Differentially Private Generative Adversarial Network

Liyang Xie[1], Kaixiang Lin[1], Shu Wang[2], Fei Wang[3], Jiayu Zhou[1]
[1]Computer Science and Engineering, Michigan State University
[2]Department of Computer Science, Rutgers University
[3]Department of Healthcare Policy and Research, Weill Cornell Medical School
{xieliyan,linkaixi}@msu.edu,sw498@cs.rutgers.edu,few2001@med.cornell.edu,jiayuz@msu.edu

ABSTRACT

Generative Adversarial Network (GAN) and its variants have recently attracted intensive research interests due to their elegant theoretical foundation and excellent empirical performance as generative models. These tools provide a promising direction in the studies where data availability is limited. One common issue in GANs is that the density of the learned generative distribution could concentrate on the training data points, meaning that they can easily *remember* training samples due to the high model complexity of deep networks. This becomes a major concern when GANs are applied to private or sensitive data such as patient medical records, and the concentration of distribution may divulge critical patient information. To address this issue, in this paper we propose a differentially private GAN (DPGAN) model, in which we achieve differential privacy in GANs by adding carefully designed noise to gradients during the learning procedure. We provide rigorous proof for the privacy guarantee, as well as comprehensive empirical evidence to support our analysis, where we demonstrate that our method can generate high quality data points at a reasonable privacy level.

CCS CONCEPTS

• **Computing methodologies** → **Neural networks**; • **Computer systems organization** → **Neural networks**; • **Security and privacy** → *Privacy-preserving protocols*;

KEYWORDS

Deep Learning; Differential Privacy; Generative model

1 INTRODUCTION

In recent years, more and more data in different application domains are becoming readily available for the rapid development of both computer hardware and software technologies. Many data mining methodologies have been developed for analyzing those big data sets. One representative example is deep learning, which typically needs a huge amount of training samples to achieve promising performance. However, there exists domains where it is impossible to get as much data as we want. Medicine and Health Informatics are such fields. On individual patient level analysis, each patient is treated as a sample in model training process. However, considering the complexity of many diseases, the number of all patients from the whole world is still very small and far from enough. Moreover, we can never get the medical data from all patients for privacy and sensitivity reasons. Further, the expensive and time-consuming data collection process also limits the amount of data. Thus, the problem of building high-quality medical analytics models remains very challenging at present.

Generative models [5, 21–23, 30] have provided us a promising direction to alleviate the data scarcity issue. By sketching the data distribution from a small set of training data, we are able to sample from the distribution and generate much more samples for our study. By combining the complexity of deep neural networks and game theory, the Generative Adversarial Network (GAN) [16] and its variants have demonstrated impressive performance in modeling the underlying data distribution, generating high quality "fake" samples that are hard to be differentiated from real ones [24, 31, 32]. Ideally, with the high quality generative distribution in hand, we can protect the privacy of raw data by releasing only the distribution instead of the raw data to the public or constrained individuals, and can even sample datasets to fit our needs and conduct further analysis.

However, the GANs can still implicitly disclose privacy information of the training samples. The adversarial training procedure and the high model complexity of deep neural networks, jointly encourage a distribution that is concentrated around training samples. By repeated sampling from the distribution, there is a considerable chance of recovering the training samples [2]. For example, Hitaj *etal.* [19] introduced an active inference attack model that can reconstruct training samples from the generated ones. Therefore, it is highly demanded to have generative models that not only generates high quality samples but also protects the privacy of the training data.

With the above considerations, in this paper we propose a *Differentially Private Generative Adversarial Network* (DPGAN). DPGAN provides proven privacy control for the training data from the sense of differential privacy [12]. Specifically, our proposed framework applies a combination of carefully designed noise and gradient clipping, and uses the *Wasserstein distance* [2] as an approximation of the distance between probability distributions, which is a more reasonable metric than JS- divergence in GAN. There are also prior works on studying differential privacy in deep learning models [1]. However, our DPGAN is different from [1] by clipping only on weights. We also proves that the gradient can be bounded at same time, which avoids unnecessary distortion of the gradient. This not only keeps the loss function with Lipschitz property but also provides a sufficient privacy guarantee. Unlike the privacy preserving deep framework mentioned in [25], whose privacy loss is proportional to the amount of data needed to be labeled in public

Conference'18, Aug 2018, London, UK

ACM ISBN 123-4567-24-567/08/06.
https://doi.org/10.475/123_4

data set, the privacy loss of our DPGAN is irrelevant to the amount of generated data. This makes our methods applicable under a wide variety of real world scenarios. We evaluate DPGAN under various benchmark datasets and network structures (fully connected networks and CNN), and demonstrate that DPGAN can generate high-quality data points with sufficient protection for differential privacy with reasonable privacy budget.

The remaining of the paper is structured as follows: first, we will briefly overview the related literature in Section 2, and then introduce the proposed DPGAN framework and theoretical properties in Section 3. Our framework is evaluated in Section 4 by the end.

2 RELATED WORK

In this section, we provide a brief literature review of relevant topics: generative adversarial network, differential privacy and differentially private learning in neural networks.

Generative Adversarial Network. GAN and its variants are developed in recent years with important advances from the theoretical perspective. Instead of clipping the weights, Gulrajani *et al.* [17] improve the training stability and performance of WGAN by penalizing the norm of the critical gradients with respect to its input. Gulrajani *et al.* [17] is aligned with our differential privacy framework due to controlled value of gradient norms.

Zhao *et al.* [36] introduces energy-based GAN (EBGAN), which views the discriminator as an energy function that attributes low energies to the regions near the data manifold and higher energies to other regions. Similar to the original GANs, a generator is seen as being trained to produce contrastive samples with minimal energies, while the discriminator is trained to assign high energies to these generated samples. The instantiation of EBGAN framework use an auto-encoder architecture, with the energy being the reconstruction error, in place of the usual discriminator. The behavior of EBGAN has shown to be more stable than regular GANs during training. Berthelot *et al.* [4] also use an autoencoder as a discriminator and developed an equilibrium enforcing method, paired with a loss derived from the Wasserstein distance. It improves over WGAN by balancing the power of the discriminator and the generator so as to control the trade- off between image diversity and visual quality. Qi [28] proposes a loss-sensitive GAN with Lipschitz assumptions on data distribution and loss function. It improves WGAN by allowing the generator to focus on improving poor data points that are far apart from real examples rather than wasting efforts on those samples that have already been well generated, and thus improving the overall quality of generated samples. Jones *et al.* [3] used differentially private version of Auxiliary Classifier GAN (AC-GAN) to simulate participants based on the population of the SPRINT clinical trial. Choi *et al.* [8] proposed medGAN, which is a generative adversarial framework that can successfully generate EHR. However, the approach may have privacy concerns as we discussed earlier.

Differential Privacy. Differential privacy (DP) [9] and related algorithms have been widely studied in the literatures. Examples include Dwork *et al.* [11] for sensitivity-based algorithm, which is among the most popular methods that protect privacy by adding noise to mask the maximum change of data related functions. This work laid the theoretical foundation of many DP studies. Chaudhuri *et al.* [6, 7] proposed DP empirical risk minimization. The general idea of our DP framework has the same spirits as the objective perturbation, which is different from adding noise directly on the output parameters. Another related framework that adds noise on gradient is Song *et al.* [35], which studied DP variants of stochastic gradient descent. In their empirical results, the practice of moderate increasing in the batch size can significantly improve the performance. Song *et al.* [34] followed their early work [35], and studied as how to use stochastic gradient to learn from models trained by data from multiple sources with DP requirements (hence multiple level of noise). A comprehensive and structured overview of DP data publishing and analysis can be found in [37], where several possible future directions and possible applications are also mentioned.

