Learning Coverage Functions and Private Release of Marginals

Vitaly Feldman
IBM Research

Pravesh Kothari
UT Austin
Coverage Functions

A function $f : 2^n \rightarrow \mathbb{R}^+$ is a coverage function if:
Coverage Functions

A function $f : 2^{[n]} \rightarrow \mathbb{R}^+$ is a coverage function if:

Ground Set $U$
Coverage Functions

A function \( f : 2^{[n]} \rightarrow \mathbb{R}^+ \) is a coverage function if:

Ground Set \( U \)

\( A_1, A_2, \ldots, A_n \subseteq U \)
Coverage Functions

A function \( f : 2^{[n]} \to \mathbb{R}^+ \) is a coverage function if:

Ground Set \( U \)

\( A_1, A_2, \ldots, A_n \subseteq U \)

For any \( S \subseteq [n] \)
Coverage Functions

A function $f : 2^{[n]} \rightarrow \mathbb{R}^+$ is a **coverage function** if:

**Ground Set** $U$

$A_1, A_2, \ldots, A_n \subseteq U$

For any $S \subseteq [n]$

$$f(S) = \left| \bigcup_{i \in S} A_i \right|$$
Coverage Functions

A function \( f : 2^{[n]} \rightarrow \mathbb{R}^+ \) is a coverage function if:

Ground Set \( U \)
\( A_1, A_2, \ldots, A_n \subseteq U \)

For any \( S \subseteq [n] \)

\[ f(S) = \left| \bigcup_{i \in S} A_i \right| \]
Coverage Functions

A function \( f : \{-1, 1\}^n \rightarrow [0, 1] \) is a coverage function if:

- Ground Set \( U \) with \( A_1, A_2, \ldots, A_n \subseteq U \)
- For any \( S \subseteq [n] \)

\[
f(S) = \left| \bigcup_{i \in S} A_i \right|
\]
Coverage Functions

\[ f : \{-1, 1\}^n \rightarrow [0, 1] \] coverage function:

\[ f(x) = \sum_{S \subseteq [n]} \alpha_S \cdot \text{OR}_S(x) \]
Coverage Functions

\[ f : \{-1, 1\}^n \rightarrow [0, 1] \]

coverage function:

\[ f(x) = \sum_{S \subseteq [n]} \alpha_S \cdot \text{OR}_S(x) \]

non-negative
Coverage Functions

\[ f : \{-1, 1\}^n \rightarrow [0, 1] \]

\[
f(x) = \sum_{S \subseteq [n]} \alpha_S \cdot \text{OR}_S(x)
\]

coverage function:

monotone disjunction on \( S \)
Coverage Functions

\[ f : \{-1, 1\}^n \rightarrow [0, 1] \] coverage function:

\[ f(x) = \sum_{S \subseteq [n]} \alpha_S \cdot \text{OR}_S(x) \]

\[ \text{OR}_S(x) = 0 \iff x_i = 1 \text{ for every } i \in S \]

“non-negative linear combination of monotone disjunctions”
Coverage Functions

\( f : \{-1, 1\}^n \rightarrow [0, 1] \)  

**coverage function:**

\[
f(x) = \sum_{S \subseteq [n]} \alpha_S \cdot \text{OR}_S(x)
\]

\[
\sum_{S \subseteq [n]} \alpha_S \leq 1
\]
Coverage Functions

Important subclass of submodular functions.
• Topic of intense research in machine learning.

Algorithmic Game Theory\Economics
• Valuation Functions - utilities, prices.
Lehmann-Nisan-06, Dobzinski et. al.-05, Vondrák-08,
Papadimitriou et. al.-08, Dughmi et. al. 11

Differential Privacy: in this talk
Our Results

1. Learnability of Coverage Functions.
2. Applications to Private Data Release
Learning Coverage Functions

“PAC Learning with $\ell_1$ error”

Learner sees random examples labeled the target coverage function $f$.

Must return a hypothesis $h$ such that

$$\mathbb{E}[||h(x) - f(x)||] \leq \epsilon$$
Learning Coverage Functions

“PAC Learning with $\ell_1$ error”

Coverage functions are learnable on the uniform distribution in time $\tilde{O}(n)\text{poly}(1/\epsilon)$ and samples $\log(n)\text{poly}(1/\epsilon)$.

Previous Work: Submodular Functions


Best: $2^{O(1/\epsilon^2)} \cdot \text{poly}(n)$

exponential dependence necessary.[Feldman-K-Vondrak 2013]
Learning Coverage Functions

**PMAC Model** [Balcan and Harvey 2011]

“Probably Mostly Approximately Correct”

Learner sees random examples from the target coverage function \( f : \{-1, 1\}^n \rightarrow \mathbb{R}^+ \).

