Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

DL-DPGAN: A Correlation-Regularized Differentially Private GAN for Privacy-Utility Balanced Synthetic Data Generation

Mohammad Emadi
Shahid Beheshti University
Vahideh Moghtadaiee
vmoghtadaiee@gmail.com

Shahid Beheshti University
Mina Alishahi
Open Universiteit

Research Article

Keywords: Data Privacy, GAN, DPGAN, DL-DPGAN, Synthetic Data, Classifier

Posted Date: April 14th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-9250688/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

DL-DPGAN: A Correlation-Regularized Differentially Private GAN for Privacy-Utility Balanced Synthetic Data Generation

Mohammad Emadi[1], Vahideh Moghtadaiee[1*], Mina Alishahi[2]

[1]Cyberspace Research Institute, Shahid Beheshti University, Tehran, Iran.

[2]Department of Computer Science, Open Universiteit, Amsterdam, The Netherlands.

*Corresponding author(s). E-mail(s): v_moghtadaiee@sbu.ac.ir;

Abstract

Balancing data utility and privacy in machine learning is crucial,particularly in settings where large volumes of sensitive data are collected and analyzed. Generative Adversarial Networks (GANs) and their privacy-preserving variants, Differentially Private GANs (DPGANs), have been employed to create synthetic data that supports privacy protection. However, prior work has shown that, in practice, synthetic data generated by such models can still exhibit privacy risks, especially when trained on confidential data. In this study, we propose Double Loss DPGAN (DL-DPGAN), an enhanced framework that incorporates a correlation-based privacy loss together with the Wasserstein distance to reduce information leakage and improve training stability. To evaluate the model's performance, we compare the empirical privacy behavior and the quality of the synthetic data generated by DL-DPGAN with those produced by standard GAN and DPGAN baselines. Experiments on multiple benchmark datasets indicate that DL-DPGAN generates synthetic data with strong resistance to distinguishability while maintaining competitive classifier performance compared to these baselines. Overall, DL-DPGAN offers a reasonable balance between privacy protection and model utility, providing a practical approach for privacy-preserving and accurate machine learning.

Keywords: Data Privacy, GAN, DPGAN, DL-DPGAN, Synthetic Data, Classifier.

1 Introduction

The rapid deployment of machine learning systems across sensitive domains such as healthcare, finance, and critical infrastructures has led to the large-scale collection and processing of personal and confidential data. Leveraging this data for intelligent services requires balancing utility and privacy, especially when sharing information across interconnected devices and platforms [1]. Deep learning, a prominent technique for processing large datasets, generally performs best when provided with abundant and diverse data [2]. However, in environments where data is sensitive, distributed, and subject to strict privacy requirements, collecting large-scale high-quality datasets is costly and challenging, which significantly hinders model development and deployment.

To overcome these challenges, synthetic data generation has emerged as a viable solution. By generating artificial datasets that resemble real data, Generative Adversarial Networks (GANs) [3] have gained popularity as an effective technique for producing high-quality synthetic data while retaining the key statistical features of the original datasets. These synthetic data can subsequently be used to train machine learning models without directly accessing sensitive or private information, thus supporting privacy protection. However, traditional GANs and even their differentially private variants (DPGANs) [4] may inadvertently leak sensitive patterns, as they often generate synthetic data that remains highly correlated with the real datasets. Recent empirical studies have shown that privacy risks may still manifest in practice even when formal guarantees are claimed, motivating additional empirical privacy auditing and mitigation mechanisms [5]. This vulnerability raises significant privacy concerns in data-sensitive environments, where unintended information leakage may result in legal, financial, or reputational harm. While differential privacy provides formal guarantees under a well-defined threat model, recent studies suggest that empirical leakage may still arise due to distributional similarity between real and synthetic data. This gap between formal guarantees and practical privacy behavior motivates the present study.

In this work, we address these limitations by proposing Double Loss Differentially Private GAN (DL-DPGAN), a novel extension of DPGAN designed to address privacy and utility challenges. Our approach introduces a correlation-based privacy loss to reduce the similarity between real and synthetic data, thereby mitigating information leakage, while leveraging the Wasserstein distance to improve training stability. The proposed framework is particularly suited for privacy-critical data sharing scenarios, where synthetic data must preserve utility while minimizing empirical privacy risks beyond formal differential privacy guarantees. The main contributions of this paper are summarized as follows:

- We design DL-DPGAN, integrating a correlation-based privacy loss into the generator and discriminator to reduce data correlation and enhance protection against privacy attacks.
- We empirically show that, on the evaluated datasets, the proposed approach offers a competitive privacy–utility trade-off compared to standard GAN and DPGAN models.

- We investigate the impact of DPGAN-generated data on classifier performance and compare it to classifiers trained on standard GAN-generated data, offering insights into the practical utility of privacy-preserving synthetic data.
- We provide a comprehensive evaluation of DL-DPGAN on both the Adult dataset and the TON IoT Linux Process dataset, covering general tabular data and IoT network environments.
- We analyze privacy behavior through differential privacy budgets, entropy, and real-synthetic distinguishability (RSD), providing insights into the model's empirical resistance to privacy risks.

The structure of the paper is as follows: section 2 reviews previous studies and section 3 provides background information on DP, GANs, and DPGANs. Section 4 describes the proposed system model, including the DL-DPGAN architecture and the methods used to ensure DP. Section 5 defines the datasets, classifiers, and evaluation metrics used in this paper. Section 6 presents the experimental results on two real-world datasets. Lastly, section 7 concludes the paper, summarizing the key findings and their implications.

2 Related Work

In recent years, privacy-preserving ML has gained significant attention. Various privacy-enhancing techniques, including encryption, anonymization, (local) DP, and GANs models have proven to be effective solutions for safeguarding users' privacy [6][7]. In particular, GANs models have been employed in several real-world applications [8]. For instance, authors in [9] introduce DPGANs designed to create synthetic indoor location data. The approach incorporates DP to generate data that remains useful for analysis while ensuring privacy. By adding noise to the GAN's training process, the model provides privacy guarantees for sensitive location data. Zhang et al. [10] explore the use of DPGAN within a federated learning framework for detecting COVID-19 pneumonia. This approach enables the generation of private and diverse patient data, allowing hospitals to collaboratively train COVID-19 models without sharing the original data. The results demonstrate the high accuracy of this method in diagnosing COVID-19 while preserving patient privacy.

Privacy and security challenges in large-scale distributed and data-driven systems have been widely studied. Prior surveys emphasize risks such as information leakage, utility degradation, and computational overhead in privacy-preserving machine learning frameworks. However, comparatively less attention has been paid to the empirical privacy behavior of synthetic data generated under formal differential privacy guarantees, particularly regarding distributional similarity and memorization effects.

