Characterization of Private Learnability via Graph Theory

Abstract

In this work based on (Alon et al., 2024), we will look to answer the question of which class of functions have a pure/approximate DP algorithm and which class of functions donot. Turns out that for a given class of functions $\mathcal{H}$, the answer to whether $\mathcal{H}$ is learnable under pure DP and approximate DP is equivalent to analyzing the maximum fractional clique value and maximum clique in the contradiction graph of $\mathcal{H}$ respectively. We will look at the relation pertaining to the clique value.

1. Preliminaries

Notation. We denote set of all functions from a domain $\mathcal{X}$ to $\{0,1\}$ by $\{0,1\}^{\mathcal{X}}$. Let $\mathcal{X}$ be the domain for the class of functions that we describe. If $\mathcal{D}$ is a probability distribution on a countable set Ω, and $x \in \Omega$, we denote by $\mathcal{D}(x)$ the probability mass of $\mathcal{D}$ on x. We use the notation $x \leftarrow \mathcal{D}$ to denote that x is sampled according to $\mathcal{D}$. For a (possibly randomized) function f over distribution $\mathcal{D}$, the distribution $f(\mathcal{D})$ is induced by evaluating f on independent samples of $\mathcal{D}$.

Definition 1.1 (Supremum)**.** Supremum denoted by $\sup$ of a partially ordered set S is defined as the smallest value greater than (or) equal to all the elements of the set i.e,

$$\forall s \in S,\ \sup(S) \triangleq \min_{a}(a \geq s)$$

If $\sup(S) \in S$ then it is also known as maximum element in the set.

Definition 1.2 (Infimum)**.** Infimum denoted by $\inf$ of a partially ordered set S is defined as the greatest value smaller than (or) equal to all the elements of the set i.e,

$$\forall s \in S,\ \inf(S) \triangleq \max_{a}(a \leq s)$$

If $\inf(S) \in S$ then it is also known as minimum element in the set.

One place where the difference matters is when the infimum (or) supremum doesnot belong to the set. For example, for the set $\inf((0,1)) = 0$ and $\sup((0,1)) = 1$.

1.1. PAC Learning

PAC (Probably Approximately Correct) learning model is a framework developed by (Valiant, 1984; Vapnik & Chervonenkis, 2015) which is useful to characterize the learnability of a class of functions. In general, learning a function colloquially means predicting what the function could be from (usually limited) input and output pairs of the function (training data). Below, we define the common terminology used in the PAC learning literature which would be helpful in understanding the rest of the paper.

Definition 1.3. A set of functions $\mathcal{F} \subseteq \{0,1\}^{\mathcal{X}}$ is said to be consistent with respect to a dataset $S \subseteq \mathcal{X} \times \{0,1\}$ iff there exists at least one function $f \in \mathcal{F}$ such that $S = \{(x_i, f(x_i))\}_i$.

Note: An element s of the dataset S here is assumed to be a tuple (x, y) where x is the set of instances/input to the concept function and y is it's labeled output. The best way to interpret S is as training data.

Definition 1.4 (Concept class)**.** A concept class $\mathcal{C} \subseteq \{0,1\}^{\mathcal{X}}$ is a set of functions that are consistent with the given training data.

Definition 1.5 (Hypothesis class)**.** A hypothesis class $\mathcal{H} \subseteq \{0,1\}^{\mathcal{X}}$ is a set of functions that can be used to predict the concept class of the training data.

Note: $\mathcal{C}$ and $\mathcal{H}$ can be different. For example, $\mathcal{C}$ can be set of all quadratic functions over $\mathbb{Z}$ but our limitation might be to use a linear function over $\mathbb{Z}$ to fit the training data, in which case $\mathcal{H}$ will be set of all linear functions over $\mathbb{Z}$.

Definition 1.6 (Population loss of h)**.** The population loss of a hypothesis $h \in \mathcal{H}$ wrt to a (training data) set S is defined as $L_S(h) := \frac{1}{|S|} \sum_{i \in [m]} \mathbb{1}[h(x_i) \neq y_i]$[1].

