_{|S| \leq 2} \hat{h}(S)^2 \geq \frac{1}{\mathrm{poly}(n)}?$$

Or even $\sum_{|S|\leq 2} \hat{h}(S)^2 > 0$?

Thank you.

Any question?

References

[BT96] N. Bshouty and C. Tamon. On the Fourier spectrum of monotone functions. *Journal of the ACM*, 43(4):747–770, 1996.

[FLS11] V. Feldman, H. K. Lee, and R. A. Servedio. Lower bounds and hardness amplification for learning shallow monotone formulas. *Journal of Machine Learning Research - Proceedings Track*, 19:273–292, 2011.

[LMN93] N. Linial, Y. Mansour, and N. Nisan. Constant depth circuits, Fourier transform and learnability. *Journal of the ACM*, 40(3):607–620, 1993.

[Mar57] A. A. Markov. On the inversion complexity of systems of functions. *Doklady Akademii Nauk SSSR*, 116:917–919, 1957. English translation in [**?**].

[MO03] E. Mossel and R. O'Donnell. On the noise sensitivity of monotone functions. *Random Structures and Algorithms*, 23(3):333–350, 2003.