Differentially Private Learning in Neural Network. The applications of DP in deep learning have been studied recently in several literatures: Abadi *et al.* [1] studied a gradient clipping method that imposed privacy during the training procedure. Shokri and Shmatikov [33] for multi-party privacy preserving neural network with a parallelized and asynchronous training procedure. Papernot *et al.* [25] combined Laplacian mechanism with machine teaching framework. Phan *et al.* [27] developed "adaptive Laplace Mechanism" that could be applied in a variety of different deep neural networks while the privacy budget consumption is independent of the number of training step. Phan *et al.* [26] developed a private convolutional deep belief network by leveraging the functional mechanism to perturb the energy-based objective functions of traditional CDBNs.

We propose DPGAN to address the challenges appeared in the previous works. In [25] the privacy loss is proportional to the amount of data labeled in that public data set, which may bring about unbearable privacy loss. We solve this problem by training a differentially private generator and can generate infinite number of data points without violating the privacy of training data. Shokri and Shmatikov [33] requires the transmission of updated local parameters between server and local task, which is at risk of leakage of private information. Our framework addressed this issue by avoiding a distributed framework. Also, our work is different from [26] by adding noise within the training procedure instead of adding noise on both energy functions and an extra softmax layer.

3 METHODOLOGY

In this section, we elaborate the proposed privacy preserving framework DPGAN. Without loss of generality, we discuss the DPGAN in the context of the WGAN framework [2] while we note that the proposed DPGAN technique can also be easily extended to other GAN frameworks. We firstly introduce differential privacy and then conduct a brief review of GAN and WGAN. We then introduce moments accountant [1], which is the key technique in our framework to set a bound to the probability ratio so as to guarantee the privacy in the iterative gradient descent procedure.

3.1 Differential Privacy

The privacy model used in our approach is differential privacy [10]. Denote an algorithm with the differential privacy property by $A_p(\cdot)$. The algorithm is randomized in order to make it difficult for an

observer to re-identify the input data, where an observer is anyone who gets outputs of algorithms using the data. Differential privacy (DP) is defined by [12]:

Definition 3.1. (Differential Privacy, DP) A randomized algorithm $\mathcal{A}_p$ is (ϵ, δ)-differentially private if for any two databases $\mathcal{D}$ and $\mathcal{D}'$ differing in a single point and for any subset of outputs S:

$$\mathbb{P}(\mathcal{A}_p(\mathcal{D}) \in \mathrm{S}) \leq e^{\epsilon} \cdot \mathbb{P}(\mathcal{A}_p(\mathcal{D}') \in \mathrm{S}) + \delta, \quad (1)$$

where $\mathcal{A}_p(\mathcal{D})$ and $\mathcal{A}_p(\mathcal{D}')$ are the outputs of the algorithm for input databases $\mathcal{D}$ and $\mathcal{D}'$, respectively, and $\mathbb{P}$ is the randomness of the noise in the algorithm.

It can be shown that the definition is equivalent to:

$$\left|\log\left(\frac{P(\mathcal{A}_p(\mathcal{D}) = s)}{P(\mathcal{A}_p(\mathcal{D}') = s)}\right)\right| \leq \epsilon,$$

with probability $1-\delta$ for every point s in the output range, where ϵ reflects the *privacy level*. A small ϵ (≤ 1.0) means that the difference of algorithm's output probabilities using $\mathcal{D}$ and $\mathcal{D}'$ at s is small, which indicates high perturbations of ground truth outputs and hence high privacy, and vice versa. The non-private case is given by $\epsilon = \infty$. δ measures the violation of the "pure" differential privacy. That is, there exists a small output range associated with probability δ such that for some fixed point s in this area, no matter what the value of ϵ is, one can always find a pair of datasets $\mathcal{D}$ and $\mathcal{D}'$ so that the inequality $|\log(\frac{P(\mathcal{A}_p(\mathcal{D})=s)}{P(\mathcal{A}_p(\mathcal{D}')=s)})| \geq \epsilon$ holds. Typically we are interested in values of δ so that are less than the inverse of any polynomial in the size of the database.

According to Def. 3.1 and the intuition above, the noise protects the membership of a data point in the dataset. For example, when conducting a clinical experiment, sometimes a person does not want the observer to know that he or she is involved in the experiment. This is due to the fact that observer may link the test result to the appearance/disappearance of certain person and harm the interest of that person. A proper membership protection would ensure that replacing this person with another one will not affect the result too much. This property holds only if the algorithm itself is randomized, i.e. the output is associated with a distribution. And this distribution will not change too much if certain data point is perturbed or even removed. This exactly what the differential privacy tries to achieve.

3.2 GAN and WGAN

Generative adversarial nets [16] simultaneously train two models: a generative model G that transforms input distribution to output distribution that approximates the data distribution, and a discriminative model D that estimates the probability that a sample came from the training data rather than the output of G. Let $p_z(\mathbf{z})$ be the input noise distribution of G and $p_{data}(\mathbf{x})$ be the real data distribution, GAN aims at training G and D to play the following two-player minimax game with value function $V(G, D)$:

$$\min_G \max_D V(G,D) = E_{\mathbf{x}\sim p_{data}(\mathbf{x})}[log(D(\mathbf{x}))] + E_{\mathbf{z}\sim p_z(\mathbf{z})}[log(1 - D(G(\mathbf{z})))]. \quad (2)$$

WGAN [2] improves GAN by using the Wasserstein distance instead of the Jensen–Shannon divergence. It solves a different two-player minimax game given by:

$$\min_G \max_{w\in W} E_{\mathbf{x}\sim p_{data}(\mathbf{x})}[f_w(\mathbf{x})] - E_{\mathbf{z}\sim p_z(\mathbf{z})}[f_w(G(\mathbf{z}))], \quad (3)$$

where functions $\{f_w(\mathbf{x})\}_{w\in W}$ are all K-Lipschitz (with respect to x) for some K. Our approach exploits such K-Lipschitz property in WGAN and solves Formula 3 in a differentially private manner.

3.3 DPGAN framework

Our method focuses on preserving the privacy during the training procedure instead of adding noise on the final parameters directly, which usually suffers from low utility. We add noise on the gradient of the Wasserstein distance with respect to the training data. The parameters of discriminator can be shown to guarantee differential privacy with respect to the sample training points. We note that the privacy of data points that havenâĂŹt been sampled for training is guaranteed naturally. This is because replacing these data won't cause any change in output distribution, which is equivalent to the case of $\epsilon = 0$ in Definition 3.1. The parameters of generator can also guarantee differential privacy with respect to the training data. This is becuase there is a post-processing property of differential privacy [12], which says that any mapping (operation) after a differentially private output will not invade the privacy. Here the mapping is in fact the computation of parameters of generator and the output is the differentially private parameter of discriminator. Since the parameters of generator guarantee differential privacy of data, it is safe to generate data after training procedure. In short, we have: differentially private discriminator + computation of generator → differentially private generator. This also means that even if the observer gets generator itself, there is no way for him/her to invade the privacy of training data.

