Generative Models of Huge Objects

Lunjia Hu
Department of Computer Science, Stanford University, CA, USA

Inbal Rachel Livni Navon
Department of Computer Science, Stanford University, CA, USA

Omer Reingold
Department of Computer Science, Stanford University, CA, USA

Abstract

This work initiates the systematic study of explicit distributions that are indistinguishable from a *single* exponential-size combinatorial object. In this we extend the work of Goldreich, Goldwasser and Nussboim (SICOMP 2010) that focused on the implementation of huge objects that are indistinguishable from the uniform distribution, satisfying some global properties (which they coined truthfulness). Indistinguishability from a single object is motivated by the study of generative models in learning theory and regularity lemmas in graph theory. Problems that are well understood in the setting of pseudorandomness present significant challenges and at times are impossible when considering generative models of huge objects.

We demonstrate the versatility of this study by providing a learning algorithm for huge indistinguishable objects in several natural settings including: dense functions and graphs with a truthfulness requirement on the number of ones in the function or edges in the graphs, and a version of the weak regularity lemma for *sparse* graphs that satisfy some global properties. These and other results generalize basic pseudorandom objects as well as notions introduced in algorithmic fairness. The results rely on notions and techniques from a variety of areas including learning theory, complexity theory, cryptography, and game theory.

2012 ACM Subject Classification Theory of computation → Pseudorandomness and derandomization; Theory of computation → Random network models; Theory of computation → Generating random combinatorial structures

Keywords and phrases pseudorandomness, generative models, regularity lemma

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.5

Related Version *Full Version*: `https://arxiv.org/abs/2302.12823` [17]

Funding *Lunjia Hu*: Supported by the Simons Foundation Collaboration on the Theory of Algorithmic Fairness, Omer Reingold's NSF Award IIS-1908774, and Moses Charikar's Simons Investigators award.
Inbal Rachel Livni Navon: Supported by the Simons Foundation Collaboration on the Theory of Algorithmic Fairness, the Sloan Foundation Grant 2020-13941, and the Zuckerman STEM Leadership Program.
Omer Reingold: Supported by the Simons Foundation Collaboration on the Theory of Algorithmic Fairness and the Simons Foundation Investigators award 689988.

1 Introduction

A pseudorandom distribution is indistinguishable from the uniform distribution to a set of computationally bounded distinguishers. Pseudorandomness is a cornerstone of many areas of computer science and mathematics. The variability of pseudorandom distributions stems from the different objects they can generate (bit strings, functions, permutations and more) and the different computational bounds that can be imposed on the distinguishers. In the area of cryptography, it is typical to consider powerful distinguishers that are at least

Table 1 Comparison between problem settings.

	What do we imitate?	What do we construct?
Pseudorandomness	distribution of objects	distribution of objects
Explicit construction (e.g. expander graphs)	distribution of objects	single object
Our setup: generative models	single object	distribution of objects

polynomial time, giving rise to central notions such as pseudorandom generators [3, 41], pseudorandom functions [8] and pseudorandom permutations [27]. More limited distinguishers give rise to other fundamental notions such as k-wise independent hashing and ε-biased distributions (cf. [40, 29]). In the area of explicit combinatorial constructions, we typically try to emulate the uniform distribution by a single object, rather than with a distribution. A primer example is the fundamental notion of expander graphs (see [16, 39] for surveys), with its multiple variants (including various notions of randomness extractors). These are graphs that are indistinguishable from a uniformly selected graph to a limited set of distinguishers (such as distinguishers that check if a random edge crosses a given cut).

In these classic areas of pseudorandomness, a distribution, or even a single object, is constructed to emulate a distribution (typically, the uniform distribution). In this paper we ask for a distribution to emulate *a single object* (Table 1). This reversal may seem absurd from the perspective of pseudorandomness but makes perfect sense from the perspective of generative models. An early exposure of the TOC community to generative models was with respect to the World Wide Web. These were models that produce distributions of graphs that imitate some properties of the Web, such as power law on the degrees of nodes (see [28] for a survey). At any given point, the web is a single graph, but it is also a very large graph that does not have a simple description. Generative models gave a useful way to analyze, or estimate through experimentation, the expected performance of protocols on the Web.

Other well studied generative models are the stochastic block model [15] and the more elaborate mixed membership stochastic block model [1]. Consider a graph representing some connections between individuals, such as the connectivity of the social network. The stochastic block model partitions the vertices into disjoint communities and for every two communities assigns a probability of connection. This model represents a distribution over graphs where for each two vertices, an edge is placed independently with the probability assigned to the pair of communities of its end points. (In the mixed-membership model, each vertex is assigned a distribution over communities.) These models help identify useful substructures within a social structure such as sub-communities or different social roles. But given a single social network, B^*, what is an appropriate model to capture it? After all, a model describes a distribution over networks rather than the single network we are trying to explain. A prevalent approach is to aim at the maximum likelihood model. Out of all models, the probability of sampling B^* is maximized under the maximum likelihood model. Heuristics for estimating the maximum likelihood model have been playing a major roles in the generative-model literature and in its application in practice. It should be noted that the probability that the model would produce B^* is often very small. In this light, the meaningfulness of a maximum-likelihood models may be debated and may depend on a particular setting.

From the perspective of indistinguishability, it may be more natural to seek a model that produce a distribution that is indistinguishable from B^* to a meaningful set of distinguishers. For example, in the case of the stochastic block model, natural distinguishers are defined

by two sets of vertices U and V and ask what is the probability that a random edge in the graph crosses from U to V. A stochastic block model that fool all such distinguishers is exactly what is given by the Frieze-Kannan regularity lemma (also known as the weak regularity lemma) [7]. The indistinguishability perspective on generative models and known connections between learning and pseudrandomness, which we will discuss shortly, are both a motivation as well as the starting point of this work.

1.1 Overall Goal: Indistinguishable Generative Models of Huge Objects

In many of the applications of generative models, such as modeling the Web or a social network, the objects being modeled are huge. In this paper, we aim at a *systematic theory of efficiently learning and implementing huge generative models*. Our models will generate a distribution of objects satisfying some global properties that are indistinguishable from a fixed combinatorial object. Such a theory presents non-trivial challenges that do not manifest themselves neither when generating huge pseudorandom objects, nor in generative models of polynomial-size objects.

Concretely, we assume that we have access to an object B^* with exponential size. For example, B^* could be a function with an exponentially large domain, or a graph with exponentially many vertices and edges. We are most interested in the case where the object B^* is too large to read or process as a whole, and we have to access it by sampling: for example, when B^* represents a function $f : X \to \{0,1\}$ with $|X|$ being exponentially large, we may access B^* by asking for random pairs $(x, f(x)) \in X \times \{0,1\}$ (sample access) or random inputs $x \in X$ conditioned on $f(x) = 1$ (support access). Given access to the huge object B^*, our goal is to create a generative model M for B^*. Here, our model M represents a distribution over objects, and we want to ensure that this distribution is indistinguishable from B^* to all distinguishers D in a class $\mathcal{D}$. Specifically, if we use $D^B \in \{\text{“accept”}, \text{“reject”}\}$ to denote the output of distinguisher D given sample/support access to object B, our indistinguishability requirement is that for every $D \in \mathcal{D}$,

$$|\Pr[D^{B^*} = \text{“accept”}] - \mathbb{E}_{B\sim M}[\Pr[D^B = \text{“accept”}]]| \leq \varepsilon.$$

We aim for building an *efficient* learner L that can output a model M satisfying the indistinguishability requirement above when given sample/support access to the true object B^*. When B^* is exponentially large (which is the case we are interested in), the output model M also needs to generate exponentially large objects, and thus we cannot expect an efficient learner to directly output M. Instead, we want our learner to output an *efficient implementation* of M, which, roughly speaking, is a randomized algorithm that can efficiently provide sample/support access to objects drawn from M (Definitions 13 and 14).

Our goal of learning a generative model M indistinguishable from the true object B^* is analogous to the problem addressed by Goldreich, Goldwasser and Nussboim [9] in the area of pseudorandomness. They study the problem of efficiently implementing a distribution of huge objects, satisfying some global properties, that are indistinguishable from the uniform distribution of such objects. Follow-up works of [9] such as [31, 30] study efficient implementations that are indistinguishable from certain distributions of huge random graphs. All these works aim to achieve indistinguishability from a known distribution of objects, whereas in our problem of learning a generative model, we assume that the true object B^* is initially *unknown*, and to collect information about B^*, we additionally need a learner L that can use sample/support access to B^* to *efficiently* construct an implementation of an indistinguishable model.

