imdpGAN: Generating Private and Specific Data with Generative Adversarial Networks

Anonymous Authors

***Abstract*—Generative Adversarial Network (GAN) and its variants have shown promising results in generating synthetic data. However, the issues with GANs are: (i) the learning happens around the training samples and the model often ends up remembering them, consequently, compromising the privacy of individual samples - this becomes a major concern when GANs are applied to training data including personally identifiable information, (ii) the randomness in generated data - there is no control over the specificity of generated samples. To address these issues, we propose imdpGAN - an information maximizing differentially private Generative Adversarial Network. It is an end-to-end framework that simultaneously achieves privacy protection and learns latent representations. With experiments on MNIST dataset, we show that imdpGAN preserves the privacy of the individual data point, and learns latent codes to control the specificity of the generated samples. We perform binary classification on digit pairs to show the utility versus privacy trade-off. The classification accuracy decreases as we increase privacy levels in the framework. We also experimentally show that the training process of imdpGAN is stable but experience a 10-fold time increase as compared with other GAN frameworks. Finally, we extend imdpGAN framework to CelebA dataset to show how the privacy and learned representations can be used to control the specificity of the output.**

***Index Terms*—privacy-preserving learning, learning latent representations, generative adversarial networks**

I. Introduction

The world of today is moving towards more personalized hardware and software, collecting sensitive information with multiple Personally Identifiable Information (PII) attributes, especially in domains like healthcare and Internet-of-Things (IoT). Often times, deep learning techniques are used to solve problems like detecting cancer patterns [1], diabetic retinopathy [2], and so on. But deep learning typically needs huge amount of data to achieve promising performance. However, in domains like healthcare and IoT (with a lot of PII attributes), it is impossible to get as much data as we want. Also, such models learn finer details in training data and are shown to compromise privacy of individuals. One such example is successful recovery of individual samples from the training set, by using hill climbing on output probabilities [3]. Therefore, enforcing privacy while using deep learning techniques to analyze such data has become an absolute necessity. In short there are two challenges: the availability of a huge amount of data and protecting the privacy of individual users.

Generative models have mitigated the data scarcity issue by successfully generating patient records, sensor data, medical records, tabular data [4]–[8]. Using the combination of game theory and deep learning, GANs and its many other variants, have demonstrated promising performance in modeling the underlying data distribution [9]. These generative models can generate high quality "fake" samples that are hard to differentiate from the real ones [10]–[12]. Ideally, we can generate these "fake" data samples to fit our needs and conduct the desired analysis without privacy implications. Although, the generation process is random and we cannot implicitly control the variation in type or style of data we want to generate.

Privacy is being enforced on sensitive data using several anonymization techniques. Some examples include k - anonymity [13], l-diversity [14], t-closeness [15], which are effective but vulnerable to de-anonymization attacks [16]. Since, these techniques do not solve the data scarcity issue, researchers are trying to introduce privacy preservation in generative models [17]. The generation of "fake" samples is not self-sufficient and is prone to disclosure of private information about the individual training samples. The adversarial training procedure with high model complexity often leads to learning a distribution that just copies the training samples. Repeated sampling from such distributions increases the chance of recovering the training samples, hence, compromising the privacy of the data. [18] demonstrated an inference attack that uses generated samples to recreate the training samples.

Our contribution. With the above considerations, we try to learn meaningful latent representations of variation, known as latent codes, to have control over the specificity of the generator output and use a private training procedure to preserve the privacy of the individual training samples. Therefore, in this paper, we present an amalgamation of techniques from Information Maximizing Generative Adversarial Network, (used to learn interpretable latent representations in an unsupervised manner) and Differentially Private Generative Adversarial Network (used to preserve privacy of the training samples). The models are built using the machine learning framework Pytorch [19].

We propose imdpGAN, a unified framework to:

1) *Protect privacy of training samples.* Protecting privacy in images simply means that one will not be able to recognize what is there in the image, i.e., the generator will generate blurry images as we increase privacy giving rise to a privacy versus accuracy trade-off. To demonstrate the trade-off, we train a binary classifier on digit pairs and find accuracy on corresponding test samples (discussed in Section IV-B2). Results show that as we increase privacy, the accuracy of binary classifier decreases.
2) *Control specificity of generated samples.* There are two kinds of variations in a data set: discrete and continuous.

Discrete variations are represented by different classes. For example, MNIST dataset has 10 classes representing one digit per class. Changing one class to another is a discrete variation. The dataset has digits positioned at varying angles and having different widths representing continuous variation. We learn tunable latent codes to control both types of variations.

We evaluate our proposed approach on MNIST dataset and extend the imdpGAN framework to complex CelebA [20] dataset. Results show that imdpGAN preserves privacy and learns meaningful latent codes, which are varied to show class and style variations while generating new images. Although, the classification accuracy decreases as we increase privacy.

As privacy concerns are rising up there are multiple use cases of our framework. For example, popular face recognition systems (FRS) claim that they store only a representation of users' faces and not the actual image[1]. However, while operating they require a complete face image as input to auhenticate an user. The proposed framework, imdpGAN, can be used to create anonymized face images that are closer to the real face representations by learning meaningful latent codes while generating private faces to preserve user's privacy. Although, there will be a trade-off between the accuracy of the FRS and privacy of the face image, which can further be adjusted by tunable parameters.