The DPGAN procedure is summarized in Algorithm 1. In line 9, the clipping guarantees that $\{f_w(x)\}_{w\in W}$ are all K_w-Lipschitz with respect to x for some unknown K_w and act in a way to bound the gradient from each data point. The RMSProp in line 8 and line 13 is

Algorithm 1 Differentially Private Generative Adversarial Nets

Require: α_d, learning rate of discriminator. α_g, learning rate of generator. c_p, parameter clip constant. m, batch size. M, total number of training data points in each discriminator iteration. n_d, number of discriminator iterations per generator iteration. n_g, generator iteration. σ_n, noise scale. c_g, bound on the gradient of Wasserstein distance with respect to weights

Ensure: Differentially private generator θ.

1: Initialize discriminator parameters w_0, generator parameters θ_0.
2: **for** $t_1 = 1, \ldots, n_g$ **do**
3: **for** $t_2 = 1, \ldots, n_d$ **do**
4: Sample $\{\mathbf{z}^{(i)}\}_{i=1}^m \sim p(\mathbf{z})$ a batch of prior samples.
5: Sample $\{\mathbf{x}^{(i)}\}_{i=1}^m \sim p_{data}(\mathbf{x})$ a batch of real data points.
6: For each i, $g_w(\mathbf{x}^{(i)}, \mathbf{z}^{(i)}) \leftarrow \nabla_w \left[f_w(\mathbf{x}^{(i)}) - f_w(g_\theta(\mathbf{z}^{(i)}))\right]$
7: $\bar{g}_w \leftarrow \frac{1}{m}(\sum_{i=1}^m g_w(\mathbf{x}^{(i)}, \mathbf{z}^{(i)}) + N(0, \sigma_n^2 c_g^2 I))$.
8: $w^{(t_2+1)} \leftarrow w^{(t_2)} + \alpha_d \cdot RMSProp(w^{(t_2)}, \bar{g}_w)$
9: $w^{(t_2+1)} \leftarrow clip(w^{(t_2+1)}, -c_p, c_p)$
10: **end for**
11: Sample $\{\mathbf{z}^{(i)}\}_{i=1}^m \sim p(\mathbf{z})$, another batch of prior samples.
12: $g_\theta \leftarrow -\nabla_\theta \frac{1}{m} \sum_{i=1}^m f_w(g_\theta(\mathbf{z}^{(i)}))$
13: $\theta^{(t_1+1)} \leftarrow \theta^{(t_1)} - \alpha_g \cdot RMSProp(\theta^{(t_1)}, g_\theta)$
14: **end for**
15: **return** θ.

Let (M, d) be a metric space for which every probability measure on M is a Radon measure (a so-called Radon space). For $p \geq 1$, let $P_p(M)$ denote the collection of all probability measures μ on M with finite p^{th} moment. Then, there exists some x_0 in M such that:

$$\int_M d(x, x_0)^p \, \mathrm{d}\mu(x) < +\infty.$$

The p^{th} **Wasserstein distance** between two probability measures μ and ν in $P_p(M)$ is defined as

$$W_p(\mu,\nu) := \left(\inf_{\gamma\in\Gamma(\mu,\nu)} \int_{M\times M} d(x,y)^p \, \mathrm{d}\gamma(x,y)\right)^{1/p},$$

where $\Gamma(\mu, \nu)$ denotes the collection of all measures on $M \times M$ with marginals μ and ν on the first and second factors respectively. (The set $\Gamma(\mu, \nu)$ is also called the set of all **couplings** of μ and ν.)

The above distance is usually denoted $W_p(\mu, \nu)$ (typically among authors who prefer the "Wasserstein" spelling) or $\ell_p(\mu, \nu)$ (typically among authors who prefer the "Vaserstein" spelling). The remainder of this article will use the W_p notation.

The Wasserstein metric may be equivalently defined by

$$W_p(\mu,\nu) = \left(\inf \mathbf{E}\left[d(X,Y)^p\right]\right)^{1/p},$$

where $\mathbf{E}[Z]$ denotes the expected value of a random variable Z and the infimum is taken over all joint distributions of the random variables X and Y with marginals μ and ν respectively.

Lipschitz Function

A function f such that

$$|f(x) - f(y)| \leq C|x - y|$$

for all x and y, where C is a constant independent of x and y, is called a Lipschitz function. For example, any function with a bounded first derivative must be Lipschitz.

an optimization algorithm that can adaptively adjust the learning rate according to the magnitude of gradients [18].

3.4 Privacy Guarantees of DPGAN

To show the DPGAN in Algorithm 1 indeed protects the differential privacy, we demonstrate that the parameters of generator θ (through discriminator parameters w) guarantee differential privacy with respect to the sample training points. Hence any generated data from G will not disclose the privacy of training points. Through the *moment accountant* mechanism, we can compute the final composition result ϵ. By treating parameters of discriminator $w^{(t_2+1)}$ (line 9 in Algorithm 1) as one point in the output space, it is easy to see that the procedure of updating w for fixed t_2 in any loop is just the algorithm $\mathcal{A}_p$ in definition 3.1. Here the input of $\mathcal{A}_p$ is real data and noise and the output is the updated w. So we have $\mathcal{A}_p(D) = M(aux, D)$ where aux is an auxiliary input, which in our algorithm refers to the previous parameters $w^{(t_2)}$. Hence the update of $w^{(t)}$ (line 3 to 10 in Algorithm 1) is an instance of adaptive composition. Together with definition 3.1, it is natural to define the following privacy loss at o:

Definition 3.2. (Privacy loss)

$$c(o; M, aux, \mathcal{D}, \mathcal{D}') \triangleq \log \frac{\mathbb{P}[M(aux, D) = o]}{\mathbb{P}[M(aux, \mathcal{D}') = o]},$$

which describes the difference between two distributions caused by changing data. The privacy loss random variable is given by $C(M, aux, \mathcal{D}, \mathcal{D}') = c(M(\mathcal{D}); M, aux, \mathcal{D}, \mathcal{D}')$, which is defined by evaluating the privacy loss at an outcome sampled from $M(\mathcal{D})$. Note that we assume the supports of 2 distributions associated with $M(aux, D)$ and $M(aux, \mathcal{D}')$ are generally the same so it is safe to evaluate them at same point o. This is a critical assumption since if there is an area s in support $M(aux, D)$ but not in $M(aux, \mathcal{D}')$, then evaluating $C(M, aux, \mathcal{D}, \mathcal{D}')$ in s will result in ∞ and violate the privacy. We define the log of the moment generating function of the privacy loss random variable and moments accountant as:

Definition 3.3. (Log moment generating function)

$$\alpha_M(\lambda; aux, \mathcal{D}, \mathcal{D}') \triangleq \log \mathbb{E}_{o\sim M(aux,D)}[exp(\lambda C(M, aux, \mathcal{D}, \mathcal{D}'))].$$

Definition 3.4. (Moments accountant)

$$\alpha_M(\lambda) \triangleq \max_{aux, \mathcal{D}, \mathcal{D}'} \alpha_M(\lambda; aux, \mathcal{D}, \mathcal{D}').$$

Moments accountant can be seen as the "worst situation" of the moment generating function. The definition of moments accountant enjoys good properties as mentioned in [1] (Theorem 2), where the composability property shows that the overall moments accountant can be easily bounded by the sum of moments accountant in each iteration, which brings about a result that privacy is proportional to iterations. The tail bound can also be applied in the privacy guarantee (Theorem 1 in same paper). We will use this theorem to deduce our own result. Comparing with strong composition theorem [14], moments accountant saves a factor of $\sqrt{\log(n_g/\delta)}$. According to the definition 3.1, for a large iteration n_g, this is a significant improvement.