Beyond indistinguishability, we also aim to achieve the notion of *truthfulness* introduced in [9]. To demonstrate this notion, consider pseudorandom permutations $f_s : \{0,1\}^n \mapsto \{0,1\}^n$ [27]. A distribution of permutations is pseudorandom if it is indistinguishable from the uniform distribution of permutations. It should be noted that a pseudorandom $\{0,1\}^n \mapsto \{0,1\}^n$ function [8] is also indistinguishable from a random permutation over $\{0,1\}^n$ (as long as the number of queries are sufficiently smaller than $2^{n/2}$). Nevertheless, insisting that the pseudorandom objects satisfy the global condition of being a permutation is critical in the applications of pseudorandom permutations. This motivates the distinction of [9] between indistinguishability (that the pseudorandom objects are indistinguishable from a uniform object to a class of distinguishers) and *truthfulness* which is a global property that needs to hold exactly or approximately in a statistical sense. In our setup, a generative model M is truthful if every object B drawn from the distribution represented by M satisfies a certain global property. For example, when the true object B^* is a function $f^* : X \to \{0,1\}$ with support size $|\{x \in X : f^*(x) = 1\}|$ being k, a truthful requirement on a generative model M for B^* may restrict M to always generate functions with support size k.

The study of implementing huge pseudorandom objects [32, 9, 31, 30] has pseudorandom functions and permutations as vital building blocks. Besides these building blocks, our techniques for learning generative models of huge objects also come from connections to the regularity lemma and especially the work of Trevisan, Tulsiani and Vadhan [37]. In [37], they construct an efficiently-implementable function $f : X \mapsto [0,1]$ which is indistinguishable from some $f^* : X \mapsto [0,1]$ to a family of distinguishers represented by functions $g : X \mapsto [0,1]$. Indistinguishability here means that $|\mathbb{E}[f(x)g(x)] - \mathbb{E}[f^*(x)g(x)]|$ is smaller than some error parameter ε. After [37], the problem of creating indistinguishable functions and its applications to cryptography are further studied in [38, 22, 34, 35, 36, 4]. These works assume that the true function f^* is known and they do not explicitly deal with the problem of learning f^*, but the corresponding learning task has been studied in the algorithmic fairness literature through the notions of multi-accuracy, multi-calibration, and outcome indistinguishability [14, 23, 5, 13, 6]. When applying techniques from these works to solve some problems in our setting, we need to deal with additional challenges such as the truthfulness requirement that we want our generative model to satisfy.

1.2 Our Results

The main conceptual contribution of this paper is in suggesting a new frontier for the study of indistinguishability, which is highly motivated and technically challenging. As we introduce in Section 1.1, the notion of indistinguishability from a single huge object combines at its core the areas of learning theory and pseudorandomness which, as recent research uncovered, have deep connections, providing a way to describe and address a rich landscape of natural problems. Below we summarize the main problems we address in this new framework.

Truthful Learning That Preserves Support Size

Suppose we have sample access to a function $f^* : \{0,1\}^n \to \{0,1\}$ and we want to build an indistinguishable generative model for f^*. Here sample access allows us to observe pairs $(x, f^*(x))$ with x drawn uniformly at random from $\{0,1\}^n$, and accordingly, we assume that every distinguisher also decides to accept or reject based on such a random pair $(x, f(x))$ from a function f that may or may not be the true f^*. This task of learning a generative model for a binary function is closely related to the task of *no-access outcome indistinguishability* studied in [5], and it has been observed that the task can be reduced to multi-accuracy. Indeed,

assuming that the distinguishers have bounded complexity and can be learned efficiently, using previous algorithms in [14, 23, 5], we can design an efficient learner that constructs a generative model indistinguishable from f^* (Theorem 19). The model constructed this way is specified using a predictor $p : \{0,1\}^n \to [0,1]$, and the model represents the distribution of functions $f : \{0,1\}^n \to \{0,1\}$ where the function value $f(x)$ is distributed independently for every $x \in \{0,1\}^n$ according to the Bernoulli distribution $\mathsf{Ber}(p(x))$ with mean $p(x)$.

Learning generative models for binary functions becomes a more challenging task when we additionally enforce truthfulness requirements. A natural choice of truthfulness requirement is to preserve the support size of the function. Assuming that we know the support size $|\{x \in \{0,1\}^n : f^*(x) = 1\}|$ of f^* is k, we would like our model to only generate functions that also have support size k. We show how to build an efficient learner that can output such a truthful model which is also indistinguishable from the true function f^*:

▶ **Theorem 1** (Informal statement of Theorem 20). *Let $\mathcal{B}$ be the class of sample-access objects induced by binary functions $f : \{0,1\}^n \to \{0,1\}$ satisfying $|\mathsf{supp}(f)| = k$, where $\mathsf{supp}(f) := \{x \in \{0,1\}^n : f(x) = 1\}$. Let $\mathcal{D}$ be a class of distinguishers that is efficiently learnable. There exists an efficient (ε, δ)-learner L for $\mathcal{B}$ w.r.t. $\mathcal{D}$ and the learner always outputs an efficient implementation of a model M that is truthful w.r.t. $\mathcal{B}$.*

Note that the truthfulness requirement on the support size cannot be enforced simply using computationally-bounded distinguishers, because computing the support size of a function f exactly requires reading the values $f(x)$ for all the exponentially many inputs $x \in \{0,1\}^n$. Also, this truthfulness requirement cannot be satisfied directly by a generative model specified by a predictor p, where the function value $f(x)$ is distributed according to $\mathsf{Ber}(p(x))$ independently of the function values $f(x')$ of other individuals $x' \neq x$. To enforce a fixed support size, the function values of different individuals must coordinate in a global manner, requiring us to use new techniques. We create a binary tree with leaves corresponding to the function domain $\{0,1\}^n$, and following ideas in previous work such as [9], we assign support size budgets from the root to the leaves. However, the true function f^* is a single unknown object and is very different from the uniform distribution considered in [9], so there are no closed-from distributions (such as the binomial distributions used in [9]) that can guide us to distribute the support size budget from a node two its two children. Instead, we need to *estimate* how the budget should be divided, and this leads to accumulated error towards the leaves and forces us to stop before reaching the leaves. To efficiently propagate the budgets to the leaves, we solve a zero-sum game where player C chooses the budgets for the leaves and player D distinguishes them from the target. We show that if player D uses the multiplicative weights algorithm to minimize regret, we can create an indistinguishable and truthful model from the empirical distribution over the optimal responses from player C.

Learning a Function with Support Access

In the classic setting when learning a function $f : \{0,1\} \to \{0,1\}$, the learner receives random samples of form $(x, f(x))$ for a uniform $x \in \{0,1\}$. In this work, we also consider a function object in which we receive a random *positive entry*. That is, the learner receives random x's such that $f(x) = 1$. This type of random access is natural is certain situations, for example when we have information on the individuals that graduated some program, but not on those that did not.

▶ **Theorem 2** (Informal statement of Theorem 21). *Let $\alpha > 0$, and let $f : \{0,1\} \to \{0,1\}$ be a function such that $\Pr[f(x) = 1] = \alpha$. Let $\mathcal{D}$ be a collection of distinguishers, each $D \in \mathcal{D}$ associated with a set $S_D \subseteq [N]$ and accepts x if $x \in S_D$. If there exists a weak agnostic learner for $\mathcal{D}$, then there exists a learning algorithm L running in time $\mathrm{poly}(n)$, that receives random elements from the set $\{x|f(x) = 1\}$ and outputs a model M that is indistinguishable from f to all $D \in \mathcal{D}$.*

The theorem holds when there is a weak agnostic learner for the collection of distinguishers $\mathcal{D}$ under the distribution of a random support element (i.e. random x s.t. $f(x) = 1$). In the full version [17] we show that if a collection of distinguishers $\mathcal{D}$ has a weak agnostic learner over the standard sample access distribution, and the learner is a statistical query algorithm, then there is a learner for $\mathcal{D}$ also under the distribution of random support element.

The proof of the theorem is similar to the classic boosting argument, with an additional step that the learner performs of keeping the support size of the model approximately the same as support size of f. This step is necessary because under the distribution of a random support element, the boosting algorithm is only promised to work when the support sizes of f and the model are approximately equal.

Learning an object under the distribution of random support element is potentially very useful in the case of *sparse objects*. For a sparse function f, if we choose a uniform $x \in \{0,1\}$, then $f(x) = 0$ with high probability, and a learner cannot hope to learn anything non-trivial with random samples of form $(x, f(x))$. Unfortunately, the above theorem does not hold for sparse functions, but in the next part we show how this theorem can be used to learn different sparse objects - sparse graphs.

Learning Sparse Graphs Without Dense Subgraphs

Suppose $G = ([N], E)$ is a graph represented by the N^2 length string of its adjacency matrix. In this representation, receiving a random edge from G is equivalent to receiving a random support element from the function representing its adjacency matrix. Therefore, Theorem 2 implies that we can learn a model for G that is indistinguishable for a set of distinguishers $\mathcal{D}$ that have a weak agnostic learner. The theorem only holds for functions f with a constant fraction of 1 entries, which corresponds to a dense graph. What about sparse graphs?

