Attribute-Efficient Learning and Weight-Degree Tradeoffs for Polynomial Threshold Functions

Rocco Servedio, Columbia University
Li-Yang Tan, Columbia University
Justin Thaler, Harvard University
Attribute-Efficient Learning

- Attribute-efficient learning is a clean framework capturing the problem of learning in the presence of irrelevant information.
  - Especially important in the age of Big Data.

- Consider a scientist trying to identify genetic causes of a disease.
  - The disease depends on the interaction of a small number of genes.
  - The scientist collects a massive amount of genetic data from participants.
  - Only a small amount of this information is actually relevant to the function being learned (the mapping of genes to a subject's phenotype).
Attribute-Efficient Learning

- Goal of an algorithm for attribute-efficient learning:
  - Run in time poly(n), where n is total number of attributes.
  - Use a number of examples which is polynomial in the description length of the function f to be learned.
  - The latter can be substantially smaller than n if most of the attributes are irrelevant.
Comparison to Junta Problem

- The most general version of the problem of learning in the presence of irrelevant information is called the “Junta Problem” [Blum-Langley 1997, Mossel-O’Donnell-Servedio 2004].
  - Assume nothing about $f$ other than that it depends on $k << n$ attributes.
  - Uniform-distribution variant of Junta Problem called “the most important open question in uniform distribution learning” by MOS.

- Our goal is both more and less ambitious than the uniform-distribution Junta Problem.
  - We want to learn under *arbitrary distributions.*
  - But are willing to assume the relevant attributes interact in structured ways.
  - We focus on attribute-efficient learning of *decision lists.*
Decision Lists

- A length k decision list of $x_1, \ldots, x_n$ is a sequence of “if-then-else” statements:

  \[ x_1 \rightarrow x_5 \rightarrow x_9 \rightarrow x_3 \rightarrow x_7 \rightarrow 1 \]
  
  \[
  \begin{array}{cccccc}
  0 & 1 & 1 & 0 & 1 \\
  \end{array}
  \]

- Attribute-efficiently learning DLs is a well-studied and challenging open problem.


- DLs are PAC-learnable in poly(n) time, but seem to lie on boundary of tractability in the attribute-efficient setting.
Mistake-Bounded Learning

- We establish our results in the *mistake-bounded model*.

- Mistake-Bounded model:
  - Learning consists of a sequence of trials. In each trial, the learner is given some $x$ from $\{0,1\}^n$ and outputs $h(x)$, her guess as to what $f(x)$ is.
    - If $h(x) = f(x)$, great!
    - If $h(x) \neq f(x)$, learner is charged a mistake.

- Goal: design an efficient algorithm that minimizes number of mistakes over all possible (infinite) sequences of trials.
Algorithmic Machinery

- Theorem (Expanded-Winnow Algorithm) [Klivans-Servedio 2004]: Let \( f(x) = \text{sgn}(p(x_1, \ldots , x_n)) \), where \( p \) is a degree-\( d \) polynomial with integer coefficients whose absolute values sum to \( W \). Then we can learn \( f \) in time \( n^{O(d)} \) per example and mistake bound \( O(W^2 d \log(n)) \).

- \( p \) is called a polynomial threshold function (PTF) for \( f \), and \( W \) is called the weight of \( p \).

- Corollary: Attribute-efficient learning of DLs reduces to showing that every length \( k \) decision list has a low-degree, low-weight PTF.
What was known?

- Theorem [Klivans-Servedio 2004]: Let $f$ be a length $k$ DL. For every $d \leq k^{1/3}$, there is a degree $d$, weight $2^{O(k/d^2)}$ PTF computing $f$.

- Theorem [Beigel 1994]: There is a length $k$ decision list $f$ such that for any $d \leq k$, any degree $d$ PTF computing $f$ requires weight $2^{\Omega(k/d^2)}$.

- So both theorems are tight at low degrees ($d<k^{1/3}$). But it was open what happens at higher degrees.

- We show that at higher degrees, neither theorem is tight!
New Results

- Theorem: Let $f$ be a length $k$ DL. For every $d \geq k^{1/3}$, there is a degree $d$, weight $2^{O((k/d)^{1/2})}$ GPTF* computing $f$.
  
  *A GPTF is slightly more expressive than a PTF, and just as useful for learning purposes.

- Theorem: There is a length $k$ DL such that for any $d \leq k$, any degree $d$ PTF computing $f$ requires weight $2^{\Omega((k/d)^{1/2})}$.

- Both of these theorems improve on prior work when the degree is relatively high ($d > k^{1/3}$).

- The main remaining gap is that our upper bound uses GPTFs while our lower bound applies only to PTFs.
Comparison of New Upper Bound to Prior Work [Klivans-Servedio 2004]

PTF Weight Upper Bound for DLs of length $k=1,000,000$

- Blue line is $2^{O(k/d^2)}$ PTF weight upper bound of [Klivans-Servedio 2004] (holds for $d < k^{1/3}$).
- Red Line is our new $2^{O((k/d)^{1/2})}$ GPTF weight upper bound (holds for $d \geq k^{1/3}$).
## Comparison of Our Algorithm to Prior Work

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Run Time</th>
<th>Mistake Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>Winnow Algorithm [Littlestone 1988]</td>
<td>$n$</td>
<td>$2^k \log(n)$</td>
</tr>
<tr>
<td>Halving Algorithm [Littlestone 1988]</td>
<td>$n^k$</td>
<td>$k \log(n)$</td>
</tr>
<tr>
<td>Klivans-Servedio (for every $d \leq k^{1/3}$)</td>
<td>$n^d$</td>
<td>$2^{O((k/d)^{1/2})} \log(n)$</td>
</tr>
<tr>
<td>Servedio-Tan-Thaler (for every $d \geq k^{1/3}$)</td>
<td>$n^d$</td>
<td>$2^{O((k/d)^{1/2})} \log(n)$</td>
</tr>
</tbody>
</table>
Comparison of New Lower Bound to Prior Work [Beigel 1994]