In order to use moments accountant we need $g_w(\mathbf{x}^{(i)}, \mathbf{z}^{(i)})$ to be bounded (by clipping the norm in Algorithm 1 in [1]) and add noise according to this bound. We do not clip the norm of $g_w(\mathbf{x}^{(i)}, \mathbf{z}^{(i)})$, instead we show that by only clipping on w can we automatically guarantee a bound of the norm of $g_w(\mathbf{x}^{(i)}, \mathbf{z}^{(i)})$.

Lemma 3.5. *Under the condition of Alg. 1, assume that the activation function of the discriminator has a bounded range and bounded derivatives everywhere: $\sigma(\cdot) \le B_\sigma$ and $\sigma'(\cdot) \le B_{\sigma'}$, and every data point $\mathbf{x}$ satisfies $\|\mathbf{x}\| \le B_x$, then $\|g_w(\mathbf{x}^{(i)}, \mathbf{z}^{(i)})\| \le c_g$ for some constant c_g.*

Proof. Without loss of generality, we assume f_w is implemented using a fully connected network. Let H be the number of layers except input layer. Let $\mathbf{W}^{(l)}$ be the l-th weight matrix ($l = 1, \ldots, H$) whose element $\mathbf{W}^{(l)}_{ij}$ is the weight connecting j-th node in layer $l-1$ to i-th node in layer l. Let $\mathbf{D}^{(l)}$ be the diagonal Jacobian of nonlinearities of l-th layer. We thus have:

$$\mathbf{D}^{(l)}_{ij} = \begin{cases} \sigma'(\mathbf{w}^{(l)}_{i,:}\sigma(\mathbf{z}^{(l-1)})) & \text{if} \quad i = j \\ 0 & \text{if} \quad i \neq j \end{cases} \tag{4}$$

where $\mathbf{w}^{(l)}_{i,:}$ is the ith row of $W^{(l)}$ and $\sigma(\mathbf{z}^{(l-1)})$ is the output of the $l-1$-th layer. The following fact is well known from the back-propagation algorithm on a fully connected network:

$$\boldsymbol{\delta}^{(H)} = \nabla_{\mathbf{a}} C \odot \sigma'(\mathbf{z}^{(H)}), \tag{5}$$

$$\boldsymbol{\delta}^{(l)} = ((\mathbf{W}^{(l+1)})^T \boldsymbol{\delta}^{(l+1)}) \odot \sigma'(\mathbf{z}^{(l)}), \tag{6}$$

$$\frac{\partial C}{\partial \mathbf{W}^{(l)}_{jk}} = \mathbf{a}^{(l-1)}_k \boldsymbol{\delta}^{(l)}_j, \tag{7}$$

where C is the cost function, $\mathbf{z}^{(l)}$, $\mathbf{a}^{(l)}$ and $\boldsymbol{\delta}^{(l)}$ are the input, output and error vector of layer l, respectively. From 7 we have for $l = 2, \ldots, H$:

$$\begin{aligned} \frac{\partial C}{\partial \mathbf{W}^{(l)}} &= \boldsymbol{\delta}^{(l)}(\mathbf{a}^{(l-1)})^T \\ &= (\mathbf{D}^{(l)}(\mathbf{W}^{(l+1)})^T \boldsymbol{\delta}^{(l+1)})(\mathbf{a}^{(l-1)})^T \\ &= (\mathbf{D}^{(l)}(\mathbf{W}^{(l+1)})^T \ldots \mathbf{D}^{(H-1)}(\mathbf{W}^{(H)})^T \boldsymbol{\delta}^{(H)}) \\ &\quad * (\mathbf{a}^{(l-1)})^T \\ &= (\mathbf{D}^{(l)}(\mathbf{W}^{(l+1)})^T \ldots \mathbf{D}^{(H-1)}(\mathbf{W}^{(H)})^T) \\ &\quad * (\mathbf{a}^{(l-1)})^T \sigma'(z^{(H)}). \end{aligned} \tag{8}$$

Take $\frac{\partial C}{\partial \mathbf{W}^{(l_0)}}$ as an example:

$$[\mathbf{D}^{(l)}(\mathbf{W}^{(l+1)})^T]_{ij} \le c_p B_{\sigma'}$$

$$[\mathbf{D}^{(l)}(\mathbf{W}^{(l+1)})^T \mathbf{D}^{(l+1)}(\mathbf{W}^{(l+2)})^T]_{ij} \le (c_p B_{\sigma'})^2 m_{l+1},$$

where we assume that $c_p \le \frac{1}{m_{l+1} B_{\sigma'}}$. Here m_{l+1} is the number of nodes in the $l+1$th layer. And thus we have:

$$\left[\prod_{l=l_0}^{H-1} \mathbf{D}^{(l)}(\mathbf{W}^{(l+1)})^T\right]_{ij} \le (c_p B_{\sigma'})^{H-l_0} \prod_{l=l_0}^{H-2} m_{l+1}. \tag{9}$$

Because of the assumption that $\sigma(\cdot) \le B_\sigma$, we have $a^{(l-1)}_j \le B_\sigma$. Combining it with 8, we have $[\frac{\partial C}{\partial \mathbf{W}^{(l)}}]_{ij} \le c_p B_\sigma B^2_{\sigma'}$ and therefore

we have:

$$\begin{aligned}\|g_w(\mathbf{x}^{(i)}, \mathbf{z}^{(i)})\| &= \left\|\nabla_w\left(f_w(\mathbf{x}^{(i)}) - f_w(g_\theta(\mathbf{z}^{(i)}))\right)\right\| \\ &\le 2\left\|\nabla_w f_w(\mathbf{x}^{(i)})\right\| = 2\sum_l \sum_{ij}\left[\frac{\partial C}{\partial \mathbf{W}^{(l)}}\right]_{ij} \\ &\le 2c_p B_\sigma B_{\sigma'}^2 \sum_{k=1}^{H-1} m_k m_{k+1} = c_g,\end{aligned}$$

where the boundness of $g_\theta(\mathbf{z}^{(i)})$ comes from the choice of sigmoid activation in the last layer of generator. Note that when computing c_g, we need to take into consideration the dropout rate, weight sparsity, connection percentage of convolutional nets, and other factors. □

Remark 1. *Note that activation functions like ReLU (and its variants) and Softplus have unbounded B_σ. This will not affect our result because both the data points and weights are bounded, which guarantees that the output of each node in each layer is bounded. The boundness of data comes from a common fact that each data element has a bounded range.*

We have the following lemma which guarantees DP for discriminator training procedure.

