Differentially Private GAN for Time Series

Pepijn te Marvelde , Lydia Y. Chen[1] **, Aditya Kunar**[2] **, Zilong Zhao**[2]
TU Delft
[1]Responsible Professor, [2]Supervisor
{P.H.temarvelde, A.Kunar}@student.tudelft.nl, {Y.Chen-10, Z.Zhao-8}@tudelft.nl

Abstract

Generative Adversarial Networks (GANs) are a modern solution aiming to encourage public sharing of data, even if the data contains inherently private information, by generating synthetic data that looks like, but is not equal to, the data the GAN was trained on. However, GANs are prone to remembering samples from the training data, therefore additional care is needed to guarantee privacy. Differentially Private (DP) GANs offer a solution to this problem by protecting user privacy through a mathematical guarantee, achieved by adding carefully constructed noise at specific points in the training process. A state-of-the-art example of such a GAN is Gradient Sanitized Wasserstein GAN, (GS-WGAN), [1]. This model is shown to create higher quality synthetic images than other DP GANs. To extend the applicability of GS-WGAN we first reproduce and extend the evaluation, verifying that the model outperforms DP-CGAN by an average of 40% when assessed across three qualitative metrics and two datasets. Secondly we propose improvements to the architecture and training procedure to make GS-WGAN applicable on timeseries data. The experimental results show that GS-WGAN is fit for generating synthetic timeseries through promising experimental results.

1 Introduction

The rise of big and open data is provoking the use of machine learning models in an ever-growing set of domains [1]. To encourage growth and stimulate fast progress there is a need for data sharing. However, those datasets most valuable for corporations and research frequently contain private data, thus they cannot be distributed due to various privacy laws (e.g., GDPR [2]). A technique proposed to enable sharing of valuable datasets without violating users' privacy is leveraging Generative Adversarial Networks (GAN) [3] in combination with Differential Privacy [4] to generate synthetic differentially private data.

GANs have been shown to be effective for numerous data(set) augmentation tasks like increasing image resolution and enriching datasets by generating new samples indistinguishable from the real data. As a result of this, models trained on the augmented data will perform better [5]. GANs achieve this by using two models that train in parallel. The first is a Generative model G, which aims to capture the distribution of the data, and the second is a Discriminatory model D, which tries to differentiate between samples from the training data and samples generated by G [3]. During the joint training D gets better at differentiating real from fake and punishes G if it creates bad samples, thus G will create more realistic samples over time. Unfortunately GANs are shown to be susceptible to attacks aiming to recover training samples [6], making standard GANs unsuitable for usecases where privacy needs to be guaranteed.

Differential Privacy (DP) is a methodology that preserves the privacy of a dataset by injecting calibrated statistical noise [4]. There exists different DP techniques that differ in what type of noise to inject, and when to inject it, e.g., directly to the data, or during the learning process. The first approach works by adding noise to the answers returned by queries from a database, but using this approach the dataset itself is not differentially private and therefore cannot be released in full. In contrast, adding this noise during the learning process of a GAN will result in a privatized dataset, which can be released to the public with no additional privacy loss.

Achieving differentially private GAN training is done by using a DP Stochastic Gradient Descent algorithm, proposed by Abadi et. al. in [7], which carefully controls the influence a single training sample can have during training. DP SGD preserves privacy by carefully clipping, and adding Gaussian noise to, the gradients. This gradient descent algorithm is typically applied only in the discriminator, since this part of the GAN is trained on real data. But if only the generator, or synthetic data generated by the generator, will be released after training only the generator can be privately trained [1]. As a result of only privately training the generator the discriminator will be more reliable which improves the combined training process.

Two state of the art examples of GANs capable of generating synthetic data with DP guarantees are GS-WGAN [1] and DP-CGAN [8]. Both models use the previously mentioned DP SGD algorithm during training. GS-WGAN applies this algorithm only to the generator, DP-CGAN only to the discriminator. The second key difference is the loss

function used in the two models. DP-CGAN uses a minimax loss whereas GS-WGAN uses a Wasserstein-1 metric as loss which generates gradients with norms close to 1 resulting in less information being lost during gradient clipping.

GS-WGAN and DP-CGAN are both shown to be able to generate realistic MNIST and FashionMNIST images, GS-WGAN creating the more realistic samples out of the two [1; 9]. But, sensitive datasets that would benefit from being distributed in a private manner are rarely MNIST-like images. Therefore it is interesting to explore GS-WGAN performance on other types of data. An example of another data type that would benefit greatly from methods to generate synthetic data with rigorous privacy guarantees is time series data, e.g., location traces, browsing history, or credit card transactions. These all contain incredibly personal information and are thus not suited for release to the public, limiting the advance in industries dealing with these kinds of data. There exists GANs shown to be capable of generating such timeseries, examples are RDP-CGAN [10], [11] and [12], all differing slightly how they apply Differential Privacy during learning and their model architecture.

In this paper, we ask two research questions: (i) the exhaustive quantitative and qualitative difference between DP-CGAN and GS-WGAN, and (ii) if and how to apply GS-WGAN on time series data.

i) To fairly compare the performance an exhaustive quantitative and qualitative evaluation on MNIST and Fashion-MNIST data has been carried out, as is done in [1]. The quantitivate results show performance by means of the Inception Score [13], Frechet Inception Distance [14] and the average classification accuracy of downstream classifiers. Since quantitative metrics cannot capture all relative aspects of images a qualitative evaluation is carried out by means of a comparison of the generated samples.

ii) Exploring GS-WGANs capabilities regarding synthetic time series data generation was done by altering the model to take in inputs with other dimensions than 28x28 values (MNIST dimensions) and evaluating its performance in generating time series data for two datasets containing ECG heartrhythms: PTB [15] and MIT-BIH [16]. Data generated by GS-WGAN will be evaluated and performance will be compared to the three models last mentioned in the previous paragraph.

This paper organized as follows. We first explain the preliminary background of GANs and Differential Privacy in section 2 and give an overview of the different GANs mentioned in section 3. These sections are followed by the main contributions of this research:

- In section 4 the experiment setup and results for part (i) are laid out.
- The proposed changes to GS-WGAN to facilitate generation of synthetic time series data, and the results of time series generation for part (ii) are discussed next in section 5.

After that the ethical implications of this research are discussed, and last, the conclusion is given in section 7.

2 Preliminaries

This section briefly introduces Generative Adversarial Networks and Differential Privacy, followed by how these concepts are combined in subsection 2.3.

2.1 Generative Adversarial Networks

The GAN framework, proposed by Goodfellow et al. [3], works by having two neural networks compete in a minimax game with the goal to create a generator that is capable of generating samples indistinguishable from the training samples. The two networks competing are called the discriminator (D) and the generator (G). The output of each of these networks serves as input for the other: G creates fake samples that become training examples for D, which tries to differentiate real and fake samples. D's classification is fed back into G to update its weights through backpropagation. After sufficient training G is capable of generating realistic samples. These samples can then be used to enrich existing datasets for improved modelling on this data, or for releasing datasets without sensitive information.

