Distributed Learning, Communication Complexity, and Privacy

Maria-Florina Balcan, Georgia Tech

Joint with Avrim Blum, Shai Fine, Yishay Mansour

Distributed Learning

Many ML problems today involve massive amounts of data distributed across multiple locations.

Often would like low error hypothesis wrt the overall distrib.

Distributed Learning

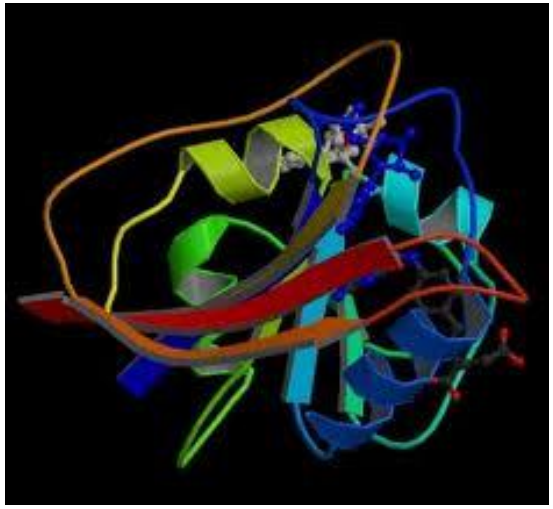
Data distributed across multiple locations.

E.g., medical data

Distributed Learning

Data distributed across multiple locations.

E.g., scientific data



Distributed Learning

- Data distributed across multiple locations.
- Each has a piece of the overall data pie.

- To learn over the combined D, must communicate.

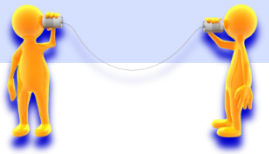
Important question: how much communication?

Plus, privacy & incentives.

Our distributed learning model

- X – instance space. k players.
- Player i can sample from D_i, samples labeled by f.
- Goal: find h that approximates f w.r.t. $D=1/k\ (D_1 + \ldots + D_k)$
- Fix C of VCdim d. Assume $k << d$. [realizable: $f \in C$, agnostic: $f \notin C$]

Goal: learn good h over D, as little communication as possible



- Total communication (bits, examples, hypotheses)
- Rounds of communication.

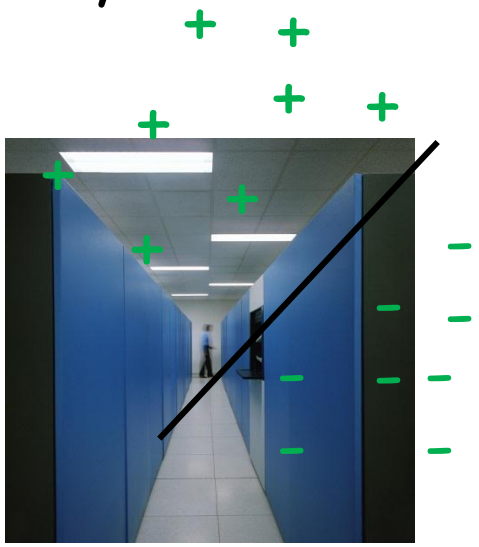
Efficient algos for problems when centralized algos exist.

Interesting special case to think about

k=2. One has the positives and one has the negatives.

- How much communication, e.g., for linear separators?

Player 1



Player 2

Overview of Our Results

Introduce and analyze Distributed PAC learning.

- Generic bounds on communication.
- Broadly applicable communication efficient distributed boosting.
- Tight results for interesting cases (conjunctions, parity fns, decision lists, linear separators over "nice" distrib).

Analysis of privacy guarantees achievable.

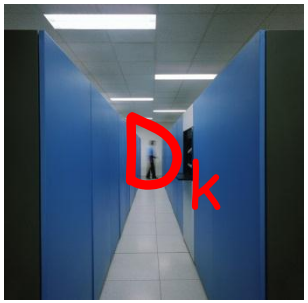
Some simple communication baselines.

Baseline #1

d/ϵ log(1/ϵ) examples, 1 round of communication

- Each player sends d/(ϵk) log(1/ϵ) examples to player 1.
- Player 1 finds consistent h $\in$ C, whp error $\leq \epsilon$ wrt D

...