Lemma 1. *Given the sampling probability $q = \frac{m}{M}$, the number of discriminator iterations in each inner loop n_d and privacy violation δ, for any positive ϵ, the parameters of discriminator guarantee (ϵ, δ)-differential privacy with respect to all the data points used in that outer loop (fix t_1) if we choose:*

$$\sigma_n = 2q\sqrt{n_d \log(\frac{1}{\delta})}\Big/\epsilon. \tag{10}$$

Proof. The DP guarantee for the discriminator training procedure follows from the intermediate result [1] (Theorem 1). We need to find an explicit relation between σ_n and ϵ, i.e., how much noise standard deviation σ_n we need to impose on the gradient so that we can guarantee a privacy level ϵ, with small violation δ. Combine inequality $n_d q^2 \lambda^2/\sigma^2 \le \lambda\epsilon/2$ and inequality $e^{-\lambda\epsilon/2} \le \delta$ in Theorem 1, we can get the result by letting the equality hold. □

Lemma 1 quantifies the relation between noise level σ_n and privacy level ϵ. It shows that for fixed perturbation σ_n on gradient, larger q leads to less privacy guarantee (larger q). This is indeed true since when more data are involved in computing discriminator w, less privacy is assigned on each of them. Also, more iterations (n_d) leads to less privacy because the observer gives more information (specifically, more accurate gradient) for data. This requires us to choose the parameters carefully in order to have a reasonable privacy level. Finally we have the following theorem as the privacy guarantee of the parameters of the generator:

Theorem 1. *The output of generator learned in Algorithm 1 guarantees (ϵ, δ)-differential privacy.*

The privacy guarantee a direct consequence from Lemma 1 followed by the post-processing property of differential privacy [12].

4 EXPERIMENT

In this section, we will present extensive experiments to investigate how the noise will affect the effectiveness of generative network on two benchmark datasets (MNIST and MIMIC-III)[1]. There are several notable findings that are worth highlighting. The Wasserstein distance converges as the training procedure goes on and exhibits fluctuation in the late stage in the case of privacy. This fluctuation correlates well with the quality of generated data and reflects the privacy level. In addition, our framework can be generalized under various network structures and applied on many benchmark datasets.

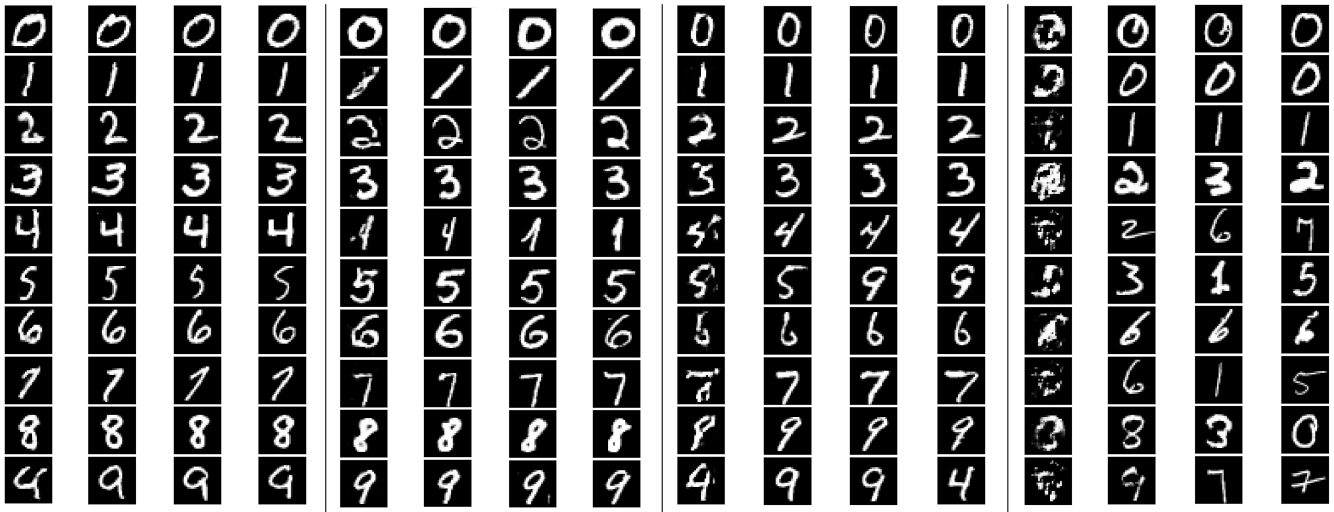
4.1 Relationship between Privacy Level and Generation Performance

We conduct experiments on MNIST dataset to illustrate the relationship between the privacy level and the quality of output images from the generator.

In this experiment, we set both the learning rate of discriminator α_d and generator α_g to be 5.0×10^{-5}. The parameter clip constant c_p is 1.0×10^{-2} such that the weights of discriminator will be clipped back to $[-c_p, +c_p]$. We use MNIST's training data with data size $M = 6 \times 10^4$ and the batch size m is set to be 64. Hence the sample probability q is $\frac{m}{M} = \frac{64}{6\times 10^4} \approx 1.1 \times 10^{-3}$. The noise scale δ is 10^{-5}, and the number of iterations on discriminator (n_d) and generator (n_g) are 5 and 5×10^5, respectively. Since we use leaky ReLU as the activation function on discriminator network and ReLU on generative network, we have $B_{\sigma'} \le 1$, where $B_{\sigma'}$ is the bound on the derivative of the activation function. Dimension of $\mathbf{z}$ is 100 and every coordinate is within $[-1, 1]$. We adopt similar network structure of DCGAN [29] with noise generation and inference parts to protect data privacy, of which the effectiveness has been verified in [2]. To impose a certain level of noise on the network, we choose Gaussian noise with zero mean (hence no bias) and multiple values of standard deviation. Gaussian distribution is widely used in privacy-preserving algorithm (see Gaussian mechanism and its variants in [12]) and usually results in (ϵ, δ)-differential privacy. We add L_2-regularization on the weights of generator and discriminator, which has little impact on our bound in Lemma 3.5.

In the first experiment we investigate how the change in noise level affects the image quality. Four groups of the generated images are plotted and shown in Figure 1, corresponding to 4 different ϵ values. In each group, the leftmost column shows the generated images for a certain ϵ value. The rest three columns are the corresponding nearest neighbor images from the training set, which demonstrates that the distortions of images are caused by noise instead of bad training images. The distance between training images and generated images is Euclidean norm. Comparing the generated images with their nearest neighbors, it is clear to see that our model is not simply to memorize the training data but to be capable of generating photographic samples with unique details. As mentioned in [16], these images indeed come from actual samples of the model distributions, rather than the conditional means given samples of hidden units. Most importantly, the generated images of each group in Figure 1 shows that, the larger the variance of

[1]Code and experiment scripts are available at: https://github.com/illidanlab/dpgan



(a) ϵ=∞ (b) ϵ = 29.0 (c) ϵ = 14.0 (d) ϵ = 9.6

Figure 1: Generated images with four different ϵ on MNIST dataset are plotted in leftmost column in each group. Three nearest neighbors of generated images are plotted to illustrate the generated data is not memorizing the real data and the privacy is preserved. We can see that the images get more blurred as more noise is added.

noise is, the blurrier the generated images would be, when all other conditions are the same. In the sense of differential privacy, any observer who gets the generated images can hardly know whether a data point is involved in the training procedure or not, as elaborated in Theorem 1 and illustrated by the generated images in Figure 1. The observer has no way to reconstruct the training images in such case and hence the privacy of data is protected. This demonstrates that our model successfully addresses the privacy issue mentioned previously. The noise level (ϵ) is recommended to be tuned in a large range to guarantee good quality of generated images. In addition, it can be seen from the results that our method does not suffer from mode collapse or gradient vanishing, which is an advantage that is inherited from the WGAN network structure.