Must return a hypothesis \( h \) such that

\[
\Pr[h(x) \leq f(x) \leq (1 + \gamma)h(x)] \geq 1 - \delta
\]

More challenging model to learn than PAC.
Learning Coverage Functions

PMAC Model [Balcan and Harvey 2011]
“Probably Mostly Approximately Correct”

Coverage functions are PMAC learnable on the uniform distribution in time $\tilde{O}(n)\text{poly}\left(\frac{1}{\delta \cdot \gamma}\right)$ and samples $\log(n)\text{poly}\left(\frac{1}{\delta \cdot \gamma}\right)$.

Previous Work: Submodular Functions:
[Balcan and Harvey 2011] [Feldman and Vondrak 2013]

Best: $2^{O\left(\frac{1}{(\delta \gamma)^2}\right)}\text{poly}(n)$

First fully polynomial time PMAC learning algorithm for an interesting class.
Learning Coverage Functions

“Agnostic Learning with $\ell_1$ error”

learner gets random examples from an arbitrary $f$.

Must return $h$:

$$\mathbb{E}[|h(x) - f(x)|] \leq opt + \epsilon$$
Learning Coverage Functions

“Agnostic Learning with $\ell_1$ error”

learner gets random $x$

Must return $h$:

$$\mathbb{E}\left[|h(x) - f(x)|\right] \leq opt + \epsilon$$

$$opt = \min_{\text{coverage } \ c} \mathbb{E}[|c(x) - f(x)|]$$
Coverage functions are agnostically learnable over all product and symmetric distributions in time and samples $n^{O(\log (1/\epsilon))}$. 
Learning Coverage Functions

Coverage functions are agnostically learnable over all product and symmetric distributions in time and samples $n^{O(\log (1/\epsilon))}$.

Hardness assumption: Sparse $LPN$
Learning Coverage Functions

*distribution independent setting:
*reduction from Disjoint-DNFs*

\[ \text{poly}(n, 1/\epsilon) \quad \text{PAC algorithm for coverage functions} \]

\[ \Downarrow \]

\[ \text{poly}(n, 1/\epsilon) \quad \text{PAC algorithm for Disjoint-DNFs}. \]

*DNFs where any assignment satisfies at most one term.*
Learning Coverage Functions

distribution independent setting:

reduction from DISJOINT-DNFs

Best: $n^{O(n^{1/3})}$

[Klivans-Servedio 2001]

poly$(n, 1/\epsilon)$ PAC algorithm for coverage functions

poly$(n, 1/\epsilon)$ PAC algorithm for DISJOINT-DNFs.
Application: Private Data Release

Database $D \subseteq \{-1, 1\}^n$
Application: Private Data Release

Database \( D \subseteq \{-1, 1\}^n \)

Counting Query \( q : \{-1, 1\}^n \to \{0, 1\} \)
Application: Private Data Release

Database $D \subseteq \{-1, 1\}^n$

Counting Query $q : \{-1, 1\}^n \rightarrow \{0, 1\}$

Reply $\frac{1}{|D|} \sum_{r \in D} q(r)$
Application: Private Data Release

Database $D \subseteq \{-1, 1\}^n$

Counting Query $q : \{-1, 1\}^n \rightarrow \{0, 1\}$

Reply $\frac{1}{|D|} \sum_{r \in D} q(r)$

goal: answer queries accurately while ensuring “privacy” of participants of the database
Application: Private Data Release

Differential Privacy

[Dwork, McSherry, Nissim, Smith 2006]

Adjacent Databases: \( |D \Delta D'| = 1 \)

Algorithm \( A : 2^{\{-1,1\}^n} \rightarrow R \) is \( \kappa \)-differentially private if for any adjacent databases \( D \) and \( D' \), if for every \( T \subseteq R \)

\[
\Pr[A(D) \subseteq T] \leq e^{\kappa} \Pr[A(D') \subseteq T]
\]

“algorithm behavior doesn’t change much on adjacent databases.”
Application: Private Data Release

**Marginal Queries**

$q$: monotone conjunction
Application: Private Data Release

Releasing Marginals

- simple and natural query class.
Application: Private Data Release

Releasing Marginals

- simple and natural query class.
- official stats of US census, Dept. of Labor and IRS.
Application: Private Data Release

Releasing Marginal

- simple and natural query class.
- official stats of US census, Dept. of Labor and IRS.
- well studied.

Barak et. al.07, UllmanVadhan11, Cheraghchi et. al. 12,
Hardt. et. al.12, Thaler et. al. 12, Bun et. al.13,
Chadrasekharan et. al.14, Dwork et. al. 13
Counting queries with monotone conjunction predicate.

