Properly Learning Decision Trees in Almost-Polynomial Time

Li-Yang Tan
Joint work with:

Guy Blanc  Stanford
Jane Lange  MIT
Mingda Qiao  Stanford

(Slides available on my webpage)
This talk:
Turning black boxes into decision trees

Given queries to a function $f: \{0,1\}^n \rightarrow \{0,1\}$

Output decision tree representation of $f$
Decision trees: simple and effective

- Fast to evaluate
- Easy to understand its structure,
  Easy to explain its behavior

```
0 1 x2 1
1 0
```

```
0 1 x7 x11
```

```x
x
x
```
Want fast and accurate algorithms

Assumption about $f$:
Can be represented by a small decision tree

Fast algorithm

Accurate tree $T$:
Pr[$T(x) \neq f(x)$] ≤ $\varepsilon$

uniform $x$
Our result: A new algorithm for this task

Algorithm with runtime
\[ \text{poly}(n) \cdot s^{O(\log \log s)} \]

Queries to \( f: \{0,1\}^n \rightarrow \{0,1\} \)

\( f \) has a size-\( s \) DT

Size-\( s \) DT \( T \) such that:
\[ \Pr[T(x) \neq f(x)] \leq \varepsilon \]
uniform \( x \)
Our algorithm is furthermore agnostic

Algorithm with runtime
\[\text{poly}(n) \cdot s^{O(\log \log s)}\]

Queries to \(f\): \{0,1\}^n \to \{0,1\}

\(f\) has a size-\(s\) DT

\(f\) is \textbf{opt-close} to size-\(s\) DT

Size-\(s\) DT \(T\) such that:
\[\Pr[T(x) \neq f(x)] \leq \varepsilon\]

uniform \(x\)

\(\leq \text{opt} + \varepsilon\)

Equivalently, our algorithm is noise tolerant
Comparison with prior work

- For $s = \text{poly}(n)$, our runtime is:

  $$\text{poly}(n) \cdot s^{O(\log \log s)} = n^{O(\log \log n)}$$

- Previous fastest runtime $n^{O(\log n)}$, even in the realizable setting
  - 3 different algorithms:
    [ Ehrenfeucht–Haussler 89, Mehta–Raghavan 02, Blanc–Lange–T. 20 ]
  - Analyses of all 3 algorithms tight: $n^{\Omega(\log n)}$ lower bounds
Formal statement of our result

Given queries to $f: \{0,1\}^n \to \{0,1\}$ that is $\text{opt}_s$-close to size-$s$ DT, our algorithm runs in time:

$$\tilde{O}(n^2) \cdot (s/\varepsilon)^O(\log((\log s)/\varepsilon))$$

and outputs a size-$s$ DT that is $(\text{opt}_s + \varepsilon)$-close to $f$.

In the language of learning theory:

We give a new query algorithm for properly and agnostically learning decision trees under the uniform distribution.
Two general approaches to learning DTs

**Bottom up**

- Ehrenfeucht–Haussler 89
- Mehta–Raghavan 02

**Top down**

- Blanc–Lange–T. 20

Our algorithm combines both approaches: “[BLT] meets [MR]”
Outline for the rest of this talk

- [BLT]’s top-down algorithm and its limitations
- Our algorithm and its key ingredient: decision tree pruning lemma

Strengthens theorem of [O’Donnell, Saks, Schramm, Servedio 05], which was key ingredient in [BLT]

- Challenges in the agnostic setting and how we overcome them
Top-down algorithms for learning DTs

1. Choose a “good” variable $x_i$ to query as root

2. Recurse on $f_{x_i=0}$ and $f_{x_i=1}$

- Definition of “good” = splitting criterion of algorithm
- Intuitively, want most “important”/“informative” variable
Influence as a splitting criterion

[BLT]’s splitting criterion:

**Definition.** The influence of a variable $i$ on $f: \{0,1\}^n \rightarrow \{0,1\}$ is the quantity:

$$\text{Inf}_i(f) := \Pr[f(x) \neq f(x \oplus i)]$$

where $x \oplus i$ denotes $x$ with its $i$-th coordinate flipped.

Splitting criteria used in practice: Gini impurity, information gain, etc.
See [BLT] for detailed discussion.

“Top-down induction of decision trees: rigorous guarantees and inherent limitations” Blanc, Lange, Tan. ITCS 2020
[BLT]’s top-down algorithm

1. Using queries to $f$, identify the variable $i$ with (approximately) the largest influence on $f$

2. Query $x_i$ as the root of tree

3. Build left and right subtrees by recursing on $f_{x_i=0}$ and $f_{x_i=1}$

4. When tree reaches user-specified depth $d$, label each leaf $\ell$ with round($\mathbb{E}[f_\ell]$) $\in \{0,1\}$
Guarantees and limitations of this algorithm

**Theorem [BLT].** Let $f : \{0,1\}^n \rightarrow \{0,1\}$ be a size-$s$ DT.

Growing this “most-influential-at-the-root” tree to depth

$$d \leq O((\log(s/\varepsilon))^2)$$

guarantees accuracy $\geq 1-\varepsilon$ with respect to $f$.