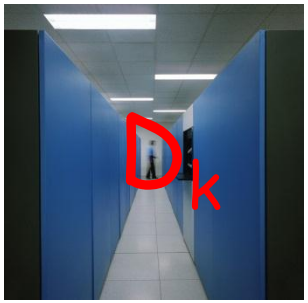
Some simple communication baselines.

Baseline #2 (based on Mistake Bound algos):

M rounds, M examples & hyp, M is mistake-bound of C.

- In each round player 1 broadcasts its current hypothesis.
- If any player has a counterexample, it sends it to player 1. If not, done. Otherwise, repeat.

...



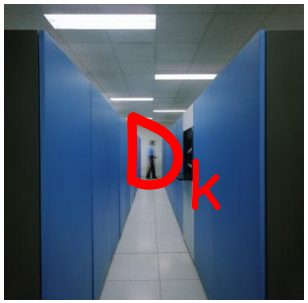
Some simple communication baselines.

Baseline #2 (based on Mistake Bound algos):

M rounds, M examples, M is mistake-bound of C.

- All players maintain same state of an algo A with MB M.
- If any player has an example on which A is incorrect, it announces it to the group.

...

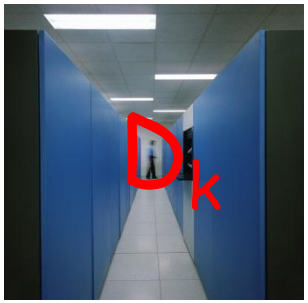


Improving the Dependence on $1/\epsilon$

Baselines provide linear dependence in d and $1/\epsilon$, or M and no dependence on $1/\epsilon$.

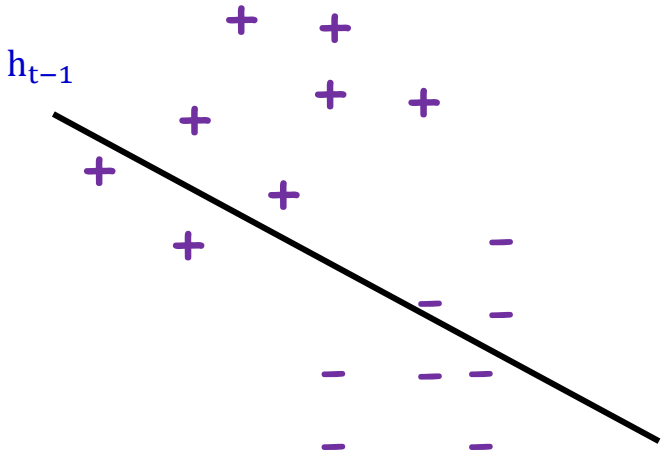
Can get better $O(d \log 1/\epsilon)$ examples of communication!

...



Recap of Adaboost

- Weak learning algorithm A.
- For t=1,2, … ,T
  - Construct D_t on $\{x_1, \ldots, x_m\}$
  - Run A on D_t producing h_t



- D_1 uniform on $\{x_1, \ldots, x_m\}$
- D_{t+1} increases weight on x_i if h_t makes a mistake on x_i ; decreases it on x_i if h_t correct.

Key points:

- $D_{t+1}(x_i)$ depends on $h_1(x_i), \ldots, h_t(x_i)$ and normalization factor that can be communicated efficiently.
- To achieve weak learning it suffices to use O(d) examples.

Distributed Adaboost

- Each player i has a sample S_i from D_i.
- For t=1,2, … ,T
  - Each player sends player 1, enough data to produce weak hypothesis h_t. [For t=1, O(d/k) examples each.]
  - Player 1 broadcasts h_t to all other players.
  - Each player i reweights its own distribution on S_i using h_t and sends the sum of its weights $w_{i,t}$ to player 1.
  - Player 1 determines the #of samples to request from each i [samples O(d) times from the multinomial given by $w_{i,t}/W_t$].

Distributed Adaboost

Can learn any class C with $O(\log(1/\epsilon))$ rounds using $O(d)$ examples + $O(k \log d)$ bits per round.