Learning a sparse graph, or a sparse object in general, is a very challenging task because of the huge domain. The weak regularity lemma [7] has error that is proportional to N^2, which is too much in the case of sparse graphs (an empty graph is indistinguishable from a sparse graph with this error). Therefore in the setting of a sparse graphs it is more natural to require an ε error from the distinguisher under the distribution of receiving a random edge. Under this distribution, the error of the distinguishers scales with the number of edges. We show a learner for a specific class of sparse graphs, those that have no dense subgraphs. We note that a random sparse graph has no dense subgraphs, so many graphs have this property.

▶ **Theorem 3** (Informal statement of Theorem 25). *Let $G = ([N], E)$ be a sparse graph with no dense subgraphs. Let $\mathcal{D}$ be a collection of distinguishers, each $D \in \mathcal{D}$ associated with two sets $U_D, V_D \subseteq [N]$ and accepts an edge (u, v) if $u \in U_D, v \in V_D$. If there exists a weak agnostic learner for $\mathcal{D}$, then there exists a learning algorithm L running in time $\mathrm{polylog}(N)$, that receives random edges from G and outputs a model M that is indistinguishable from f to all $D \in \mathcal{D}$.*

The model that the learner outputs is *dense*, i.e. the model outputs graphs with $\Theta(N^2)$ many edges. This is done because of technical reasons – to allow us to use rejection sampling when training the model. This brings us to the question, is there a dense graph that is

indistinguishable from our sparse graph G? The answer to this question depends on G, and in the full version [17] we show that if a sparse graph G has a very dense subgraph, then there is no dense graph that is indistinguishable from G.

The proof of the theorem has two parts, in the first part we show that for every sparse graph G with no dense subgraphs, there exists a dense graph H that is indistinguishable from G. In this part of the proof we apply the strong regularity lemma for sparse graphs [26, 33] on G, and use the resulting partition to build the dense indistinguishable graph H. This part of the proof is existential, and we do know how to find H efficiently, as the strong regularity lemma does not have an efficient algorithm for finding the partition. It is not possible to use the weak regularity lemma or its variants [7], because its error is too large. In the second part of the proof, we reduce the learning G to learning H, and show that the resulted model M is indistinguishable from G to all distinguishers $D \in \mathcal{D}$.

Other Results on Learning Generative Models

In this work we also show indistinguishable models in several other settings

- Let $f : \{0,1\}^n \to \{0,1\}^n$ be a function. Learning such function is harder than learning a binary function because the large domain makes f a sparse object (when viewed as a graph for example it is an out-degree one graph). For such functions, we show that there exists a learner that given samples from the distribution $(x, f(x))$, outputs a model that is indistinguishable against the following set of distinguishers $\mathcal{D} = \{(S_D, j_D) | S_D \subset \{0,1\}^n, j_D \in [n]\}$ such that $D = 1 \iff x \in S_D, f(x)_j = 1$. This appears on Section 3.4.
- In Section 4.1 we apply the theorems for functions on the adjacency matrix of a dense graph $G = ([N], E)$. For a set of distinguishers $\mathcal{D}$ that have a weak agnostic leaner, we have an efficient learner that outputs an indistinguishable model when G is:
  1. A dense graph when the learner receives random adjacency matrix entries.
  2. A dense graph with a fixed total number of edges $m = \Theta(N^2)$.
  3. A directed graph with a fixed out-degree $m = \Theta(N)$.
  4. A dense graph when the learner receives random edges.
- For a directed graph $G = (\{0,1\}^n, E)$ with constant out-degree d, we can treat each of the d outgoing degrees as a function $f_i : \{0,1\}^n \to \{0,1\}^n$. For the same set of distinguishers that we can handle in the case of a length-preserving functions (the first item in this list), we provide an efficient learner.
- In the case of a uniform degree *undirected graph*, we provide in Section 4.3 a learning algorithm for an indistinghuishable model, albeit for a somewhat limited set of distinguishers.

Impossibility Results

As we discussed earlier, our goal of learning a generative model is closely related to the goal in [9] of implementing huge random objects, but a key difference is that we assume the groundtruth is a single unknown object B^*, whereas [9] considers a known uniform distribution of objects. This means that we need an additional learning procedure to collect information about B^*, and we show in Section 5 that our task of efficiently learning a generative model is only possible when the distinguisher class is efficiently learnable.

Besides the requirement of learning, our setting is more challenging than the setting in [9] in many other ways. We demonstrate this by another two impossibility results on fooling entry-access distinguishers and fooling stronger distinguishers than the model.

When we consider a pseudorandom function, f_s, the function is indistinguishable from the uniform distribution to distinguishers that have entry access to the function (allowed to ask for an arbitrary string x and get $f_s(x)$). Furthermore, while f_s is computable in a fixed polynomial time, the distinguishers can run in any polynomial time (and under reasonable assumptions, even exponential time). [9] and subsequent work inherit these two properties - indistinguishability to distinguishers that are computationally more complex than the models and have entry access to the model. In Section 5 we argue that neither of these properties is achievable in our setting.

For the impossibility of fooling distinguishers with entry access, in Theorem 27 we give the example of a class $\mathcal{D}$ that contains a distinguisher D_x for every input $x \in \{0,1\}^n$ which queries the function value $f(x)$ for a function f and outputs "accept" if and only if $f(x) = 1$. We argue that every model M that is indistinguishable from the true f^* for the set of distinguishers $\mathcal{D}$ has to be very close to f^*. Since the size of f^* is exponential and f^* is unknown, no efficient learner can output a model that is close to f^*. We also show an example, using an idea from [37], of a distinguisher and a true function f^*, such that the distinguisher can tell apart f^* from any model M with a low complexity compared to the distinguisher (Theorem 28). This highlights the fact that in our setting, the generative model and the learner constructing the model have to be computationally comparable or stronger than the distinguishers.

1.3 Related Work

As mentioned in Section 1.1, [9] introduced the problem of creating an indistinguishable implementation of a random object. [9] as well as follow-up works [31, 30] also present a collection of positive results for dense and sparse graphs or functions with a variety of truthfulness conditions and access models of the distinguishers.

The connection between generative models and indistinguishability has been manifested through the invention of generative adversarial networks (GANs) [10, 2]. Intuitively, a GAN is trained to imitate a distribution of objects (say images). The generator is trained in concert with a discriminator that could be interpreted as a distinguisher. Through a sequence of rounds, the generator is trained to fool the discriminator which is then trained to fail the generator. GANs highlight the connection between generative models and indistinguishability [21], but they do not naturally fall into our framework as they are more directly described in terms of indistinguishability of two distributions.

The connection between indistinguishability and learning theory has been established in many previous works (e.g. [37] applies the boosting technique from learning theory). More recently, in the context of algorithmic fairness, the relation between learning theory and indistinguishability has been dramatically expanded in the notions of multicalibration and outcome indistinguishability [14, 5], in applications to learning and statistical inference through the notions of omnipredictors and universal adapatability [12, 24, 18, 11, 25] and in the emergence of research uncovering intricate and exciting connections while studying the sample complexity of indistinguishability from a learning-theoretic perspective [20, 19].

It is possible to view our learning setting as a 2-players zero-sum game, between the learner and the distinguishers, in which the learner's goal is to output a model for an indistinguishable object and the distinguishers try to tell apart the input and the model. In this setting, there is a relation between min-max theorems and regularity-lemma theorems. Such theorems prove that it is possible to express a complex object f by a function of a few simpler objects $g_1, \ldots g_t$ that, in our setting, represent the distinguishers [37, 38]. There have been works improving the parameters and also using such theorems for applications in cryptography [38, 22, 34, 35, 36, 4]. In this work, our setting is slightly different, as we assume that the object f is complex and unknown, and the learner has to learn it. The proof

of Theorem 3 has an intermediate step that is existential and has a similar structure to a weak regularity lemma theorems, but since the required error there is too small, we derive it from the sparse strong regularity lemma.

▶ Note 4. This is an abridged version of the paper. We refer the readers to the full version [17] for proofs and other contents that are omitted in this version.

2 Preliminaries

Throughout the paper, we are interested in learning objects such as functions and graphs, and we are particularly interested when these objects have exponential sizes (e.g. functions with exponentially large domains and graphs with exponentially many vertices and edges). We typically use B to denote an object, and use $\mathcal{B}$ to denote a class of objects. We view an object B as a function $B : Q \to \Delta_A$ that maps a query $q \in Q$ to a distribution $B(q)$ over answers in A.

2.1 Functions

When the object is a function $f : X \to Y$, we consider three access types. For sample access, B returns a random pair $(x, f(x))$. For support access, it returns a random x such that $f(x) = 1$. For entry access, upon querying x, B returns $f(x)$.

