Principled Private Data Release with Deep Learning

Michael Fine Harvard University
Cambridge MA, 02138
mfine@college.harvard.edu

1 Background

1.1 Query Release Problem

We study the problem of privately generating synthetic data to answer statistical queries over a data universe $\mathcal{X}$. Formally, a statistical query over $\mathcal{X}$ is a function $q : \mathcal{X} \to \{0, 1\}$. Given a a dataset $S \in \mathcal{X}^n$, we define $q(S) = \sum_{s \in S} q(s)$.

Our goal is to produce a synthetic dataset that, for every query in some family of queries, takes approximately the same value as the true dataset.

Definition 1 ($\alpha-$approximate)**.** *We say a synthetic dataset S $\alpha-$approximates a true dataset $\hat{S}$ w.r.t a family of statistical queries $\mathcal{Q}$ if*

$$\forall q \in \mathcal{Q} : \quad |q(S) - q(\hat{S})| \leqslant \alpha \tag{1}$$

[TODO finish up]

1.2 Game Theoretic Formulation

One can formulate the problem of producing an $\alpha-$approximate dataset as a two-player, zero sum game [5] between a discriminator D and a generator G. The generator has an action set $\mathcal{X}$, while the discriminator has an action set $\mathcal{Q}$. The generator aims to output a dataset $S \in \mathcal{X}$ that maximally agrees with $\hat{S}$, while the discriminator aims to find queries $q \in \mathcal{Q}$ that distinguish $\hat{S}$ and S.

Formally, given a play $S \in \mathcal{X}$ and $q \in Q$, the discriminator gets payoff $V(S, q)$ and the generator gets payoff $V(S, q)$, where $V(S, q)$ denotes:

Definition 2 (Payoff)**.**

$$V(S, q) := |q(S) - q(\hat{S})| \tag{2}$$

The goal of both G and D is to maximize their worst case payoffs, thus

$$\max_{q \in \mathcal{Q}} \min_{S \in X} V(S, q) \ \text{(Goal of } D) \quad \textit{and} \quad \min_{S \in X} \max_{q \in \mathcal{Q}} V(S, q) \ \text{(Goal of } G) \tag{3}$$

If there exists a point (S^*, q^*) such that neither G nor D can improve their payoffs by playing a different move, we call that a *Pure Nash Equilibrium*. Unfortunately, a pure equilibrium is not always guaranteed to exist (and likely does not in the case of synthetic data generation).

However, the seminal work of Nash et. al showed that there always exists a *Mixed Nash Equilibrium (MNE)*, where the players play *probability distributions* over their action sets, instead of fixed actions.

Let $\Delta(\mathcal{X})$ and $\Delta(\mathcal{Q})$ denote the set of probability distribution over $\mathcal{X}$ and $\mathcal{Q}$. Formally, if G plays a strategy $u \in \Delta(\mathcal{X})$ and D plays $w \in \Delta(\mathcal{Q})$, we define the payoff to be the expected value of a single draw:

Preprint. Under review.

$$V(u, w) := \mathbb{E}_{S\sim u, q\sim w} V(S, q) \tag{4}$$

Thus, a pair of strategies $u^* \in \Delta(\mathcal{X})$ and $w^* \in \Delta(\mathcal{Q})$ forms an $\alpha-$approximate mixed nash equilibrium if for all strategies $u \in \Delta(\mathcal{X})$ and $w \in \Delta(\mathcal{Q})$

$$V(u^*, w) \leqslant V(u, w) + \alpha \quad and \quad V(u, w^*) \leqslant V(u, w) - \alpha \tag{5}$$

Moreover, Gaboardi et. al showed how to reduce the problem of finding an $\alpha-$approximate dataset to the problem of finding an an $\alpha-$equilibrium in the query release game:

Theorem 1. *Let (u, w) be the $\alpha-$approximate MNE in a query release game for a dataset $\hat{S} \in \mathcal{X}$ and a query universe $\mathcal{Q}$. If $\mathcal{Q}$ is closed under negation, then the dataset S sampled from u α-approximates $\hat{S}$ over Q. [3]*

Hence, our task is to provide an algorithm to private reach an $\alpha-$MNE in the query release game. In the following section, we will provide the background for how this can be done with GANs.