However, for this last point a stronger privacy guarantee is necessary since GANs may generate samples not present in the training data. This makes GANs susceptible to membership inference attacks [17]. To improve resilience against such attacks, and introduce a privacy guarantee, Differential Privacy [4] can be used.

2.2 Differential Privacy

To fully understand the intricacies of Differential Privacy in GANs we first need to understand the core of the DP mathematical framework which is defined as follows in [4; 1]:

Definition 2.1 (Differential Privacy)**.** A randomized algorithm $\mathcal{M}$ with range $\mathcal{M}$ is (ϵ, δ)-DP, if

$$Pr[\mathcal{M}(S) \in \mathcal{O}] \leq e^{\epsilon} \cdot Pr[\mathcal{M}(S') \in \mathcal{O}] + \delta \qquad (1)$$

holds for any subset of outputs $\mathcal{O} \subseteq \mathcal{R}$ and for any adjacent datasets S and S', where S and S' differ from each other with only one training example. ϵ corresponds to the upper bound of privacy loss, and δ is the probability of breaching DP constraints.

In simpler terms: DP ensures and quantifies privacy through a mathematical guarantee. The privacy of a system is quantified through ϵ and δ. ϵ is a metric for privacy loss, lower ϵ means more private data. δ is the probability of breaking the DP guarantee (accidental information leakage) and is often set to a maximum of the inverse of the number of records $\frac{1}{|N|}$.

2.3 Differential Privacy in GANs

In DP GANs the randomized algorithm $\mathcal{M}$ is (a variation on) a differentially private stochastic gradient descent algorithm proposed in [7], also shown in algorithm 1 for the readers convenience. This algorithm works by bounding its sensitivity (i.e. the influence any single training sample has on the whole model in a step) through clipping gradients and adding

Model	Loss	DP site	Noise mechanism
GS-WGAN [1]	Wasserstein Loss & gradient penalty	Generator	Gaussian
DP-CGAN [8]	Minimax Loss	Discriminator	Gaussian
RDP-CGAN [10]	Wasserstein Loss & gradient penalty	Discriminator & Autoencoder/decoder	Gaussian
PATE-GAN [11]	KL Divergence Loss	Discriminator	Laplacian
DPGAN [12]	Wasserstein Loss & weight clipping	Discriminator	Gaussian

Table 1: Overview of the GANs mentioned in this research.

Algorithm 1: Differentially Private SGD

Input Samples $\{x_1, ..., x_N\}$, loss function $\mathcal{L}(\theta) = \frac{1}{N}\sum_i \mathcal{L}(\theta, x_i)$. Hyper-parameters: learning rate η_t, noise scale σ, batchsize L, gradient norm bound C.

Initialize θ_0 randomly;

for $t \in [T]$ **do**

- Take a random sample : L_t with sampling probability B/N;
- **Compute Gradient** For each $i \in L_t$, compute $g_t(x_i) \leftarrow \Delta_{\theta_t}\mathcal{L}(\theta_t, x_i)$
- **Clip gradient** $\bar{g}_t \leftarrow g_t(x_i)/\max(1, \frac{||g_t(x_i)||_2}{C})$
- **Add noise** $\tilde{g}_t \leftarrow \frac{1}{L}(\sum_i \bar{g}_t(x_i) + \mathcal{N}(0, \sigma^2 C^2 I))$
- **Descent** $\theta_{t+1} \leftarrow \theta_t - \eta_t \tilde{g}_t$

Output θ_T and overall privacy cost (ϵ, δ) computed using a privacy accountant.

random normally distributed noise to those gradients resulting in "a sanitized version of the gradients in which the influence of the input data is bounded, guaranteeing privacy" [18]. Every training step introduces a privacy loss, which needs to be carefully monitored to keep track of the spent privacy budget ϵ.

Both GS-WGAN and DP-CGAN use a refinement of differential privacy called Rényi Differential Privacy (RDP) [19]. This refinement is used since it provides composition which is convenient for keeping track of the spent privacy over multiple steps of gradient descent [1]. It also introduces a tighter bound on the privacy budget when compared to the classically used Moment Accountant which reduces the amount noise needed to add to the gradients, while not compromising privacy [20; 8].

3 Related work

To create a better understanding of the GANs GS-WGAN will be compared to, a summary of all mentioned models is given here. Starting with the GS-WGAN itself in subsection 3.1, followed by DP-CGAN, the model GS-WGAN will be compared with on MNIST data. Last, three GANs shown to be capable of generating high quality synthetic timeseries will be summarised from subsection 3.3. An overview of the key characteristics for each model is given in Table 1

3.1 Gradient-Sanitized Wasserstein GAN

The main model evaluated in this research is the Gradient-Sanitized Wasserstein GAN proposed by Chen in [1] which addresses two shortcomings in standard DP-SGD, namely (i) that gradient clipping destroys valuable gradient information, and thus impacts utility; and (ii) that the search for a reasonable clipping value is intensive.

The impact of gradient clipping is reduced by only applying gradient sanitization in the generator, since only that model will be released. The discriminator does not to be privately trained as is done in standard DP GANs. This reduces the limitations posed on the discriminator and allows a more complex and better model architecture, as well as discriminator warm-start. This warm-starting process comprises of pre-training discriminators with a non-private generator for some iterations, these pre-trained discriminators are then used to train the private generator without compromising privacy.

The search for a good clipping gradient is eliminated by bounding sensitivity using the Wasserstein-1 metric [21] as the loss function. This results in bounded gradients with norms close to 1 and avoids the search for hyper-parameters like the clipping bound. To further increase privacy GS-WGAN supports using multiple discriminators during training. The dataset is subsampled into a seperate subset for each discriminator. At each training step a single discriminator is chosen and used for training. This increases privacy since any sample that is not in the subset has no risk of being leaked.

GS-WGAN was released with two generator architectures, a large learning framework with state-of-the-art performance on image datasets: ResNet [22], and DCGAN, a simpler convolutional architecture. Since ResNet performs better on images this architecture is used for part (i). In part (ii) both architectures are used and compared.

3.2 Differentially Private Conditional GAN

The main idea of DP-CGAN [8] is adding gaussian noise to the sum of the separately clipped real and fake discriminator loss. It is also capable of generating sample labels along with the samples itself.

3.3 Timeseries GANs

The following GANs are used as baselines in part (ii) for assessing GS-WGAN performance on time series. These particular models were chosen as they are part of the evaluation from [10] that will be extended by adding GS-WGAN.

RDP-CGAN [10]: Combines convolutional autoencoders and convolutional GANs to preserve temporal dynamics and critical characteristics for the generated synthetic data.

PATE-GAN [23]: Modifies the Private Aggregation of Teacher Ensembles (PATE) framework [11] and applies it to GANs by replacing the discriminator with a PATE mechanism to make it DP. Also uses multiple teacher discriminators on subsamples of the training data, similar to GS-WGAN.

DPGAN [12]: DPGAN uses the Wasserstein distance as the loss function, like GS-WGAN, but, instead of clipping gradients to limit the effect of any one training sample, the weights of the model are clipped to provide privacy guarantees.