This paper proceeds as follows: we start with a review of the background to explain the relevant work done on differential privacy for deep learning and learning latent representations in Section II. In Section III, after defining the privacy model and mutual information maximization used to learn latent codes, we introduce the imdpGAN framework followed by the differential privacy guarantees. In Section IV, experiments on MNIST dataset are described, followed by the extended experiments on the CelebA dataset. Then we discuss the shortcomings of imdpGAN framework in Section V and finally conlcude in Section VI.

II. Background

In this section, we provide a brief literature review of relevant topics: differential privacy for deep learning and learning disentangled representations.

A. Differential Privacy for Deep Learning

In the works that study differential privacy in deep learning, [21] change the model's training algorithm to make it private by clipping and adding noise to the gradients. Authors also propose a privacy accounting technique and introduce a moments accountant that computes the privacy costs. In [22], authors use differential privacy with a parallel and asynchronous training procedure for a multi-party privacy-preserving neural network. It involves transmitting local parameters between server and local task, which has a high risk of information leakage. [23] models a private convolutional deep belief network by adding noise on its objective functions and an extra softmax layer. [17] leverages the moments accountant and the private training procedure from [21] to train a differentially private generator. Authors add noise to the training procedure and avoid a distributed framework to prevent any information leaks. Advantages of DPGAN's techniques over other methods made them a salient choice for privacy preservation in the proposed framework [17].

B. Learning Latent Representations

Learning latent representations in a supervised, unsupervised, and semi-supervised manner is attempted by a lot of studies. Works that use supervision (labeled data): bilinear models [24] to separate style and content; multi-view perceptron [25] to separate face identity from the viewpoint, train a subset of representation to match some supplied label using supervised learning. Then there are semi-supervised methods developed to eliminate the need for labels of variations. To disentangle representations, [26] proposes a higher-order Boltzmann machine, which uses a clamping technique. [27] uses the clamping idea with variational autoencoders (VAEs) to learn codes that can represent pose and light in 3D rendered images. The model proposed by [28] can learn a representation that supports basic linear algebra on code space using GANs [9]. InfoGAN [29] learns disentangled representations with no supervision of any kind. Unlike hossRBM [30], which can learn only the discrete latent factors and has high complexity, InfoGAN can learn both discrete and continuous latent representations and can scale to complicated data. We use the techniques from InfoGAN to learn factors of variation, called the latent codes, to control specificity in the proposed framework as it is easy to incorporate along with the privacy framework.

III. Methodology

We unify the techniques to stabilize the GAN training [31] with techniques to make a differentially private generator [17], and learning latent representations [29]. In this paper, we propose an information maximizing differentially private Generative Adversarial Network (imdpGAN) that preserves privacy of the training samples and learns latent codes to control specificity of generated output. We discuss the privacy model in Section III-A. Then we discuss the mutual information regularization used to learn factors of variation from the data in Section III-B. Then we explain imdpGAN framework, the objective function, and the private training procedure used to introduce differential privacy in Section III-C. Finally, we explain the privacy guarantees of imdpGAN in Section III-D.

A. Differential Privacy

We used differential privacy, as defined by [32], as the privacy model for imdpGAN framework:

Definition 1. *(Differential Privacy, DP) A randomized algorithm A_P is (ε, δ)-differentially private if for any two databases*

[1]Apple tweeted, "Face ID only stores a mathematical representation of your face on iPhone, not a photo.", https://twitter.com/apple/status/1215224753449066497

D and D' differing in a single point and for any subset of outputs S:

$$P(A_P(D) \in S) \leq e^{\varepsilon} . P(A_P(D') \in S) + \delta$$

where $A_P(D)$ and $A_P(D')$ are the outputs of the algorithm for input databases D and D', respectively, and P is the randomness of the noise in the algorithm.

[17] shows that definition in Theorem 1 is equivalent to:

$$\left| \log \frac{P(A_P(D) = s)}{P(A_P(D') = s)} \right| \leq \varepsilon$$

with probability $1 - \delta$ for every $s \in output$, where ε is the privacy level. A small ε value indicates the A_P's output probabilities differ by a small value at s indicating high fluctuations of ground truth outputs and hence high privacy. On the other hand, $\varepsilon = \infty$ means no noise or simply the non-private case.

According to Definition 1, and the above intuition, the ε values represent what level of privacy is protected of individual sample from the dataset. For example, when collecting sensitive information for some experiment, sometimes an individual does not want an observer to know their involvement in the experiment. This is because then the observer can harm that individual's interest. Preserving this involvement would ensure the protection of individual's privacy. It will also make sure that the result will not affect too much if we replace this individual with someone else, which is what we plan to achieve using differential privacy.

B. Mutual Information Regularization

In traditional GANs [9], the generator uses a simple input noise vector z and imposes no constraints on how the noise is used. As a result, noise is used in a highly entangled way and prevents from learning mappings of z to semantic features in the data. The results from the generator of such GANs is, therefore, highly random. However, many datasets decompose into a set of meaningful factors of variation. For example, the MNIST dataset has ten classes, and within the dataset, the thickness and angle of the digits vary. Ideally, a model should automatically learn a discrete random variable to represent the classes and a continuous random variable to represent the thickness and angle properties for such dataset.

To target the structured semantic features of data distribution, [29] decomposes the input noise vector into two parts: the noise vector z and the latent code c. The output of the generator becomes $G(z,c)$. In traditional training, the generator can freely ignore the additional latent code, c. Therefore, to introduce dependency between c and $G(z,c)$, a mutual information term, $I(c;G(z,c))$ is used as a regularization term given as:

$$\begin{aligned} I(c;G(z,c)) &= H(c) - H(c|G(z,c)) \\ &= H(G(z,c)) - H(G(z,c)|c) \end{aligned} \tag{1}$$

where $H(c|G(z,c))$, $H(G(z,c)|c)$ are conditional entropies and $H(c)$, $H(G(z,c))$ are entropies.