In this work, we will assume that for any given training data S, $\mathcal{H} = \mathcal{C}$ (i.e, there exists at least one function in our hypothesis class that fully characterizes any training data given). Formally, $\exists h \in \mathcal{H}$ with $L_S(h) = 0$. And therefore, we start using the words concept class and hypothesis class interchangeably hereafter.

One of the interesting questions to pose would be, given that the model performs really well on the training data S (i.e, it finds a hypothesis $h \in \mathcal{H}$ with $L_S(h) = 0$), we would like

[1] $\mathbb{1}$ denotes the indicator random variable

to characterize how it would perform on the entire domain. The below theorem captures this when $\mathcal{H}$ is finite.

Theorem 1.7. *If $\mathcal{H}$ is finite and S be a sequence of $m \geq 1$ independent random examples of some target function h' then for any $0 \leq \epsilon \leq 1$, given $h \ni L_S(h) = 0$,*

$$\Pr[\sum_{x \in \mathcal{X}} |h'(x) - h(x)| \geq \epsilon] \leq |\mathcal{H}| \cdot e^{-\epsilon m}$$

Proof. Let's say the true error rate of h is ϵ i.e,

$$\mathsf{BAD} = \{x \in \mathcal{X} \mid h(x) \neq h'(x)\} \ni |\mathsf{BAD}| = \epsilon |\mathcal{X}|$$

If a sample were picked uniformly and independently from $\mathcal{X}$ then

$$\Pr_{x \xleftarrow{\$} \mathcal{X}}[x \notin \mathsf{BAD}] = 1 - \epsilon$$

From this, we can bound the probability of all m training samples not being from the bad set as[2]

$$\Pr[\bigwedge_{i \in [m]} x_i \notin \mathsf{BAD}] = (1 - \epsilon)^m \leq e^{-\epsilon \cdot m}$$

The theorem statement then follows from a simple union bound. □

The above result, by a simple rearrangement of terms, gives a bound on the sample complexity of the concept class i.e,

Corollary 1.8. *For a given hypothesis class $\mathcal{H}$, for a given $\epsilon > 0$ and $\delta < 1$, one has to see at least $m \geq \frac{1}{\epsilon} \ln(\frac{|\mathcal{H}|}{\delta})$ number of samples if one has to predict the a concept $h' \in \mathcal{H}$ such that the probability of having more than ϵ misclassifications is less than (or) equal to δ.*

It is worth noting that in many cases, our hypothesis class can be anything. In short, $\mathcal{H}$ is more often than not, very large. In which case, the above result becomes vacuous. So, we need to start looking at the next best thing which gives information about learnability even when $\mathcal{H}$ is infinite.

Given $\mathcal{H}$ can be infinite, we now try to argue about how many samples do we need to see to be able to predict something non-trivial about the concept at hand. Intuitively, we try to capture this idea that if no matter how many samples we see, it could still be any function in the hypothesis space, then the concept is complex and not learnable. But, before that let's formulate what it means to learn $\mathcal{H}$.

Definition 1.9 (Learning Algorithm (Alon et al., 2019))**.** We say that an algorithm $\mathcal{A}$ learns a class $\mathcal{H}$ with α-error, $(1 - \beta)$ confidence and sample-complexity m if for every realizable set of independently sampled training data of 'm' examples $S \ni |S| = m$[3]:

$$\Pr[L_S(h) > \alpha] \leq \beta$$

where the probability is taken over the random coins of $\mathcal{A}$ and the random coins of sampling S[4].

Theorem 1.10. *A hypothesis class $\mathcal{H} \subseteq \{0,1\}^{\mathcal{X}}$ is PAC-learnable $\iff$ $\exists$ a learning algorithm $\mathcal{A}$ wrt $\mathcal{H}$.*

Definition 1.11 (Dichotomy of a Set)**.** Any (disjoint) partition of a set is called a dichotomy of a set. In essence, a dichotomy can be thought of as labeling the data in a set as 0 and 1.