1.3 Generative Adversarial Networks

Generative Adversarial Networks (GANs) TODO:

1.4 Online Learning

To efficiently find equilibrium in the zero-sum GAN game, we draw on results from online learning. In the online learning setting, in each of T rounds a player is given a loss function f_t , possibly adversarial chosen. The players goal is to chose an action $x_{t+1} \in \mathcal{X}$ in order to minimize the cumulative *regret*.

Definition 3 (Regret)**.** *The regret measures the cumulative loss of the player, compared to the best fixed decision in hindsight.*

$$\textit{Regret}_T(f_1, ..., f_T) = \sum_{t=1}^{T} f_t(x_t) - min_{x^*\in\mathcal{X}} \sum_{t=1}^{T} f_t(x^*) \tag{6}$$

When a strategy provably leads to regret is sublinear in T, we call that *no-regret*, as regret $\to 0$ as $T \to \infty$.

One approach to regret minimization is to choose the action x_{t+1} that minimizes the cumulative loss over all past loss functions

$$x_{t+1} = \arg\min_{x\in\mathcal{X}} \sum_{t=1}^{T} \tag{7}$$

This approach is known as *Follow-The-Leader*. While natural, this approach is easily exploitable by an adversary. At a high level, this is because it *overfits* to past outcomes, allowing it to optimize between suboptimal strategies. To rectify this, a powerful strategy is *Follow-the-Regularized-Leader*

Definition 4 (Follow-The-Reguralized-Leader (FTRL))**.** *Given a reguralization function $R(x)$ and a regularization weight η_T, at each step choose $x_{t+1} \in \mathcal{X}$ to minimize the regularized cumulative loss:*

$$x_{t+1} = \arg\min_{x\in\mathcal{X}} \sum_{t=1}^{T} + \eta_T R(X) \tag{8}$$

One common regularization function is the l_2 norm $R(x) = ||x||_2$.

When the loss function f_T is convex, FTRL can be shown to be no-regret [4]. Unfortunately, in the GAN setting, where the loss function f_T is defined by a highly non-convex deep network, we don't have that guarantee.

1.4.1 Online Learning for Non-convex losses

In general, finding the minimum of a sum of non-convex functions is hard. However, in practice gradient descent over neural networks has proven to be remarkably effective at approximately solving non-convex loss functions. This leads to the natural question: assuming we have an offline non-convex optimization oracle $\mathcal{O}$, can we use that to find a no-regret strategy in the online setting? Agarwal et al showed that we can, using a Follow-The-Leader variant known as Follow-The-Perturbed-Leader [1]. Formally:

Definition 5 (Offline optimization oracle)**.** *Let $\mathcal{O}$ take a sequence of (possibly non-convex) loss functions $(f_1 ... f_T) \in \mathcal{L}^T$ and a d-dimensional vector d, and output $x^* \in \mathcal{X}$*

$$x^* \in \arg\min_{x \in \mathcal{X}} \sum_{t=1}^{T} f_t + \sigma^T x \tag{9}$$

We can use this offline oracle $\mathcal{O}$ to minimize regret in the online case:

Definition 6 (Follow-The-Perturbed-Leader (FTPL))**.** *Given an offline oracle $\mathcal{O}$ and a parameter η, at each step t FTPL draws a random vector $\sigma_t \sim Exp(\eta)^d$. It then outputs*

$$x^* \in \arg\min_{x \in \mathcal{X}} \sum_{t=1}^{T} f_t + \sigma^T x \tag{10}$$

Theorem 2. *FTPL has sublinear regret. [1]*

Building on the work of Freund and Schapire, we can show that if G and D both play FTPL for T rounds, they will converge to an $\alpha-$approximate equilibrium. Formally:

Theorem 3. *Suppose that G and D play according to FTPL. We can choose $T \in poly(d)/\alpha^3$ such that the expected average regret of FTPL is at most α. Then, G_T and D_T produce strategies in an $\alpha-$approximate equilibrium [1].*

2 Results

TODO: summarize approach Theorem 3 requires the existence of an actual oracle $\mathcal{O}$ that can find the minimum of a perturbed sum of non-convex functions. In this case, we need an oracle that can minimize the sub of deep neural networks. While SGD is remarkably effective in practice, we are unable to provide guarantees (probabilistic or otherwise) about how close the convergent solution SGD outputs is to the global minima. Recent work suggests that this is not a problem in practice, as spurious local minima (local minima significantly worse than the global minima) get exponentially rarer as the network gets larger [2]. However, these results still rely on too many impractical assumptions to make them relevant in practice.

However, while we cannot guarantee (or even certify) convergence to an approximate global minima in general, we can take advantage of the specific structure of the query release problem. In the query release problem (unlike most deep learning problems) we know *exactly* what the global minima of loss for the generator is: 1/2 – the loss from outputting the true, sensitive dataset. TODO: prove this.

We can use this to track the regret bounds TODO: Explain this better in prose

Algorithm 1: QueryGAN

Input: one-layer discriminator D_θ, deep generator G_ϕ, offline optimization oracle $\mathcal{O}$, Rounds T, noise η, output dimension d, game objective M

Result: Trained generator G_T, Accuracy α

for $t \in 1...T$ **do**

Draw discriminator and generator perturbations

$\sigma_1 \sim Exp(\eta)^d \;\; and \;\; \sigma_2 \sim Exp(\eta)^d$

Update D and G according to FTPL:

$\theta_{t+1} \leftarrow \theta_t - \nabla_{\theta_t} \left(\sum_{t=1}^{T} f_t + \sigma_1^T x \right) \quad \text{and} \quad \phi_{t+1} \leftarrow \phi_t - \nabla_{\phi_t} \left(\sum_{t=1}^{T} g_t + \sigma_2^T x \right)$

Update losses:

$f_{t+1}(\cdot) = M(\cdot, D_{\phi_{t+1}}) \;\; and \;\; g_{t+1}(\cdot) = M(D_{\phi_{t+1}}, \cdot)$

Store generator loss:

$\alpha_{t+1} \leftarrow f_{t+1}(G_{\phi_{t+1}}) - 1/2$

return *Mixed strategy:* $G \sim Unif\{G_{\theta_1}, G_{\theta_T}\}$, *Regret:* $\sum_{t \in T} \alpha_T$

Theorem 4. *Let G, α be the results of running QueryGAN with inputs TODO: What inputs. Let x be the dataset sampled from G*

$$x := \{G(z_1), ..., G(z_n)\} \quad \textit{where } z \sim P_z^n \tag{11}$$

TODO: what is P_z

Then x is $\alpha-$approximate with respect to all queries representable by D.

- Replace D with multiplicative weights
- Replace D with

References

[1] Naman Agarwal, Alon Gonen, and Elad Hazan. Learning in Non-convex Games with an Optimization Oracle. *arXiv:1810.07362 [cs, stat]*, October 2018.

[2] Anna Choromanska, Mikael Henaff, Michael Mathieu, Gérard Ben Arous, and Yann LeCun. The Loss Surfaces of Multilayer Networks. *arXiv:1412.0233 [cs]*, November 2014.

[3] Marco Gaboardi, Emilio Jesús Gallego Arias, Justin Hsu, Aaron Roth, and Zhiwei Steven Wu. Dual Query: Practical Private Query Release for High Dimensional Data. *arXiv:1402.1526 [cs]*, February 2014.

[4] Elad Hazan. Introduction to Online Convex Optimization. *arXiv:1909.05207 [cs, math, stat]*, September 2019.

[5] Justin Hsu, Aaron Roth, and Jonathan Ullman. Differential Privacy for the Analyst via Private Equilibrium Computation. *Proceedings of the 45th annual ACM symposium on Symposium on theory of computing - STOC '13*, page 341, 2013.