4.2 Relationship between Privacy Level and the Convergence of Network.

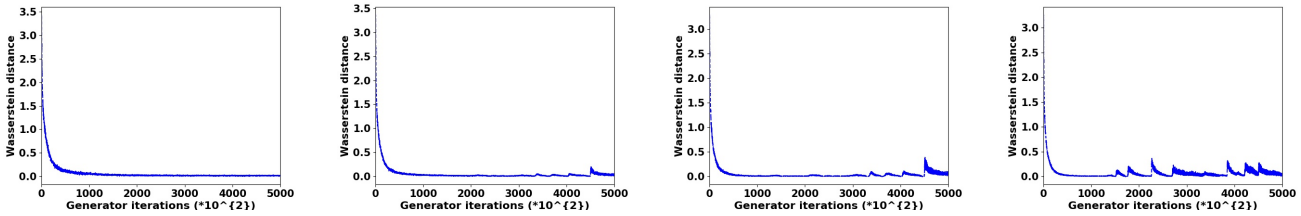
In the second experiment, we plot the Wasserstein distance for every 100 generator iterations. The result shows that the Wasserstein distance decreases during training and converges in the end, which also correlates well with the visual quality of the generated samples [2]. The corresponding results are shown in Figure 2. As expected, the Wasserstein distance decreases as the training procedure goes on and converges, which is the result of joint effect of discriminator and generator.

Despite the fluctuation caused by the min-max training itself, we can also observe that, a smaller ϵ (hence larger noise) leads to more frequent fluctuation and larger variance, which is especially clear in the latter half of the curves. This conforms to the common intuition that more noise will results in a more blurry image, which is also consistent with the results of the previous experiment. One interesting phenomena is that the peaks often appear after the convergence of Wasserstein distance. More evidences show that this might be caused by clipping the weight. The reason is that clipping weights is equivalent to adjusting the gradient g_i in directions whose the corresponding gradient w_i magnitude is too large ($|w_i| > c_p$). Different from gradient descent step (even with noise) which always changes the weight towards the optimal solution, the effect of such adjustment is hard to predict and hence might cause instability. This is especially clear when network converges. However, these peaks can be quickly eliminated during the training procedure and the network may maintain a numerical stability. This is due to the fact that the generator is in convergence stage, which is one of the advantages of adversarial networks. Hence our system does not suffer from divergence problem. Again, this experiment demonstrates the most important property of a learning system with differential privacy consideration: there exists a trade-off between learning performance and privacy level.

4.3 Classification on MNIST Data

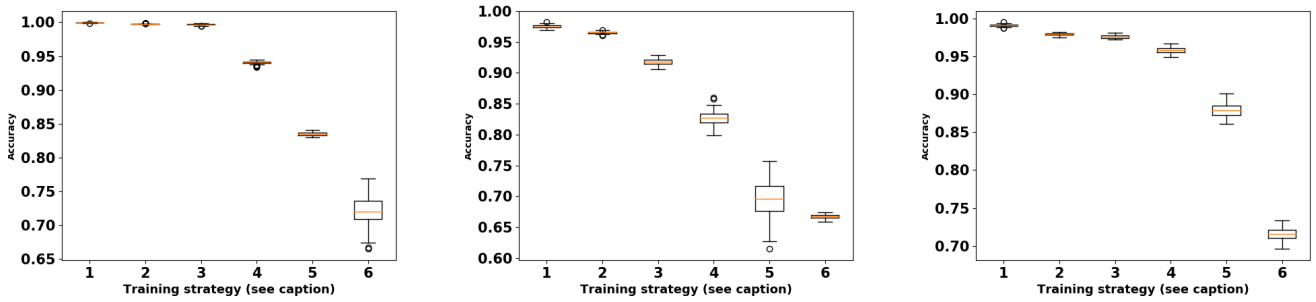
In this section we conduct a binary classification task to further evaluate the quality of the generated MNIST data. Here we use the same settings as in subsection 1 4.1. Take a pair of digits 0 and 1 as an example, we generate 0s and 1s from their own training samples (use all samples) separately, with different ϵ values. For each digit, we generate equal number of data as training samples. Then for fixed ϵ (and for training set), we randomly select 4000 samples from generated data (contains 2000 for both 0 an 1), build classifiers on them and test on MNIST's testing set. Then we repeat this for 100 times and show the accuracy (Figure 3) on testing set with classifiers built from training data and generated ones with different standard deviations. Finally we run the same procedures for digit pairs 23 and 45, as well.

The results are shown in Figure 3. Despite the fact that smaller noise makes the accuracy higher (better generated quality), the variance of plot also decreases generally. The generate quality is little affected below some threshold (for example, somewhere between 3.0 and 11.0 for digit 01). Thus it is recommended to choose an ϵ larger than that threshold (add less noise) so that the generated



(a) ϵ=∞ (b) $\epsilon = 29.0$ (c) $\epsilon = 14.0$ (d) $\epsilon = 9.6$

Figure 2: Wasserstein distance for different privacy levels when applying DPGAN on MINST. We can see that the curves converge and exhibit more fluctuations as more noise is added.



(a) Digit 0 and 1 (b) Digit 2 and 3 (c) Digit 4 and 5

Figure 3: Binary classification task on MNIST database with different training strategies. From left to right we use training data, generated data without noise, generated data with $\epsilon = 11.5, 3.2, 0.96, 0.72$. We can see that as less noise is added, the accuracy of classifier build on generated data gets higher, which indicates that the generated data has better quality.

data will not be affected much. Note that a threshold between 3.0 and 11.0 is quite promising privacy level. Comparing among three figures, digit pairs 01 performs better than the rest two, which is due to the reason that the shapes of digit 0 and 1 make them easy to be separated. This experiment use classification task to demonstrate the trade-off between learning performance and privacy level.