Definition 1.12 (Shattering)**.** A set $S \subseteq \mathcal{X}$ is shattered by a hypothesis space $\mathcal{H}$ iff $\forall$ dichotomies of S $\exists h \in \mathcal{H}$ that is consistent with the given dichotomy.

Now that we have defined the tools, let us look at a quantity of a hypothesis class $\mathcal{H}$ which tries to capture the intuition from before.

Definition 1.13 (VC-Dimension (Vapnik & Chervonenkis, 2015))**.** VC Dimension of a hypothesis class $\mathcal{H} \subseteq \{0,1\}^{\mathcal{X}}$ denoted by $\mathsf{VC}(\mathcal{H})$ is the size of the largest finite subset of $S \subseteq \mathcal{X}$ that is shattered by $\mathcal{H}$.

Now, notice that if $\mathsf{VC}(\mathcal{H}) = d$ where d is finite then it means that there exists one labeling of the data which is not possible with the current concept class and therefore we have learned something about our concept class by looking at a finite number of training instances. More formally,

Theorem 1.14. *A hypothesis class $\mathcal{H} \subseteq \{0,1\}^{\mathcal{X}}$ is PAC-learnable $\iff$ $\mathsf{VC}(\mathcal{H})$ is finite*

In essence, this shows that looking at a particular property of $\mathcal{H}$ (VC-dimension in this case) can conclusively tell us if something is learnable or not even if $\mathcal{H}$ is infinite[5].

Now that we have a characterization that tell us if something is learnable. A natural question would be to ask what is privately learnable.

1.2. Differential Privacy

We will now define what it means to learn a function privately. For this we will look to the differential privacy.

Definition 1.15 ((ϵ, δ)-DP)**.** A function f is (ϵ, δ) − DP (Differentially Private) if $\forall\, x \sim x'$ and all events E,

$$\Pr[f(x) \in E] \leq e^{\epsilon} \cdot \Pr[f(x') \in E] + \delta$$

[2] Assuming, $1 - \epsilon \leq e^{-\epsilon}$ for small enough ϵ

[3] $\ni$ denotes such that

[4] Ideally $\alpha = O(1)$ and $\delta = m^{-\omega(1)}$

[5] It is worth noting that it might not be efficient to compute the VC-dimension for a given hypothesis class.

When $\delta = 0$, we will use the short-hand ϵ-DP for $(\epsilon, 0)$-DP and it is called a pure DP otherwise it is called approximate DP.

Now, given that we have defined what it means for a function to be differentially private. We will ask if a function class $\mathcal{H}$ is privately learnable.

Definition 1.16 ((Alon et al., 2019; Bun et al., 2020))**.** A class of functions $\mathcal{H} \subseteq \{0,1\}^{\mathcal{X}}$ is said to be *privately learnable* if there exists a learnable algorithm $\mathcal{A}$ such that $\mathcal{A}$ is (ϵ, δ)-DP for some $\epsilon = O(1)$, $\delta \leq m^{-\omega(1)}$.

In this work, we will mainly focus on characterizing functions that satisfy the approximate-DP requirement (i.e, $\delta \neq 0$). However similar ideas can be used to give an equivalent characterization of pure DP.

1.3. Motivation

There have been different and seemingly disjoint characterizations of pure DP ($\delta = 0$) and approximate DP ($\delta \neq 0$) in the literature. However (Alon et al., 2024) show that two naturally close variants in graph theory (clique and fractional clique of a graph based on a function class $\mathcal{H}$) characterize private learnability of that class wrt approximate and pure DP definition. Other than very valid reason of a purely theoretical pursuit, these kind of unifying characterizations of an already known concept are important as they often lead to a better understanding about that concept. And given the unification is in terms of well-studied combinatorial objects (like graphs), it will enable us to make use of the very rich plethora of well-studied tools and results to further the concept at hand.

Coming back to (Alon et al., 2024). It builds on two of seemingly disjoint characterizations of pure DP and approximate DP called representation dimension and Littlestone dimension. As we will focus on approximate DP, we will restrict our focus to the Littlestone dimension.