Beyond surveys, a number of studies have analyzed privacy-preserving mechanisms in IoT applications. Wakili and Bakkali [11] provide a comparative assessment of IoT security approaches, including cryptography, blockchain, machine learning, and fog/edge computing, emphasizing trade-offs and the potential of hybrid models for domains such as smart healthcare, smart grids, and smart cities. Similarly, Boujemaa et al. [12] explore the integration of AI into IoT ecosystems, discussing how

machine learning, deep learning, and reinforcement learning enhance automation and decision-making across sectors like healthcare, transportation, and agriculture. In parallel, several works have investigated synthetic data generation as a privacy-preserving strategy in IoT. Alabdulwahab et al.[13] propose a CTGAN-based approach for intrusion detection systems (IDS) in sensor networks, integrating DP with techniques such as quantile matching and dynamic KS adjustment to balance privacy and utility while mitigating risks like singling out, linkability, and inference attacks. Likewise, Han et al.[14] introduce a DPGAN framework that injects Gaussian noise during training to protect sensitive information while maintaining the usability of the generated data. Their approach demonstrates strong applicability to real-world IoT use cases, including healthcare and smart city applications.

Several studies have compared classifier performance under various privacy-preserving mechanisms. For example, Lopuhaä-Zwakenberg et al. [15] systematically evaluate classifiers under differential privacy, and Emadi et al. [16] investigate the impact of GAN- and DPGAN-generated data on classifier accuracy. The study presented in [17], investigates the impact of anonymization techniques on classifiers' performance. Training classifiers using synthetic data generated by GANs is examined in [18] and [16]. These research aims to evaluate the performance of classifiers trained on synthetic data compared to real data. The results indicate that data generated by GANs can successfully be used to train classifiers and achieve satisfactory performance. Rashid et al. in [19] use artificial data generated by GAN to improve the classification accuracy of skin lesions. This study shows that employing synthetic data generated by GAN can significantly enhance the performance of classification models. They have trained different classification models using real and synthetic data and compared the results. The authors in [20] also examines the impact of feature selection and privacy-preserving mechanisms. This research compares anonymization, DP, GAN, and DPGAN, and shows that the selection and arrangement of features in privatized data can affect the accuracy of ML models. The results of this study suggest a balance between privacy preservation and model accuracy, providing valuable insights for research related to the privacy of medical data.

Considering optimization algorithms in DPGANs, a framework based on DPGAN is presented in [4], which uses the RMSProp algorithm along with a DP mechanism to generate synthetic data. This study investigates the impact of noise injection on the GAN training process and demonstrates that appropriately adjusting the noise level can strike a balance between preserving privacy and maintaining the quality of synthetic data. Following this, the study [21] introduces the ADAM-DPGAN model with the aim of improving this approach, where ADAM is used as the optimization algorithm instead of RMSProp. This change leads to improved training stability and helps achieve more optimal performance in both privacy preservation and classifier accuracy.

Although numerous studies have explored privacy-preserving synthetic data generation using GANs and their differentially private variants, most approaches emphasize formal differential privacy guarantees, optimization strategies, or noise calibration techniques. Recent evidence suggests, however, that formal guarantees alone may not fully capture empirical privacy risks arising from distributional similarity, overfitting,

or memorization in generative models. This gap between formal privacy guarantees and practical privacy behavior motivates a closer investigation into empirical leakage in differentially private generative frameworks.

3 Preliminaries

This section explains the background on the proposed DL-DPGAN framework designed for privacy preservation. We examine DL-DPGAN within the context of the Wasserstein GAN (WGAN) model while highlighting its adaptability to other GAN architectures. To establish a foundation, we first introduce the concept of DP as a widely adopted technique for safeguarding data privacy, followed by a concise overview of WGAN and DPGAN.

3.1 Differential Privacy (DP)

Differential Privacy (DP) is a powerful framework for quantifying privacy loss in individuals when algorithms process aggregated data. The concept is defined in terms of adjacent databases, which, in machine learning, refers to two datasets that differ by just a single record. In essence, a differentially private algorithm should produce nearly identical outputs when given these two similar datasets. Below is the formal definition of DP.

ε-DP [22]: Let the privacy budget be denoted as $\varepsilon \in \mathbb{R}_{\geq 0}$, and let M represent an algorithm operating on datasets. We say that M satisfies ε-DP if, for any two adjacent datasets d and d', and for all possible output sets S, the following condition holds:

$$Pr[M(d) \in S] \leq e^{\varepsilon} Pr[M(d') \in S] \tag{1}$$

By minimizing the differences in the probability distributions of the outputs derived from two closely related input datasets, ε-DP ensures plausible deniability for the value of any individual record, even if other records are compromised. A smaller value of ε provides stronger privacy protection under ε-DP.

(ε, δ)-DP [22]: A randomized mechanism M defined over a set of databases D satisfies (ε, δ)-DP if, for any two neighboring databases d and d' in D that differ by only one record, and for any subset of possible outputs $S \subseteq R$, the following inequality holds:

$$Pr[M(d) \in S] \leq e^{\varepsilon} Pr[M(d') \in S] + \delta \tag{2}$$

Here, ε and δ represent the privacy budget and the confidence parameter, respectively. The privacy parameter δ provides more flexibility to DP because it allows for greater differences in the outputs between two adjacent datasets d and d', thereby enabling a balance between improving the utility of the generated data and maintaining privacy simultaneously. The Gaussian mechanism is commonly used to achieve (ε, δ)-DP. For a function $f : d \to R$, its L_2 sensitivity is defined as:

$$\Delta_2 f = \max_{d,d'} \|f(d) - f(d')\|_2 \tag{3}$$

To ensure (ε, δ)-DP, Gaussian noise is added to the output of f, that is, $M(d) = f(d) + \mathcal{N}(0, (\sigma \Delta_2 f)^2)$, where [23]:

$$\sigma^2 > \frac{2 \ln(1.25/\delta)}{\varepsilon} \tag{4}$$

3.2 GANs

Generative Adversarial Networks (GANs), introduced by Goodfellow et al. in 2014 [3], are designed to generate synthetic data that closely mirrors a given dataset by learning the distribution of its features through a game-theoretic approach. GANs consist of two main components: a **generator** (G) and a **discriminator** (D). The generator creates synthetic data from noise inputs sampled from a latent space, while the discriminator attempts to distinguish between real and fake data. The discriminator evaluates both real and synthetic data, classifying the synthetic data as either fake or real. This classification generates a loss value, which is then used to optimize both the generator and discriminator. The generator's goal is to produce data that the discriminator cannot distinguish from real data, while the discriminator strives to correctly identify whether the data is real or generated.

The training process of GANs follows a Minmax game framework, where the goal is to minimize the generator's loss while maximizing the discriminator's loss. The mathematical formulation of this dynamic is represented as:

$$\min_G \max_D \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}(\mathbf{x})}[\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z} \sim p_z(\mathbf{z})}[1 - \log D(G(\mathbf{z}))] \tag{5}$$

In this equation, $\mathbf{x}$ and $\mathbf{z}$ are random samples drawn from the data distribution $p_{\text{data}}(\mathbf{x})$ and the noise distribution $p_{\mathbf{z}}(\mathbf{z})$ (the *prior*), respectively. The first term of the equation represents the discriminator's ability to correctly classify real data, while the second term assesses its ability to distinguish between real and synthetic data generated by the generator.