[efficient if can efficiently weak-learn from $O(d)$ examples]

Proof:

- As in Adaboost, $O(\log 1/\epsilon)$ rounds to achieve error ϵ.
- Per round: $O(d)$ examples, $O(k \log d)$ extra bits for weights, 1 hypothesis.

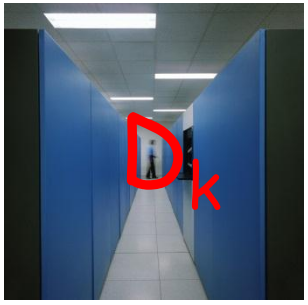
Dependence on $1/\epsilon$, Agnostic learning

Distributed implementation of Robust halving [Balcan-Hanneke'12].

- error $O(OPT)+\epsilon$ using only $O(k \log|C| \log(1/\epsilon))$ examples.

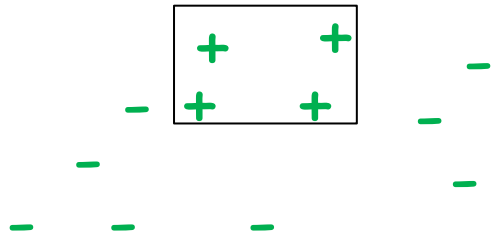
Not computationally efficient in general, but says $O(\log(1/\epsilon))$ possible in principle.

...



Better results for special cases

Intersection-closed when fns can be described compactly .



C is intersection-closed, then *C* can be learned in one round and k hypotheses of total communication.

Algorithm:

- Each i draws S_i of size $O(d/\epsilon \log(1/\epsilon))$, finds smallest h_i in *C* consistent with S_i and sends h_i to player 1.
- Player 1 computes smallest h s.t. $h_i \subseteq h$ for all i.

Key point:

h_i, h never make mistakes on negatives, so $\mathrm{err}_{D_i}(h) \leq \mathrm{err}_{D_i}(h_i) \leq \epsilon$.

Better results for special cases

E.g., conjunctions over $\{0,1\}^d$ [$f(x) = x_2x_5x_9x_{15}$]

- Only $O(k)$ examples sent, $O(kd)$ bits.
  - Each entity intersects its positives.
  - Sends to player 1.
  - Player 1 intersects & broadcasts.

1101111011010111
1111110111001110
1100110011001111
1100110011000110

[Generic methods $O(d)$ examples, or $O(d^2)$ bits total.]

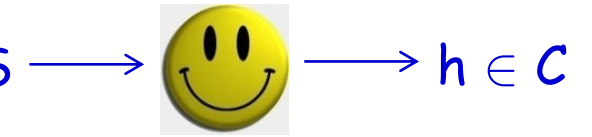
Interesting class: parity functions

- $k = 2, X = \{0,1\}^d$, C = parity fns, $f(x) = x_{i_1} \text{XOR } x_{i_2} \ldots \text{XOR } x_{i_l}$
- Generic methods: O(d) examples, $O(d^2)$ bits.
- Classic CC lower bound for determining if two subspaces intersect implies $\Omega(d^2)$ bits LB for proper learning.

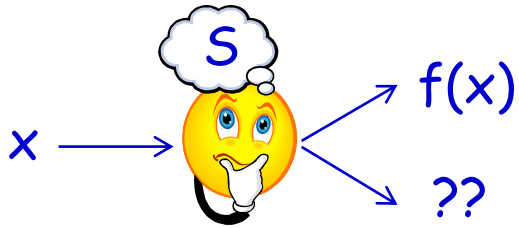
Improperly learn C with O(d) bits of communication!

Key points:

- Can properly PAC-learn C.
  [Given dataset S of size $O(d/\epsilon)$, just solve the linear system]



- Can non-properly learn C in reliable-useful manner [RS'88]



