

On Noise Tolerant Learning of Sparse Parities and Related Problems

ALT 2011

Lev Reyzin

Georgia Institute of Technology

work Joint with Elena Grigorescu and Santosh Vempala

PAC Learning Parities

- [**Def.**] For $x \in \{0,1\}^n$ and $c \in \{0,1\}^n$, let
$$\text{parity}_c(x) = c \cdot x \pmod{2}.$$
- [**Def.**] If c has 1s in only r positions, we call c an **r -parity**.
 - This is known as the “sparse” case.
- [**Model**] For an unknown target c , learner sees labeled examples $(x, \text{parity}_c(x))$ from unknown distribution. i.e.
 $(00110101,1), (10011010,1), (00101111,0), \dots$
- [**Model**] Learner needs to determine c (or, more generally, predict labels of future examples).

PAC Learning Parities is Easy

- draw some examples x:

$(0110, 1)$, $(1001, 1)$, $(1101, 1)$, $(1110, 0)$

- Solve the linear system (mod 2):

$$(1110, 0) - (0110, 1) = (1000, 1)$$

$$(1001, 1) - (1000, 1) = (0001, 0)$$

$$(1101, 1) - (1000, 1) - (0001, 0) = (0100, 0)$$

$$(0110, 1) - (0100, 0) = (0010, 1)$$

$$\text{so, } c = (1010)$$

- For any parity size r , can be done via Gaussian elimination, using n linearly independent examples, taking n^3 time.

PAC Learning Parities with Noise

- Consider the case when **labels** of the examples are **flipped** with probability $\eta < \frac{1}{2}$, say $\eta = 1/10$.
 - called “white label noise” (Angluin & Laird ’87)
 - ideally want algorithm running in time $\text{poly}(n, 1/\epsilon, 1/\delta, 1/(1-2\eta))$

(00110101...,1), (10011010...,1), (00101111...,0), (00110101...,1), (11001010...,0),
(01101111...,0), (01111101...,1), (00111011...,0), (10101111...,0), (00110101...,1),
(10011010...,1), (00110001...,0), (11001100...,1), (01110011...,0), (01101111...,0),
(01111101...,1), (00111011...,0), (10100110...,0), (00111101...,1), (01010000...,0),
(00110100...,0), (00000100...,1), (01110111...,1), (00101111...,0), (10100001...,1),
(10011010...,1), (00110001...,0), ...

- Gaussian elimination completely fails, because to get a (10000000...) vector, need n steps. Even for $\eta = 1/\text{poly}(n)$, likely to make an error.
 - **main issue: we don't know where the errors occur**

PAC Learning “Noisy Parities” Isn’t Easy

- For r -parities, nothing better than n^r (brute force) was known previously.
 - Even for noise rates $1/1000000$ or even $1/\text{poly}(n)$
- Parity has high “statistical query dimension” (Blum et al. ’94) – an unconditional impediment for most learning algorithms.
- Similar or equivalent to hard problems in many other fields: the LWE problem, the shortest vector problem, cryptographic primitives, ...
- Feldman et al (’09), showed that many difficult learning problems “reduce” to learning parities.
 - Learning juntas, learning DNF, learning Decision trees, learning parities under adversarial noise, ...

Some Hope

- When parities are of unrestricted size, the best algorithm (Blum & Kalai & Wasserman '00) takes **time $2^{n/\lg n}$** (2^n is brute-force).
 - Parities cannot be learned faster than brute force in the statistical query model (Kearns '93).
 - This celebrated result separated the classes “noisy PAC” from SQ.
- Gives hope something can be done to improve the n^r in the “sparse” case.
 - We borrow ideas from Hopper & Blum ('01), Klivans & Servedio ('04), and Burhman et al. ('10).