Both the generator and discriminator are trained using separate loss functions, which assess how close the generator's outputs are to the actual labels (real or fake) of the data. The training process is iterative: in each cycle, the generator creates synthetic data, and the discriminator evaluates both the synthetic data and real data to classify them. The loss values from these evaluations are then used to update the weights of both the generator and discriminator. This process repeats for a predefined number of epochs.

While GANs have proven effective, their initial formulation tends to be unstable and inefficient during training. As a result, numerous studies have proposed modifications to improve training procedures and architectures, such as the Wasserstein GAN (WGAN) [24], to increase stability and enhance convergence.

Wasserstein GAN (WGAN) [24]: WGAN was proposed to improve training stability by replacing the Jensen–Shannon divergence with the Wasserstein distance. The objective is to reduce the distance between p_z and p_{data} through the following expression:

$$\min_G \max_{w \in W} \mathbb{E}_{x \sim p_{\text{data}}}[f_w(x)] - \mathbb{E}_{z \sim p_z}[f_w(G(z))] \tag{6}$$

Here, the functions $\{f_w(x)\}_{w \in W}$ are K-Lipschitz continuous (with respect to x) for a certain constant K. Our approach leverages the K-Lipschitz property of WGAN and solves Eq. 6 in a manner that guarantees DP.

3.3 Differentially Private GAN (DPGAN)

Differentially Private GAN (DPGAN) [4] offers a technique for achieving a balance between privacy and utility by generating synthetic data that mimics real data while ensuring that individual data points' specific information is not exposed. The DPGAN model used in this study adopts the WGAN architecture [24]. Building on the work of [25], several approaches have applied DP-SGD and GANs to generate synthetic data while preserving DP [4, 26]. In DPGAN, the DP mechanism introduces noise into the discriminator's gradients during training to protect the privacy of the training data. Ensuring the privacy of the discriminator also guarantees that the generator's parameters (which learn to produce new data) maintain privacy as well. This is based on the post-processing property of DP [23], which asserts that any operation applied to a differentially private output will not compromise privacy. The privacy budget in DP controls the level of noise added to the discriminator's gradients, represented by ε. Typically, reducing ε increases the noise, which could potentially affect the quality of the generated data.

Algorithm 1, outlines the steps involved in training the DPGAN using the Wasserstein distance. This algorithm ensures DP by introducing controlled noise into the training process while maintaining stability using gradient clipping and weight constraints. The key aspects of this approach are as follows:

- **Adding noise for DP:** In line 6 of algorithm, Gaussian noise with variance σ_n^2 is added to each individual gradient before computing their average. This ensures that the contribution of any single data point remains obscured, thereby satisfying DP requirements.
- **Weight clipping for Lipschitz continuity:** In line 9 of algorithm, the weights of the discriminator are clipped within a specified range $[-c_p, c_p]$. This weight clipping enforces a constraint on the discriminator's function values $\{f_w(x)\}$ to maintain a form of continuity called K_w-Lipschitz continuity with respect to the input x. This continuity constraint, controlled by an unknown constant K_w, helps control the sensitivity of the discriminator's gradients for each data point, further supporting DP.
- **RMSProp optimization for stability:** Steps 8 and 13 employ the RMSProp optimization algorithm. RMSProp dynamically adjusts the learning rate based on the magnitude of gradients, ensuring that the learning rate remains higher for smaller gradients and lower for larger gradients. This prevents extreme parameter updates and facilitates smoother convergence.

By integrating these techniques and leveraging the Wasserstein distance efficiently while maintaining DP guarantees, DPGAN seeks a balance between privacy preservation and stable GAN training.

Algorithm 1 Differentially Private GAN Training (DPGAN)

Input: α_d (discriminator learning rate), α_g (generator learning rate), c_p (weight clipping threshold), m (batch size), n_d (discriminator steps per generator step), n_g (total generator steps),
σ_n (noise scale for privacy), c_g (gradient clipping bound).
Output: Generator parameters θ.

Initialize discriminator weights w_0; Generator parameters θ_0.
for $t = 1$ to n_g **do**
  for $k = 1$ **to** n_d **do**
    Draw mini-batch of prior samples $\{z^{(i)}\}_{i=1}^{m} \sim p(z)$.
    Draw mini-batch of real samples $\{x^{(i)}\}_{i=1}^{m} \sim p_{\text{data}}(x)$.
    Compute gradient for each i:

$$g_w(x^{(i)}, z^{(i)}) \leftarrow \nabla_w \left(f_w(x^{(i)}) - f_w(g_\theta(z^{(i)})) \right)$$

    Average and add noise:

$$\bar{g}_w \leftarrow \frac{1}{m} \left(\sum_{i=1}^{m} g_w(x^{(i)}, z^{(i)}) + \mathcal{N}(0, \sigma_n^2 c_g^2 I) \right)$$

    Update discriminator: $w \leftarrow w + \alpha_d \cdot \text{RMSProp}(w, \bar{g}_w)$
    Clip weights: $w \leftarrow \text{clip}(w, -c_p, c_p)$
  end for
  Draw mini-batch of prior samples $\{z^{(i)}\}_{i=1}^{m} \sim p(z)$.
  Compute generator gradient:

$$g_\theta \leftarrow -\nabla_\theta \frac{1}{m} \sum_{i=1}^{m} f_w(g_\theta(z^{(i)}))$$

  Update generator: $\theta \leftarrow \theta - \alpha_g \cdot \text{RMSProp}(\theta, g_\theta)$
end for
return θ

4 Proposed Method

Here, we first explain the proposed DL-DPGAN, followed by a description of its overall architecture. We then present the privacy guarantee of the proposed DL-DPGAN.

4.1 Proposed DL-DPGAN

In the proposed model of this study, the core structure of the DPGAN Algorithm (Algorithm 1) based on the Wasserstein distance is preserved. The training process

of suggested DL-DPGAN (DoubleLoss-DPGAN) is outlined in algorithm 2, introducing an additional component called *correlation-based privacy loss* to enhance privacy protection to the objective functions of both the discriminator and the generator.

The addition of a correlation-based privacy loss is motivated by the need to mitigate indirect information leakage. Even when differential privacy guarantees are formally satisfied, synthetic data that exhibits high statistical correlation with the original dataset can enable reconstruction or inference of sensitive attributes, posing an empirical privacy risk. By explicitly reducing these dependencies during training, the model aims to lower such risks beyond the guarantees provided by differential privacy alone. To operationalize this, the correlation between corresponding real and synthetic samples within each mini-batch is computed and penalized. Minimizing this penalty encourages the generator to produce outputs that are less linearly correlated with the real data, thereby reducing the memorization of specific training samples and enhancing empirical privacy protection.