▶ **Definition 5** (Function-induced sample-access object). *Let $f : X \to Y$ be a function and let $B : Q \to \Delta_A$ be an object. We say B is the* sample-access *object* induced *by f if $Q = \{\perp\}, A = X \times Y$, and $B(\perp)$ is the distribution of $(x, f(x)) \in A$ where x is drawn uniformly from X.*

▶ **Definition 6** (Function-induced support-access object). *Let $f : X \to \{0, 1\}$ be a binary function. We define the support of f to be $\mathsf{supp}(f) := \{x \in X : f(x) = 1\}$. Let $B : Q \to \Delta_A$ be an object. Assuming $\mathsf{supp}(f) \neq \emptyset$, we say B is the* support-access *object induced by f if $Q = \{\perp\}, A = X$, and $B(\perp)$ is the uniform distribution over $\mathsf{supp}(f) \subseteq X$.*

▶ **Definition 7** (Function-induced entry-access object). *Let $f : X \to Y$ be a function and let $B : Q \to \Delta_A$ be an object. We say B is the* entry-access *object* induced *by f if $Q = X, A = Y$, and for every $q \in Q$, $B(q)$ is the singleton distribution such that $a \sim B(q)$ equals to $f(q)$ deterministically.*

In this paper, we show positive results for learning generative models of functions with sample access and support access (Section 3) whereas we show impossibility results for entry access (Section 5). This separation is mainly because entry access makes the distinguishers stronger and thus makes indistinguishability harder to achieve (see Definitions 10 and 11 below).

2.2 Graphs

For a graph $G = (V, E)$ where we assume V has exponential size, we define two access types, sample-access which corresponds to a random adjacency matrix entry, and support-access which corresponds to a random edge in the graph.

▶ **Definition 8** (Graph-induced sample-access object). *Let $G = (V, E)$ be a directed or undirected graph and let $B : Q \to \Delta_A$ be an object. We say B is the* sample-access *object induced by G if $Q = \{\perp\}, A = V \times V \times \{0, 1\}$, and $B(\perp)$ is the distribution of $(u, v, y) \in A$ where (u, v) is drawn uniformly from $V \times V$, $y = 1$ if $(u, v) \in E$ and $y = 0$ otherwise.*

▶ **Definition 9** (Graph-induced support-access object). *Let $G = (V, E)$ be a directed or undirected graph and let $B : Q \to \Delta_A$ be an object. Assuming $E \neq \emptyset$, we say B is the* support-access *object induced by G if $Q = \{\perp\}, A = V \times V$, and $B(\perp)$ is the uniform distribution over $E \subseteq V \times V$.*

2.3 Indistinguishability

Each learner we design in this paper has access to a ground-truth object B^*, and it aims to output an object B that is *indistinguishable* from B^*. In many cases, the learner does not just output a single object B, but a distribution over objects, and we refer to such distributions as *models*. Below we formally define the notion of indistinguishability.

▶ **Definition 10** (Distinguisher). *A* distinguisher *D is an algorithm that when given access to an object $B : Q \to \Delta_A$, outputs* "accept" *or* "reject"*. That is, the distinguisher is allowed to make queries $q \in Q$ to the model, and for each query q the distinguisher receives an answer $a \in A$ drawn independently from $B(q) \in \Delta_A$. We allow the distinguisher D itself to be randomized, and we use random variable D^B to denote the output of the distinguisher D in* {"accept", "reject"} *when given access to B.*

▶ **Definition 11** (Indistinguishability). *Let $B^* : Q \to \Delta_A$ be an object, and let model M be a distribution over objects $B : Q \to \Delta_A$. We say model M is* ε-indistinguishable *from object B^* w.r.t. a distinguisher D if*

$$|\Pr[D^{B^*} = \text{“accept”}] - \mathbb{E}_{B\sim M}[\Pr[D^B = \text{“accept”}]]| \leq \varepsilon. \tag{1}$$

We say model M is ε-indistinguishable from object B^* w.r.t. a class $\mathcal{D}$ of distinguishers if (1) *holds for every $D \in \mathcal{D}$.*

2.4 Truthfulness

In addition to indistinguishability, another desirable property of a model is *truthfulness* introduced in [9]. Truthfulness requires every object generated from the model to satisfy a certain (usually global) property which we formalize using an object class $\mathcal{B}$:

▶ **Definition 12** (Truthfulness). *We say a model M is* truthful *w.r.t. an object class $\mathcal{B}$ if*

$$\Pr_{B\sim M}[B \in \mathcal{B}] = 1.$$

2.5 Implementations

Our goal is to design efficient learners, and thus we cannot expect the learner to output a model M explicitly, especially when the objects drawn from M are huge. Instead, our learner outputs an efficient *implementation* of a model, defined as follows.

▶ **Definition 13** (Ordinary Implementation). *For $\ell \in \mathbb{Z}_{\geq 0}$, let T be a randomized algorithm that takes $(r, q) \in \{0,1\}^\ell \times Q$ as input, and outputs $T(r, q) \in A$. We say T is an* ordinary implementation *of a model M with* seed length *ℓ if for every* seed *$r \in \{0,1\}^\ell$ there exists an object $B_r : Q \to \Delta_A$ such that*

1. *for every $q \in Q$, $T(r, q)$ is distributed according to $B_r(q)$, where the randomness in $T(r, q)$ comes from the internal randomness in algorithm T;*
2. *B_r is distributed according to M when r is drawn uniformly from $\{0,1\}^\ell$.*

While our goal is to output an ordinary implementation with a polynomial-length seed, following the approach in [9], it is more convenient to first build implementations using a random oracle and then transform the implementation to an ordinary one using Lemma 15.

▶ **Definition 14** (Random-Oracle Implementation). *Let T be a randomized algorithm that takes a function $r : \{0,1\}^* \to \{0,1\}$ as an oracle. On an input $q \in Q$, the algorithm T outputs $T^r(q) \in A$. We say T is a* random-oracle implementation *of a model M if for every $r : \{0,1\}^* \to \{0,1\}$ there exists an object $B_r : Q \to \Delta_A$ such that*

1. *for every $q \in Q$, $T^r(q)$ is distributed according to $B_r(q)$, where the randomness in $T^r(q)$ comes from the internal randomness in algorithm T;*
2. *B_r is distributed according to M when r is a uniformly random function from $\{0,1\}^*$ to $\{0,1\}$.*

▶ **Lemma 15** (Theorem 2.9 in [9]). *Suppose that one-way functions exist. There exists an algorithm H with the following properties. Let $\mathcal{D}$ be a class of distinguishers where each $D \in \mathcal{D}$ is a circuit of size at most W for some $W \geq 1$. Let T be a random-oracle implementation of a model M with circuit complexity at most W. Given W, T and an arbitrary $\varepsilon \in (0,1)$ as input, the algorithm H runs in time $\mathrm{poly}(W, 1/\varepsilon)$ and outputs an ordinary implementation T' of a model M' where T' has seed length and circuit complexity both being $\mathrm{poly}(W, 1/\varepsilon)$, and*

$$|\mathbb{E}_{B' \sim M'} \Pr[D^{B'} = \text{“accept”}] - \mathbb{E}_{B \sim M} \Pr[D^{B} = \text{“accept”}]| \leq \varepsilon \quad \text{for every } D \in \mathcal{D}.$$

Moreover, if M is truthful w.r.t. an object class $\mathcal{B}$, then M' is also truthful w.r.t. $\mathcal{B}$.

Lemma 15 can be proved by using a pseudorandom function to emulate the random oracle.

2.6 Learning

We describe the learners we aim to design in the definition below.

▶ **Definition 16** (Learner). *Let $\mathcal{B}$ be a class of objects $B : Q \to \Delta_A$ and $\mathcal{D}$ be a class of distinguishers. An (ε, δ)-*learner *L for the class $\mathcal{B}$ w.r.t. $\mathcal{D}$ is an algorithm with the following properties. For any $B^* \in \mathcal{B}$, given access to B^*, the learner outputs an implementation T of a model M such that with probability at least $1 - \delta$, M is ε-indistinguishable from B^* w.r.t. $\mathcal{D}$.*

2.7 Other Notations

For $v \in \mathbb{R}$, we define $\mathsf{cap}(v)$ by capping its value into $[0,1]$, i.e.,

$$\mathsf{cap}(v) = \begin{cases} v, & \text{if } 0 \leq v \leq 1; \\ 1, & \text{if } v > 1; \\ 0, & \text{if } v < 0. \end{cases}$$

Given a list of values $(v_1, \ldots, v_t)$ we define $\mathsf{Lcap}(v_1, \ldots, v_t)$ by summing over the list and capping the value to $[0,1]$ in every iteration. We formally define it recursively:

$$\begin{aligned} \mathsf{Lcap}(v_1) &= \mathsf{cap}(v_1), \\ \mathsf{Lcap}(v_1, \ldots, v_t) &= \mathsf{cap}(\mathsf{Lcap}(v_1, \ldots, v_{t-1}) + v_t). \end{aligned} \tag{2}$$

3 Learning Functions with Exponentially Large Domains

The goal of this section is to efficiently learn a generative model that is indistinguishable from a target function $f^* : X \to Y$ to a class $\mathcal{D}$ of distinguishers. We allow the domain X of the function to have exponential size $N := |X|$, and require our learner to run in time $\text{polylog}(N)$. This means that the learner cannot read the entire function f^*, and can only access it via random sample. Throughout the paper, our learners output an efficient random-oracle implementation T of a model M, which can be turned in to an efficient ordinary implementation by Lemma 15.