4 Reproducing

Part (i) of this research: the exhaustive quantitative and qualitative difference between GS-WGAN and DP-CGAN is laid out in this section. This evaluation was performed to verify claims by Chen et. al. [1] stating that GS-WGAN outperforms other state of the art DP GANs when tasked with creating synthetic MNIST images with a robust privacy guarantee. This evaluation shows the reproduced results alongside an extra dimension of the comparison: performance of downstream classifiers versus privacy cost ϵ. This evaluation is structured as follows: the metrics used are discussed in subsection 4.1. All details to be able to reproduce the evaluation are given in subsection 4.2. Last, the results of the evaluation are laid out in subsection 4.3. Since not only results, but also ease of use of these models is important, a more subjective view is given in Appendix C.

4.1 Method

To compare the two models the generated images are compared qualitatively and quantitatively across iterations, and thus across privacy cost ϵ. For the qualitative evaluation the final generated images are analysed. The quantitative comparison is done by means of two widely used metrics in GAN evaluations: the Inception Score (IS) [13] & Frechet Inception Distance (FID) [14], and the accuracy of classifiers trained on the generated samples. The first two aim to quantify how realistic the samples are by leveraging pre-trained classifiers to assess the generated images. The third metric shows how useful the generated samples are for training new classifiers.

Inception Score The Inception Score [13] is a commonly used metric to evaluate images generated by GANs. It aims to quantify the realism of an image and has been shown to correlate well with human scoring. To compute this a classifier [24] was trained on real MNIST data and used to compute this metric. The metric formula as follows:

$$IS = exp(\mathbb{E}_x KL(p(y|x)||p(y))) \tag{2}$$

which is the KL divergence between the marginal distribution and the conditional distribution of each image. In simpler terms: if an image contains a single clear object (number or fashion item in this case) the conditional distribution will have low entropy, and if the set of images has high variety the marginal distribution will have high entropy. Thus, synthetic image sets containing realistic samples evenly distributed over the classes get a high score.

Frechet Inception Distance Another metric used, and specifically designed, for GAN performance evaluation is the Frechet Inception distance, which improves upon the Inception Score by taking into account the statistics of real world samples, and comparing those to the statistics of generated samples [25]. The FID is given by:

$$\text{FID} = ||\mu_r - \mu_g||^2 + Tr(\Sigma_r + \Sigma_g - 2\sqrt{\Sigma_r \Sigma_g}) \tag{3}$$

where μ and Σ are the mean and covariance of the activations of the last pooling layer of the Inception network, and $_r$ denotes real & $_g$ the generated samples. Thus FID tries to capture the distance of a set of generated to a set of real images, by comparing statistics of the activations of the Inception network.

Accuracy of downstream classifiers The simplest, and arguably most meaningful, metric used in this evaluation is the accuracy of downstream classifiers trained on generated samples. This metric captures whether the generated images are of high quality by verifying if a classifier trained on those samples generalizes well on test data. To get a better idea of how the accuracy compares to the accuracy of a model trained on real samples the accuracy is divided by the accuracy of a model trained on real samples.

4.2 Experimental setup

During this evaluation the original implementation of GS-WGAN [26] using PyTorch 1.2.0 was used. The implementation of DP-CGAN [27] is written using TensorFlow 1.14.0. In an effort to increase reproducibility both models and the evaluation script can be run in Docker containers which should prevent versioning issues. The full implementation and instructions to reproduce this evaluation can be found on https://github.com/ptemarvelde/dp-timeseries. All training and evaluations were done on a Google Cloud system: e2-custom (12 vCPUs, 96 GB memory).

Training details. To reproduce the evaluation of GS-WGAN and compare it to DP-CGAN both models were trained with the same hyperparameters as in the paper, also shown in Table 8. The pre-trained discriminators published alongside the GS-WGAN implementation were also used to warm-start the model. During training intermediate samples and generators, respectively for GS-WGAN and DP-CGAN were saved to be evaluated.

Details on metric calculation. All metrics were calculated using 10k generated samples for each save step. Since the authors of GS-WGAN did not provide code for their evaluation the actual calculation differs in some places, but the relation between the metrics of DP-CGAN & GS-WGAN was preserved since they are both evaluated equally.

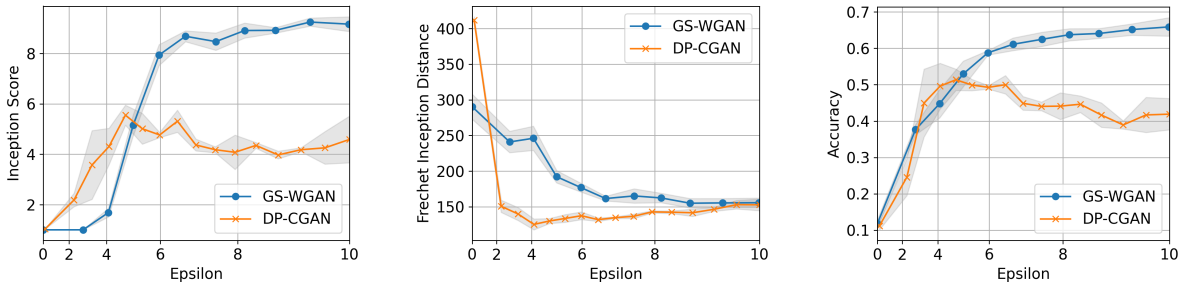
For the IS and FID the calculation was done in 10 batches using 1000 samples each, for the classifier accuracy all 10k generated samples were used at once.

Inception Score. For this metric a simplified model of the Inception network was used. This model was trained on 60k input images for 5 epochs with a final accuracy on the test set of 98% and 91% for MNIST and Fashion-MNIST respectively.

Frechet Inception Distance. FID was calculated using the implementation from [28], which uses the InceptionV1 net-

		Inception Score ↑		Frechet Inception Distance ↓		Average downstream accuracy ↑		Calibrated accuracy ↑	
		Repr.	Paper	Repr.	Paper	Repr.	Paper	Repr.	Paper
MNIST	Real	9.92	9.80	4.46	1.02	0.85	0.88	100%	100%
	GS-WGAN	**9.17 ± 0.290**	**9.23**	155.74 ± 6.658	**61.34**	**0.56 ± 0.025**	**0.60**	**66%**	**68%**
	DP-CGAN	4.59 ± 0.925	4.76	**152.78 ± 8.173**	179.16	0.36 ± 0.043	0.52	41%	45%
Fashion-MNIST	Real	8.49	8.98	9.00	1.49	0.75	0.79	100%	100%
	GS-WGAN	**5.03 ± 0.281**	**5.32**	**221.71 ± 5.880**	**131.34**	**0.51 ± 0.008**	**0.53**	**69%**	**67%**
	DP-CGAN	3.69 ± 0.750	3.55	229.08 ± 6.772	243.80	0.34 ± 0.019	0.43	45%	54%

Table 2: Comparison of quantitative results from the paper and the reproduced (Repr.) values with $\epsilon = 10$ and $\delta = 10^{-5}$. ↑ & ↓ respectively denote higher and lower values being better. Best results from both the reproduced values and the Paper's values are **bold**



(a) Inception Score (b) Frechet Inception Distance (c) Average calibrated classifier accuracy*

Figure 1: Privacy vs. utility tradeoff comparison between GS-WGAN and DP-CGAN with up to $\epsilon = 10$ ($\delta = 10^{-5}$). Corresponding graphs for Fashion-MNIST are presented in Appendix D. *Average accuracy of a classifier trained on synthetic data and tested on real data divided by the accuracy of the same classifiers trained and tested on real data.

work, opposed to the InceptionV3 network used in the original paper's evaluation since none of the three publicly available FID calculation codebases performed accurately. For the downstream classifier accuracy a collection of standard scikit-learn models trained and evaluated, as is done in [1; 9]. The only difference being the number of privately generated samples used to train the classifiers. In the GS-WGAN paper this is 60k, but in this evaluation only 10k synthetic samples was used since this reduced computational costs significantly and had no significant impact on classifier accuracy (0.003 ± 0.020).