In the DL-DPGAN algorithm, the correlation-based privacy loss component is incorporated into both the discriminator and generator update steps in order to encourage the generated data to be less similar to the real data, thereby improving privacy protection during training. First, in line 7 of algorithm, the correlation between real data $x^{(i)}$ and synthetic data $G(z^{(i)})$ is computed for each sample, and the average absolute correlation is used as the discriminator's privacy loss:

$$\text{PrivacyLoss}_d = \frac{1}{m} \sum_{i=1}^{m} \left| \text{Corr} \left(x^{(i)}, G(z^{(i)}) \right) \right| \tag{7}$$

where m is the mini-batch size and $\text{Corr}(\cdot)$ denotes the Pearson correlation coefficient. Then, in line 8 of algorithm, this privacy loss is integrated into the discriminator's gradient update using the following formulation:

$$\bar{g}_\omega = \frac{1}{m} \left(\sum_{i=1}^{m} g_\omega(x^{(i)}, z^{(i)}) + \mathcal{N}(0, \sigma_n^2 c_g^2 I) \right) + \lambda_D \cdot \text{PrivacyLoss}_d \tag{8}$$

Similarly, in line 13, the generator's privacy loss is calculated with the same correlation formula:

$$\text{PrivacyLoss}_g = \frac{1}{m} \sum_{i=1}^{m} \left| \text{Corr} \left(x^{(i)}, G(z^{(i)}) \right) \right| \tag{9}$$

And in line 14, this loss is incorporated into the generator's gradient update:

$$g_\theta = -\nabla_\theta \frac{1}{m} \sum_{i=1}^{m} f_\omega(G(z^{(i)})) + \lambda_G \cdot \text{PrivacyLoss}_g \tag{10}$$

where ∇_θ denotes the gradient with respect to the generator parameters θ. The total loss functions for the discriminator and generator can be expressed more generally as:

$$\text{Total Loss}_D = \text{Utility Loss}_D + \lambda_D \times \text{Privacy Loss}_D \tag{11}$$

$$\text{Total Loss}_G = \text{Utility Loss}_G + \lambda_G \times \text{Privacy Loss}_G \tag{12}$$

The hyperparameters λ_D and λ_G regulate the influence of the privacy loss on the discriminator and generator updates, respectively. Adjusting these values directly impacts the trade-off between privacy and utility: larger λ values strengthen privacy by further reducing the resemblance between synthetic and real samples, but may impair the utility of the synthetic data. Conversely, smaller λ values favor higher data quality at the expense of reduced privacy. Thus, careful tuning of λ_D and λ_G is essential to properly balance privacy preservation and utility.

Correlation-based privacy loss

Let $x^{(i)} \in \mathbb{R}^d$ denote a real sample in a mini-batch of size m, and let $\tilde{x}^{(i)} = G(z^{(i)}) \in \mathbb{R}^d$ be the corresponding synthetic sample generated from the noise vector $z^{(i)}$. In our implementation, the operator $\mathrm{Corr}(\cdot)$ in eqs. (7) and (9) is instantiated as a feature-wise Pearson correlation computed over the current mini-batch.

For each feature $j \in \{1, \ldots, d\}$ we first compute the batch means and standard deviations of the real and synthetic samples as

$$\mu_j^{(r)} = \frac{1}{m}\sum_{i=1}^{m} x_j^{(i)}, \qquad \mu_j^{(f)} = \frac{1}{m}\sum_{i=1}^{m} \tilde{x}_j^{(i)}, \tag{13}$$

$$\sigma_j^{(r)} = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\big(x_j^{(i)} - \mu_j^{(r)}\big)^2}, \qquad \sigma_j^{(f)} = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\big(\tilde{x}_j^{(i)} - \mu_j^{(f)}\big)^2}. \tag{14}$$

The batch-wise covariance between the j-th coordinates of real and synthetic samples is then given by

$$\mathrm{cov}_j = \frac{1}{m}\sum_{i=1}^{m}\big(x_j^{(i)} - \mu_j^{(r)}\big)\big(\tilde{x}_j^{(i)} - \mu_j^{(f)}\big), \tag{15}$$

and the corresponding Pearson correlation coefficient is

$$\rho_j = \frac{\mathrm{cov}_j}{\sigma_j^{(r)}\sigma_j^{(f)}}. \tag{16}$$

We then aggregate the feature-wise correlations into a scalar correlation-based privacy loss by averaging the absolute values:

$$\mathrm{PrivacyLoss} = \frac{1}{d}\sum_{j=1}^{d}\big|\rho_j\big|. \tag{17}$$

Minimizing this term encourages the generator to produce synthetic samples that are less linearly correlated with the corresponding real samples in each mini-batch, thereby reducing memorization and improving empirical privacy.

4.2 The Overall Architecture

Figure 1 shows the overall architecture of the proposed model. This architecture includes a comprehensive process for generating synthetic data using the DL-DPGAN

Algorithm 2 DL-DPGAN: Wasserstein-Based Differentially Private GAN with Double Loss

Input: α_d (discriminator learning rate), α_g (generator learning rate), c_p (weight clipping threshold), m (batch size), n_d (discriminator steps per generator step), n_g (total generator steps), σ_n (noise scale for privacy), c_g (gradient clipping bound), λ_D (privacy loss weight for discriminator), λ_G (privacy loss weight for generator).
Output: Generator parameters θ.

Initialize discriminator weights w_0; Generator parameters θ_0.
for $t = 1$ **to** n_g **do**
  for $k = 1$ **to** n_d **do**
    Draw mini-batch of prior samples $\{z^{(i)}\}_{i=1}^{m} \sim p(z)$.
    Draw mini-batch of real samples $\{x^{(i)}\}_{i=1}^{m} \sim p_{\text{data}}(x)$.
    Compute gradient for each i:

$$g_w(x^{(i)}, z^{(i)}) \leftarrow \nabla_w \left(f_w(x^{(i)}) - f_w(g_\theta(z^{(i)})) \right)$$

    Compute Privacy Loss for Discriminator:

$$\text{PrivacyLoss}_d \leftarrow \frac{1}{m} \sum_{i=1}^{m} \left| \text{Corr}(x_{\text{real}}^{(i)}, x_{\text{fake}}^{(i)}) \right|$$

    Average, add noise, and include Privacy Loss:

$$\bar{g}_w \leftarrow \frac{1}{m} \left(\sum_{i=1}^{m} g_w(x^{(i)}, z^{(i)}) + \mathcal{N}(0, \sigma_n^2 c_g^2 I) \right) + \lambda_D \cdot \text{PrivacyLoss}_d$$

    Update discriminator: $w \leftarrow w + \alpha_d \cdot \text{RMSProp}(w, \bar{g}_w)$
    Clip weights: $w \leftarrow \text{clip}(w, -c_p, c_p)$
  end for
  Draw mini-batch of prior samples $\{z^{(i)}\}_{i=1}^{m} \sim p(z)$.
  Compute Privacy Loss for Generator:

$$\text{PrivacyLoss}_g \leftarrow \frac{1}{m} \sum_{i=1}^{m} \left| \text{Corr}(x_{\text{real}}^{(i)}, x_{\text{fake}}^{(i)}) \right|$$

  Compute generator gradient with Privacy Loss:

$$g_\theta \leftarrow -\nabla_\theta \frac{1}{m} \sum_{i=1}^{m} f_w(g_\theta(z^{(i)})) + \lambda_G \cdot \text{PrivacyLoss}_g$$

  Update generator: $\theta \leftarrow \theta - \alpha_g \cdot \text{RMSProp}(\theta, g_\theta)$
end for
Return θ