3.1 Learning Sample-Access Binary Functions

We start by studying the case where the target object B^* is the sample-access object induced by a binary function $f^* : X \to \{0,1\}$ (Definition 5). We assume that every distinguisher $D \in \mathcal{D}$ satisfies the following: when given access to a sample-access object B induced by a function $f : X \to \{0,1\}$, the distinguisher asks a single query $\perp$, receives an answer $a = (x, y) \sim B(\perp)$, and outputs $D(x, y) \in \{\text{“accept”}, \text{“reject”}\}$. We allow the distinguisher itself to be randomized, and each distinguisher D defines a function $g_D : X \to [-1, 1]$ such that

$$g_D(x) = \Pr[D(x,1) = \text{“accept”}] - \Pr[D(x,0) = \text{“accept”}] \quad \text{for every } x \in X.$$

We use the following claim to relate a distinguisher $D \in \mathcal{D}$ to the function $g_D : X \to [-1, 1]$:

▷ Claim 17. For every distinguisher $D \in \mathcal{D}$, the following equation holds for every $x \in X$ and $y_1, y_2 \in \{0,1\}$:

$$\Pr[D(x,y_1) = \text{“accept”}] - \Pr[D(x,y_2) = \text{“accept”}] = y_1 g_D(x) - y_2 g_D(x).$$

The claim can be easily proved by considering the four possible choices of $(y_1, y_2) \in \{0,1\} \times \{0,1\}$.

As we show later in Section 5, it is necessary to impose certain learnability assumptions on the distinguishers. To that end, we assume that there is an *auditor* for the function class $\mathcal{G} := \{g_D : D \in \mathcal{D}\}$, defined as follows:

▶ **Definition 18** (Auditor). *Let $\mathcal{D}$ and $\mathcal{G}$ be defined as above. We say an algorithm Λ is an $(\varepsilon, \gamma, \delta)$-auditor for $\mathcal{G}$ if it satisfies the following property. Given access to a sample-access object B^* induced by a function $f^* : X \to \{0,1\}$ and taking a predictor $p : X \to [0,1]$ as an oracle, if there exists $g \in \mathcal{G}$ such that*

$$|\,\mathbb{E}[f^*(x)g(x)] - \mathbb{E}[p(x)g(x)]| \geq \varepsilon,$$

then Λ outputs $\hat{g} : X \to [-1,1]$ satisfying the following with probability at least $1 - \delta$:

$$\mathbb{E}[f^*(x)\hat{g}(x)] - \mathbb{E}[p(x)\hat{g}(x)] \geq \gamma.$$

The auditor defined above can be viewed as a weak agnostic learner for the class $\mathcal{G}$. When the domain X is $\{0,1\}^n$ with size $N = 2^n$, many classes allow efficient auditors that run in time $\text{poly}(n) = \text{polylog}(N)$. Using an auditor for $\mathcal{G}$, we prove the following theorem:

▶ **Theorem 19.** *Let the distinguisher class $\mathcal{D}$ and the function class $\mathcal{G}$ be defined above. Let $\varepsilon, \gamma, \delta, \delta' \in (0, 1/2)$ be parameters satisfying $\gamma \leq \varepsilon$ and $\delta' \leq c\delta\gamma^2$ for a sufficiently small absolute constant $c > 0$. Let $\mathcal{B}$ be the class of sample-access objects induced by binary functions $f : X \to \{0,1\}$. Let Λ be an $(\varepsilon, \gamma, \delta')$-auditor for $\mathcal{G}$ (Definition 18). Then there*

exists an (ε, δ)-learner L for $\mathcal{B}$ w.r.t. $\mathcal{D}$. Moreover, if the auditor Λ runs in time at most W_1 and always outputs a function with circuit size at most W_2, then the learner L runs in time $\mathrm{poly}(\gamma^{-1}, \log(\delta^{-1}), W_1)$ *and always outputs implementations with circuit complexity* $\tilde{O}(\gamma^{-2} W_2)$.

We prove the theorem by applying results from algorithmic fairness [37, 14, 23], see the full version for more details.

3.2 Truthful Learning That Preserves Support Size

Some desirable properties of a generative model are global and cannot be enforced using computationally bounded distinguishers alone. This motivated [9] to introduce the notion of *truthfulness* that ensures such global properties beyond indistinguishability. Here we focus on a natural global property of a binary function $f : X \to \{0,1\}$: the size of its support $\mathsf{supp}(f) := \{x \in X : f(x) = 1\}$. The model M we create in Section 3.1 using the multiaccurate predictor p may generate functions f with support size different from the target function f^*. Indeed, even if $\sum_{x \in X} p(x) = |\mathsf{supp}(f^*)|$, a random function f with each entry $f(x)$ independently drawn from $\mathsf{Ber}(p(x))$ is not guaranteed to satisfy $|\mathsf{supp}(f)| = |\mathsf{supp}(f^*)|$. Now we show an efficient learner that outputs truthful models that preserve the support size of the generated functions.

An overview of the proof appears in the introduction, and the proof appears in the full version [17].

▶ **Theorem 20.** *Let the distinguisher class $\mathcal{D}$ and the function class $\mathcal{G}$ be defined as in Section 3.1. Let $\varepsilon, \gamma, \delta, \delta' \in (0, 1/2)$ be parameters satisfying $\gamma \leq \varepsilon$ and $\delta' \leq c\delta\gamma^2$ for a sufficiently small absolute constant $c > 0$. Let $\mathcal{B}$ be the class of sample-access objects induced by binary functions $f : X \to \{0,1\}$ satisfying $|\mathsf{supp}(f)| = k$, where $X = \{0,1\}^n$. Let Λ be an $(\varepsilon, \gamma, \delta')$-auditor for $\mathcal{G}$ (Definition 18). Then there exists an (ε, δ)-learner L for $\mathcal{B}$ w.r.t. $\mathcal{D}$ and the learner always outputs a random-oracle implementation of a model M that is truthful w.r.t. $\mathcal{B}$. Moreover, if the auditor Λ runs in time at most W_1 and always outputs a function with circuit size at most W_2, then the learner L runs in time* $\mathrm{poly}(n, \gamma^{-1}, \log(\delta^{-1}), W_1)$ *and always outputs an implementation with circuit complexity* $\mathrm{poly}(n, \gamma^{-1}, W_2)$.

3.3 Learning Support-Access Binary Functions

In the previous sections we showed how to learn and construct an implementation of an indistinguishable model for a function induced sample-access object B^*, i.e. there exists a function $f^* : X \to \{0,1\}$, and the distinguishers and learner both receives random samples of form $(x, f^*(x))$. In this section, we learn a support-access object induced by a function see Definition 6. For a binary function $f^* : X \to \{0,1\}$, the support-access object B^* induced by f^* outputs random samples from the set $\{x | f^*(x) = 1\}$. We show how to construct an efficient implementation of a model that is indistinguishable from B^*.

Distinguishers: Let $\mathcal{D}$ be a set of distinguishers, such that each $D \in \mathcal{D}$ has an associated subset $S_D \subset X$. Given access to a sample x from a object B,the distinguisher accepts if $x \in S_D$. We assume that for every $D \in \mathcal{D}$, the set S_D has known size and an efficient description, that on input x answers if $x \in S_D$.

Auditor: Let $\mathcal{D}$ be defined as above. We say that an algorithm $\Lambda^{B^*,p}$ is an $(\varepsilon, \gamma, \delta)$ auditor for the collection of sets $\mathcal{S} = \{S_D | D \in \mathcal{D}\}$ if it has the following properties. Given access to a function-induced support access-object B^* and query access to a predictor $p : X \to [0,1]$. If there exists $S \in \mathcal{S}$ and $b \in \{-1, 1\}$ such that

$b\left(\Pr_{x\sim B^*(\perp)}[x \in S] - \Pr_{x\sim p}[x \in S]\right) > \varepsilon$. Then the auditor returns a set S' such that with probability $1-\delta$, $b\left(\Pr_{x\sim B^*(\perp)}[x \in S'] - \Pr_{x\sim p}[x \in S']\right) > \gamma$. Where $x \sim p$ is the distribution generated from the predictor p, i.e. $\Pr_{x\sim p}[x = x'] = p(x')/\sum_{x''\in X} p(x'')$.