Our results

- r -parities are learnable exactly, under the uniform distribution, in **time** $\approx n^{(1+(2\eta)^2+o(1))r/2}$
 - almost optimal sample complexity
 - extends to arbitrary distributions
 - works under adversarial noise
- Via known results by Feldman et al. ('09), our result gives improved bounds for learning:
 - noisy juntas
 - s -term DNF

Half-Parities (*Fact #1*)

Let c be a parity, and c_1 and c_2 be $r/2$ -parities s.t. $c = c_1 + c_2$

Observe: $\text{parity}_c(x) = \text{parity}_{c_1}(x) + \text{parity}_{c_2}(x)$
 $\text{parity}_{c_1}(x) + \text{parity}_{c_2}(x) + \text{parity}_c(x) = 0$
 $\text{parity}_{c_1}(x) = \text{parity}_{c_2}(x) + \text{parity}_c(x)$

Idea is to search for c_1, c_2 that “compose” c . Since c_1 and c_2 are smaller, this search should be more efficient than brute-force.

Another Useful Fact (*Fact #2*)

- Any two different parities c_1, c_2 are **uncorrelated** under the uniform distribution

$$\Pr_{x \sim U} [\text{parity}_{c_1}(x) = \text{parity}_{c_2}(x)] = \frac{1}{2}.$$

- This implies that if c_1 and c_2 are $r/2$ -parities such that $c_1 + c_2 \neq c$, then

$$\Pr_{x \sim U} [\text{parity}_{c_1}(x) = \text{parity}_{c_2}(x) + \text{parity}_c(x)] = \frac{1}{2}.$$

Algorithm

1. Draw a set S of m examples
2. Evaluate each $r/2$ -parity on S .
 - These give each $r/2$ -parity a “signature.”
 - These make $n^{r/2}$ points (one per half-parity) on the m dimensional Hamming cube. Call this set $H1$.
3. Evaluate each $r/2$ -parity on S , this time xor-ing each evaluation with the label.
 - This makes a second set on the m -dimensional Hamming cube. Call this set $H2$.
4. Find the nearest neighbors $c1, c2$ between $H1$ and $H2$.
5. Return $c1 + c2$

Why Does This Work?

- Two $r/2$ -parities that **compose** the correct r -parity will have expected Hamming **distance = η** . (*from Fact #1*)
 - time: $mn^{r/2}$
- Two $r/2$ -parities that **do not compose** the correct r -parity will have expected Hamming **distance = $1/2$** . (*from Fact #2*)
 - time: $mn^{r/2}$
- There is a **large expected “gap,”** and we take enough samples so that we see that gap w.h.p.
 - samples: $m \approx r \log n$ (*Hoeffding bounds*)
- Given this gap exists, we can use Andoni & Indyk ('06) Nearest neighbor search to find the **closest pair**.
 - takes time $n^{(1+(2\eta)^2+o(1))r/2}$.

Extensions

- We can handle **arbitrary distributions**, with a slightly harder argument.
 - bound too scary to include on this slide
- When $\eta < \varepsilon$, we can get an improved bound using **noise-tolerant Boosting** (Kalai & Servedio '05).
 - but resulting learner is “improper”
- Works for **adversarial noise** (Feldman et al. '09).

Applications to Other Problems

We use reductions from Feldman et al. ('09) to get better bounds for other classes.

1. juntas (with noise): arbitrary functions on r of the n variables
 - previous best: n^r [brute force]
 - now: $n^{(1-2^{1-r}(1-2\eta)+2^{1-2r}(1-2\eta)^2+o(1))r}$
2. s -term DNF (no noise):
 - previous best [Verbeurgt '90]: $n^{\log_2(s/\epsilon)}$
 - now: $n^{(1-\tilde{O}(\epsilon/s)+o(1))\log_2(s/\epsilon)}$

Discussion / Open Problems

- Might this idea extend to $r/3$ -parities, etc?
 - Probably not because of 3-SUM “lower bounds” Erickson ('95).
- Polynomial time algorithm for $\omega(1)$ -parities?
- Is the uniform distribution the hardest?
- Improving $2^{n/\log n}$ in the non-sparse case.
 - Many people have been trying for the last 10 years.