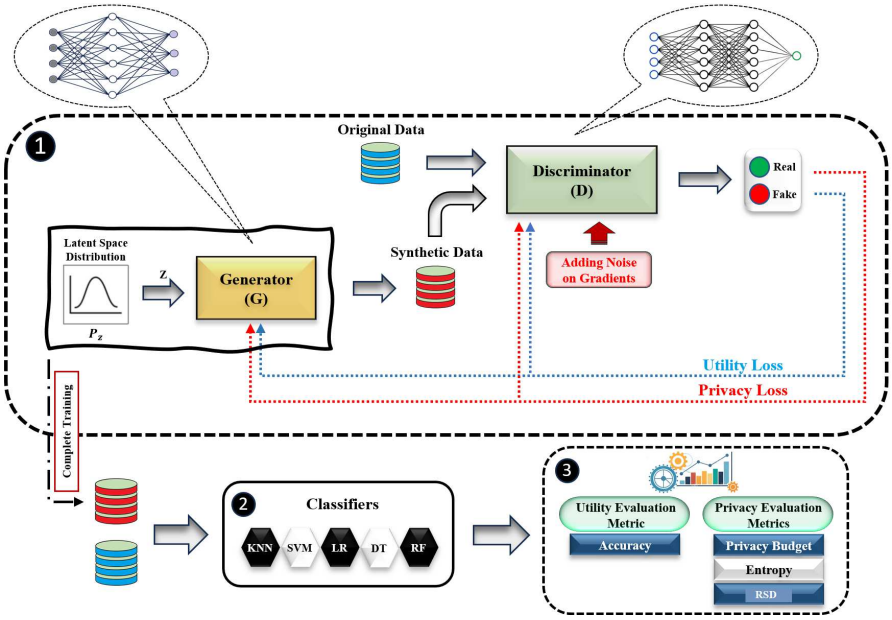


Fig. 1: The overall architecture of the suggested system model.

model and evaluating the utility and privacy of the generated data through various classifiers. The different sections of this process, labeled (1) to (3), each refer to a specific stage of the generation, usage, and evaluation of synthetic data, and they will be explained individually in the following.

- **(1) DL-DPGAN Training**: The training process follows the Wasserstein GAN framework. The generator maps random noise vectors to synthetic samples, while the discriminator tries to distinguish real from synthetic data. To enforce differential privacy, Gaussian noise is added to the discriminator's gradients. The key extension in DL-DPGAN is the introduction of a dual-loss mechanism. Alongside the utility loss (which promotes realism), a correlation-based privacy loss penalizes high linear dependency between real and synthetic samples within each mini-batch. The balance between these two losses, controlled by hyperparameters λ_D and λ_G, allows the model to seek a trade-off between data utility and empirical privacy. The complete training procedure is detailed in Algorithm 2. As mentioned, The DPGAN model adopted in this study is built on the WGAN architecture and optimized with RMSProp; this configuration serves as our primary baseline for evaluating the impact of the added correlation-based privacy loss.
- **(2) Classifiers**: In this step, the trained generator of DL-DPGAN is employed to generate synthetic data samples that not only closely resemble the real data distribution but also ensure the preservation of the privacy of the original dataset. Thereafter, classifiers are trained using the synthetic data produced by the trained DL-DPGAN model, and their performance is evaluated on real data. This approach

assesses the ability of models trained solely on synthetic data to generalize to real-world data, thereby highlighting the practical applicability of the generated synthetic datasets. For evaluation, several widely used classification algorithms are employed, including k-Nearest Neighbor (kNN), Support Vector Machine (SVM), Logistic Regression (LR), Decision Tree (DT), and Random Forest (RF).

- **(3) Evaluation Metrics**: In this section, various metrics are used to evaluate both the performance of the classifiers and the privacy preservation of the synthetic data. The utility metric include *Accuracy*, which assesses the effectiveness of the classifiers. Additionally, for evaluating the privacy preservation of the synthetic data, three metrics are employed: *Privacy Budget* (ε), *Entropy*, and *Real-Synthetic Distinguishability Test* (RSD). These metrics are specifically designed to measure the level of privacy preservation and the likelihood of sensitive information leakage in synthetic data. Detailed explanations of these metrics are provided in section 5. Subsequently, in the final, we analyze the results to assess the overall quality and utility of the synthetic data across various data analysis tasks.

4.3 Privacy Guarantee

Building on the definitions of (ε, δ)-DP and the Gaussian mechanism provided in section 3, we now derive the specific privacy guarantees of our DL-DPGAN framework.

Our privacy analysis is based on the differentially private stochastic gradient descent (DP-SGD) framework introduced by Abadi et al. [25], and follows the structure of the privacy guarantee in DPGAN [4]. In particular, we ensure (ϵ, δ)-DP by adding calibrated Gaussian noise to the discriminator's gradients. The cumulative privacy loss is tracked using the Moments Accountant technique. A detailed explanation of gradient sensitivity and the required noise level is provided below.

In our DL-DPGAN framework, the discriminator directly accesses the real training data. During training, the discriminator's gradients are clipped to a fixed norm c_g, and Gaussian noise $\mathcal{N}(0, \sigma_n^2 c_g^2 I)$ is added to ensure privacy. These updates satisfy the conditions of the Gaussian Mechanism, and thus each gradient step is differentially private. The generator is updated solely based on the output of the discriminator, which has already been privatized through noise injection. Therefore, by the post-processing property of DP, the generator's parameters θ would theoretically be differentially private. However, since our generator also incorporates a correlation-based regularization term that directly accesses real data, it does not fully satisfy DP guarantees. As such, the formal guarantee applies only to the discriminator component.

In addition to the standard DP mechanism, our algorithm includes a correlation-based privacy loss term, designed to reduce similarity between real and synthetic records. Although this term enhances empirical privacy by discouraging memorization, it is not part of the formal DP guarantee, as it does not involve any randomized mechanism or noise injection.

4.3.1 Bounding the Gradient Norm

To apply the Gaussian mechanism, we must first bound the ℓ_2-norm of the discriminator's gradients. Let $f_w(x)$ be the output of the discriminator for input x with

parameters w. The per-sample gradient is given by:

$$g_w(x^{(i)}, z^{(i)}) = \nabla_w \left(f_w(x^{(i)}) - f_w(g_\theta(z^{(i)})) \right). \tag{18}$$

We bound its norm as follows:

$$\left\| g_w(x^{(i)}, z^{(i)}) \right\| \leq \left\| \nabla_w f_w(x^{(i)}) \right\| + \left\| \nabla_w f_w(g_\theta(z^{(i)})) \right\|. \tag{19}$$

Let L denote the number of layers in the discriminator. Assuming each layer uses a Lipschitz-continuous activation function σ with bounded derivative $\|\sigma'\| \leq B_\sigma$, and weights are clipped to $[-c_p, c_p]$, we can recursively bound the gradient as:

$$\left\| \nabla_w f_w(x^{(i)}) \right\| \leq 2c_p B_\sigma^L \cdot \prod_{l=1}^{L-1} m_l, \tag{20}$$

where m_l is the number of units in layer l. Since generator outputs $g_\theta(z^{(i)})$ are also bounded due to the final sigmoid activation, the same bound holds for $\nabla_w f_w(g_\theta(z^{(i)}))$.