1.4. Littlestone Dimension

(Littlestone, 1988) captures the situation in which a learner adaptively looks at training examples in each round and keeps updating their prediction of the concept that fits well with everything that was seen until then. Naturally, the learner is allowed to make "mistakes" (wrong predictions about the actual concept class). Ideally, even the paper wants the number of mistakes to be independent of the samples seen. And, for a given class of functions, the original paper wants to find a bound on the number of mistakes that a learner would commit in the worst-case before learning about the function. This is what the Littlestone dimension captures. And, if the number of mistakes that the learner makes is infinite before learning it, then it is not learning anything.

The idea of Littlestone dimension is better understood in the setting of a (min-max) game involving the learner and the environment. The goal of the learner is to predict the concept from which the training samples are given and the environment's goal is to adaptively pick the examples and their outputs such that the learner makes maximum number of mistakes before it could learn the function. This whole game can be represented as a binary tree in which the nodes represent the training example from the domain $\mathcal{X}$ that is given to the learner by the environment and the edges represent the opposite labeling to the one picked by the learner. The goal of the environment then it is to make the tree as deep as possible by choosing samples and their labeling for the worst-case. Below, we abstract out all these details and present this idea of Littlestone dimension as a purely combinatorial object called "Mistake Tree" that seems closer to the VC dimension and the concept of shattering.

Definition 1.17 (Mistake Trees)**.** A *mistake tree* wrt a domain $\mathcal{X}$ is a binary decision tree where each vertex represents a particular instance from $\mathcal{X}$ whose edges are labeled by 0 or 1 such that each internal node has one outgoing edge labeled 0 and one outgoing edge labeled 1.

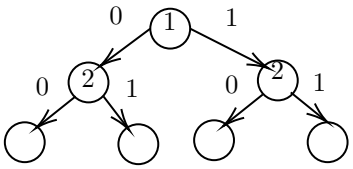

Figure 1. A mistake tree with $\mathcal{X} = \{1, 2\}$

A root-to-leaf path in a mistake tree is a sequence of labeled examples $(x_1, y_1), \ldots (x_d, y_d)$. The point x_i is the i^{th} internal node in the path, and y_i is the label of its outgoing edge to the next node in the path.

A mistake tree is said to be *shattered* by a hypothesis class $\mathcal{H}$ iff for every root to path is consistent with some function $h \in \mathcal{H}$.

Definition 1.18 (Littlestone Dimension)**.** Littlestone dimension of a hypothesis class $\mathcal{H}$ denoted by $\mathsf{LD}(\mathcal{H})$ is the largest number d such that there exists a complete binary tree that is shattered by $\mathcal{H}$.

Intuitively, one can think of Littlestone dimension capturing the idea of learning a function despite some error in the samples that it has seen. In some sense, it is providing an algorithm that is robust to the "mistakes" of the learner and as we have seen before in our class (Georgiev & Hopkins, 2022; Asi et al., 2023; Hopkins et al., 2023); Robustness of an algorithm is a form of guarantee on it's privacy. Intuitively, this is because, the samples in which two datasets

differ (as part of our definition of DP) can be thought of as "mistakes" that will be ignored and, hence an algorithm that is robust to mistakes is intuitively also private.

Theorem 1.19 ((Alon et al., 2019; Bun et al., 2020))**.** *A class of functions $\mathcal{H} \subseteq \{0,1\}^{\mathcal{X}}$ is* privately learnable $\iff$ *Littlestone dimension of $\mathcal{H}$ is finite*

The rest of the paper would be to use this result about the Littlestone dimension to show that a hypothesis class is approx-DP learnable $\iff$ a contradiction graph of $\mathcal{H}$ has a clique of $\mathsf{poly}(\mathsf{m})$. But, before seeing that we will have to define few graph theory related terms.

1.5. Graph Theory

Definition 1.20 (Clique)**.** A clique of an undirected graph $G = (V, E)$ is any subset of vertices $S \subseteq V$ such that for any two distinct vertices $u, v \in S$, $\{u, v\} \in E$.