The regularization ensures high mutual information between latent code c and generator output $G(z,c)$. It helps the model learn meaningful latent representations. The mutual information term maximizes the dependency between the latent codes and generator output. If c and $G(z,c)$ are independent, then the term $I(c;G(z,c))$ becomes zero, i.e., one variable does not reveal anything about the other. In contrast, to maximize $I(c;G(z,c))$, we relate c and $G(z,c)$ using a non-linear mapping. In the framework, the non-linear mapping is realized using a neural network, Q. The term $I(c;G(z,c))$ is hard to maximize directly, therefore, we define a variational lower bound, $L_I(G,Q)$ to maximize the mutual information [33].

C. imdpGAN framework

We started from a basic GAN architecture but replaced the K-L, J-S divergence with Wasserstein distance as it makes the training stable [31]. We passed an additional input, i.e. the latent code, to the generator and added a mutual information regularization term to introduce dependency between the latent code and the generator output to learn meaningful latent representations.

*1) **Objective Function**:* As shown in Figure 1, we passed a noise vector z and a latent code vector c as input to the generator, G. The output from the generator, $G(z,c)$, and the

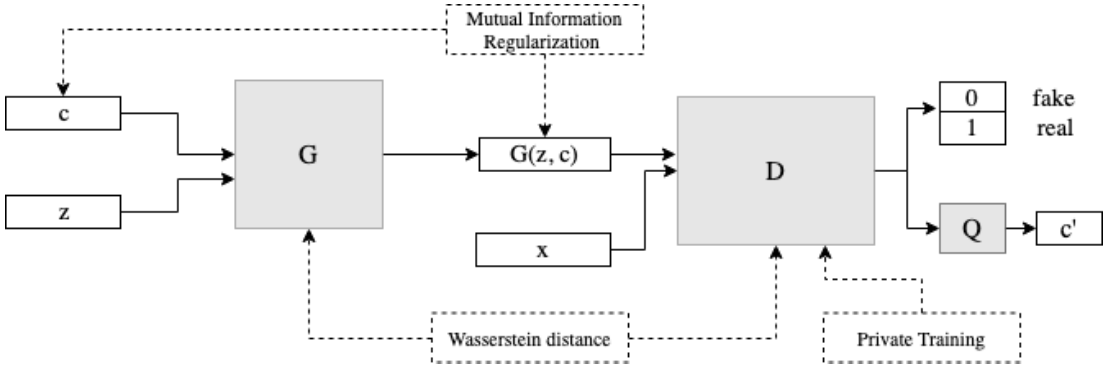


Fig. 1. imdpGAN Architecture: addition of the private training procedure, the mutual information regularization and the Wasserstein distance.

real samples, x, were given to the discriminator, D. Recall that the objective function of a traditional GAN is:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim P_{data}}[\log D(x)] + \mathbb{E}_{z\sim P_{noise}}[\log(1-D(G(z)))] \quad (2)$$

When we use Wasserstein distance [31] instead of the K-L, J-S divergence, the objective function in equation 2 changes as:

$$\min_G \max_{w\in W} \mathbb{E}_{x\sim P_{data}}[f_w(x)] - \mathbb{E}_{z\sim P_{noise}}[f_w(G(z))] \quad (3)$$

where functions $f_w(x)_{w\in W}$ are K-Lipschitz, which is a condition required to solve equation 3.

Further, when we incorporated the mutual information regularization term and the auxiliary distribution Q, the objective function in equation 3 becomes:

$$V_{imdpGAN}(f_w,G,Q) = \min_{G,Q} \max_{w\in W} \mathbb{E}_{x\sim P_{data}}[f_w(x)] - \mathbb{E}_{z\sim P_{noise}}[f_w(G(z))] - \lambda L_I(G,Q) \quad (4)$$

where $L_I(G,Q)$ is variational lower bound used to optimize the mutual information term. λ is an extra hyperparameter used to scale the mutual information according to the GAN objectives.

From Figure 1, D is trained using the private procedure explained in next section. Q is trained in a manner to maximize the mutual information. The learned latent code, c', can be used to control the specificity of the output.

*2) **Private training Procedure**:* [21] describes a differentially private training procedure for stochastic gradient descent that involves adding noise to the gradients and clipping the parameters of the discriminator. [17] extended the procedure to DPGANs. We use the extended version to formulate a private training procedure. The private procedure used in imdpGAN is summarized in Algorithm 1. Adding noise to each gradient step ensures local differential privacy. We get a differentially private generator at the end of the training.

Algorithm 1: Private Training Procedure

Input: Noise - z, real samples - x, parameter clip constant - c_p, batch size - m, total number of training samples - M, number of generator and discriminator iterations - n_g and n_d, weights of generator and discriminator - w_g and w_d, noise - N, noise scale - σ.