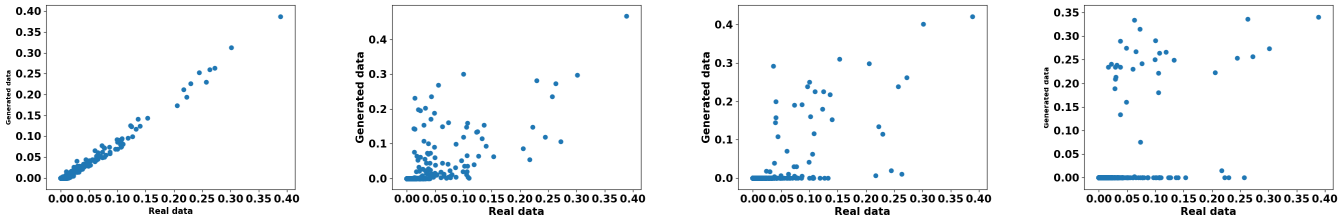
4.4 Electronic Health Records

In this section we apply DPGAN to generate Electronic Health Records (EHR) while the privacy of patients is needed to be protected. EHR is one of the most important information sources from which we can learn the genetics and biological characteristics of certain population. However, the access to EHR requires administrative permission in consideration of the privacy protection, which is very inconvenient to the research community. Choi *et al.* proposed medGAN [8], which can successfully generate EHR based on MIMIC-III critical care datasets [15, 20], while the sensitive information is not guaranteed to be protected. MIMIC-III is a well-known public EHR database consisting of the medical records of 46,520 intensive care unit (ICU) patients over 11 years old. In our experiments we use the extracted ICD9 codes[2] only, and group them using their first 3 digits. For each patient (1 out of 46520) in each admission to one hospital, we record what kind of diseases this patient has and make it into a hot vector. For example, patient A has been diagnosed with 3 diseases (with ICD9 codes 9, 42 and 146, respectively) in one admission and we use a vector to represent the patient A's visit, where the vector has digit 1 in position 9, 42 and 146, and has digit 0 in the rest positions. Then we add up all vectors (different admissions and different hospitals) of a certain patient and hence each patient has one and only one vector $x \in Z_+^{|C|}$ with $|C| = 1071$. We then binarize the data, where all non-zero elements are transferred to 1. These vectors serve as summary of historical record of each patient's health condition and can be considered as a feature for patients. Together we can also extract useful information from these vectors. Notice that we remove the patient data with missing values before feeding them into network.

Similar to previous experiments, we set the learning rates of both the discriminator α_d and generator α_g to be 5.0×10^{-4}. The parameter clip constant c_p is 0.1 and n_d is equal to 2. Also we have $m = 500$, $M = 46520$ for MIMIC-III dataset and $q = \frac{500}{46520} \approx 1.1 \times 10^{-2}$. The δ is set as 10^{-5}. We adopt the same network structure as in [8]. After generating the data, we set a threshold at 0.5 to convert the generated data matrix from continuous domain to binary domain. Since the quality of EHR cannot be observed as images directly, we adopt the dimensional wise probability (DWP) [8] as a quantitative measurement for the quality of the generated data, which is to check whether the model has learned each dimensionâĂŹs distribution correctly. Through DWP we study how the performance of DPGAN varies with the changing of noise level.

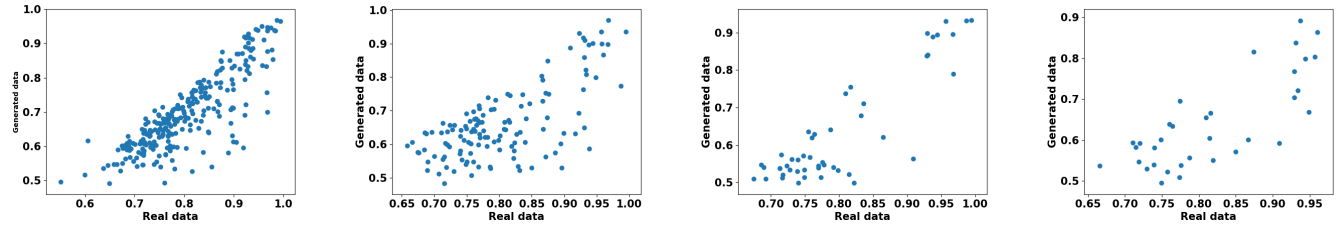
The results are shown in Figure 4 for different noise magnitudes. Each point in the figure is a pair of float numbers that represents Bernoulli success probability of real data (x-axis), and generated data (y-axis) of one dimension (corresponding to one disease). The Bernoulli success probability (of each dimension) is the sample mean of that dimension (Maximum likelihood estimation of independent of Bernoulli trials), which is a portion of 1 in that column. This characterizes the rareness of that disease and hence together

[2]International Statistical Classification of Diseases and Related Health Problems, 9th edition



(a) ϵ=∞ (b) ϵ=2.31×10^2 (c) ϵ=1.39×10^2 (d) ϵ=96.5

Figure 4: DWP evaluation on MIMIC-III database with different ϵ values (1070 points). We can see that as more noise is added, the distribution of generated data in each dimension becomes more deviated from the real training data.



(a) ϵ=∞ (b) ϵ=2.31×10^2 (c) ϵ=1.39×10^2 (d) ϵ=96.5

Figure 5: Dimension-wise prediction evaluation on MIMIC-III database with different ϵ values. We can see that as more noise is added, AUC value of classifier build from generated data gets lower and the data gets sparser.

reflects the distribution of diseases among population, which is a very important statistical characteristic and can be frequently queried. Hence there is a must to protect the people who provide this distribution by adding noise. Despite the theoretical result in 1, we can understand the privacy protection in a intuitive way: on one hand, if no noise added (Figure 4 (a)), changing database by adding one person may change the frequency of certain disease in some extent. This change is especially significant when the number of people in database is small or a group of people is changed (See "group privacy" in [13]). By looking at this change, an observer may make some conclusions and harm the interest of anyone who involves in the database. For example, adding a group of people may enlarge the frequency of certain disease, if this disease is highly related with the quality of life or it is some rare disease, health insurance company may raise people's premiums. On the other hand, if there is noise added (Figure 4 (b) to (d)), observer is not sure what is the effect by adding this person (or these people) because the output is uncertain (associated with a noise distribution) and the generated data will hardly leak any patient's privacy information. This uncertainty gets larger when more noise is added, which can be seen from Figure 4. On the whole, it can be seen from this experiment that our model indeed provides protection in the sense of differential privacy on the medical data, and solves the problem we mentioned in abstract.

Note that the rareness of diseases are also well protected due to the perturbation of noise. Assuming that there is a public-available generated EHR data that are generated based on the EHR of a certain population, the insurance company may raise the insurance premium for those who get rare diseases, based on the statistical information inferred from generated EHR data. Since DPGAN may change the rareness of diseases, the issuarance company cannot get this type of information accurately from our generated data, thus the interest of this group of people is guaranteed.

The results also indicate how well the generative model captures training data's distribution. In Figure 4 (a), most of the points are concentrated around line $y = x$, which indicates that our model captures each dimension's distribution correctly. It can also be seen from Figure 4 (left to right) that a large variance of noise makes more points deviated from line $y = x$. This means that for one disease, the rareness of generated data becomes more different from real data, which also indicates the quality of generated data is degraded. This phenomenon matches our intuition that applying a higher level of noise often leads to a worse distribution approximation, which is also consistent with evidence in Figure 2 (a) in [8].

4.5 Classification on EHR Data

Continue with previous sub-section, we use dimension-wise prediction (DWpre) [8] to evaluate how well the generative model recovers the relationship among the dimensions of the data. The basic idea of DWpre is to select the same column from training set and generated set as target and set the rest columns as feature. Then we build logistic regression classifiers on both of them and test on testing set. One assumption here is that a closer performance of two classifiers indicates better quality of the generated set. Due to the highly unbalanced testing data (0 is dominated), we use AUC as the measurement here.