▶ **Theorem 21.** *Let $\alpha \in [0,1]$ be a parameter, and let $\mathcal{B}$ be a collection of support access object induced by binary functions, such that $\forall B \in \mathcal{B}, \mathbb{E}_{x\in X}[f_B(x)] = \alpha$. Let $\mathcal{D}$ be a collection of distinguishers as described above.*

Let $\varepsilon, \gamma, \delta', \delta''$ be parameters such that $\delta' \leq c\delta\gamma^2\alpha^{-2}$ for a sufficiently small constant c. Let Λ be an $(\varepsilon, \gamma, \delta')$ auditor Λ for $\mathcal{D}$. Then there exists a $(2\varepsilon, \delta)$-learner L for $\mathcal{B}$ with respect to the distinguisher class $\mathcal{D}$. The learner L runs in time $\mathrm{poly}(\gamma^{-1}\log(\delta^{-1})\alpha^{-1}, W_1, W_2)$, where W_1 is the running time of the auditor Λ and W_2 the circuit complexity of its output. The implementation T that the learner outputs runs in time $\mathrm{poly}(\gamma^{-1}\log(\delta^{-1})\alpha^{-1}, W_2)$

The learner L in the theorem above has access to an auditor Λ that can audit support-access objects. In the full version [17] we discuss under which conditions such auditor exists. The learning algorithm is similar to the classic boosting algorithm, with an additional step of keeping the expected value of the model to be approximately α. We note that if the function is sparse, i.e. α is very small, then the algorithm is no longer efficient.

3.4 Learning Bit-String Functions

In this section we are interested in learning a function $f : \{0,1\}^n \to \{0,1\}^n$. This is a harder than learning a binary function, because the range of the function is very large. Therefor we only learn an indistinguishable model with respect to a very limited set of distinguishers, with a product structure. In this setting, the sampling distribution is a pair $(x, f(x))$ for a random input x.

Distinguishers: Let $\mathcal{D}$, such that each distinguisher $D \in \mathcal{D}$ has an set $S_D \subset \{0,1\}^n$ and a coordinate $j \in [n]$. The distinguisher D accept a sample $(x, f(x))$ if $x \in S$ and $f(x)_j = 1$.

Auditor: Let $\mathcal{D}$ be defined as above. We say that an algorithm $\Lambda^{B^*,p}$ is an $(\varepsilon, \gamma, \delta)$ auditor for the collection of sets $\mathcal{S}$ if it has the following properties. Given access to a function-induced support access-object B^* and query access to a set of n predictors $p_1, \ldots p_n$, such that $p_j : \{0,1\}^n \to [0,1]$. If there exists $S \in \mathcal{S}$ and $j \in [n]$ such that $b\left(\Pr_x[x \in S, f(x)_j = 1] - \mathbb{E}_x[p_j(x)\cdot \mathbf{1}(x \in S)]\right) > \varepsilon$. Then the auditor returns a set $S' \subseteq \{0,1\}^n$ and $j \in [n]$ such that $b\left(\Pr_x[x \in S', f(x)_j = 1] - \mathbb{E}_x[p_j(x)\cdot \mathbf{1}(x \in S')]\right) > \gamma$ with probability $1-\delta$.

▶ **Theorem 22.** *Let $\mathcal{B}$ be a collection of support access object induced by functions $f : \{0,1\}^n \to \{0,1\}^n$. Let $\mathcal{D}$ be a collection of distinguishers as described above.*

Let $\varepsilon, \gamma, \delta', \delta''$ be parameters such that $\delta' \leq c\delta\gamma^2 n^{-1}$ for a sufficiently small constant c. Let Λ be an $(\varepsilon, \gamma, \delta')$ auditor for $\mathcal{D}$. Then there exists a $(2\varepsilon, \delta)$-learner L to $\mathcal{B}$ with respect to the distinguisher class $\mathcal{D}$ and the learner L runs in time $\mathrm{poly}(\gamma^{-1}\log(\delta^{-1})\alpha^{-1}, W_1, W_2)$, where W_1 is the running time of the auditor Λ and W_2 the circuit complexity of its output. The implementation T that the learner outputs runs in time $\mathrm{poly}(\gamma^{-1}\log(\delta^{-1})\alpha^{-1}, W_2)$

4 Learning Exponential-Size Graphs

4.1 Learning Dense Graphs

The most basic setting for graphs is the dense model, where the graph-induced sample-access object B induced by a graph $G = ([N], E)$ can be thought of as getting a random entry $((u,v), b)$ for $u, v \in [N], b \in \{0,1\}$ from the adjacency matrix of G (see Definition 8). We can

think of the adjacency matrix of the graph as a function, where the graph imposes some extra structure on the distinguishers. Therefore, some of the results from Section 3 follow directly.

Distinguishers: Let $\mathcal{D}$ be a collection of distinguishers. Each distinguisher has two sets of vertices $U_D, V_D \subset [N]$. When getting a random entry $((u,v),b)$ from the graph-induced sample-access object B^*, it accepts if $u \in U_D$, $v \in V_D$ and $b = 1$.

Auditors: An algorithm Λ is an $(\varepsilon,\gamma,\delta)$-auditor for a collection of tuples $\mathcal{S}$ containing pairs $(U,V) \subset [N] \times [N]$ if it satisfies the following. The auditor received query access to a predictor $p : [N] \times [N] \to [0,1]$ and to a graph-induced sample-access object B^* induced by a graph $G^* = ([N], E^*)$. If there exists $(U,V) \in \mathcal{S}$ such that

$$b \cdot \left(\Pr_{(u,v)\in[N]\times[N]}[u \in U, v \in V, (u,v) \in E^*] - \mathbb{E}_{(u,v)\in[N]\times[N]}[\mathbf{1}(u \in U, v \in V)p(u,v)] \right) \geq \varepsilon.$$

Then with probability $(1-\delta)$, it outputs U', V' such that

$$b \cdot \left(\Pr_{(u,v)\in[N]\times[N]}[u \in U', v \in V', (u,v) \in E^*] - \mathbb{E}_{(u,v)\in[N]\times[N]}[\mathbf{1}(u \in U', v \in V')p(u,v)] \right) \geq \gamma.$$

The setting of a dense graph can be derived directly from the function theorem, by applying it on the adjacency matrix of the graph.

▶ **Corollary 23** (Corollary of Theorem 19). *Let the distinguisher class $\mathcal{D}$ be defined above. Let $\varepsilon, \gamma, \delta, \delta' > 0$ be parameters satisfying $\delta' \leq c\delta\gamma^2$ for a sufficiently small absolute constant $c > 0$. Let $\mathcal{B}$ be the class of sample-access objects induced by graphs over vertex set $[N]$. Let Λ be an $(\varepsilon,\gamma,\delta')$-auditor for $\mathcal{S} = \{(U_D, V_D) | D \in \mathcal{D}\}$. Then there exists an (ε,δ)-learner L for $\mathcal{B}$ w.r.t. $\mathcal{D}$. Moreover, if the auditor Λ runs in time at most W_1 and always outputs a function with circuit size at most W_2, then the learner L runs in time $\mathrm{poly}(\gamma^{-1}, \log(\delta^{-1}), W_1)$ and always outputs implementations with circuit complexity $O(\gamma^{-2}W_2)$.*

The same holds also for learning a graph with a fixed number of edges, by applying Theorem 20. Similarly, we can derive a theorem on directed graphs by applying Theorem 19 or Section 3.2 on the directed graph adjacency matrix. It is also possible to generate a directed graph with a fixed out-degree by applying Section 3.2 for every vertex individually.

4.2 Learning Sparse Graphs Without Dense Subgraphs

For a sparse graph, a sample-access graph object is not useful, because a random entry in the adjacency matrix of the graph is nearly always 0. We study graph-induced support-access objects (Definition 9), which corresponds to an object B^* induced by a sparse graph $G = ([N], E)$, $B(\perp)$ that outputs a random edge in the graph $(u,v) \in E$.

Applying Theorem 21 implies a corollary for support-access graphs, but the theorem is only efficient for dense graphs. In this section we show how to create *a dense model for a sparse graph*, as long as the sparse graph does not have a subgraph which is too dense. We do so by using the strong regularity lemma for sparse graphs [26, 33].

Distinguishers: Let $\mathcal{D}$ be a collection of distinguishers. Each distinguisher has two sets of vertices $U_D, V_D \subset [N]$. A distinguisher D on input (u,v) accepts if $u \in U_D$ and $v \in V_D$.

Auditors: An algorithm Λ is an $(\varepsilon,\varepsilon',\delta)$-auditor for a collection pairs of sets $\mathcal{S}$, such that $(U,V) \in \mathcal{S}$, $U, V \subset [N]$ if it satisfies the following. The auditor received query access to a predictor $p : [N] \times [N] \to [0,1]$ and access to a graph-induced support-access object

B^* representing a graph $G^* = ([N], E^*)$. If there exists a pair of sets $(U, V) \subset \mathcal{S}$ and a bit b such that $b \cdot \left(\Pr_{(u,v)\sim B^*(\perp)}[u \in U, v \in V] - \Pr_{(u,v)\sim p}[u \in U, v \in V]\right) \geq \varepsilon$, Then Λ outputs sets $U', V' \subset [N]$ such that

$$b \cdot \left(\Pr_{(u,v)\sim B^*(\perp)}[u \in U', v \in V'] - \Pr_{(u,v)\sim p}[u \in U', v \in V'] \right) \geq \varepsilon'.$$

The distribution $(u, v) \sim p$ is defined by the predictor p, i.e. for all $u, v \in [N]$ we have $\Pr_{(u',v')\sim p}[u' = u, v' = v] = p(u, v)/\sum_{u'',v''\in[N]} p(u'', v'')$.