We denoted the clique number of an undirected graph G by $\omega(G)$ and is defined as the size of maximum clique in the graph.

2. Private Learnability via Graph Theory

Definition 2.1 (Contradiction graph of $\mathcal{H}$)**.** A contradiction graph of a concept class $\mathcal{H} \subseteq \{0,1\}^{\mathcal{X}}$ of order m is denoted by $G_m(\mathcal{H}) := (V_m(\mathcal{H}), E_m(\mathcal{H}))$ where,

1. The vertices are datasets (with labeling, i.e, think of each instance of dataset to be of the form (x, b) where $x \in \mathcal{X}$ and $b \in \{0,1\}$) of size m that are consistent with $\mathcal{H}$ and,
2. There exists an edge between two vertices $S, S' \iff$ there exists a $x \in \mathcal{X}$ such that $(x, 0) \in S$ and $(x, 1) \in S'$ (or) vice versa. We say such datasets contradict.

Definition 2.2 (Clique Dimension)**.** The clique dimension of a hypothesis class $\mathcal{H}$, denoted by $\mathsf{CD}(\mathcal{H})$ is defined as

$$\mathsf{CD}(\mathcal{H}) := \sup_m \left(\{\omega(G_m(\mathcal{H})) = 2^m\} \right) \in \mathbb{N} \cup \{\infty\}$$

Theorem 2.3. *Let $\mathcal{H}$ be a hypothesis class and $m \in \mathbb{N}$ then $\omega(G_m(\mathcal{H})) \leq 2^m$.*

Proof. Let v_S be an arbitrary vertex in the contradiction graph $G_m(\mathcal{H})$ that represents dataset S of 'm' instances i.e, $S = \{(x_i, y_i)_{i \in [m]}\}$. To have a conflict with another dataset S', there must exist an x_i such that $(x_i, 1 \oplus y_i) \in S'$. And the maximum number of conflicts possible are $(\binom{m}{1} + \ldots + \binom{m}{m})$ i.e, it can have conflict with another dataset in atleast 1 and atmost m entries. This means that the degree of vertex v_S is atmost $(\binom{m}{1} + \ldots + \binom{m}{m}) = 2^m - 1$. And if a vertex in a graph can have almost degree d then there cannot be a clique of size more than d. □

Theorem 2.4. *Let $\mathcal{H}$ be an hypothesis class then* $\mathsf{LD}(\mathcal{H}) \leq \mathsf{CD}(\mathcal{H})$

Proof. Let $\mathsf{LD}(\mathcal{H}) = m$. In this case, any mistake tree that is drawn for an 'm'-sized training dataset (or) less will have a complete binary tree by definition. Each root to leaf path represents a set of training examples and any two distinct root to leaf paths differ at-least in one ancestor. Therefore, this means that any two distinct training dataset of size 'm' will have at least one data point labeled differently. That is between any two vertices in $G_m(\mathcal{H})$ there exists an edge between them i.e, for any $m \in \mathbb{N}$, $\omega(G_m(\mathcal{H})) \geq 2^m$. This means $\mathsf{CD}(\mathcal{H}) \geq m$. Therefore, $\mathsf{LD}(\mathcal{H}) \leq \mathsf{CD}(\mathcal{H})$. □

Theorem 2.5. *Let $\mathcal{H}$ be an hypothesis class and let $\mathcal{K}$ be a clique in $G_m(\mathcal{H})$. Then, there exists an $x \in \mathcal{X}$ such that:*

1. *Atleast $\frac{|\mathcal{K}|-1}{2m}$ datasets corresponding to the vertices in $\mathcal{K}$ contain $(x, 1)$*
2. *Atleast $\frac{|\mathcal{K}|-1}{2m}$ datasets corresponding to the vertices in $\mathcal{K}$ contain $(x, 0)$*

Proof. Let $k = |\mathcal{K}|$. For any $(x^\star, b^\star) \in (\mathcal{X} \times \{0,1\})$, we define

$$\mathcal{K}_{(x^\star, b^\star)} := \{S \in \mathcal{K} \mid (x^\star, b^\star) \in S\}$$

Below, we will define an algorithm that is guaranteed to terminate and then look at it's properties to argue that we have indeed proved our theorem.