Output: Differentially private *Generator*

for $Generator Iterations, n_g$ **do**

  for $Discriminator Iterations, n_d$ **do**

    Sample $\left\{z^{(i)}\right\}_{i=1}^{m} \sim p(z)$ a batch of samples ;

    Sample $\left\{x^{(i)}\right\}_{i=1}^{m} \sim p_{data}(x)$ a batch of real data points ;

    For each sample i, compute gradient of WGAN - $g_{w_d}(x^{(i)}, z^{(i)})$;

    Add noise to the gradient ;

    $g_{w_d} \leftarrow g_{w_d} + N(0, \sigma_n^2 c_p^2 I)$;

    Update discriminator weights, w_d ;

    Clip the parameters ;

    $w_d \leftarrow clip(w_d, -c_p, +c_p)$;

  end

  Sample $\left\{z^{(i)}\right\}_{i=1}^{m} \sim p(z)$ another batch of samples ;

  Update generator gradient, g_{w_g} and weights, w_g ;

end

return *Generator* ;

D. Privacy Guarantees of imdpGAN

To prove the privacy guarantees of imdpGAN, we show that the output of generator (through parameters of discriminator, w_d) guarantees differential privacy with respect to the training samples. Therefore, no generated output from G will compromise the privacy of training points. We can compute the final privacy value, ε, using the moments accountant mechanism.

Assume two generator iterations t_1 and t_2. By treating the parameters of discriminator, w_d at t_1 as one point in outer space, it can be seen that the procedure to update w_d from Algorithm 1 for fixed t_2 is just the algorithm A_p in Definition 1. So, we have $A_p(D) = M(aux, D)$, where *aux* is just auxiliary input, which refers to w_d at iteration t_1. On combining with Definition 1, the privacy loss at point o can be defined as:

Definition 2. *Privacy Loss*

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

The state of discriminator weights is updated by sequentially applying differentially private mechanisms. This is an instance of *adaptive mechanism* modelled by letting *aux* of k^{th} mechanism, M_k, to be the output of all previous mechanisms. For a given mechanism, M, we define λ^{th} moment $\alpha_M(\lambda; aux, D, D')$ as the log of moment generating function evaluated at λ:

Definition 3. *Log moment generating function*

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

The worst case scenario of moment generating function, known as the moments accountant, can be written as:

Definition 4. *Moments accountant*

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

The definition of moments accountant has properties as explained in [21] (Theorem 2): i) composability - the overall moments accountant can be bounded by the sum of moments accountant in each iteration, i.e, privacy is proportional to iterations, ii) the tail bound can also be applied in privacy guarantee.

We need $g_{w_d}(x^{(i)}, z^{(i)})$ to be bounded (by clipping the norm and adding noise according to this bound in Algorithm 1) to use the moments accountant. [17] (Lemma 3.5) proposes that by only clipping on w_d, we can automatically guarantee a bound on $g_{w_d}(x^{(i)}, z^{(i)})$. The lemma is given as:

Definition 5. *Under the condition of Algorithm 1, assume that the activation function of the discriminator has a bounded range and bounded derivatives everywhere:* $\sigma(.) \leq B_\sigma$ *and* $\sigma(.) \leq B_{\sigma'}$*, and every data point* $||x|| \leq B_x$*, then* $||g_{w_d}(x^{(i)}, z^{(i)})|| \leq c_p$ *for some constant* c_p*.*

[17] proves that the Definition 5 holds true if the following condition on derivatives of objective function is met:

$$\begin{aligned} ||g_{w_d}(x^{(i)}, z^{(i)})|| &= \left|\left|\nabla_w \left(f_w(x^{(i)}) - f_w(g(z^{(i)}))\right)\right|\right| \\ &\leq 2\left|\left|\nabla_w f_w(x^{(i)})\right|\right| \end{aligned} \tag{5}$$

The derivative of our objective function, defined in Equation 4 is:

$$\left|\left|\nabla_w \left(f_w(x^{(i)}) - f_w(g(z^{(i)})) - \lambda L_I(g(z^{(i)}, c^{(i)}), Q)\right)\right|\right|$$

Since the additional mutual information regularization term, $\lambda L_I(g(z^{(i)}, c^{(i)}), Q)$ is always positive, the condition in Equation 5 holds true for this objective function as well. Therefore, in Algorithm 1, we only clip w_d to guarantee a bound on $g_{w_d}(x^{(i)}, z^{(i)})$.

Using Definition 5, [17] (Lemma 1.) proves the following Definition 6. The definition holds true for our objective function as well,and therefore, guarantees differential privacy for discriminator training procedure.

Definition 6. *Given the sampling probability* $q = \frac{m}{M}$ *with M as total number of samples and m as batch size, 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, generator Iterations*$, n_g$ *if we choose:*

$$\sigma = \frac{2q\sqrt{n_d \log \frac{1}{\delta}}}{\varepsilon} \tag{6}$$

Equation 6 quantifies the relationship between privacy level ε and noise level σ. Tuning the noise level is required to have a reasonable privacy level. Note that for a fixed value of σ, larger q will lead to less privacy guarantee because when more samples are involved less privacy is assigned on each of them. Also, as the iterations (n_d) increase less privacy is guaranteed because the training reveals more information about the data (specifically, more accurate gradients). Therefore, there is a need to choose a reasonable privacy level.

To achieve different levels of privacy, we use a noise distribution with zero mean and varying standard deviation.

IV. Experiments

A. Experimental Setup

The training of traditional GANs [9] has problems with vanishing gradient and stability. [31] proposed the use of Wasserstein distance between the generator and the data distribution replacing the loss function such that a non-zero gradient always exists. We used the same network architecture given by [29]. However, we added the private training procedure and used Wasserstein distance for training. Note that the mutual information regularization was already available with the base architecture.