Graph Notations and Definitions

For a graph $G = ([N], E)$ and $U, V \subset [N]$, we define $E_G(U, V) = \{(u, v) \in E | u \in U, v \in V\}$ to be the set of edges between U, V in G. We denote by $\rho_G(U, V)$ the edge density between U, V in G, $\rho_G(U, V) = \frac{|E_G(U,V)|}{|U||V|}$. We denote by $\rho_G = \rho_G([N], [N])$ the edge density of the graph. We use the definition of upper-uniform graphs from [26, 33] with a small additional requirement also for small sets.

▶ **Definition 24** (Upper-uniform graphs). *A graph $G = ([N], E)$ is (η, γ)-upper uniform, if for every two disjoint sets $U, V \subset [N]$, with $\min\{|U|, |V|\} \geq \eta N$ we have that $\rho_G(U, V) \leq \gamma\rho_G$, and for U, V such that $\min\{|U|, |V|\} < \eta N$, we have that $|E(U, V)| \leq \gamma\eta\rho_G N^2$.*

We remark that a random sparse graph is upper-uniform for constants η, γ with high probability.

▶ **Theorem 25.** *For every parameter $\gamma, \varepsilon, \varepsilon', \lambda'\lambda'' > 0$, such that $\lambda' \leq c\lambda\varepsilon'^2$ for a sufficiently large constant c. Then there exists $\eta \in [0, 1]$ such that the following holds. Let $\mathcal{B}$ be a collection of graph-induced support access objects, such that for each $B^* \in \mathcal{B}$, the graph it represents G_{B^*} is (η, γ)-upper-uniform.*

Let $\mathcal{D}$ be a collection of distinguishers. If there exists an $(\varepsilon, \varepsilon', \delta')$-auditor Λ for the collection of sets $\mathcal{C} = \{(U_D, V_D) | D \in \mathcal{D}\}$, then there exists an (ε, δ'')-learning algorithm L for all $B^* \in \mathcal{B}$ with respect to $\mathcal{D}$.

The proof of the theorem appears in the full version [17]. In addition, we show how this theorem can be combined with Theorem 20 to create a sparse uniform out-degree graph.

4.3 Learning Uniform Degree Graphs

Suppose we are interested in generating a truthful model for a uniform degree d graph. That is, we want that all graph in our model has a uniform degree d. In previous sections we discussed directed graphs with uniform out-degree. For undirected graphs, in [9] the authors show a construction of a graph indistinguishable from random, by applying a random permutation on a large girth expander. In this work we restrict the set of distinguishers to those that can be described by a partition, and create a model by learning the densities of the edges between each part in the partition and permuting the edges.

Distinguishers: Then the set of distinguishers $\mathcal{D}$ contains distinguishers D with sets (U_D, V_D) such that $U, V \in \mathcal{U}$. Every distinguisher D accepts an edge (u, v) if $u \in U, v \in V$. Let $\mathcal{U} = \{U \subset [N] | \exists D \text{ s.t. } U = U_D \text{ or } U = V_D\}$. We assume that $\mathcal{U}$ is a partition with t parts, and that $|U_J|$ is linear in N.

▶ **Lemma 26.** *Let $\mathcal{B}$ be a collection of graph-induced support-access objects, such that for all $B^* \in \mathcal{B}$, the graph G_{B^*} has a uniform degree d. Let $\mathcal{D}$ be the distinguishers class defined above. Then for every constant ε there exists an $(\varepsilon, \delta)-$ learning algorithm L for the class $\mathcal{B}$ with respect to $\mathcal{D}$. The algorithm runs in time* $\text{poly}(1/\varepsilon, \log(1/\delta))$.

5 Impossibilities

A main difference in our work from [9] is in the target distribution/object we aim to be indistinguishable from. In [9], the target distribution is fixed and uniform over many objects, whereas in our setup the target is a single object which is initially unknown, and a learner is needed to access the target object to make it possible to create an indistinguishable model. This difference makes our setup challenging, and below we show example tasks that are impossible to achieve in our setup because of this difference.

5.1 Fooling Distinguishers with Entry-Access is Hard

In [9], the distinguishers can query for specific entries of an object. Such distinguishers can be impossible to fool in our setup. For example, suppose the target object B^* is the *entry-access* object induced by a function $f^* : X \to \{0,1\}$ (Definition 7), and suppose our learner aims to output a model M of entry-access objects B induced by functions $f : X \to \{0,1\}$. For every $x \in X$, suppose there is a distinguisher that queries for the value of $f(x)$ and outputs "accept" if and only if $f(x) = 1$. To fool these distinguishers, we have to learn the target function f^* exactly, which is clearly impossible if the domain X has exponential size and the learner can only make polynomially many queries.

▶ **Theorem 27.** *Let X be a non-empty finite set. Let $\mathcal{B}$ be the class of entry-access objects induced by all functions $f : X \to \{0,1\}$. Let $\mathcal{D}$ be the class of distinguishers D_x for every $x \in X$ where given an object B, the distinguisher D_x outputs* "accept" *if and only if the answer $a \sim B(x)$ is equal to 1. Let L be an (ε, δ)-learner for the class $\mathcal{B}$ w.r.t. $\mathcal{D}$ for $\varepsilon, \delta < 1/2$. Then L needs to query every input $x \in X$ in the worst case.*

5.2 Learned Model Needs to be Stronger than Distinguishers

The model learned in [9] can fool distinguishers with significantly larger circuit complexity than the model itself. Below we show that this can become impossible in our setup where the target is a single object.

▶ **Theorem 28** (Remark 1.6 in [37])**.** *Let $n, W > 1$ be positive integers satisfying $W \log W \leq 2^n/C$ for a sufficiently large absolute constant $C > 0$. There exists a sample-access object B^* induced by a function $f^* : \{0,1\}^n \to \{0,1\}$ and a distinguisher D with circuit complexity $\tilde{O}(nW)$ such that for any model M with circuit complexity at most W, it holds that*

$$|\Pr[D^{B^*} = \text{"accept"}] - \mathbb{E}_{B\sim M}[\Pr[D^B = \text{"accept"}]]| > 1/3. \tag{3}$$

5.3 The Distinguisher Class Needs to be Learnable

Since the target distribution in [9] is fixed, no learning is needed in order to produce an indistinguishable model. In our setup, the learning task is usually performed using an auditor, which can be viewed as a weak agnostic learner for the class of distinguishers. A natural question is whether we can still achieve indistinguishability if such a weak agnostic learner does not exist. Previous works [14, 11] have shown negative answers to this question for certain notions of indistinguishability (such as calibrated multiaccuracy) by showing that these notions imply (strong) agnostic learning for the distinguisher class. The indistinguishability notion we use for generative models is closer to multiaccuracy, and below we show that efficiently achieving this notion requires the distinguisher class to be efficiently realizably learnable. For a true function $f^* : X \to \{0,1\}$, multiaccuracy requires a predictor $p : X \to [0,1]$ to satisfy

$$|\,\mathbb{E}[(f^*(x) - p(x))g(x)]| \leq \varepsilon \tag{4}$$

for every function g in a class $\mathcal{G}$. Now consider the case where $\mathcal{G}$ consists of functions $g : X \to \{-1, 1\}$. For an arbitrary $g^* \in \mathcal{G}$, suppose the true function f^* satisfies $f^*(x) = 1$ if $g^*(x) = 1$ and $f^*(x) = 0$ if $g^*(x) = -1$. Then (4) implies

$$\mathbb{E}\,|f^*(x) - p(x)| \leq \varepsilon. \tag{5}$$

Now we define $\hat{g}(x) = 1$ if $p(x) \geq 1/2$, and define $\hat{g}(x) = -1$ if $p(x) < 1/2$. It is easy to check that if $\hat{g}(x) \neq g^*(x)$ for some $x \in X$, then $|f^*(x) - p(x)| \geq 1/2$, and thus (5) implies the following realizable learning guarantee for the class $\mathcal{G}$:

$$\Pr[\hat{g}(x) \neq g^*(x)] \leq 2\varepsilon.$$

References

1 Edoardo M Airoldi, David M Blei, Stephen E Fienberg, and Eric P Xing. Mixed membership stochastic blockmodels. *Journal of machine learning research: JMLR*, 9:1981–2014, 2008.

2 Sanjeev Arora, Rong Ge, Yingyu Liang, Tengyu Ma, and Yi Zhang. Generalization and equilibrium in generative adversarial nets (GANs). In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 224–232. PMLR, 06–11 August 2017. URL: https://proceedings.mlr.press/v70/arora17a.html.

3 Manuel Blum and Silvio Micali. How to generate cryptographically strong sequences of pseudorandom bits. *SIAM Journal on Computing*, 13(4):850–864, 1984. doi:10.1137/0213053.