Input: $\mathcal{K}$

As long as there is a $S \in \mathcal{K}$ such that $(x, b) \in S$ and $|\mathcal{K}_{(x, 1 \oplus b)}| < \frac{c-1}{2m}$:

1. Update $S \leftarrow S \setminus \{(x, b)\}$
2. Update the edges of $\mathcal{K}$ i.e, if any dataset S' no longer contradicts S then remove the edge between them

- Notice, that the $|\mathcal{K}| = k$ and each vertex represents a 'm' sized dataset. Therefore, total number of samples are $k \cdot m$ and if the algorithm is removing one dataset per iteration then in atmost $k \cdot m$ iterations. It will stop. So, the algorithm definitely terminates.
- At each iteration $\frac{c-1}{2m}$ edges are removed. So total number of $k \cdot m \cdot \frac{k-1}{2m} = \binom{k}{2}$ edges are removed.

- Therefore, not all edges present in the clique have been removed by the end. Because total number of edges are $\binom{k}{2}$. So, we have atleast one edge remaining in $\mathcal{K}$ and two vertices after the above algorithm terminates. That is, there are atleast two distinct datasets $S, S' \in \mathcal{K}$ that contradict each other on some example x i.e, wlog $(x, 0) \in S$ and $(x, 1) \in S'$ and because $(x, 0)$ survived the algorithm, it means atleast $\frac{k-1}{2m}$ have $(x, 0)$ and $\frac{k-1}{2m}$ have $(x, 1)$.

□

Intuitively, similar to the way of how we found out the degree of a vertex in a possibly infinite graph is bounded by 2^m combinatorially. The above algorithm, finds out a point with specific properties by exploiting the structure of the contradiction graph.

Theorem 2.6. *Let $\mathcal{H}$ be an hypothesis class then, for any $m \in \mathbb{N}$,*

$$\omega(G_m(\mathcal{H})) \leq (2m+1)^{\mathsf{LD}(\mathcal{H})}$$

Proof. Let $\mathcal{K}$ be the maximum sized clique in $G_m(\mathcal{H})$ i.e, $\omega_m(G) = |\mathcal{K}|$. From the previous theorem, we known that there exists $x \in \mathcal{X}$ which is present as $(x, 0)$ in half the datasets of the clique and. as $(x, 1)$ in half the datasets of the clique. Let,

$$Y = \{S \in \mathcal{K} \mid (x, 1) \in S\}$$
$$N = \{S \in \mathcal{K} \mid (x, 0) \in S\}$$

Note that Y and N form cliques themselves wrt $G_m(\mathcal{H})$ because $Y \subseteq \mathcal{K}$ and $N \subseteq \mathcal{K}$.

Take x to be the root of a mistake tree, and recursively keep constructing the mistake tree by using the cliques in Y, N. This way, in the i-th step we have 2^i cliques and the size of each clique is at least $\frac{\omega_m}{(2m+1)^i}$. Note that the depth of any mistake tree is bounded by $\mathsf{LD}(\mathcal{H})$. It means that after $\mathsf{LD}(\mathcal{H})$ many iterations, the above process should stop i.e, $|Y| = |N| = 1$.

$$\implies \frac{\omega(G_m(\mathcal{H}))}{(2m+1)^{\mathsf{LD}(\mathcal{H})}} \leq 1$$

From Theorem 2.6, we can say that if $\mathsf{LD}(\mathcal{H}) = c$ is finite (i.e, independent of m i.e, constant) then $\omega(G_m(\mathcal{H})) \leq (2m+1)^c = \mathsf{poly}(m) \implies \mathsf{CD} = \mathsf{poly}(m)$.

From Theorem 2.4, If $\mathsf{LD}(\mathcal{H}) \neq$ finite $\implies \mathsf{CD}(\mathcal{H}) \neq$ finite.