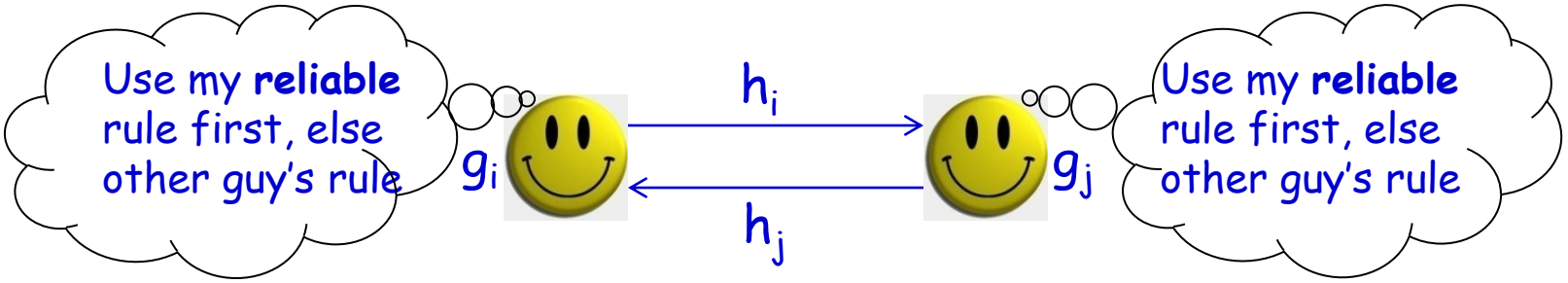
[if x in subspace spanned by S, predict accordingly, else say "?"]

Interesting class: parity functions

Improperly learn C with $O(d)$ bits of communication!

Algorithm:

- Player i properly PAC-learns over D_i to get parity h_i. Also improperly R-U learns to get rule g_i. Sends h_i to player j.
- Player i uses rule R_i: "if g_i predicts, use it; else use h_j"



Key point: low error under D_j because h_j has low error under D_j and since g_i never makes a mistake putting it in front does not hurt.

Distributed PAC learning: Summary

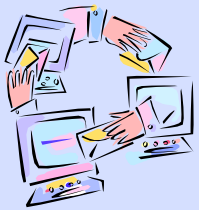
- Communication as a fundamental resource.
- General bounds on communication, communication-efficient distributed boosting.
- Improved bounds for special classes (intersection-closed, parity fns, and linear separators over nice distributions).

Linear Separators

Lemma: α-well spread S [$|\cos(x, x')| \leq \alpha$] and margin $\geq \gamma$, find a consistent hypothesis with $\leq O(k\left(1 + \frac{\alpha}{\gamma^2}\right)$ rounds, each round one hyp. (vector) communicated.

Algorithm: margin perceptron in round-robin.

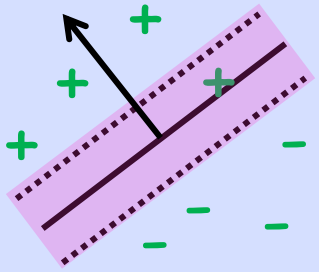
Initially w_0=0.



For t=1,2, … ,

For i=1,2, … ,k

- Player i receives hyp. from previous player.
- Update until $(w_{t,i} \cdot x)y(x) \geq 1$ for all its x.
- Then pass to next player.



Stop if number of updates in meta-round t $\leq 1/\alpha$.

Linear Separators

Lemma: α-well spread S [$|\cos(x, x')| \leq \alpha$] and margin $\geq \gamma$, find a consistent hypothesis with $\leq O(k\left(1 + \frac{\alpha}{\gamma^2}\right)$ rounds, each round one hyp. (vector) communicated.

Algorithm: margin perceptron in round-robin.

Proof Idea:

- Margin perceptron makes $\leq \frac{3}{\gamma^2}$ updates.
- If # of updates in meta-round t $\leq 1/\alpha$, then $w_{t,k}$ is consistent with all players data (every update can hurt by at most α).
- So, total # meta-rounds $\leq 1 + \frac{3\,\alpha}{\gamma^2}$.

Linear Separators

Corollary: Over any non-concentrated D [density bounded by c·unif], can learn with $O((d \log d)^{1/2})$ vectors communicated (constant k, ϵ).

Proof (sketch):

- Plug in $\alpha = O(((\log d)/d)^{1/2})$. [in a poly-size sample, whp all pairs x,x' will have $|\cos(x,x')| = O(((\log d)/d)^{1/2})$]
- Plug in $\gamma = O(1/d^{1/2})$. [most points have margin at least this γ.]
- (Just take some care since some points have small margin)

Better than $O(d)$ for baseline methods!