Protocols for Learning Classifiers on Distributed Data

Suresh Venkatasubramanian
University of Utah

Hal Daumé
Jeff Phillips
Avishek Saha
The Mechanics of Learning

![Diagram of the mechanics of learning](image)

Batch mode learning
The Mechanics of Learning

- Training Data
- Learning Algorithm
- Test Data
- Model
- Predictions

Batch mode learning
The Mechanics of Learning

Training Data → Learning Algorithm → Model → Predictions

In all cases, data is easily accessible by learning algorithm!

Online learning
In all cases, data is easily accessible by learning algorithm!
Parallel Learning

Distribute data for efficient learning.
Parallel Learning

Distribute data for efficient learning.

![Diagram of Parallel Learning](image)
Parallel Learning

Distribute data for efficient learning.

\[ D \rightarrow D_1 \rightarrow \theta_1 \quad \cdots \quad \rightarrow D_k \rightarrow \theta_k \]
Parallel Learning

Distribute data for efficient learning.
Distributed Data

courtesy the Earth Observatory

data can be distributed across geographically distinct locations
Data can be distributed across geographically distinct locations
Distributed Data

Data can be distributed even inside a single machine

Intel's Nehalem (Core I7) chip
Distributed Data

Data can be distributed even inside a single machine
Learning without chatter

- Communication is either expensive, or undesirable
- “Move analysis to the data”

How can we learn in a communication-restricted environment?
Distributed learning is not distributed computing
A simple model for distributed learning

- \(k\) “players”
- Each player owns data \(D_i\). Let \(D = \bigcup_i D_i\)
- Learning task \(T\), solution \(h\), error \(\text{err}(h, D, T)\).

**Problem**

*Given \(\epsilon > 0\), design protocol to let players agree on solution \(\tilde{h}\) such that*

\[
\text{err}(\tilde{h}, D, T) \leq \text{err}(h^*, D, T) + \epsilon
\]

*with minimum inter-player communication.*
Learning a classifier

\[ w \cdot x = b \]

misclassification error = fraction of mistakes
Learning a classifier

\[ w \cdot x = b \]

misclassification error = fraction of mistakes
Learning a classifier

\[ w \cdot x = b \]

Misclassification error = fraction of mistakes
Learning a classifier
Learning a classifier

\[ w \cdot x = b \]
Learning a classifier

\[ w \cdot x = b \]

misclassification error = fraction of mistakes
Learning a classifier

\[ w \cdot x = b \]

misclassification error = fraction of mistakes
Learning a classifier

\[ w \cdot x = b \]

misclassification error = fraction of mistakes
Learning a classifier

misclassification error = fraction of mistakes
A simple observation: random sampling of the data

**Definition**

*Given a range space* $(X, \mathcal{R})$, *a set* $S \subset X$ *is an* $\epsilon$-*net if for all* $R \in \mathcal{R}$,

$$|R \cap X| \geq \epsilon \implies R \cap S \neq \emptyset$$

**Theorem**

*Any range space* $(X, \mathcal{R})$ *of VC-dimension* $d$ *has an* $\epsilon$-*net of size* $O\left(\frac{d}{\epsilon} \log \frac{1}{\epsilon}\right)$.
A simple observation: random sampling of the data

**Definition**

Given a range space \((X, \mathcal{R})\), a set \(S \subset X\) is an \(\epsilon\)-net if for all \(R \in \mathcal{R}\),

\[
|R \cap X| \geq \epsilon \implies R \cap S \neq \emptyset
\]

**Theorem**

Any range space \((X, \mathcal{R})\) of VC-dimension \(d\) has an \(\epsilon\)-net of size
\[
O\left(\frac{d}{\epsilon} \log \frac{1}{\epsilon}\right)
\]
A simple observation: random sampling of the data