Remark: Depth $O((\log s)^2) \Rightarrow$ Size $s^{O(\log s)}$

**Matching lower bound.** There is a size-$s$ DT $f$ such that this tree must be grown to depth

$$d \geq \tilde{\Omega}(\log s)^2$$

even just to achieve accuracy $\geq 51%$. 
Outline of this talk

- [BLT]'s top-down algorithm and its limitations
- Our algorithm and its key ingredient: decision tree pruning lemma

Strengthens theorem of [O’Donnell, Saks, Schramm, Servedio 05], which was key ingredient in [BLT]

- Challenges in the agnostic setting and how we overcome them
Being less greedy

[BLT]:
The most influential variable is a “somewhat good” root

This work:
There’s an even better root among the $k$ most influential variables
The difference in more detail

Let $f$ be a size-$s$ DT.

[BLT]: Recursively querying the most influential variable yields accurate tree for $f$ if grown to depth $O(\log s)^2$.

**Our Less-Greedy Thm**: Exists variable $x_i$ among the $\text{polylog}(s)$ most influential ones s.t. there is accurate tree for $f$ of depth $O(\log s)$ with $x_i$ as its root.

[BLT] commits to the most influential variable as the root

We consider a small set of candidate roots
\textbf{Queries to }f\\
Greediness parameter }k\\
Depth budget }d\\

\textbf{Less-Greedy}(f, k, d):

1. If }d = 0, return \text{round}(\mathbb{E}[f]) \in \{0,1\}

2. Identify set }S := \{k \text{ most influential variables of } f \}\\

3. For each }i \in S, construct tree }T_i := x_i\\

\textbf{Less-Greedy}(f_{x_i=0}, k, d-1) \quad \textbf{Less-Greedy}(f_{x_i=1}, k, d-1)

4. Return tree }T_i \text{ with minimal error w.r.t. } f\\

\textbf{Time complexity:} \quad \textbf{Less-Greedy}(f, k, d) \text{ makes } (2k)^d \text{ recursive calls}\\
\quad \text{ }S \text{ can be identified in poly}(n) \text{ time}
Done, assuming Less-Greedy Theorem

**Less-Greedy Theorem:** If $f$ is a size-$s$ DT, exists a variable $x_i$ among the $\text{polylog}(s)$ most influential ones s.t. there is an accurate tree for $f$ of depth $O(\log s)$ with $x_i$ as its root.

**Equivalently:**

\[
\text{Less-Greedy}(f, k, d) \text{ with } k \coloneqq \text{polylog}(s) \text{ and } d \coloneqq O(\log s)
\]

returns an accurate tree for $f$

- # recursive calls = $(2k)^d = s^{O(\log \log s)}$
- Overall runtime = $\text{poly}(n) \cdot s^{O(\log \log s)}$
Search space of Less-Greedy

Which trees does Less-Greedy($f, k, d$) search over?

For all paths $\pi$, variable queried is one of the $k$ most influential ones of $f_\pi$
This proves the **Less-Greedy Theorem** because

1: $T'$ is an accurate tree for $T$

2+3: $T'$ is in search space of $\text{Less-Greedy}(f, k, d)$

Since $f_\pi$ is a size-$s$ tree, at most

$$k := \frac{(\log s)}{\tau} = O(\log s)^2$$

such variables.
Small depth is easy to achieve

Size-$s$ tree $T$ for $f$

Prune

Pruned tree $T'$

1. $\leq \varepsilon$ error

2. $d \leq \log(s/\varepsilon)$

3. $\forall \pi$: $\inf_i (f_\pi) \geq \tau := \frac{\varepsilon}{\log(s/\varepsilon)}$

Easy fact: Truncating size-$s$ tree at depth $\log(s/\varepsilon)$ incurs $\leq \varepsilon$ error.
We enforce high influence recursively, starting at the root.

**Q:** \( \text{Inf}_i(f) \geq \tau ? \)

**If so:** keep root; prune subtrees recursively

**If not:** Discard root + subtree that’s further from \( f \)
**Lemma:** Let \( f \) be a function and \( T \) be a depth-\( d \) DT. Then:
\[
\text{dist}\left(f, \text{Prune}_{f, \tau}(T)\right) \leq \text{dist}(f, T) + d \cdot \tau.
\]

**Proof:** Induction on \( d \). Only do harder case where \( \text{Inf}_i(f) \leq \tau \).

\[
\text{dist}(f, \text{Prune}_{f, \tau}(T)) \\
\begin{align*}
\text{(min \leq avg)} & \leq \mathbb{E}_b[\text{dist}(f, \text{Prune}_{f, \tau}(T_b))] \\
\text{(IH)} & \leq \mathbb{E}[\text{dist}(f, T_b) + (d - 1) \cdot \tau] \\
\text{(\Delta-ineq.)} & \leq \mathbb{E}[\text{dist}(f, f_{x_i=b})] + \mathbb{E}[\text{dist}(f_{x_i=b}, T_b)] + (d - 1) \cdot \tau \\
\text{(Inf}_i(f) \leq \tau) & = \text{Inf}_i(f) + \text{dist}(f, T) + (d - 1) \cdot \tau \\
\end{align*}
\]

\[
\begin{align*}
\leq \text{dist}(f, T) + d \cdot \tau.
\end{align*}
\]
Putting everything together