**Goal:** A differentially private algorithm that, given \( D \), publishes a summary, that, for any monotone conjunction \( c \), can be used to compute

\[
\frac{1}{|D|} \sum_{r \in D} c(r) \pm \beta
\]
Application: Private Data Release

Releasing Marginals

Counting queries with conjunction predicate.

**Goal**: A differentially private algorithm that, given D, publishes a summary, that, for any monotone conjunction c, can be used to compute

\[
\frac{1}{|D|} \sum_{r \in D} c(r) \pm \beta
\]
Application: Private Data Release

Releasing Marginals

Counting queries with conjunction predicate.

Goal: A differentially private algorithm that, given D, publishes a summary, that, for any monotone conjunction \( c \), can be used to compute

\[
\frac{1}{|D|} \sum_{r \in D} c(r) \pm \beta
\]

Worst Case Error: Best \( n^{O(\sqrt{n})} \)

[Thaler, Ullman, Vadhan 2012][Dwork, Nikolov, Talwar 2013]
Application: Private Data Release

*Releasing Marginals with low average error*
Application: Private Data Release

Releasing Marginals with low average error w.r.t. some distribution over queries.
Application: Private Data Release

*Releasing Marginals with low average error*

Introduced by [Gupta, Hardt, Roth, Ullman 2011]

[Cheraghchi, Klivans, K, Lee 2012], [Hardt, Rothblum, Servedio 2012],
[Dwork, Nikolov, Talwar 2013]
Application: Private Data Release

Releasing Marginals with low average error

Introduced by [Gupta, Hardt, Roth, Ullman 2011]

[Cheraghchi, Klivans, K, Lee 2012], [Hardt, Rothblum, Servedio 2012],
[Dwork, Nikolov, Talwar 2013]

Releasing Marginals: $n^O(1/\bar{\alpha}^2)$

$\bar{\alpha}$ : average error
Application: Private Data Release

**PAC Learning Algorithm**

\[
poly(n, 1/\epsilon)/\kappa \text{ algorithm for releasing conjunction counting queries privately with low average error over all conjunctions.}
\]
Application: Private Data Release

Releasing Marginals with low average error

average error over all monotone conjunctions?

Most applications require releasing only short marginals.
Application: Private Data Release

Releasing Short Marginals

“k-way marginals”

Counting queries on monotone conjunctions of length $k$.

Worst Case Error: $n^{O(\sqrt{k})}$

[Thaler,Ullman,Vadha 2012] [Dwork,Nikolov,Talwar 2013]
Application: Private Data Release

*Releasing Marginals with low average error*

“k-way marginals”

average error over all monotone conjunctions *of length k*?

previous results not applicable.
Application: Private Data Release

Releasing k-way marginals

Agnostic learning algorithm on symmetric distributions.

\[ k\text{-way marginals can be released with average error } \bar{\alpha} \text{ in time } n^{O(\log 1/\bar{\alpha})}. \]
Application: Private Data Release

Counting Query Function

\( c_D : C \rightarrow [0, 1] \)

“maps queries to their correct answers on a database D”.
Application: Private Data Release

Counting Query Function

\[ c_D : \mathcal{C} \rightarrow [0, 1] \]

Counting Query Class

\[ \mathcal{Q}(\mathcal{C}) = \{ c_D \mid D \subseteq \{-1, 1\}^n \} \]
Application: Private Data Release

Counting Query Class

\[ \mathcal{Q}(C) = \{ c_D \mid D \subseteq \{-1, 1\}^n \} \]

For \( C \) = monotone conjunctions,

\[ \mathcal{Q}(C) = \text{coverage functions.} \]

Learning on a distribution

\[ \downarrow \]

private release of marginals with low average error.

used in previous works on PDR.
PAC and PMAC Learning coverage functions.
PAC Learning
On the Uniform Distribution*
Simple properties of the Fourier spectrum
f: coverage function.
PAC Learning
On the Uniform Distribution
Simple properties of the Fourier spectrum

\( f: \) coverage function.

Monotonicity of Fourier coeffs:

\[
\text{For } T \subseteq V, \quad |\hat{f}(T)| \geq |\hat{f}(V)|
\]
PAC Learning

On the Uniform Distribution

Simple properties of the Fourier spectrum

\( f: \) coverage function.

Monotonicity of Fourier coeffs:

\[
\text{For } T \subset V, \quad |\hat{f}(T)| \geq |\hat{f}(V)|
\]

“find large Fourier coeffs efficiently”
PAC Learning
On the Uniform Distribution
Simple properties of the Fourier spectrum

\( f \): coverage function.

Monotonicity of Fourier coeffs:

\[
\text{For } T \subseteq V, \quad |\hat{f}(T)| \geq |\hat{f}(V)|
\]

L1 spectral norm of \( f \) is bounded above by 2.
PAC Learning
On the Uniform Distribution
Simple properties of the Fourier spectrum

\( f \): coverage function.