1) ***Introducing Privacy*** *:* We make the training procedure private by adding noise to discriminator gradients and clipping parameters of the discriminator as explained in Algorithm 1. Instead of adding noise on the final output directly (global differential privacy), we focused on preserving privacy during the training (local differential privacy) as it generally results in high utility. We used Gaussian noise with zero mean (no bias) and varying standard deviation, σ. Gaussian noise is a popular choice for privacy preservation [34] and usually results in (ε, δ)-differential privacy. From equation 6, σ and ε are inversely corelated. Therefore, more noise results in smaller ε values. The batch size for all experiments is 64, and the number of samples is 60,000 and 202,599 for MNIST and CelebA, respectively. We set the sampling probability to $q = 64/number\_of\_samples$, the clip constant (c_p) to 0.01 such that weights of discriminator are clipped back to $[-c_p, c_p]$, privacy violation (δ) to 10^{-5}, and number of discriminator iterations (n_d) to 5.

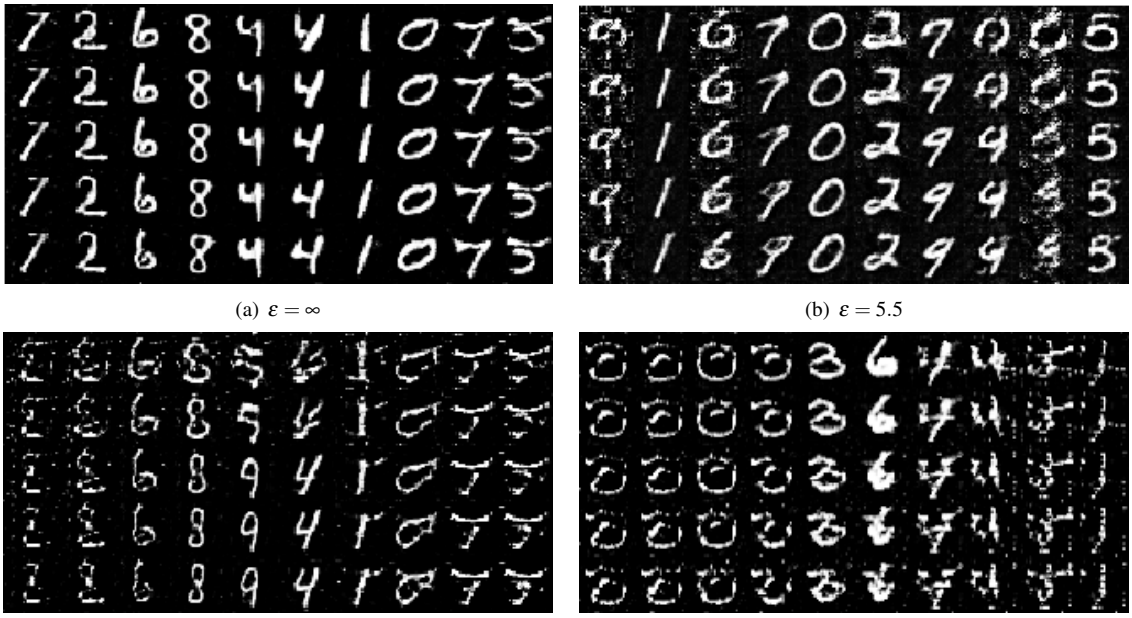
2) ***Learning Latent Representations*** *:* The term $I(c; G(z, c))$, from equation 1, is hard to maximize directly. Therefore, we defined an auxiliary distribution, Q, and used variational information maximizing term, $L_I(G, Q)$ [33] to maximize the mutual information. In imdpGAN, we formulated Q as a neural network. Q and the discriminator share all convolutional layers. To learn latent representations, we modeled discrete codes with uniform categorical distribution, $Cat(K = k, p = k/100)$ that model the discontinuous variation in data, i.e., classes. We also modeled continuous codes with a uniform continuous distribution, $Unif(-1, 1)$ that capture continuous variations in data, like writing style (width) in MNIST, color variation in CelebA, etc.

B. imdpGAN Generation

In this experiment, we study the effect of noise[2] on the generator samples. We use MNIST data with 60,000 training samples to train imdpGAN. The experiment is run several times with varying privacy levels, $1 \leq \varepsilon \leq 10$. The samples generated with $\varepsilon \geq 10$ looked similar to the samples generated with $\varepsilon = \infty$. To show the results we pick three ε values, 1.22, 2.2 and 5.5 as the samples generated with these values looked significantly different.

We wanted to see the output change as we make changes to the latent codes. The experiment is performed on MNIST dataset, with 60,000 training samples, 10 classes, and continuous variations like the rotation and width of the digit. We use a discrete-valued latent code, $c1 \sim Cat(K = 10, p = 0.1)$, which models the discontinuous variation in data, i.e., classes and a continuous latent code, $c2 \sim Unif(-1, 1)$, which captures continuous variation in data such as the writing style. The generator is differentially private, so the samples generated

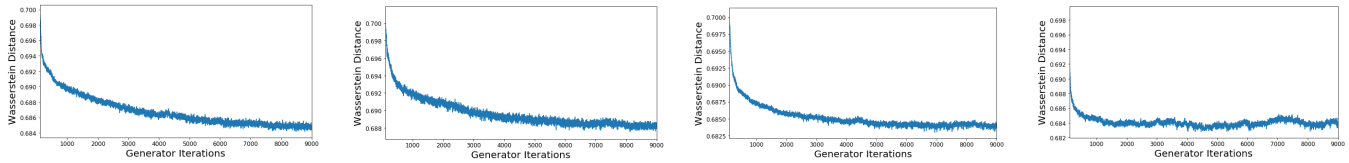
[2]Noise here refers to the Gaussian noise we add to the training procedure to introduce local differential privacy and not the noise vector z we pass as input to the generator.