Therefore, $\mathsf{LD}(\mathcal{H})$ is finite $\iff \mathsf{CD}(\mathcal{H})$ is finite. □

3. Thoughts and Contributions

Contributions. The paper provides an important result but it treats Littlestone dimension and other aspects as combinatorial object rather than providing insight into their use. Therefore, I believe this write up would be a nice accompanying work to it that builds intuiton and the required background with almost no assumptions.

Future work: There are few things that the paper itself leaves for future work such as showing tighter bounds between Littlestone dimension and clique dimension. Currently, a Littlestone dimension of value d translates to $(2m+1)^d$ is terms of clique dimension as seen in Theorem 2.6. A natural question to ask therefore would be to see if a tighter relation could be established between LD and CD. A similarly interesting thing to explore is to see if other notions of privacy can also be unified under the umbrella of graph theory by defining new variants of clique. Currently, it is clear that there will function classes for which computing clique dimension (or) Littlestone dimension (or) even VC dimension might be intractable. In which, can we find some sub class of contradiction graphs in which this dimension is easy to compute (maybe, for example: sparse graphs, expander graphs) and then see which kind of functions will have such graph variants in their contradiction graphs and see if that is giving more insight into learning those specific class of functions. Another interesting question is to see if contradiction graphs can directly establish some relation with VC Dimension (why stop at private learning, can we use this to generalize this to different kinds of learning).

Personal Thoughts: During this, I read the 1988 paper of Littlestone dimension which looks at this problem very differently and I wonder if given our updated knowledge about relations such as robustness and privacy, is it a good time to look back at some of the fundamental papers again and rethink them. If yes, then what are the good old papers to read. I felt that the 2024 paper (Alon et al., 2024) almost removes most of the intuiton and compared to that the 1988 paper does a really good job at the reasoning of the combinatorial object (i.e, mistake trees), of-course because it was seminal introductory work. Also, there was a simple proof for bounding the size of clique (Theorem 2.3) compared to the one given in paper. So, I wonder if we can find simpler proofs for the fractional clique as their clique counterparts seem to have simpler proofs.

Note: There is appendix too!

References

Alon, N., Livni, R., Malliaris, M., and Moran, S. Private pac learning implies finite littlestone dimension. In *Proceedings of the 51st Annual ACM SIGACT Symposium*

on Theory of Computing, pp. 852–860, 2019.

Alon, N., Moran, S., Schefler, H., and Yehudayoff, A. A unified characterization of private learnability via graph theory. In *The Thirty Seventh Annual Conference on Learning Theory*, pp. 94–129. PMLR, 2024.

Asi, H., Ullman, J., and Zakynthinou, L. From robustness to privacy and back. In *International Conference on Machine Learning*, pp. 1121–1146. PMLR, 2023.

Bun, M., Livni, R., and Moran, S. An equivalence between private classification and online prediction. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 389–402. IEEE, 2020.

Georgiev, K. and Hopkins, S. Privacy induces robustness: Information-computation gaps and sparse mean estimation. *Advances in neural information processing systems*, 35:6829–6842, 2022.

Hopkins, S. B., Kamath, G., Majid, M., and Narayanan, S. Robustness implies privacy in statistical estimation. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, pp. 497–506, 2023.

Littlestone, N. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine learning*, 2(4):285–318, 1988.

Valiant, L. A theory of the learnable‖ communications of the acm, 27 (11): 1134-1142. *Google Scholar Google Scholar Digital Library Digital Library*, 1984.

Vapnik, V. N. and Chervonenkis, A. Y. On the uniform convergence of relative frequencies of events to their probabilities. In *Measures of complexity: festschrift for alexey chervonenkis*, pp. 11–30. Springer, 2015.

A. Fractional Clique

In this section, we will give a flavor of results related to clique (i.e, approx-DP) getting translated to fractional clique (i.e, pure DP).

Definition A.1 (Independent Set)**.** An independent set on a undirected graph $G = (V, E)$ is any subset of vertices $S \subseteq V$ such that for any two distinct vertices $u, v \in S$, $\{u, v\} \notin E$.