**Definition**

Given a range space $(X, \mathcal{R})$, a set $S \subset X$ is an $\epsilon$-net if for all $R \in \mathcal{R}$,

$$|R \cap X| \geq \epsilon \implies R \cap S \neq \emptyset$$

**Theorem**

Any range space $(X, \mathcal{R})$ of VC-dimension $d$ has an $\epsilon$-net of size $O\left(\frac{d}{\epsilon} \log \frac{1}{\epsilon}\right)$.
A simple observation: random sampling of the data

**Definition**

Given a range space $(X, \mathcal{R})$, a set $S \subset X$ is an $\epsilon$-net if for all $R \in \mathcal{R}$,

$$|R \cap X| \geq \epsilon \quad \implies \quad R \cap S \neq \emptyset$$

**Theorem**

Any range space $(X, \mathcal{R})$ of VC-dimension $d$ has an $\epsilon$-net of size $O\left(\frac{d}{\epsilon} \log \frac{1}{\epsilon}\right)$

Random partitioning gives easy communication bounds: $O\left(\frac{d}{\epsilon} \log \frac{1}{\epsilon}\right)$
A Lower bound

Lemma

Any one-way protocol for learning an $\epsilon$-error classifier requires $\Omega(1/\epsilon)$ communication.
A dialogue is exponentially more informative than a monologue.

Theorem

- A one-way protocol for classification requires at least $O\left(\frac{1}{\epsilon}\right)$ examples.

- A two-way protocol for classification requires exchanging only $O(d \log \frac{1}{\epsilon} \log \log \frac{1}{\epsilon})$ examples between two parties, or an additional $k \log \frac{1}{\epsilon}$ examples for $k$ players.
Binary Search
Binary Search
Multiplicative Weight Updates
Multiplicative Weight Updates
Multiplicative Weight Updates
Algorithm

Player 1
computes
classifier

Player 2
finds mistakes
updates weights
picks sample

Theorem

After $O(d \log 1/\epsilon)$ points have been exchanged, we have an \( \epsilon \)-optimal classifier.
Multiple Data Sources

All nodes talk to the coordinator simultaneously.
Multiple Data Sources

All nodes talk to the coordinator simultaneously.
Multiple Data Sources

All nodes talk to the coordinator simultaneously.
Multiple Data Sources
Multiple Data Sources
Multiple Data Sources

All nodes talk to the coordinator simultaneously.
Multiple Data Sources

All nodes talk to the coordinator simultaneously
In each round:

1. Each player reports their “total weight” to coordinator
2. Coordinator tells each player what fraction of total weight they “own”
3. Each player sends weighted sample to coordinator
4. Coordinator computes classifier on all data received thus far and sends to players
5. Players update their weight vectors.

**Theorem**

In \( T = O(\log 1/\epsilon) \) rounds with \( k + d \log \log 1/\epsilon \) vectors sent in each round, we can learn an \( \epsilon \)-error classifier jointly among \( k \) players.
Strong relationship between learning and optimization

Learning task with training data $\iff$ (non?)-convex optimization

$$A_1 x \leq b_1$$
$$A_2 x \leq b_2$$
$$A_k x \leq b_k$$

$x^*$ 

?
Distributed Optimization

**Theorem**

A streaming algorithm that uses \(\ell\) passes with \(s\) memory can be converted into a distributed algorithm on \(k\)-players that uses \(k\ell s\) words of communication.

**Corollary**

There is a \(k\)-player distributed algorithm for linear programming with \(n\) constraints that finds a point satisfying all but an \(\epsilon\)-fraction of the constraints using \(O\left(\frac{k}{\epsilon^2}\right)\) words of communication (via [CC07])

Variations on this result apply for semidefinite programming as well.
Conclusions and Directions

- A model for thinking about large-scale distributed learning that focuses on **communication**
- Relevant to both big-data and “big-iron”.
- Design principles for algorithms that use the MWU and streaming algorithms

Future directions:

- distributed clustering
- multitask learning
- dimensionality reduction (PCA) [Bai, Chan, Luk 2005]
- general optimization (SDPs ?)
- limits of the model (lower bounds)


Questions?