(a) $\varepsilon = \infty$ (b) $\varepsilon = 5.5$

(c) $\varepsilon = 2.2$ (d) $\varepsilon = 1.22$

Fig. 2. Varying latent codes to generate images on MNIST dataset. In all the images, the categorical discrete latent code is varied from left to right. For different ε values the change in the discrete latent code can be seen as the class of generated image is changing. The continuous latent code $c2$ is varied from -1 to +1 (top to bottom). A change in width of the generated samples can be observed with varying $c2$.



(a) $\varepsilon = \infty$ (b) $\varepsilon = 5.5$ (c) $\varepsilon = 2.2$ (d) $\varepsilon = 1.22$

Fig. 3. Wasserstein distance versus generator iterations on MNIST dataset. As ε values decrease, curves exhibit more fluctuations (due to more noise being added) and larger variance but converge proving training stability.

with the model trained with smaller ε values (more noise added during training) are highly distorted. Due to distortion, the specificity in such images is not clearly visible. Although the learned representations are evident in at least one or more columns, as shown in Figure 2.

*1) **Convergence of network**:* The experiment is performed on MNIST dataset with 60,000 training samples to see the behavior of Wasserstein distance. We plot the Wasserstein distance versus generator iterations for every batch in MNIST dataset. The min-max training causes some fluctuations itself; therefore, to perform this experiment, only the amount of noise is changed, keeping the rest of the parameters fixed. As shown in Figure 3, the Wasserstein distance decreases during the training and converges in the end. We observed that a smaller ε, indicating more noise, leads to more fluctuations and larger variance. The trend is visible in the later half of the plot. Intuitively, more noise should result in more fluctuations and hence, blurry images, which shows the experiment's consistency with the results of the previous experiment.

*2) **Binary Classification on Digit Pairs**:* In this experiment, we use a binary classification task to demonstrate the trade-off between utility and privacy of the framework. The classification acts as a quantitative measure to evaluate the utility of proposed framework. Since the images are blurred due to private training, we choose the classes which are still somewhat recognizable from all cases of ε. We make pairs of these, viz. 3-8, 9-1, to perform binary classification. We choose 2,000 samples for each class for the no noise case and for $\varepsilon = 5.5, 2.2, 1.22$. Then we build binary classifiers using the 4,000 samples (2,000 of each class) for class pairs 3-8 and 9-1. We perform the training for 100 epochs and report the accuracy of built classifiers on MNIST's test set. The results are shown in Figure 4. As expected, as the privacy level (ε) increases, the accuracy of the classifiers decrease.

*3) **Mutual Information Maximization**:* The experiment demonstrates the changes in the mutual information, between the generator and the latent codes, with changing privacy levels. We trained imdpGAN on MNIST dataset with 60,000 training samples. We defined the latent code using a categorical uniform distribution, $c \sim Cat(K = 10, p = 0.1)$. We trained imdpGAN with different privacy levels. As shown in Figure 5, as we add more noise (decreasing ε values), the

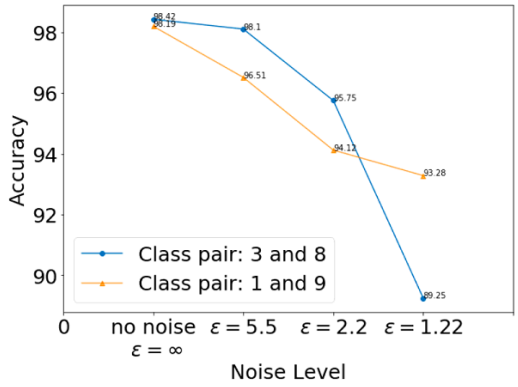


Fig. 4. Binary Classification task on MNIST test set using samples generated with different ε values as training set. From left to right, we use generated data $\varepsilon = \infty$ (without noise), and generated data with values $\varepsilon = 5.5, 2.2, 1.22$ (with noise). We can see as more noise is added, i.e., more privacy is introduced, the accuracy of classifier decreases, which indicates a trade-off between choosing a promising privacy level and an acceptable accuracy threshold.

mutual information gain decreases. The pattern demonstrates that the lower bound, L_I, is quickly maximized but due to the addition of noise at each iteration, its value decreases. Therefore, noise must be chosen very carefully to maintain a good value of L_I, which ensures the learning of meaningful latent representations.

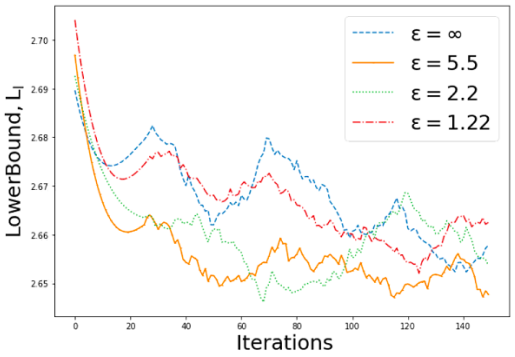
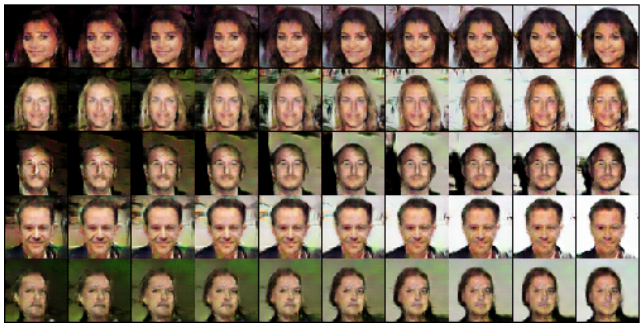


Fig. 5. The Lower Bound, L_I, versus Iterations on MNIST dataset. The lower bound decreases with decreasing ε values. The plot demonstrates that ε values must be chosen carefully to learn meaningful latent representations.