4.3 Results and findings

The quantitative results in Table 2 show that GS-WGAN outperforms DP-CGAN in all metrics. For the Inception Score GS-WGAN almost scores twice as high as DP-CGAN on MNIST data, while the improvement on Fashion-MNIST is less big (36%, 5.03 v.s. 3.69). GS-WGAN also scores worse than DP-CGAN on FID, likely due to the use of the old InceptionV1 network as mentioned in Figure 4.2.

Arguably the most impressive improvement is the performance of classifiers trained on privately generated samples. This metric translates the best to real life performance since it shows how well models trained on the generated data perform, essentially telling us how well the generated data captures the statistical properties of the training data [1]. GS-WGAN shows a performance increase over DP-CGAN of 37% and 67% for MNIST and FashionMNIST respectively. Important however, is to note that GS-WGAN only performs better with higher privacy budgets of $\epsilon > 4$.

Data generated by GS-WGAN performing better than DP-CGAN is also supported by the qualitative results shown in Table 3. GS-WGAN's images are clearer and have significantly less noise than DP-CGAN's counterparts.

An interesting observation is that both models stop approving after spending a privacy budget of $\epsilon \approx 7$. As shown in Figure 1 the metrics improve rapidly at the start of training, and slow down around the $\epsilon = 6$ mark, rarely improving after $\epsilon = 8$ Therefore, stopping training earlier will not have significant impact on quality of generated samples, but will yield a stronger privacy guarantee.

In conclusion we can confidently say GS-WGAN is capable of generating higher quality differentially private images than DP-CGAN. Since we, and Chen et. al. in the proposal of GS-WGAN [1] draw the same conclusions this is an accurate reproduction substantiating the performance of GS-WGAN.

5 Supervised generation of synthetic time series data using GS-WGAN

The second research question addressed by this study is whether GS-WGAN is capable of creating realistic differ-

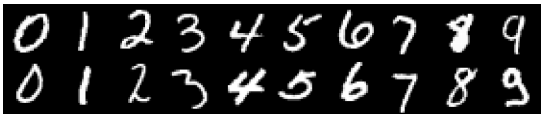
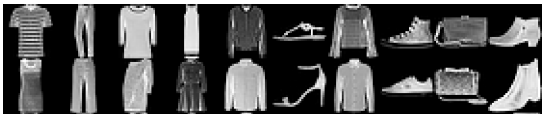
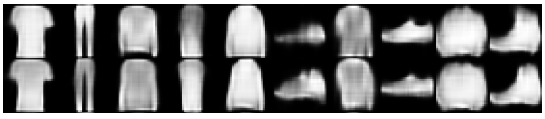
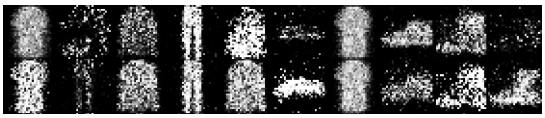
Model	MNIST	Fashion-MNIST
Real		
GS-WGAN		
DP-CGAN		

Table 3: Generated and real samples for MNIST & Fashion-MNIST datasets. $(\epsilon, \delta) = (10.0, 10^{-5})$

entially private synthetic time series data, as ResNet architecture has proven effective at classifying timeseries [29]. This question is answered in this section, while simultaneously comparing the different generator architectures of GS-WGAN. Comparing the architectures is done to see if increased performance of ResNet over DCGAN on image synthesis carries over to time series. This enables better reasoning about the general setting of applying image-based deep learning models to time series. This evaluation is done by training GS-WGAN on two time series datasets and evaluating the synthetic data. The precise approach taken, and changes made to the model, to do this are explained in subsection 5.1. In subsection 5.3 the performance is evaluated against baselines from Torfi et. al. [10] laid out in the proposal of RDP-CGAN.

5.1 Method

This section presents how GS-WGAN is applied to timeseries and how the generated data is evaluated to conclude whether the model is fit for generation of synthetic time series under strict privacy guarantees. First, the necessary changes to the model architecture are described, after that an explanation of how the data is prepared enabling the use of GS-WGAN is given, followed by an overview of the datasets used, baselines compared against and the metrics that are part of the comparison.

Model changes. GS-WGAN did not have the ability to be trained on images of arbitrary (square) sizes, the model only supported 28x28 pixel images. This restriction limits the applicability of the model immensely. Therefore, any hard-coded inner model layer dimension that limited the input size was converted to dynamic dimensions, thus scaling the model with the input image size.

Data preparation Since GS-WGAN has proven to be effective on image datasets, we want to convert the problem of generating time series to the problem of generating images, enabling direct application of GS-WGAN to the data. A similar approach was shown to be effective in [30]. To achieve this, we first convert the time series data into an image, and then train GS-WGAN model on that image. The step by step process is:

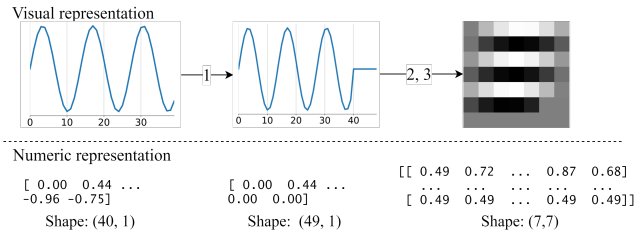


Figure 2: Diagram of data preparation steps with an explanatory sine sample of length 40, converted to an image of 7x7 pixels.

1. The input time series of length n is padded with zeros up to length $N = \lceil (n^2) \rceil$
2. The padded data is reshaped into a square array of values with width & height equal to $\sqrt{N}$
3. If necessary this square array is normalized , resulting in sample that could be interpreted as a grayscale image of $\sqrt{N}$ x $\sqrt{N}$ with pixel values between 0 and 1. This is equal to the MNIST images GS-WGAN was evaluated on for $\sqrt{N} = 28$

	PTB Diagnostic [15]	MIT-BIH Arrhythmia [16]
Samples	14,550	109,444
Sequence length	187	187
No. of classes	2	5[1]
Label distribution[2]	10506 / 4048	90587 / 18857
Train/test split	random, 80%/20%	predefined, 87553/21891
Accuracy	0.951	0.959
AUROC	0.989	0.965
AUPRC	0.995	0.927

Table 4: Statistics of datasets used for comparison alongside performance of the baseline model. [1] 1 label for normal heartbeats, 4 types of abnormal heartbeats. [2] normal / abnormal

After these steps the model can be trained on these images using the process shown to be effective on MNIST images.

