- Generator is neural network trained by gradient descent
    - Can verify optimality with near zero loss on queries
- Discriminator is simply multiplicative weights over query set
    - This lets us verify optimality for free
    - Maybe mixture of discriminators, some neural networks etc
    - Alternatively, train on empirical samples and boost
        - Probability of generalization => generation bounds, use differential privacy (nice little combo herer)
- Online algorithm trained like mixgan

Verifiable Private Data Release with Generative Adversarial Networks

Query Release Game

Suppose we want to generate synthetic data for a dataset T to approximately answer a set of queries $\mathcal{Q}$. We can view this as a zero-sum game between a discriminator D and a generator G. The generator aims to output a dataset X that maximally agrees with $\hat{X}$, while the discriminator aims to find queries that distinguish X and $\hat{X}$.

Formally, given a play $x \in \mathcal{X}$ and $q \in \mathcal{Q}$, we define the payoff $A(x, q)$

$$A(x, q) := q(T) - q(x) \tag{1}$$

where we extend queries to apply to multiple rows as the average value the query takes over that row. We let G and D play probabilty distributions over X and Q, where

$$A(u, w) := \mathbb{E}_{x \sim u, q \sim w} A(x, q) \tag{2}$$

Notice we can phrase this payoff in an equivalent form to the Wasserstein GAN training loss, up to log factors [**TODO** this is all really hand-wavy need to distinguish between empirical datasets and actually the whole dataset, as well as the fact that D doesn't have to output a fixed query]

$$A(u, w) := \mathbb{E}_{x \sim T}[D(x)] - \mathbb{E}_{z \sim p_z}[D(G(z))] \tag{3}$$

Defn: Let A be the payoffs for a two player, zero-sum game with action sets $\mathcal{X}$, $\mathcal{Q}$. Then a pair of strategies $u^* \in \Delta\mathcal{X}$ and $w^* \in \Delta\mathcal{Q}$ form an $\alpha-$approximate mixed nash equilibrium (concisely, an $\alpha-$MNE) if

$$A(u^*, w) \leq v_A + \alpha \quad \text{and} \quad A(u, w^*) \geq v_A - \alpha \tag{4}$$

Game

Let G be the generator, a deep neural network parametrized by ϕ. For a fixed discriminator D, G has loss

$$L(\phi; D) = A(u, w) \tag{5}$$

Let the discriminator D be a single layer neural network parametrized by $\theta \in \mathbb{R}^d$, with a sigmoid activation function. Given a sample $x \sim G$, define

$$D(x) := \sigma(\theta^T x + b) \tag{6}$$

Let $\mathcal{Q}$ be the set of all marginals. A **marginal** q over a dataset $X \in \{0,1\}^{d \times n}$ is a monotone conjunction over some subset of attributes S, defined by $q_S(X) = \prod_{i \in S} x_i$ for $x \in X$.

Equivalently, we can define a marginal q_θ parametrized by a boolean vector $\theta \in \{0,1\}^d$ as

$$q_\theta(X) = \sum_{x \in X} \theta^T x \tag{7}$$

[TODO: show that $\mathbb{E}[D_\theta] \approx q_\theta$]

As such, training G and D to an $\alpha-$MNE implies that G will with high probability produce a dataset D that accurately answers all marginal queries to within α error.

Training

Criteria for GAN convergence or optimality is not well understood in general. However, a few features of this formulation allow us to provide optimality guarantees, so long as the training converges.

We use a result from [@GLL+17] that training each player in a semi-concave zero sum game with instatiations of Follow-the-Regularized-Leader for T steps outputs mixed strategies that are $\alpha-$MNE where $\alpha = O(1/\sqrt{T})$.

Note that this at each step this relies on finding the optimal weighted sum of the generator neural networks

$$\theta_t = \arg\min_\theta \sum_{i=0}^{t-1} f_i(\theta) \tag{8}$$

where the cost function $f_i(\theta) = A(G_\theta, D_i)$.

This relies on the existence of an oracle that can find the minimum over a sum of generative networks. While hard in the worst case, in practice SGD is able to find good local minima. Moreover, unlike in most cases of deep learning, where we are unable to tell how close our solution is to the true global minima, here we know the global minima: 0.5. Assuming the discriminator is optimal (which, given its convex loss function, is fair), an optimal generator should reduce the discriminator to performing no better than chance. [TODO more to do with capacity etc]. We know this is obtainable because a the best any discriminator could do against a generator that output the true dataset is 0.5. As such, we are able to track exactly how suboptimal θ_t is.

Tracking cumulative regret

At each iteration i, define γ_i as the cumulative regret vs. the theoretically optimal generator:

$$\begin{aligned}\gamma_i &:= \sum_{i=0}^{t-1} f_i(\theta_t) - \min_{\theta} \sum_{i=0}^{t-1} f_i(\theta) \\ &= \sum_{i=0}^{t-1} f_i(\theta_t) - 1/2\end{aligned}$$

[TODO not clear we can actually always reach 1/2 because of noise in discriminator, need to prove this with ERM].

We know from [@FS99] that if γ_i is 0 at every $i \in [0...T]$ (eg we play a true no-regret algorithm), then we will reach an α-MNE. What can we say about the regret for *non-zero* γ_i?

[TODO]

Outline

- Abstract

Nesterov EDT

Near-Optimal No-Regret Algorithms for Zero-Sum Games

Looks at smooth approximations