C. E3: Results on CelebA dataset

We extended the experiments on CelebA dataset, which includes 202,599 celebrity face images with variations like pose and brightness. The experiment is performed several times with different values of ε. We show the generated samples for $\varepsilon = 5.5$ as the generated images are lesser likely to be visually identifiable. As shown in Figure 6, varying latent codes on CelebA dataset can generate images with varying faces, hair styles, and brightness. We used a uniform categorical latent code to capture variations in the face, and a uniform continuous latent code to capture style variations. Figure 6(a) shows the variation in continuous code brings change in the hairstyles of generated images. In Figure 6(b), the continuous code captures the brightness of generated images, and varying it allows us to generate images with specific brightness.

(a) Changing hair style



(b) Changing brightness

Fig. 6. Varying latent codes on CelebA dataset. The images in both sub-figures are generated using a fixed, $\varepsilon = 5.5$. In both sub-figures, the discrete latent code is varied from top to bottom to generate a new face. In (a), the continuous latent code is varied from -1 to +1 (left to right) and shows variation in the hairstyle of generated images. In (b), the continuous latent code is varied from -1 to +1 (left to right) and shows variation in the brightness of generated images.

V. Discussion

imdpGAN framework can alleviate problems faced while working with imbalanced data by controlling the specificity of the generator output. For example, if the number of instances belonging to some class in a dataset is small, we can selectively generate samples (using the representative latent code) for that class to increase its samples. But the problem is, imdpGAN framework itself needs a sufficient amount of samples to learn the distributions before generating good quality data. Therefore, to solve the class imbalance problem using imdpGAN, we must have adequate samples belonging to the class having less number of samples in the dataset.

[29] shows that mutual information regularization only adds a negligible complexity to GAN training. Therefore, to see how adding the private procedure effects the training time, we observed the training time without the private procedure, and with the private procedure for different noise levels. As shown in Figure 7, the private training procedure introduces an approximately 10-fold increase in training time. Without the private procedure, the model takes 955 seconds for 50 epochs to train. On the other hand, with the private procedure added

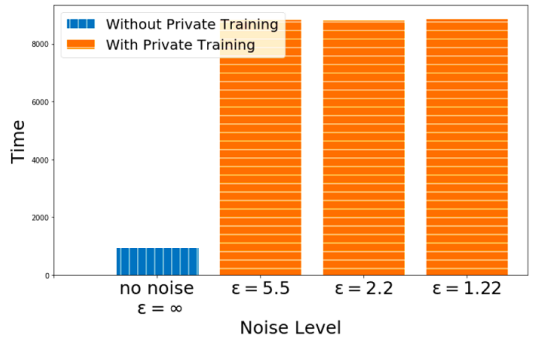


Fig. 7. The time taken by the model with and without the private training procedure. Training time increases 10-fold with the private procedure. However, it is not affected by changing values of ε.

to the training, the same model takes 8,850 seconds for the same number of epochs. Note that the training time does not change much with varying noise levels ($\varepsilon = \infty$ is the no noise case).

VI. Conclusion

In this paper, we have proposed an Information maximizing Differentially Private Generative Adversarial Network (imdpGAN), a unified framework to simultaneously preserve privacy and learn latent representations. imdpGAN preserves privacy and successfully learns meaningful latent representations. However, the private procedure added to imdpGAN's training results in a 10-fold increase in the training time.

For future work, the core idea of using the private procedure and mutual information to learn latent representation can be applied to other datasets, which do not necessarily contain images. imdpGAN shows promising results on the image datasets: MNIST and CelebA. The architecture of the Generator and the Discriminator can be changed to extend imdpGAN to other data types.

References

[1] K. Munir, H. Elahi, A. Ayub, F. Frezza, and A. Rizzi, "Cancer diagnosis using deep learning: A bibliographic review," *Cancers*, vol. 11, no. 9, 2019. [Online]. Available: https://www.mdpi.com/2072-6694/11/9/1235

[2] F. Arcadu, F. Benmansour, A. Maunz, J. Willis, Z. Haskova, and M. Prunotto, "Deep learning algorithm predicts diabetic retinopathy progression in individual patients," *npj Digital Medicine*, vol. 2, no. 1, p. 92, 2019. [Online]. Available: https://doi.org/10.1038/s41746-019-0172-3

[3] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '15. New York, NY, USA: ACM, 2015, pp. 1322–1333. [Online]. Available: http://doi.acm.org/10.1145/2810103.2813677

[4] E. Choi, S. Biswal, B. Malin, J. Duke, W. F. Stewart, and J. Sun, "Generating multi-label discrete patient records using generative adversarial networks," in *Proceedings of the 2nd Machine Learning for Healthcare Conference*, ser. Proceedings of Machine Learning Research, F. Doshi-Velez, J. Fackler, D. Kale, R. Ranganath, B. Wallace, and J. Wiens, Eds., vol. 68. Boston, Massachusetts: PMLR, 18–19 Aug 2017, pp. 286–305. [Online]. Available: http://proceedings.mlr.press/v68/choi17a.html

[5] M. Alzantot, S. Chakraborty, and M. B. Srivastava, "Sensegen: A deep learning architecture for synthetic sensor data generation," *2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pp. 188–193, 2017.