Datasets Since GS-WGAN relies on having training labels two supervised problems, making use of PTB [15] and MIT-BIH [16] datasets, are used in this comparison. Both contain electrocardiogram (ECG) recordings with the task to classify the recording as normal or abnormal. An overview of each set's statistics is given in Table 4. These two datasets were chosen since there are extensive baselines available for an accurate comparison.

Evaluation metrics To asses the quality of the synthetic data a Gradient Boosting Classifier from sklearn with the number of estimators set to 100 is used. This classifier is tasked with classifying input samples as 'normal' or 'abnormal' heartbeats. To reason about how well the generated data captures the feature distribution of the real data the model is trained on the synthetic data, and then tasked with classifying real test samples. This classification is converted into four metrics: the first being the Operating Characteristic Curve AUC (AUROC), second is the Precision-Recall curve AUC (AUPRC) which correlates better with performance on imbalanced datasets [10], these two metrics are used to compare GS-WGAN to the baselines. To further evaluate performance the F1 score, which shows how well precision and recall are balanced, and accuracy are also shown. Higher is better for all four metrics.

5.2 Experimental setup

The experiment is carried out as follows. After preparing the data as described in subsection 5.1 the GAN is trained. Since parameters (Number of discriminators & noise scale) directly influencing the spent privacy budget ϵ have tremendous effect on utility, a (naive) hyperparameter search was performed on the PTB dataset to find good values for these parameters, this can be found in subsection E.2 and results are presented in the next paragraph. During these experiments the observation that using the same setting for high and low privacy budgets was difficult. For low privacy one wants a low number of iterations coupled with a high number of discriminators and high noise scale, i.e., the GAN should learn very quickly and you do not have to worry much about getting stuck in local optima, since the chance that finding any optimum in such little iterations is small. In a high privacy setting the GAN should converge more gradually and it should converge to a global optimum. Therefore finding one set of hyperparameters for both settings is extremely difficult. Thus, we use two sets of hyperparameters for the different privacy budgets. The values can be found in Table 8. The hyperparameter search was not performed again on the MIT-BIH dataset to see if near optimal parameters for PTB also result in high utility on MIT-BIH.

The results of the search for optimal noise scale σ for the low and high ϵ settings are:

- **High** ($\epsilon > 1.0$): For this setting the number of pretrained discriminators was set to 100, since this amount of discriminators is still practical to pretrain. Any more would take multiple days. After this six different noise scales were tried between 1.0 and 2.25, with $\sigma = 1.25$ being best, resulting in the configuration shown in Table 8

	Low $\epsilon \leq 1.0$	High ϵ ¿ 10.0
Pretrained discriminators	✗	✓
Number of discriminators	400	100
Noise scale σ	4.0	1.25
Learning rate	1.0^{-2}	1.0^{-4}
Save Iterations	ϵ = 0.1: 9, ϵ = 1.0: 937	ϵ = 10.0: 312, ϵ = 100.0: 7915

Table 5: Hyperparemeters used for GS-WGAN training. Low ϵ corresponds to privacy budget ≤ 1.0, high ϵ to budgets of > 1.0.

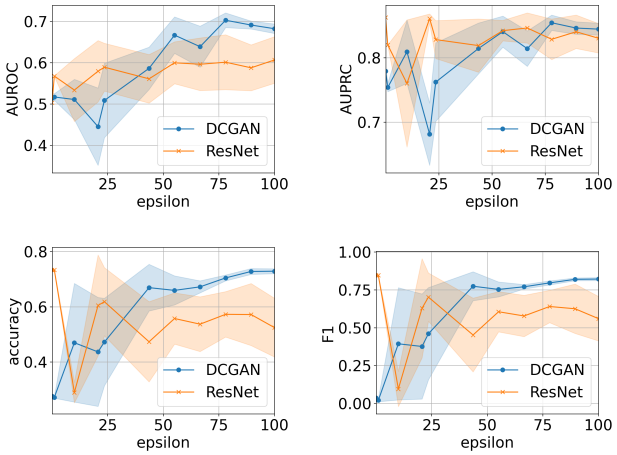


Figure 3: Comparison of GS-WGAN generator architectures by showing utility versus privacy budget. Higher values signify higher quality synthetic data. All graphs for PTB dataset.

- **Low** ($\epsilon \leq 1.0$): The search for good noise scale for low privacy budget is more difficult due to the way privacy budget is spent in GS-WGAN. The spent privacy budget per step decreases across iterations (shown visually in subsection E.1) making it difficult to get good results for low privacy budgets, since there is no time to train. The slow learning at the start of training is also prevalent in MNIST training as shown in Figure 1, DP-CGAN outperforms GS-WGAN only for low privacy budgets. To enforce some training before the spent privacy budget reaches $\epsilon = 0.1$, noise scale, number of discriminators and learning rate need to be increased. The best values found are shown in Figure 8. A higher number of discriminators could perform better but could not be tried due to GPU memory constraints.

All experiments were run five times with both DCGAN and ResNet generator architectures, and averages and standard deviations are shown. δ is always set to 10^{-5}, other hyperparameters are those shown in Table 8 unless stated otherwise. Important to note is that all values for PATE-GAN, RDP-CGAN and DPGAN are taken from the paper proposing RDP-CGAN [10]. The evaluation procedure is clearly described there, thus enabling values presented for GS-WGAN are computed using the same method making them fit for direct comparison. Training was done on an NVIDIA 2080Ti.

Dataset	Metric	PATE-GAN	RDP-CGAN	DPGAN	GS-WGAN ResNet	GS-WGAN DCGAN
PTB	AUROC	$\underline{0.75 \pm 0.012}$	$\mathbf{0.79 \pm 0.009}$	0.71 ± 0.012	0.47 ± 0.051	0.50 ± 0.121
	AUPRC	0.76 ± 0.011	$\underline{0.80 \pm 0.008}$	0.71 ± 0.018	$\mathbf{0.83 \pm 0.069}$	0.80 ± 0.083
MIT-BIH	AUROC	0.73 ± 0.006	0.77 ± 0.003	0.69 ± 0.004	0.42 ± 0.051	0.56 ± 0.055
	AUPRC	0.73 ± 0.016	0.78 ± 0.008	0.68 ± 0.023	0.17 ± 0.031	0.37 ± 0.169

Table 6: GS-WGAN comparison versus baselines under $(\epsilon, \delta) = (1, 10^{-5})$-DP. Best and second best are respectively **bold** and underlined.