Size-\(s\) tree \(T\) for \(f\)  

Depth \(d \leq \log(s/\varepsilon)\)  

\(\forall \pi: \inf_i (f_{\pi}) \geq \tau := \frac{\varepsilon}{d}\)

Final tree falls into search space of \textbf{Less-Greedy}(\(f, k, d\)) where:

\[k \leq (\log s)/\tau = O(\log s)^2, \quad d \leq O(\log s)\]

\textbf{Less-Greedy} finds it in time \(\text{poly}(n) \cdot (2k)^d = \text{poly}(n) \cdot s^{O(\log \log s)}\)
Zooming out: Our algorithm vs. previous ones

Our algorithm: Less-Greedy\((f, k=\text{polylog}(s), d=\log s)\)

Runtime: \((2k)^d = s^{O(\log \log s)}\)

Greediness parameter \(k\)

\[ k = \text{polylog}(s) \]

[BLT]: Less-Greedy\((f, k=1, d=(\log s)^2)\)

Runtime: \((2k)^d = s^{O(\log s)}\)

[MR]: Less-Greedy\((f, k=n, d=\log s)\)

Runtime: \((2k)^d = n^{O(\log s)}\)

“Decision tree approximations of Boolean functions” Mehta and Raghavan. COLT 2000
Outline of this talk

- [BLT]'s top-down algorithm and its limitations
- Our algorithm and its key ingredient: decision tree pruning lemma

Strengthens theorem of [O’Donnell, Saks, Schramm, Servedio 05], which was key ingredient in [BLT]

- Challenges in the agnostic setting and how we overcome them
Pruning lemma and the OSSS theorem

Size-$s$ tree $T$ for $f$

Depth $d \leq \log(s/\varepsilon)$

$\forall \pi$: $\text{Inf}_i(f_\pi) \geq \tau := \frac{\varepsilon}{d}$

Pruning lemma easily implies:

**Theorem [OSSS].** Every size-$s$ DT $T$ is either $\varepsilon$-close to constant, or

$$\exists i \in [n] \text{ s.t. } \text{Inf}_i(T) \geq \varepsilon / \log(s).$$

**Proof.** If $T$ has no influential variables, our pruning algorithm prunes everything away, resulting in a constant that’s $\varepsilon$-close to $T$.

“Every decision tree has an influential variable” O’Donnell, Saks, Schramm, Servedio. FOCS 2005
Agnostic Learning

Assumption about $f$:

Realizable setting: $f$ is a size-$s$ DT

Agnostic setting: $f$ is opt-close to a size-$s$ DT
Why our current algorithm fails in the agnostic setting

Realizable setting: Since $f_\pi$ is size-$s$ DT, $\leq (\log s)/\tau = O(\log s)^2$ variables with influence $\geq \tau$.

**Proof.** Every size-$s$ DT has total influence $\leq \log s$.

**False if** $f_\pi$ is merely close to size-$s$ DT
Solution: Smoothing things out

**Definition:** For a function $f : \{0,1\}^n \to \{0,1\}$, its \(\delta\)-smoothed version is the function $f_\delta : \{0,1\}^n \to [0,1]$, 

$$f_\delta(x) := \mathbb{E}[f(y)]$$

where $y = x$ with each bit flipped w.p. $\delta$.

**Why is this useful for us?**

Standard Fourier fact: For all functions $f$, total influence of $f_\delta$ is $\leq O(1/\delta)$. 
Our algorithm for the agnostic setting

- Key difference: now can bound \# variables with $\inf_i((f_\delta)_\pi) \geq \tau$ by $1/(\delta \tau)$.

- **Less-Greedy**$(f_\delta, k, d)$ finds this tree, which we prove is $(\text{opt} + \varepsilon)$-close to $f$
Recap and future directions
Proper learning in polynomial time?

Our main result:

Given queries to $f: \{0,1\}^n \rightarrow \{0,1\}$ that is opt$_s$-close to size-$s$ DT, our algorithm runs in time:

$$\tilde{O}(n^2) \cdot (s/\varepsilon)^{O(\log((\log s)/\varepsilon))}$$

and outputs a size-$s$ DT that is (opt$_s + \varepsilon$)-close to $f$.

- A poly$(n, s)$-time algorithm would be great, even just in the realizable setting
- ... via a new splitting criterion?
Further applications of our pruning lemma?

Size-$s$ tree $T$ for $f$

Depth $d \leq \log(s/\epsilon)$

$\forall \pi$: $\inf_i (f_\pi) \geq \tau := \frac{\epsilon}{d}$

Applications of the O’Donnell–Saks–Schramm–Servedio theorem:

- Original application: Evasiveness of graph properties
- Subsequent applications in percolation theory
Thank you for listening.