Therefore, the total per-sample gradient is bounded as:

$$\left\| g_w(x^{(i)}, z^{(i)}) \right\| \leq c_g, \tag{21}$$

where $c_g = 2c_p B_\sigma^L \cdot \prod_{l=1}^{L-1} m_l$ is the clipping threshold applied before adding Gaussian noise.

Lemma 1. *Given a sampling probability $q = \frac{m}{M}$, number of discriminator iterations per generator step n_d, and a target privacy failure probability δ, the discriminator satisfies (ϵ, δ)-DP for any $\epsilon > 0$ if the Gaussian noise added to discriminator gradients has standard deviation*

$$\sigma_n = \frac{2q\sqrt{n_d \log(1.25/\delta)}}{\epsilon}. \tag{22}$$

Proof. This result is derived by applying the Gaussian mechanism in conjunction with the Moments Accountant method [25]. Specifically, we combine the inequality $n_d q^2 \lambda^2 / \sigma^2 \leq \epsilon\lambda/2$ with the tail bound $e^{-\lambda\epsilon/2} \leq \delta$, and solve for σ under equality. This yields the required noise scale σ_n that guarantees (ϵ, δ)-DP. □

Theorem 2. *If the discriminator is trained according to algorithm 2 with per-sample gradient clipping bound c_g and Gaussian noise with standard deviation σ_n as defined in Lemma 1, then the discriminator satisfies (ϵ, δ)-DP with respect to the training dataset.*

However, since the generator incorporates a correlation-based regularization term that accesses real data, the generator does not satisfy DP. The overall privacy guarantee of the model is therefore limited to the discriminator component. It is important to note that, in the considered application scenario, the trained generator is not intended to be released. Instead, the generator is used internally to produce synthetic

datasets, which are then shared for downstream analysis. Therefore, privacy evaluation focuses on the risk of information leakage in the released synthetic data rather than on providing end-to-end differential privacy guarantees for the model itself.

Remark. Lemma 1 shows that larger sampling rates q or more discriminator steps n_d increase the privacy cost, requiring more noise to achieve a given ϵ. On the other hand, smaller q and fewer updates reduce privacy cost. Although the generator uses real data through a correlation-based regularizer, which prevents full DP guarantees, this mechanism improves empirical privacy. Future extensions could explore noise injection into the generator or redesign the correlation term to achieve end-to-end formal DP.

5 Experimental Setup and Metrics

This section presents the experimental setup and results of our study. We conducted experiments to evaluate the performance of various classification models on multiple datasets. In our evaluation, classifiers were trained exclusively on synthetic data generated by the proposed DL-DPGAN model and tested on real data, with their performance measured in terms of Accuracy to assess the practical utility of the synthetic samples. Moreover, the privacy-preserving capabilities of the generated data were evaluated using multiple privacy metrics, including Privacy Budget (ε), Entropy, and Real-Synthetic Distinguishability Test (RSD). All reported results correspond to the average of five independent evaluation runs under identical hyperparameter settings. In each run, downstream classifiers were evaluated using 5-fold cross-validation with independently generated data splits to reduce the effect of stochastic variability.

It should be noted that all generative models adopted in this study, including GAN, DPGAN, and the proposed DL-DPGAN, are implemented within the Wasserstein GAN (WGAN) framework, with RMSProp used as the default optimizer across all models, following the canonical DPGAN training setup. In addition, Adam is applied once to all models solely for comparison with RMSProp, while keeping the model architecture, training objective, and privacy budget unchanged.

All models were implemented and trained using PyTorch with CPU backend. The experiments were conducted on a system equipped with an Intel Core i7-13650HX CPU (14 cores, 20 threads, 2.6 GHz) and 16 GB of RAM, running Windows 11 Enterprise. No GPU acceleration was used (CUDA was disabled), and all training was performed solely on the CPU. Models were trained using the RMSProp optimizer for 100 epochs across different architectures (GAN, DPGAN, and DL-DPGAN).

5.1 Datasets

In this study, we perform our analysis using two datasets, the Adult [1] and the TON_IoT Linux Process [2] datasets, each characterized by its size, number of features, and class labels, as described below:

Adult dataset: This dataset includes 48,842 instances and 14 features, consisting of both numerical (e.g., age, hours worked per week, education number) and categorical

[1] https://archive.ics.uci.edu/datasets
[2] https://research.unsw.edu.au/projects/toniot-datasets

features (e.g., work class, marital status, occupation, native country). It is binary-class, with labels ' $\leq 50K$' for individuals with income less than or equal to \$50,000 and ' $> 50K$' for those with income greater than \$50,000.

TON_IoT Linux Process (Linuxprocess) dataset: This dataset contains 90,112 instances and 14 features describing Linux process execution statistics, such as process identifiers, CPU usage, memory consumption, and I/O activity, collected from an IoT network environment. While originally multi-class, covering different types of cyberattacks (e.g., DoS, DDoS, injection, password, scanning, XSS, MITM) and normal activity, it is used here as a binary classification task, distinguishing only between attack and non-attack cases. For this study, a balanced random subset of 12,112 instances was selected to ensure equal representation between the two classes.

Both datasets are relatively large and contain a mix of numerical and categorical features; this mixture allows entropy to capture the structural complexity and variability present in the data. Moreover, the balanced class distributions in both datasets ensure that performance evaluations are not biased toward majority classes, while the diversity of features provides a more realistic testbed for evaluating synthetic data generation under privacy constraints. The selected datasets provide complementary characteristics in terms of sample size, feature diversity, and structural complexity. Evaluating the proposed approach on datasets with varying statistical properties allows us to assess its robustness under different data distributions. Evaluating the proposed method on both a recognized benchmark and a larger, more intricate dataset enables an assessment of its robustness and generalization across diverse data characteristics and scales. This approachs justifies the use of these two datasets.

5.1.1 Data preprocessing

For all tabular datasets considered in this work, we apply a unified preprocessing pipeline before training the generative models and downstream classifiers. Each record is transformed into a fixed-length numerical feature vector $x^{(i)} \in \mathbb{R}^d$.

Given a raw dataset, we first select the attributes relevant to the prediction task and split them into continuous and categorical features. Missing values are handled by replacing special markers (such as question mark "?") with `NaN` and imputing continuous features with their empirical mean and categorical features with the most frequent category (mode) in the corresponding column.

All continuous features are then standardized using the `StandardScaler` implementation from scikit-learn, so that each continuous dimension has approximately zero mean and unit variance on the training data. For categorical features, we enumerate the distinct categories of each attribute and apply a simple one-hot encoding. The resulting one-hot vectors are concatenated across all categorical attributes and appended to the standardized continuous features, yielding a combined real-valued representation that serves as input to the discriminator and as the target space for the generator.

When generating synthetic samples, we invert this preprocessing pipeline to obtain data in the original tabular format. Continuous features are mapped back to their original scale using the inverse transform of the fitted scaler, and categorical features

are reconstructed from their one-hot encodings by selecting the most likely category for each attribute.