5.3 Results and findings

Generator architecture for GS-WGAN. First we compared GS-WGAN generator architectures: DCGAN vs ResNet on PTB data. The metrics over time, shown in Figure 3, illustrate that using DCGAN generator architecture results in higher quality synthetic data (for higher privacy budgets). All metrics show clear convergence by diminishing standard deviation as epsilon grows, which is not the case for ResNet, since the ResNet architecture significantly more trainable parameters (>40M v.s. 1M for DCGAN) convergence will take longer.

GS-WGAN v.s. baselines - fixed privacy. The performance of GS-WGAN on PTB data for a fixed privacy budget of $(\epsilon, \delta) = (1.0, 10^{-5})$ (shown in Table 6, is better than, or on par with, baselines when looking at AUPRC. But, AUROC is much lower for both GS-WGAN architectures, showing suboptimal performance for this privacy budget. This imbalance between the metrics is due to GS-WGAN generating samples that cause the downstream classifier to classify all test samples as 'normal'. This is the most prevalent class resulting in high AUPRC.

Impact of privacy budget on GS-WGAN. Although the quality of data generated for the low privacy setting is not ideal, GS-WGAN with DCGAN architecture shows promising results for higher privacy budgets, see Figure 4. For $\epsilon > 10$ the model performs close to baselines, not only with competitive AUPRC, but also with good AUROC (Figure 4), showing that the proposed method of converting time series into images is a valid technique to leverage image based GANs for synthesizing time series.

Performance on MIT-BIH. Performance on MIT-BIH data is much worse, even for large ϵ, than baselines. Thus we can confidently sufficiently good parameters from PTB do not translate to high quality synthetic MIT-BIH data. This is likely due to the large discrepancy in the number of normal v.s. abnormal samples, preventing the discriminator from learning an accurate representation of the abnormal samples, and thus the generator from generating high quality abnormal samples in the limited number of iterations enforced by the privacy budget.

Summary. GS-WGAN is capable of creating high quality synthetic time series data in a differentially private setting after the made changes to the architecture and way data is prepared. But for low privacy settings the model is outperformed by other state of the art DP GANs, only having better performance when focussing on AUPRC. Also important to note is that it might be possible to get high quality results, for all metrics, for low privacy budgets, but this would require substantial computing power to pretrain discriminators and be able to hold all discriminators in memory.

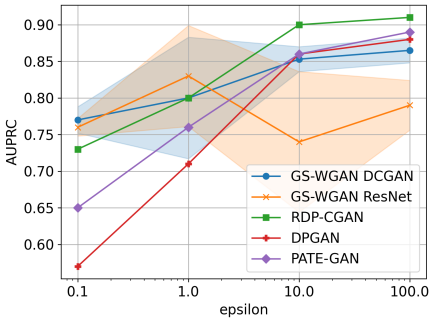


Figure 4: The effect of spent privacy budget ϵ on the quality of generated data measured by AUPRC for the PTB Diagnostic dataset.

6 Responsible Research

This section detail the ethical considerations alongside the efforts to increase reproducibility of this research.

6.1 Ethical implications

Differentially Private GANs enables distribution of valuable data in a robust privacy preserving way. This encourages advances in fields where sharing data is difficult due to the privacy implications and restrictions. It is, however, important to remember that the data is created by a deep learning model. This makes it hard to reason about possible biases in downstream models since the bias could be rooted in either the DP GAN that generated the data, or the machine learning model that was trained on the synthetic data. Therefore, much care should go into controlling the quality of the synthetic data in order to minimize bias introduced by the GAN.

6.2 Reproducibility

All material needed to reproduce the presented results can be found, alongside the raw results, on GitHub. Docker and Docker-compose files for both models and the evaluation are provided that can be used to run the training and evaluation with two commands. This allows anyone with access to the necessary computing power to reproduce all results without having to worry about dependency conflicts.

Providing all these materials to encourage and simplify reproducing of results does, unfortunately, not mean that everyone will get equal results. The exact values and synthetic data will differ slightly due to randomness in GANs. To try and reduce this a random seed was set during training and

evaluation, but since no two computers are equal this does not take away all randomness.

7 Conclusions and future work

This research consisted of two parts. For part (i) We have shown qualitatively and quantitatively that GS-WGAN outperforms DP-CGAN when tasked with synthetic MNIST images in a differentially private setting. Also extending the original evaluation of GS-WGAN by showing average downstream classifier accuracy over spent privacy budget. For using GS-WGAN to generate time series (ii) we propose using the convolutional architecture of GS-WGAN for time series generation by converting time series into images and using these to train our GAN. This approach enables the use of image based GANs for time series synthesis. We have shown that using GS-WGAN to generate Differentially Private time series yields results on par with state of the art time series based DP GANs for privacy budgets $\epsilon > 10.0$ on the PTB dataset. For low privacy budgets ($\epsilon < 1.0$ the data generated by the data boasts higher AUPRC, but significantly worse AUROC. This shows that the model does not learn feature distribution of underrepresented classes quickly enough to generate high quality samples for such classes early in training.

For future work, considering how to increase performance early on in training to get high quality results for settings with a tight privacy budget would increase usefulness of GS-WGAN greatly. Also interesting would be exploring the possibility of exploiting more features of image based GANs on time series, like using the different channels of colored images to represent different features in multivariate data.

References

[1] D. Chen, T. Orekondy, and M. Fritz, "Gs-wgan: A gradient-sanitized approach for learning differentially private generators," 2021.

[2] General data protection regulation. European Commission. [Online]. Available: https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679

[3] 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*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, Eds., vol. 27. Curran Associates, Inc., 2014. [Online]. Available: https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf

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

[5] F. H. K. dos Santos Tanaka and C. Aranha, "Data augmentation using gans," *CoRR*, vol. abs/1904.09135, 2019. [Online]. Available: http://arxiv.org/abs/1904.09135

[6] D. Chen, N. Yu, Y. Zhang, and M. Fritz, "Gan-leaks: A taxonomy of membership inference attacks against generative models," *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, Oct 2020. [Online]. Available: http://dx.doi.org/10.1145/3372297.3417238

[7] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, Oct 2016. [Online]. Available: http://dx.doi.org/10.1145/2976749.2978318

[8] R. Torkzadehmahani, P. Kairouz, and B. Paten, "Dp-cgan: Differentially private synthetic data and label generation," 2020.

[9] F. Harder, K. Adamczewski, and M. Park, "Dp-merf: Differentially private mean embeddings with random features for practical privacy-preserving data generation," 2021.

[10] A. Torfi, E. A. Fox, and C. K. Reddy, "Differentially private synthetic medical data generation using convolutional gans," 2020.

[11] N. Papernot, M. Abadi, Úlfar Erlingsson, I. Goodfellow, and K. Talwar, "Semi-supervised knowledge transfer for deep learning from private training data," 2017.

[12] L. Xie, K. Lin, S. Wang, F. Wang, and J. Zhou, "Differentially private generative adversarial network," 2018.