[6] R. Camino, C. A. Hammerschmidt, and R. State, "Generating multi-categorical samples with generative adversarial networks," *CoRR*, vol. abs/1807.01202, 2018.

[7] J. Guan, R. Li, S. Yu, and X. Zhang, "Generation of synthetic electronic medical record text," *2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pp. 374–380, 2018.

[8] L. Xu and K. Veeramachaneni, "Synthesizing tabular data using generative adversarial networks," *CoRR*, vol. abs/1811.11264, 2018.

[9] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 2672–2680. [Online]. Available: http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf

[10] O. Mogren, "C-rnn-gan: Continuous recurrent neural networks with adversarial training," *CoRR*, vol. abs/1611.09904, 2016.

[11] M. Saito and E. Matsumoto, "Temporal generative adversarial nets," *CoRR*, vol. abs/1611.06624, 2016.

[12] T. Salimans, I. J. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *NIPS*, 2016.

[13] L. Sweeney, "K-anonymity: A model for protecting privacy," *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, vol. 10, no. 5, pp. 557–570, Oct. 2002. [Online]. Available: http://dx.doi.org/10.1142/S0218488502001648

[14] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam, "L-diversity: privacy beyond k-anonymity," in *22nd International Conference on Data Engineering (ICDE'06)*, April 2006, pp. 24–24.

[15] N. Li, T. Li, and S. Venkatasubramanian, "t-closeness: Privacy beyond k-anonymity and l-diversity," in *2007 IEEE 23rd International Conference on Data Engineering*, April 2007, pp. 106–115.

[16] A. Narayanan and V. Shmatikov, "Robust de-anonymization of large sparse datasets," in *Proceedings of the 2008 IEEE Symposium on Security and Privacy*, ser. SP '08. Washington, DC, USA: IEEE Computer Society, 2008, pp. 111–125. [Online]. Available: https://doi.org/10.1109/SP.2008.33

[17] L. Xie, K. Lin, S. Wang, F. Wang, and J. Zhou, "Differentially private generative adversarial network," *CoRR*, vol. abs/1802.06739, 2018.

[18] B. Hitaj, G. Ateniese, and F. Perez-Cruz, "Deep models under the gan: Information leakage from collaborative deep learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '17. New York, NY, USA: ACM, 2017, pp. 603–618. [Online]. Available: http://doi.acm.org/10.1145/3133956.3134012

[19] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.

[20] L. Ziwei, L. Ping, W. Xiaogang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of International Conference on Computer Vision (ICCV)*, Washington, DC, USA, December 2015, p. 3730–3738.

[21] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '16. New York, NY, USA: ACM, 2016, pp. 308–318. [Online]. Available: http://doi.acm.org/10.1145/2976749.2978318

[22] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '15. New York, NY, USA: ACM, 2015, pp. 1310–1321. [Online]. Available: http://doi.acm.org/10.1145/2810103.2813687

[23] N. H. Phan, X. Wu, and D. Dou, "Preserving differential privacy in convolutional deep belief networks," pp. 1681–1704, 06 2017.

[24] J. B. Tenenbaum and W. T. Freeman, "Separating style and content with bilinear models," *Neural Comput.*, vol. 12, no. 6, pp. 1247–1283, Jun. 2000. [Online]. Available: http://dx.doi.org/10.1162/089976600300015349

[25] Z. Zhu, P. Luo, X. Wang, and X. Tang, "Multi-view perceptron: A deep model for learning face identity and view representations," in *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 1*, ser. NIPS'14. Cambridge,

MA, USA: MIT Press, 2014, pp. 217–225. [Online]. Available: http://dl.acm.org/citation.cfm?id=2968826.2968851

[26] S. Reed, K. Sohn, Y. Zhang, and H. Lee, "Learning to disentangle factors of variation with manifold interaction," in *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ser. ICML'14. JMLR.org, 2014, pp. II–1431–II–1439. [Online]. Available: http://dl.acm.org/citation.cfm?id=3044805.3045052

[27] T. D. Kulkarni, W. F. Whitney, P. Kohli, and J. Tenenbaum, "Deep convolutional inverse graphics network," in *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Curran Associates, Inc., 2015, pp. 2539–2547. [Online]. Available: http://papers.nips.cc/paper/5851-deep-convolutional-inverse-graphics-network.pdf

[28] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 11 2015.

[29] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "Infogan: Interpretable representation learning by information maximizing generative adversarial nets," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, ser. NIPS'16. USA: Curran Associates Inc., 2016, pp. 2180–2188. [Online]. Available: http://dl.acm.org/citation.cfm?id=3157096.3157340

[30] G. Desjardins, A. Courville, and Y. Bengio, "Disentangling factors of variation via generative entangling. arxiv:1210.5474, 2012. learning to disentangle factors of variation with manifold interaction," 2012.

[31] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," *CoRR*, vol. abs/1701.07875, 2017.

[32] C. Dwork, "Differential privacy," in *Proceedings of the 33rd International Conference on Automata, Languages and Programming - Volume Part II*, ser. ICALP'06. Berlin, Heidelberg: Springer-Verlag, 2006, pp. 1–12. [Online]. Available: http://dx.doi.org/10.1007/11787006_1

[33] D. Barber and F. V. Agakov, "The im algorithm: A variational approach to information maximization," in *NIPS*, 2003.

[34] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Found. Trends Theor. Comput. Sci.*, vol. 9, no. 3–4, pp. 211–407, aug 2014. [Online]. Available: http://dx.doi.org/10.1561/0400000042