We will now prove a simple but useful result from graph theory which lets us look at a clique from the perspective of independent sets of the graph.

Theorem A.2. *For a given undirected graph $G = (V, E)$, $\delta \subseteq V$ is a clique iff for any independent set $I \subseteq V$,*

$$\sum_{v \in I} \delta(v) \leq 1$$

where we overload the notation of δ to represent the indicator variable of clique δ i.e, $\delta : V \to \{0, 1\}$ such that $\delta(v) = 1 \iff v \in \delta$.

Proof. ($\implies$) Let δ be the clique. This means that for any distinct vertices in δ there exists an edge between them and therefore both v_1 and v_2 cannot be part of any independent set at the same time.

($\impliedby$) (Proof by Contradiction). Let δ be a subset of vertices such that it's not a clique. This means that there exists two distinct vertices v_1 and v_2 that donot have an edge between them. Implies $\{v_1, v_2\}$ is an independent set such that $\delta(v_1) + \delta(v_2) = 2$. □

Definition A.3 (Fractional clique)**.** Stated in terms of Theorem A.2, For a given undirected graph $G = (V, E)$, $\delta \subseteq V$ is a fractional clique iff for any independent set $I \subseteq V$,

$$\sum_{v \in I} \delta(v) \leq 1$$

where we overload the notation of δ again but this time to represent a function δ i.e, $\delta : V \to [0, 1]$ where fractional clique assigns some value in $[0, 1]$ to each vertex. This can be seen as a linear programming relaxation of an otherwise integer programming problem.

We denoted the fractional clique number of an undirected graph G by $\omega^\star(G)$ and is defined as $\max_{\delta} \sum_{v \in I} \delta(v)$.

Corollary A.4. *Because of the way it is defined (i.e, fractional clique being an LP-relaxation of the clique), it follows directly that for any undirected graph G, $\omega(G) \leq \omega^\star(G)$.*

Below, we re-write the actual proof given in the paper instead of the proof given in Theorem 2.3[6]

Theorem A.5. *Let $\mathcal{H}$ be a hypothesis class and $m \in \mathbb{N}$ then $\omega^\star(G_m(\mathcal{H})) \leq 2^m$.*

Proof. We will now prove that $\omega^\star(G_m(\mathcal{H})) \leq 2^m$ and then the result that $\omega(G_m(\mathcal{H})) \leq 2^m$ follows from Theorem A.4.

Now, first we notice that given any fractional clique δ of any undirected graph and an independent set I of the graph, $\delta(v) \leq 1$ (follows from Theorem A.3).
$v \in I$

Now, we rewrite the above observation for $G_m(\mathcal{H})$, note that for any $h \in \mathcal{H}$, the vertices corresponding to the set of all 'm' sized datasets that are consistent with h form an independent set. Therefore we can say, for any $h \in \mathcal{H}$

$$X = \sum_{S \in V_m(\mathcal{H})} \delta(S) \cdot \mathbb{1}[S \text{ is consistent wrt } h] \leq 1$$

[6]Please note that this is a stronger statement than the one proved in Theorem 2.3

Now, we will sample a random h uniformly and independently from the hypothesis space $\mathcal{H}$. We do this by randomly assigning a particular label to every element of the domain $\mathcal{X}$,

$$\begin{aligned} 1 \leq \mathop{\mathbb{E}}_{h \xleftarrow{\$} \mathcal{H}}[X] &= \sum_{S \in V_m(\mathcal{H})} \mathbb{E}\big[\delta(S) \cdot \mathbb{1}[S \text{ is consistent wrt } h]\big] \\ &= \sum_{S \in V_m(\mathcal{H})} \delta(S) \cdot \Pr\big[S \text{ is consistent wrt } h\big] \\ &= \sum_{S \in V_m(\mathcal{H})} \delta(S) \cdot 2^{-m} \\ &= 2^{-m}|\delta| \end{aligned}$$

$$\implies |\delta| \leq 2^m \qquad \square$$

.**AUTHORERR: Missing \icmlcorrespondingauthor.**

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.