5.2 Classification Models

We evaluate the performance of five widely recognized classification algorithms: k-Nearest Neighbors (kNN), Support Vector Machine (SVM), Logistic Regression (LR), Decision Tree (DT), and Random Forest (RF). A brief introduction to these classifiers is provided below, with further details available in [27].

Support Vector Machines (SVM) is powerful algorithm for data classification. SVM aims to find the best line or hyperplane (in higher dimensions) that separates the data of two classes. This line is chosen in such a way that the margin between the closest samples of each class (support vectors) is maximized, which helps reduce classification error on new data.

k-Nearest Neighbors (kNN) algorithm is a simple and intuitive method for classification based on sample similarity. In kNN, to predict the class of a new sample, the k nearest samples in the feature space are searched, and the majority class of these k samples is chosen as the predicted class. This algorithm requires no training, and all computations are performed at prediction time.

Logistic Regression (LR) is a statistical model for binary classification that models the probability of a sample belonging to one of two classes. This model uses the logistic (sigmoid) function to transform the linear combination of features into a probability value. Logistic Regression is widely used due to its simplicity and effectiveness in many classification problems.

Decision Tree (DT) is a classification model that uses a tree structure for decision making. Each internal node in the tree represents a feature (or condition), and each branch represents the outcome of a condition. The leaves of the tree represent the final class. The decision tree is constructed by splitting the data based on various features such that each internal node provides the most information about the class of the samples.

Random Forest (RF) is a classification algorithm that combines multiple decision trees trained randomly and with replacement from the data to improve accuracy and reduce overfitting. Each tree in the random forest is trained independently, and during prediction, the predicted classes by the trees are aggregated, and the majority class is chosen as the final output. This algorithm is known for its robustness and stability due to the ensemble of multiple models.

5.3 Utility Evaluation Metric

This section introduces the evaluation metric used to assess the model's performance. The dataset is first split into separate training and testing sets. The model is trained on the training set and evaluated on the testing set based on accuracy, which measures the proportion of correctly classified instances (both positive and negative) out of all predictions. The formula for calculating accuracy is given by:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{23}$$

where:

- **True Positives (TP)**: The number of instances correctly predicted as positive.
- **True Negatives (TN)**: The number of instances correctly predicted as negative.
- **False Positives (FP)**: The number of instances incorrectly predicted as positive.
- **False Negatives (FN)**: The number of instances incorrectly predicted as negative.

5.4 Privacy Evaluation Metrics

In this study, three privacy evaluation metrics are utilized: the privacy budget (ε), entropy, and Real-Synthetic Distinguishability (RSD). Below, we provide detailed descriptions of each metric and explain their significance in measuring the privacy protection level of the generated synthetic data.

5.4.1 Privacy Budget (ε)

The privacy budget, represented by ε, is a fundamental concept in DP [22] that manages the trade-off between privacy preservation and data utility. Within the DPGAN framework, ε determines the magnitude of noise injected into the discriminator's gradient updates during training to obscure individual data contributions. A smaller ε value corresponds to stronger privacy protection through the addition of more noise, thereby reducing the risk of revealing sensitive information. On the other hand, larger ε values result in less noise, enhancing the utility of the generated data but potentially weakening privacy safeguards.

The concept of the privacy budget and its role in maintaining data privacy is formally defined in section 3. In our experiments, we assess multiple ε values to examine how different privacy levels influence the utility of the synthetic data. This evaluation highlights the trade-off between privacy preservation and classifier performance as the privacy budget varies.

5.4.2 Entropy

Entropy [28], a fundamental concept in information theory introduced by Claude Shannon, quantifies the amount of uncertainty or randomness in a probability distribution. For a discrete random variable X with possible outcomes $x_1, x_2, \ldots, x_n$ and associated probabilities $P(x_1), P(x_2), \ldots, P(x_n)$, the entropy $H(X)$ is defined as:

$$H(X) = -\sum_{i=1}^{n} P(x_i) \log(P(x_i)) \tag{24}$$

where $P(x_i)$ represents the probability of occurrence of x_i. The logarithmic base chosen in entropy calculations determines its units. Base 2 (measured in bits) is commonly used in digital contexts, where each bit represents information needed to distinguish between two outcomes. Natural logarithm (base e) is more suited for continuous probability distributions, often found in statistical modeling. In this study, we use the natural logarithm to align entropy units with the probabilistic nature of our synthetic data. Eq. 24 measures the expected amount of information or unpredictability

associated with the random variable X. A higher entropy value signifies increased uncertainty in outcome prediction, indicating that the distribution is more uniform and diverse. In the context of synthetic data, elevated entropy levels suggest greater variability among data samples, which can enhance privacy by minimizing the risk of reidentifying individual records.

In this work, entropy is employed to assess the level of privacy protection in both real and synthetic datasets. For each feature, entropy is individually computed, and the mean of these values reflects the overall dataset entropy. By comparing the average entropy between real and synthetic data, we evaluate the degree of similarity between them. A higher entropy value in the synthetic data suggests stronger privacy preservation, as greater diversity makes it more difficult to infer specific information about individuals.

5.4.3 Real–Synthetic Distinguishability (RSD)

To assess empirical privacy behavior, we employ a metric referred to as Real–Synthetic Distinguishability (RSD). The objective of this evaluation is to measure how easily synthetic samples can be distinguished from real samples. In the context of privacy-preserving synthetic data generation, a lower distinguishability between real and synthetic datasets suggests that the generated data more closely approximates the statistical characteristics of the original dataset, thereby reducing the risk that synthetic samples reveal identifiable information about the underlying data distribution.

Specifically, as illustrated in Algorithm 3, synthetic samples generated by DL-DPGAN are combined with the corresponding real dataset. Real samples are labeled as 1, while synthetic samples are labeled as 0. The combined dataset is then randomly partitioned into training and testing subsets.

A Multilayer Perceptron (MLP) classifier is trained to differentiate between real and synthetic samples using binary cross-entropy loss. After training, the classifier is evaluated on the test set, and the resulting classification accuracy is reported as the Real–Synthetic Distinguishability (RSD) score.

An RSD value close to 50% indicates that the classifier cannot reliably distinguish between real and synthetic samples, suggesting low separability between the two distributions. In contrast, higher RSD values indicate that real and synthetic samples are more easily distinguishable, implying greater distributional differences and a potentially higher risk of empirical information leakage.

6 Experimental Results

This section reports model utility, privacy, optimizer sensitivity, and computational cost. We evaluate on Adult and TON IoT Linuxprocess with the same train/test splits across models and privacy budgets. Unless noted, results are averaged across 5 runs.