PTF Weight Lower Bound for DLs of length $k=1,000,000$

- Blue line is $2^{O(k/d^2)}$ PTF weight lower bound of [Beigel 1994].
- Red Line is our new $2^{O((k/d)^{1/2})}$ PTF weight lower bound.
Upper Bound Proof Sketch

- Given: a length k DL f.
- Break f into k/b “blocks” of length b.
- Closely approximate each block i in the $L_\infty$-norm with a low-degree polynomial $p_i(x)$.
  - If block i “makes a decision”, $p_i(x)$ outputs a value close to $\pm 1$.
  - Otherwise, $p_i(x)$ outputs 0.
- Put the approximations together to get a PTF p for the entire decision list f.
  - $p(x) = \sum_i 3^i p_i(x)$.
  - The highest block i to “make a decision” will dominate the output of p, so $f = \text{sgn}(p(x))$. 

Given: a length k DL f.

Break f into k/b “blocks” of length b.

Closely approximate each block i in the $L_\infty$-norm with a low-degree polynomial $p_i(x)$.
- If block i “makes a decision”, $p_i(x)$ outputs a value close to $\pm 1$.
- Otherwise, $p_i(x)$ outputs 0.

Put the approximations together to get a PTF $p$ for the entire decision list $f$.
- $p(x) = \sum_i 3^i p_i(x)$.
- The highest block i to “make a decision” will dominate the output of $p$, so $f=\text{sgn}(p(x))$.
- Degree of $p$ equals degree of the $p_i$’s.
- Weight of $p$ depends on the number of blocks and the weight of the $p_i$’s. Choose block length to balance these contributions.
Upper Bound Proof Sketch

- Klivans-Servedio use degree $d$ Chebyshev polynomials to construct each approximating polynomial $p_i(x)$.
- But when the $d$ is relatively large, the degree $d$ Chebyshev polynomials have very high weight.
  - Instead, we use lower degree Chebyshev polynomials, composed with a high-degree monomial.
  - This allows us to achieve lower weight approximating polynomials $p_i(x)$ than those obtained by Klivans-Servedio for the same degree.
Lower Bound Proof Sketch

- We prove a lower bound for a specific decision list, ODD-MAX-BIT (OMB).
- Look at the right-most bit set to 1. If it is at an odd coordinate, output 1, else output 0.
Lower Bound Proof Sketch

- Lower bound argument shows that “block-based” approach of our upper bound is intrinsic.
- Break the OMB function into $k/b$ blocks of length $b$.
- Show that you can take any PTF $p$ for OMB and turn it into a polynomial $q$ closely approximating each block.
  - $q$ has the same degree and weight as $p$.
- Beigel used Markov’s inequality from approximation theory to conclude that $q$ has to have high degree, and hence $p$ has to have high degree as well.
Lower Bound Proof Sketch

- Markov’s inequality bounds the derivative of a polynomial $q$ in terms of its degree.
- We prove a new Markov-type inequality which takes into account both the degree of $q$ and the size of its coefficients.
Lower Bound Proof Sketch

- **Markov's Inequality:** Let \( q : [-1,1] \rightarrow [-1,1] \) be a real polynomial with \( \deg(q) \leq d \). Then \( \max_{|x| \leq 1} |q'(x)| \leq d^2 \).

- **Our Markov-type Inequality:** Let \( q : [-1,1] \rightarrow [-1,1] \) be a real polynomial with \( \deg(q) \leq d \) and coefficients of absolute value at most \( W \). If \( \frac{1}{2} \leq \max_{|x| \leq 1} |q(x)| \), then
  \[
  \max_{|x| \leq 1} |q'(x)| = O(d \cdot \max\{d, \log(W)\}).
  \]
- If \( W \ll 2^d \), our inequality is tighter than Markov’s.
- This allows us to improve Beigel’s lower bound for OMB when \( d \) is relatively large.
Lower Bound Proof Sketch

• **Markov's Inequality**: Let \( q : [-1,1] \rightarrow [-1,1] \) be a real polynomial with \( \deg(q) \leq d \). Then \( \max_{|x| \leq 1} |q'(x)| \leq d^2 \).

• **Our Markov-type Inequality**: Let \( q : [-1,1] \rightarrow [-1,1] \) be a real polynomial with \( \deg(q) \leq d \) and coefficients of absolute value at most \( W \). If \( \frac{1}{2} \leq \max_{|x| \leq 1} |q(x)| \), then
  \[
  \max_{|x| \leq 1} |q'(x)| = O(d \cdot \max \{d, \log(W)\}).
  \]
  If \( W \ll 2^d \), our inequality is tighter than Markov’s.

• This allows us to improve Beigel’s lower bound for OMB when \( d \) is relatively large.

• Tight example for Markov: degree \( d \) Chebyshev polynomials. Tight example for our inequality degree \( d \) Chebyshev polynomials composed with a high-degree monomial.

• Same intuition applied for our upper bound.
Conclusions

• We provide new positive and negative results for attribute-efficient learning of decision lists.

• Our results rely on a careful study of PTF weight-degree tradeoffs for decision lists.
  • Both our upper and lower bounds improve over prior work when the allowed degree of the (G)PTF is relatively high.

• Open questions:
  • Cryptographic hardness of true attribute-efficient learning of length k decision lists? [Servedio 2000] has partial results in this direction.
  • New algorithms: beyond PTFs?
  • Moving beyond DLs: Attribute-efficient learning of more expressive concept classes like decision trees and DNFs?
Thank you!