Monotonicity of Fourier coeffs:

\[ |\hat{f}(T)| \geq |\hat{f}(V)| \]

sum of absolute values of Fourier coefficients.

L1 spectral norm of \( f \) is bounded above by 2.
PAC Learning

On the Uniform Distribution

Simple properties of the Fourier spectrum

\( f \): coverage function.

Monotonicity of Fourier coeffs:

\[
\text{For } T \subseteq V, \quad |\hat{f}(T)| \geq |\hat{f}(V)|
\]

L1 spectral norm of \( f \) is bounded above by 2.

“not too many large Fourier coeffs”
PAC Learning
On the Uniform Distribution
Simple properties of the Fourier spectrum

\( f \): coverage function.

Monotonicity of Fourier coeffs:

\[
\text{For } T \subseteq V, \quad |\hat{f}(T)| \geq |\hat{f}(V)|
\]

L1 spectral norm of \( f \) is bounded above by 2.

“return linear combination of large Fourier coefficients”
For a coverage function $c$, there is a coverage function $c'$, that depends on $O\left(\frac{1}{\epsilon^2}\right)$ variables and $L1$ approximates $c$ within $\epsilon$ on the uniform distribution.
For a coverage function $c$, there is a coverage function $c'$, that depends on $O(1/\epsilon^2)$ variables and L1 approximates $c$ within $\epsilon$ on the uniform distribution.

“reduce sample complexity - attribute efficiency”
Coverage functions are PAC learnable over the uniform distribution in time $\tilde{O}(n) \cdot \text{poly}(1/\epsilon)$ and samples $\log(n) \cdot \text{poly}(1/\epsilon)$. 
PMAC Learning
On the Uniform Distribution

if values of target coverage functions are “large”, then
additive error $\Rightarrow$ multiplicative error.
if values of target coverage functions are “large”, then additive error $\Rightarrow$ multiplicative error.

- Decompose target coverage function into “regions”.
if values of target coverage functions are “large”, then additive error \Rightarrow multiplicative error.

- Decompose target coverage function into “regions”.
- In each region, value of coverage function large compared to the max \textit{in the region}.
For any coverage* function with maximum M, either

\[ \Pr[c(x) \geq M/4] \geq 1 - \delta/2 \]

or

\[ \exists i \in [n] \quad f(x) \geq \frac{M}{16 \ln (2/\delta)} \]

for every x:

\[ x_i = -1 \]

Recurse on x:

\[ x_i = 1 \]
PMAC Learning
on the Uniform Distribution

• Concentration of Lipschitz submodular functions.
  [Boucheron, Lugosi and Massart 2000]

• for monotone submodular $f$ \[ \mathbb{E}[f] \geq M/4 \]
  [Feige 2006]
Coverage functions are PMAC learnable over the uniform distribution in time $\tilde{O}(n) \cdot \text{poly}\left(\frac{1}{\gamma \delta}\right)$ and samples $\log(n) \cdot \text{poly}\left(\frac{1}{\gamma \delta}\right)$. 
Summary

Learning Coverage Functions

- Fully polynomial PAC and PMAC learning on uniform distribution.
- Agnostic, Product\Symmetric distributions: $n^{O(\log (1/\epsilon))}$
- Arbitrary Distributions: Reduction from Disjoint-DNFs

Applications

- Faster private data release algorithms for marginals.
Future Work

Learning on more general distributions?
Questions?
Application: Private Data Release

Proper PAC Learning Algorithm

- Fully polynomial time algorithm for releasing sanitized database to answer conjunction counting queries privately with low average error over all conjunctions.
PAC Learning
On the Uniform Distribution

Algorithm: Key Idea

• Find variable set $I$ of the junta approx.: estimate singleton Fourier coeffs. monotonicity+ small spectral norm
PAC Learning
On the Uniform Distribution

Algorithm: Key Idea

• Find variable set $I$ of the junta approx.: estimate singleton Fourier coeff. $\text{monotonicity} + \text{small spectral norm}$
• Estimate Large Fourier coeff. with indices subsets of $I$.

$$|\hat{f}(T)| \leq \epsilon \Rightarrow |\hat{f}(V)| \leq \epsilon \text{ for every } V \supseteq T$$
PAC Learning

On the Uniform Distribution

Algorithm: Key Idea

• Find variable set \( I \) of the junta approx.: estimate singleton Fourier coeffs. \text{monotonicity+ small spectral norm}

• Estimate Large Fourier coeffs with indices subsets of \( I \).

\[
|\hat{f}(T)| \leq \epsilon \Rightarrow |\hat{f}(V)| \leq \epsilon \text{ for every } V \supseteq T
\]

• Return linear combination of large Fourier coeffs.