4 Yi-Hsiu Chen, Kai-Min Chung, and Jyun-Jie Liao. On the complexity of simulating auxiliary input. In *Advances in cryptology—EUROCRYPT 2018. Part III*, volume 10822 of *Lecture Notes in Comput. Sci.*, pages 371–390. Springer, Cham, 2018. doi:10.1007/978-3-319-78372-7_12.

5 Cynthia Dwork, Michael P Kim, Omer Reingold, Guy N Rothblum, and Gal Yona. Outcome indistinguishability. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 1095–1108, 2021.

6 Cynthia Dwork, Daniel Lee, Huijia Lin, and Pranay Tankala. New insights into multicalibration. *arXiv preprint*, 2023. arXiv:2301.08837.

7 Alan Frieze and Ravi Kannan. Quick approximation to matrices and applications. *Combinatorica*, 19(2):175–220, 1999.

8 Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM*, 33(4):792–807, October 1986.

9 Oded Goldreich, Shafi Goldwasser, and Asaf Nussboim. On the implementation of huge random objects. *SIAM Journal on Computing*, 39(7):2761–2822, 2010.

10 Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL: https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf.

11 Parikshit Gopalan, Lunjia Hu, Michael P. Kim, Omer Reingold, and Udi Wieder. Loss Minimization Through the Lens Of Outcome Indistinguishability. In Yael Tauman Kalai, editor, *14th Innovations in Theoretical Computer Science Conference (ITCS 2023)*, volume 251 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 60:1–60:20, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ITCS.2023.60.

12 Parikshit Gopalan, Adam Tauman Kalai, Omer Reingold, Vatsal Sharan, and Udi Wieder. Omnipredictors. In Mark Braverman, editor, *13th Innovations in Theoretical Computer Science Conference, ITCS 2022, January 31 – February 3, 2022, Berkeley, CA, USA*, volume 215 of *LIPIcs*, pages 79:1–79:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.ITCS.2022.79.

13 Parikshit Gopalan, Michael P Kim, Mihir A Singhal, and Shengjia Zhao. Low-degree multicalibration. In Po-Ling Loh and Maxim Raginsky, editors, *Proceedings of Thirty Fifth Conference on Learning Theory*, volume 178 of *Proceedings of Machine Learning Research*, pages 3193–3234. PMLR, 02–05 July 2022. URL: https://proceedings.mlr.press/v178/gopalan22a.html.

14 Ursula Hébert-Johnson, Michael Kim, Omer Reingold, and Guy Rothblum. Multicalibration: Calibration for the (computationally-identifiable) masses. In *International Conference on Machine Learning*, pages 1939–1948. PMLR, 2018.

15 Paul W. Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. *Social Networks*, 5(2):109–137, 1983. doi:10.1016/0378-8733(83)90021-7.

16 Shlomo Hoory, Nathan Linial, and Avi Wigderson. Expander graphs and their applications. *Bulletin of the AMS*, 43(4):439–561, 2006.

17 Lunjia Hu, Inbal Livni-Navon, and Omer Reingold. Generative models of huge objects, 2023. arXiv:2302.12823.

18 Lunjia Hu, Inbal Livni-Navon, Omer Reingold, and Chutong Yang. Omnipredictors for constrained optimization. *arXiv preprint*, 2022. arXiv:2209.07463.

19 Lunjia Hu and Charlotte Peale. Comparative Learning: A Sample Complexity Theory for Two Hypothesis Classes. In Yael Tauman Kalai, editor, *14th Innovations in Theoretical Computer Science Conference (ITCS 2023)*, volume 251 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 72:1–72:30, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ITCS.2023.72.

20 Lunjia Hu, Charlotte Peale, and Omer Reingold. Metric entropy duality and the sample complexity of outcome indistinguishability. In Sanjoy Dasgupta and Nika Haghtalab, editors, *Proceedings of The 33rd International Conference on Algorithmic Learning Theory*, volume 167 of *Proceedings of Machine Learning Research*, pages 515–552. PMLR, 29 March–01 April 2022. URL: https://proceedings.mlr.press/v167/hu22a.html.

21 Russell Impagliazzo. Lecture on learning models: connections between boosting, hard-core distributions, dense models, GAN, and regularity I. https://www.ias.edu/video/csdm/2017/1113-RussellImpagliazzo, 2017.

22 Dimitar Jetchev and Krzysztof Pietrzak. How to fake auxiliary input. In *Theory of cryptography*, volume 8349 of *Lecture Notes in Comput. Sci.*, pages 566–590. Springer, Heidelberg, 2014. doi:10.1007/978-3-642-54242-8_24.

23 Michael P Kim, Amirata Ghorbani, and James Zou. Multiaccuracy: Black-box post-processing for fairness in classification. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pages 247–254, 2019.

24 Michael P. Kim, Christoph Kern, Shafi Goldwasser, Frauke Kreuter, and Omer Reingold. Universal adaptability: Target-independent inference that competes with propensity scoring. *Proceedings of the National Academy of Sciences*, 119(4):e2108097119, 2022. doi:10.1073/pnas.2108097119.

25 Michael P. Kim and Juan C. Perdomo. Making Decisions Under Outcome Performativity. In Yael Tauman Kalai, editor, *14th Innovations in Theoretical Computer Science Conference (ITCS 2023)*, volume 251 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 79:1–79:15, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ITCS.2023.79.

26 Yoshiharu Kohayakawa and Vojtech Rödl. Szemerédi's regularity lemma and quasi-randomness. *Recent advances in algorithms and combinatorics*, pages 289–351, 2003.

27 Michael Luby and Charles Rackoff. How to construct pseudorandom permutations from pseudorandom functions. *SIAM Journal on Computing*, 17(2):373–386, 1988.

28 Michael Mitzenmacher. A Brief History of Generative Models for Power Law and Lognormal Distributions. *Internet Mathematics*, 1(2):226–251, 2003.

29 Joseph Naor and Moni Naor. Small-bias probability spaces: Efficient constructions and applications. *SIAM Journal on Computing*, 22(4):838–856, August 1993.

30 Moni Naor and Asaf Nussboim. Implementing huge sparse random graphs. In Moses Charikar, Klaus Jansen, Omer Reingold, and José D. P. Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, 10th International Workshop, APPROX 2007, and 11th International Workshop, RANDOM 2007, Princeton, NJ, USA, August 20-22, 2007, Proceedings*, volume 4627 of *Lecture Notes in Computer Science*, pages 596–608. Springer, 2007. doi:10.1007/978-3-540-74208-1_43.

31 Moni Naor, Asaf Nussboim, and Eran Tromer. Efficiently constructible huge graphs that preserve first order properties of random graphs. In Joe Kilian, editor, *Theory of Cryptography, Second Theory of Cryptography Conference, TCC 2005, Cambridge, MA, USA, February 10-12, 2005, Proceedings*, volume 3378 of *Lecture Notes in Computer Science*, pages 66–85. Springer, 2005. doi:10.1007/978-3-540-30576-7_5.

32 Moni Naor and Omer Reingold. Constructing pseudo-random permutations with a prescribed structure. *J. Cryptol.*, 15(2):97–102, January 2002. doi:10.1007/s00145-001-0008-5.

33 Alexander Scott. Szemerédi's regularity lemma for matrices and sparse graphs. *Combinatorics, Probability and Computing*, 20(3):455–466, 2011.

34 Maciej Skórski. Simulating auxiliary inputs, revisited. In *Theory of cryptography. Part I*, volume 9985 of *Lecture Notes in Comput. Sci.*, pages 159–179. Springer, Berlin, 2016. doi:10.1007/978-3-662-53641-4_7.

35 Maciej Skórski. A subgradient algorithm for computational distances and applications to cryptography. *Cryptology ePrint Archive*, 2016.

36 Maciej Skórski. A cryptographic view of regularity lemmas: simpler unified proofs and refined bounds. In *Theory and applications of models of computation*, volume 10185 of *Lecture Notes in Comput. Sci.*, pages 586–599. Springer, Cham, 2017. doi:10.1007/978-3-319-55911-7.

37 Luca Trevisan, Madhur Tulsiani, and Salil P. Vadhan. Regularity, boosting, and efficiently simulating every high-entropy distribution. In *Proceedings of the 24th Annual IEEE Conference on Computational Complexity, CCC 2009, Paris, France, 15-18 July 2009*, pages 126–136. IEEE Computer Society, 2009. doi:10.1109/CCC.2009.41.

38 Salil Vadhan and Colin Jia Zheng. A uniform min-max theorem with applications in cryptography. In *Advances in Cryptology–CRYPTO 2013: 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I*, pages 93–110. Springer, 2013.

39 Salil P. Vadhan. Pseudorandomness. *Foundations and Trends® in Theoretical Computer Science*, 7(1–3):1–336, 2012. doi:10.1561/0400000010.

40 Mark N. Wegman and J. Lawrence Carter. New hash functions and their use in authentication and set equality. *Journal of Computer and System Sciences*, 1981.

41 Andrew C. Yao. Theory and applications of trapdoor functions. In *23rd annual symposium on foundations of computer science (Chicago, Ill., 1982)*, pages 80–91. IEEE, New York, 1982.