The results are shown in Figure 5. Despite the fact that in most cases, classifiers trained from real data perform better than classifiers trained from generated data, the AUC values of generated data decrease as the decreasing of the ϵ (more noise added). This is due to the reason that noise perturbs the training of discriminator and affects the generator indirectly, which leads to the deviation of output distribution from the real one and can results in poor testing

performance. It can also be seen that there is not much decreasing in the performance, which is one of the advantages of our model. The points get sparser as more noise is added, which reflects another impact of noise on data. This is due to the reason that we use logistic regression to perform binary classification, which does not allow uni-label column. The sparse column are widely exists in original data and it is harder for the generative model to capture the sparsity of certain column of original data if there is more perturbation. More columns are learned as all-zero and discarded when selected as target in classification task. In summary, higher privacy results in less ability for generative model to capture the inter- dimensional relationship. Also our framework successfully addresses the issue in differential privacy system that adding noise will cause too much decreasing in system performance.

5 CONCLUSION

In this paper, we proposed a privacy preserving generative adversarial network (DPGAN) that preserves privacy of the training data in a differentially private sense. Our algorithm is proved rigorously to guarantee the (ϵ, δ)-differential privacy. We conducted two experiments to show that our algorithm can generate data points with good quality and converges under the condition of both noisy and limitation of training data, with meaningful learning curves useful for tunning hyperparameters. For future work we will consider reducing the privacy budget by trying different ways of clipping, and also tighten the utility bound.

ACKNOWLEDGMENTS

This research is supported in part by National Science Foundation under Grant IIS-1565596 (JZ), IIS-1615597 (JZ), IIS-1650723 (FW) and IIS-1716432 (FW). and the Office of Naval Research under grant number N00014-14-1-0631 (JZ) and N00014-17-1-2265 (JZ).

REFERENCES

[1] Martín Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 308–318.

[2] Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein gan. *arXiv preprint arXiv:1701.07875* (2017).

[3] Brett K Beaulieu-Jones, Zhiwei Steven Wu, Chris Williams, and Casey S Greene. 2017. Privacy-preserving generative deep neural networks support clinical data sharing. *bioRxiv* (2017), 159756.

[4] David Berthelot, Tom Schumm, and Luke Metz. 2017. Began: Boundary equilibrium generative adversarial networks. *arXiv preprint arXiv:1703.10717* (2017).

[5] Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. 2015. Importance weighted autoencoders. *arXiv preprint arXiv:1509.00519* (2015).

[6] Kamalika Chaudhuri and Claire Monteleoni. 2009. Privacy-preserving logistic regression. In *Advances in Neural Information Processing Systems*. 289–296.

[7] Kamalika Chaudhuri, Claire Monteleoni, and Anand D Sarwate. 2011. Differentially private empirical risk minimization. *The Journal of Machine Learning Research* 12 (2011), 1069–1109.

[8] Edward Choi, Siddharth Biswal, Bradley Malin, Jon Duke, Walter F Stewart, and Jimeng Sun. 2017. Generating Multi-label Discrete Electronic Health Records using Generative Adversarial Networks. *arXiv preprint arXiv:1703.06490* (2017).

[9] Cynthia Dwork. 2006. Differential privacy. In *Automata, languages and programming*. Springer, 1–12.

[10] Cynthia Dwork. 2011. Differential privacy. In *Encyclopedia of Cryptography and Security*. Springer, 338–340.

[11] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. 2006. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography*. Springer, 265–284.

[12] Cynthia Dwork and Aaron Roth. 2013. The algorithmic foundations of differential privacy. *Theoretical Computer Science* 9, 3-4 (2013), 211–407.

[13] Cynthia Dwork, Aaron Roth, et al. 2014. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science* 9, 3–4 (2014), 211–407.

[14] Cynthia Dwork, Guy N Rothblum, and Salil Vadhan. 2010. Boosting and differential privacy. In *Foundations of Computer Science (FOCS), 2010 51st Annual IEEE Symposium on*. IEEE, 51–60.

[15] Ary L Goldberger et al. 2000. Components of a new research resource for complex physiologic signals, physiobank, physiotoolkit, and physionet, american heart association journals. *Circulation* 101, 23 (2000), 1–9.

[16] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*. 2672–2680.

[17] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. 2017. Improved training of wasserstein gans. *arXiv preprint arXiv:1704.00028* (2017).

[18] Geoffrey Hinton, NiRsh Srivastava, and Kevin Swersky. [n. d.]. Neural Networks for Machine Learning Lecture 6a Overview of mini–batch gradient descent. ([n. d.]).

[19] Briland Hitaj, Giuseppe Ateniese, and Fernando Perez-Cruz. 2017. Deep Models Under the GAN: Information Leakage from Collaborative Deep Learning. *arXiv preprint arXiv:1702.07464* (2017).

[20] Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. MIMIC-III, a freely accessible critical care database. *Scientific data* 3 (2016).

[21] Yujia Li, Kevin Swersky, and Rich Zemel. 2015. Generative moment matching networks. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*. 1718–1727.

[22] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. 2015. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644* (2015).

[23] Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. 2017. Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks. *arXiv preprint arXiv:1701.04722* (2017).

[24] Olof Mogren. 2016. C-RNN-GAN: Continuous recurrent neural networks with adversarial training. *arXiv preprint arXiv:1611.09904* (2016).

[25] Nicolas Papernot, Martín Abadi, Úlfar Erlingsson, Ian Goodfellow, and Kunal Talwar. 2017. Semi-supervised knowledge transfer for deep learning from private training data. *stat* 1050 (2017), 3.

[26] NhatHai Phan, Xintao Wu, and Dejing Dou. 2017. Preserving differential privacy in convolutional deep belief networks. *Machine Learning* 106, 9-10 (2017), 1681–1704.

[27] NhatHai Phan, Xintao Wu, Han Hu, and Dejing Dou. 2017. Adaptive Laplace Mechanism: Differential Privacy Preservation in Deep Learning. *arXiv preprint arXiv:1709.05750* (2017).

[28] Guo-Jun Qi. 2017. Loss-sensitive generative adversarial networks on lipschitz densities. *arXiv preprint arXiv:1701.06264* (2017).

[29] Alec Radford, Luke Metz, and Soumith Chintala. 2015. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434* (2015).

[30] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082* (2014).

[31] Masaki Saito and Eiichi Matsumoto. 2016. Temporal generative adversarial nets. *arXiv preprint arXiv:1611.06624* (2016).

[32] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. 2016. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*. 2234–2242.

[33] Reza Shokri and Vitaly Shmatikov. 2015. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*. ACM, 1310–1321.

[34] Shuang Song, Kamalika Chaudhuri, and Anand Sarwate. 2015. Learning from data with heterogeneous noise using sgd. In *Artificial Intelligence and Statistics*. 894–902.

[35] Shuang Song, Kamalika Chaudhuri, and Anand D Sarwate. 2013. Stochastic gradient descent with differentially private updates. In *Global Conference on Signal and Information Processing (GlobalSIP), 2013 IEEE*. IEEE, 245–248.

[36] Junbo Zhao, Michael Mathieu, and Yann LeCun. 2016. Energy-based generative adversarial network. *arXiv preprint arXiv:1609.03126* (2016).

[37] Tianqing Zhu, Gang Li, Wanlei Zhou, and S Yu Philip. 2017. Differentially private data publishing and analysis: a survey. *IEEE Transactions on Knowledge and Data Engineering* 29, 8 (2017), 1619–1638.