[13] T. Salimans, I. J. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," *CoRR*, vol. abs/1606.03498, 2016. [Online]. Available: http://arxiv.org/abs/1606.03498

[14] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, G. Klambauer, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a nash equilibrium," *CoRR*, vol. abs/1706.08500, 2017. [Online]. Available: http://arxiv.org/abs/1706.08500

[15] R.-D. Bousseljot, D. Kreiseler, and A. Schnabel, "The ptb diagnostic ecg database," 2004. [Online]. Available: https://physionet.org/content/ptbdb/

[16] G. B. Moody and R. G. Mark, "Mit-bih arrhythmia database," 1992. [Online]. Available: https://physionet.org/content/mitdb/

[17] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *2017 IEEE Symposium on Security and Privacy (SP)*, 2017, pp. 3–18.

[18] L. Frigerio, A. S. de Oliveira, L. Gomez, and P. Duverger, "Differentially private generative adversarial networks for time series, continuous, and discrete open data," 2019.

[19] I. Mironov, "Rényi differential privacy," *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, Aug 2017. [Online]. Available: http://dx.doi.org/10.1109/CSF.2017.11

[20] Y.-X. Wang, B. Balle, and S. Kasiviswanathan, "Subsampled rényi differential privacy and analytical moments accountant," 2018.

[21] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proceedings of the 34th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, D. Precup and Y. W. Teh, Eds., vol. 70. PMLR, 06–11 Aug 2017, pp. 214–223. [Online]. Available: http://proceedings.mlr.press/v70/arjovsky17a.html

[22] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[23] J. Yoon, J. Jordon, and M. van der Schaar, "PATE-GAN: Generating synthetic data with differential privacy guarantees," in *International Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=S1zk9iRqF7

[24] M. Fabien, "A guide to inception model in keras," 2019. [Online]. Available: https://maelfabien.github.io/deeplearning/inception/#in-keras

[25] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," 2018.

[26] D. Chen, T. Orekondy, and M. Fritz, "DP-CGAN," 1 2020. [Online]. Available: https://github.com/DingfanChen/GS-WGAN

[27] F. Harder, K. Adamczewski, and M. Park, "dp-merf," 10 2020. [Online]. Available: https://github.com/frhrdr/dp-merf

[28] S. Tang, "Frechet-Inception-Distance," 6 2020. [Online]. Available: https://github.com/tsc2017/Frechet-Inception-Distance

[29] S. de Roever, "Using resnet for ecg time-series data," 2020. [Online]. Available: https://towardsdatascience.com/using-resnet-for-time-series-data-4ced1f5395e3

[30] E. Brophy, Z. Wang, and T. Ward, "Quick and easy time series generation with established image-based gans," 02 2019.

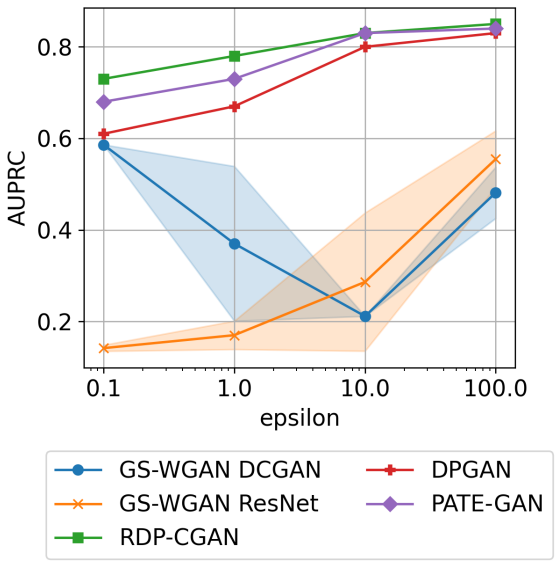


Figure 5: The effect of spent privacy budget on the quality of generated data measured by AUPRC for the MIT-BIH dataset. AUPRC calculated on by using real test samples to evaluate a classifier trained on synthetic data. Higher AUPRC signifies better classifier performance and thus higher quality generated data.

A List of abbreviations

See Table 7

Abbreviation	Meaning
GAN	Generative Adversarial Network
DP	Differential Privacy
RDP	Rényi Differential Privacy
SGD	Stochastic Gradient Descent
GS-WGAN	Gradient Sanitized Wasserstein GAN
DP-CGAN	Differentially Private Conditional GAN
PATE	Private Aggregation of Teacher Ensembles
AUROC	Area Under the Receiver Operating Characteristic Curve
AUPRC	Area Under the Precision-Recall Curve

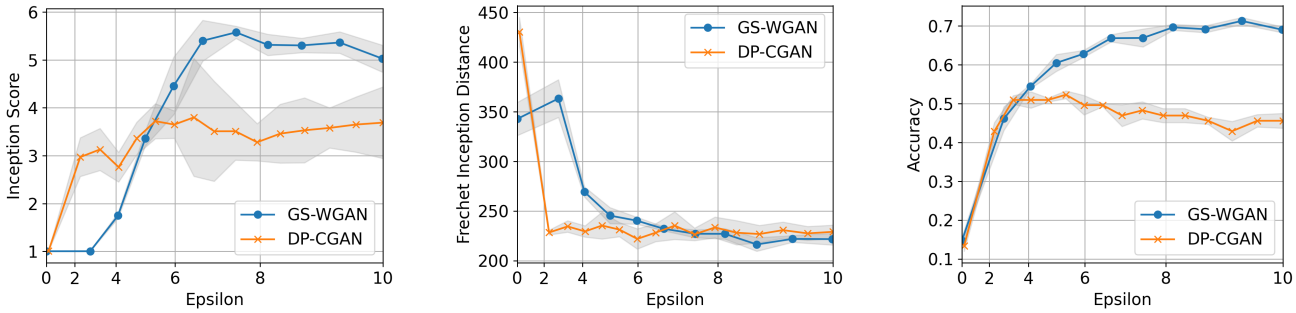
Table 7: List of used abbreviations

B MNIST training hyperparameters

See Table 8

C GS-WGAN vs DC-GAN: usability

GANs are known for intensive and long training. This also is the case for GS-WGAN and DP-CGAN. On the system used in this evaluation training takes upwards of a day to complete. This will probably be much lower on a system with a powerful GPU, but due to the current shortage those were not available in the Google Cloud Compute Engine



(a) Inception Score

(b) Frechet Inception Distance

(c) Average calibrated classifier accuracy*

Figure 6: Privacy vs. utility tradeoff comparison, on Fashion-MNIST dataset, between GS-WGAN and DP-CGAN with maximal $\epsilon = 10$ and $\delta = 10^{-5}$. *Average accuracy of a classifier trained on synthetic data and tested on real data divided by the accuracy of the same classifiers trained and tested on real data.

	GS-WGAN	DP-CGAN
(ϵ, δ)-Differential Privacy	(10, 10^{-5})	(10, 10^{-5})
(Maximum) training iterations	20,000	150,000
Number of pre-trained discriminators	1,000	-
Batch size	32	600
(σ, C) (noise scale, clipping value)	(1.07, 1.0)	(2.1, 1.1)
Save step	2,000	10,000

Table 8: Hyperparameters used in the evaluation for GS-WGAN and DP-CGAN

Also noteworthy is the ease of use of both models. GS-WGAN is much easier to configure and use than DP-CGAN. The code is more structured and understandable. Command line options and examples on how to run training are given. DP-CGAN took much more time to get working since the original implementation did not give good results. The only two advantages over GS-WGAN: First, regarding usability, is that the model will automatically end training when the predefined privacy budget has been spent. Second is the memory usage during training. GS-WGAN is incredibly memory intensive since all discriminators are kept there, requiring upwards of 22GBs of RAM for 1000 discriminators.

D Privacy v.s. utility tradeoff for GS-WGAN and DP-CGAN on FashionMNIST data

See Figure 6

E Supplementary materials for applying GS-WGAN on timeseries

E.1 Influence of number of discriminators and noise on privacy spending

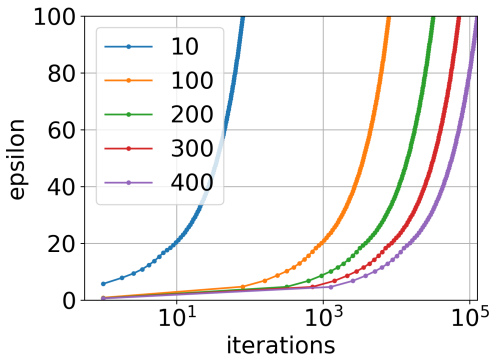
See Figure 7

E.2 PTB hyperparameter search

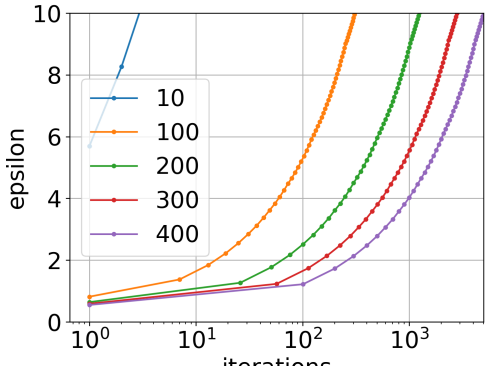
To find an appropriate value for noise scale σ for the high epsilon setting GS-WGAN was trained on PTB dataset for 6 values between 1.0-2.25, these values were chosen since they give a good balance between privacy budget and training iterations. A higher noise scale would increase training time too much. Each noise scale was used for 5 training runs and the averages and standard deviations are shown in Figure 8. The effect of the noise on synthetic data quality is most obvious in the AUROC plot. $\sigma = 1.25$ consistently shows above average synthetic data quality and was therefore chosen as the noise scale for the high ϵ setting.

E.3 MIT-BIH AUPRC plot

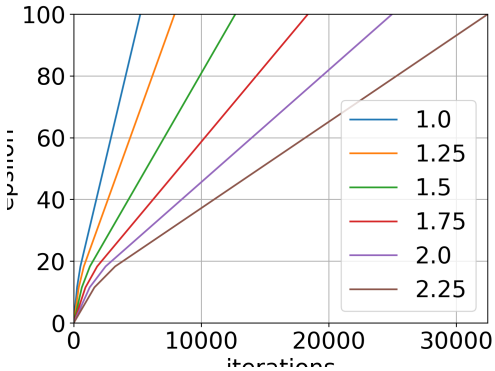
See Figure 5



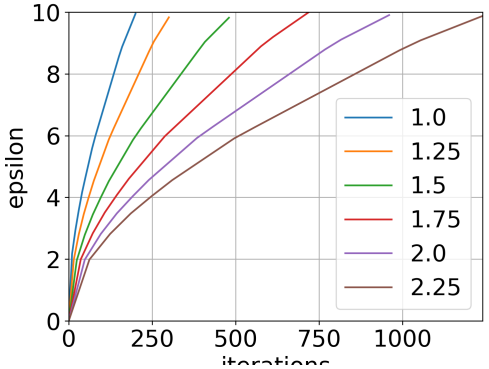
(a) Effect of number of discriminators (log scale)



(b) Effect of number of discriminators for lower privacy budget

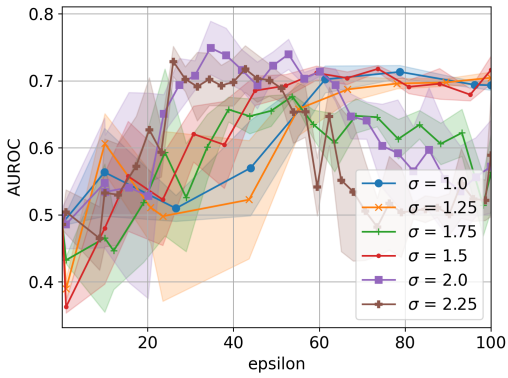


(c) Effect of noise (log scale)

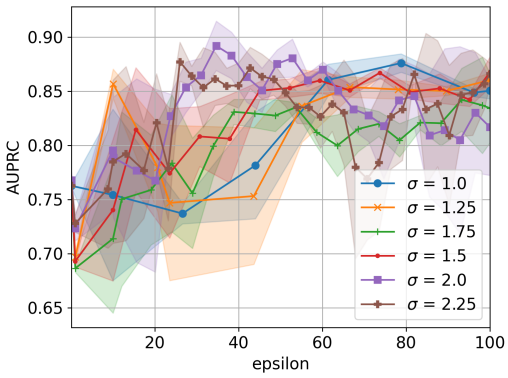


(d) Effect of noise of discriminators for lower privacy budget

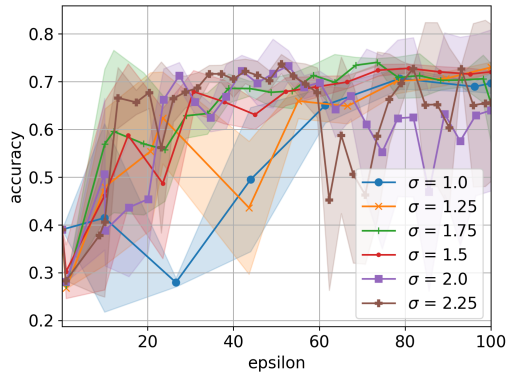
Figure 7: Effects of noise scale on utility, $1.0 \leq \sigma \leq 2.25$. Averages of 5 runs.



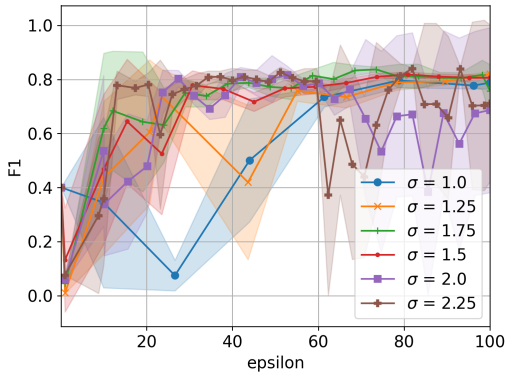
(a) Area Under Receiver Operating Characteristic Curve



(c) Downstream Classifier Accuracy



(b) Area Under Receiver Precision-Recall Curve



(d) F1 Score

Figure 8: Effects of noise scale on utility, $1.0 \leq \sigma \leq 2.25$. Averages of 5 runs.