6.1 Accuracy of DL-DPGAN

To assess the effect of the proposed correlation-based privacy loss in the DL-DPGAN framework, experiments were conducted on both datasets. The privacy budget was

Algorithm 3 Real–Synthetic Distinguishability (RSD)

Inputs: $\mathcal{D}_{\text{real}}$, $\mathcal{D}_{\text{fake}}$, f_{RSD}, n_{epochs}, m_{batch}
Outputs: Acc_{RSD}

Data Preparation:
Assign labels $\mathcal{Y}_{\text{real}} \leftarrow 1$, $\mathcal{Y}_{\text{fake}} \leftarrow 0$. Combine and split into $\mathcal{D}_{\text{train}}$, $\mathcal{D}_{\text{test}}$.
for $t = 1$ **to** n_{epochs} **do**
    Sample mini-batch $\{x^{(i)}, y^{(i)}\}_{i=1}^{m_{\text{batch}}}$ from training data.
    Compute loss:

$$\mathcal{L} = -\frac{1}{m_{\text{batch}}} \sum_{i=1}^{m_{\text{batch}}} \left(y^{(i)} \log(\hat{y}^{(i)}) + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)}) \right)$$

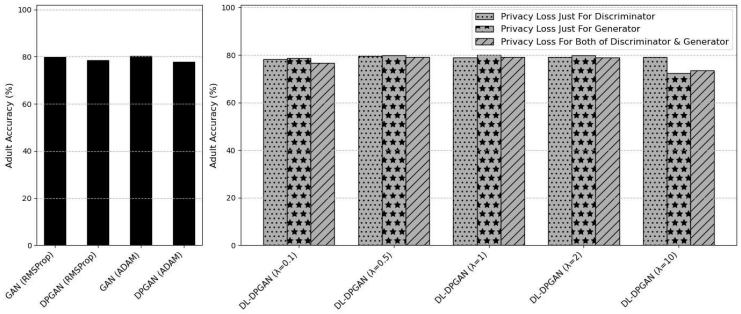
    Update f_{RSD} weights: $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}$
end for
Evaluate:
Compute $\text{Acc}_{\text{RSD}} = \frac{1}{|\mathcal{D}_{\text{test}}|} \sum_{i=1}^{|\mathcal{D}_{\text{test}}|} \mathbb{I}[\hat{y}^{(i)} = y^{(i)}]$

first fixed at $\varepsilon = 1$ to isolate the impact of the new privacy loss term from varying noise levels. The total loss functions for the discriminator and generator combine utility and privacy loss components, as shown in Eqs. 11 and 12, where λ_D and λ_G are tunable hyperparameters that control the relative contribution of the privacy loss in each update step, respectively. To analyze the sensitivity of the model to these hyperparameters, experiments were conducted with different values of λ (0.1, 0.5, 1, 2, and 10). In the baseline setting, where no privacy loss terms are applied, two optimization algorithms, RMSProp and Adam (with $\beta_1 = 0.5$ and $\beta_2 = 0.9$), were used based on the ADAM-DPGAN [21] configuration to enable a performance comparison between the two optimizers. However, for all DL-DPGAN experiments, only RMSProp was employed to specifically focus on the effect of the proposed correlation-based privacy loss mechanism on model performance.

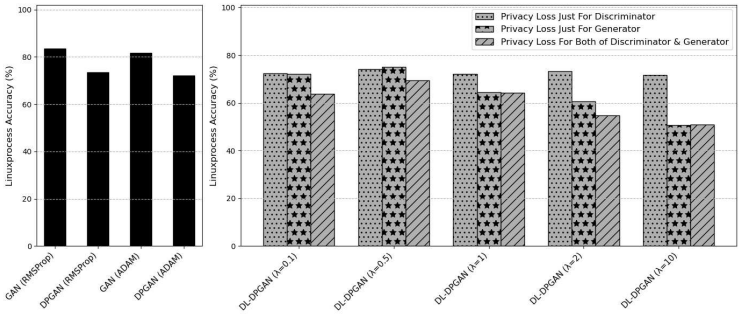
Three configurations for applying the privacy loss term were evaluated:

- Discriminator-only: Privacy loss applied only to the discriminator ($\lambda_G = 0$).
- Generator-only: Privacy loss applied only to the generator ($\lambda_D = 0$).
- Both networks: Privacy loss applied to both discriminator and generator ($\lambda_D, \lambda_G \neq 0$).

Figures 2a and 2b show the classification accuracy, while Figures 3a and 3b present the RSD results. For classification accuracy, the right panels in Figures 2a and 2b indicate that the generator-only configuration maintains slightly better performance compared to other privacy-loss settings. This suggests that applying the privacy loss exclusively to the generator enhances privacy while effectively preserving data utility across both datasets. In contrast, the left panels (GAN and DPGAN without privacy loss) show a slight decrease in accuracy when moving from GAN to DPGAN due to the noise introduced by differential privacy. On the Adult dataset, ADAM marginally outperforms RMSProp for both GAN and DPGAN, whereas RMSProp



(a) Adult



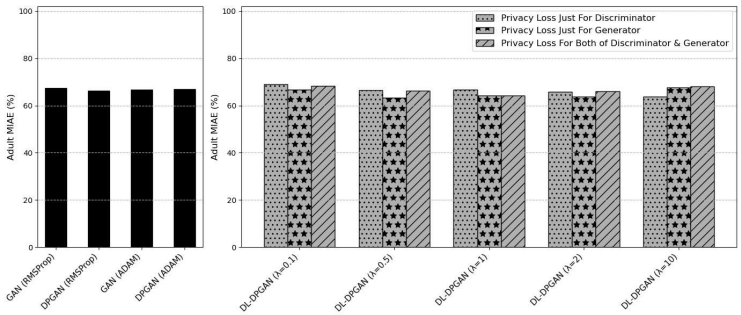
(b) Linuxprocess

Fig. 2: Accuracy comparison of classifiers trained on synthetic data generated from different GAN-based models for (a) the Adult dataset and (b) the Linuxprocess dataset. The left chart shows the performance of standard GAN and DPGAN models under two optimizers (RMSProp and ADAM), while the right chart presents the accuracy of synthetic data generated by DL-DPGAN under different values of generator privacy loss intensity (λ_G), considering three settings: privacy loss applied to the generator only, the discriminator only, and both.

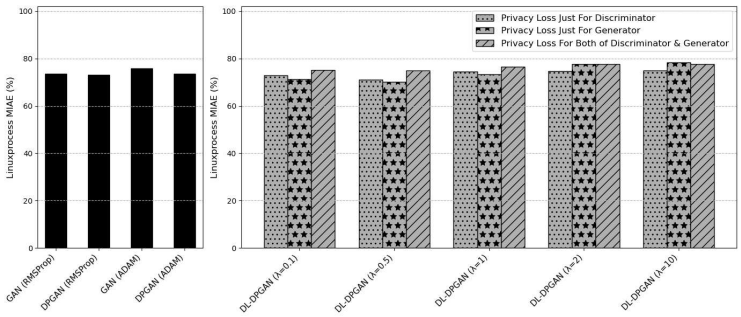
performs slightly better on the Linuxprocess dataset, reflecting that optimizer effectiveness can be dataset-dependent. Overall, the differences are minor, demonstrating stable performance across optimizers.

Regarding RSD, values closer to 50% indicate stronger resistance to distinguishability, representing improved privacy protection. As shown in the right panels of Figures 3a and 3b, the generator-only configuration (star-patterned bars) achieves the lowest RSD among all settings for both datasets. This means that synthetic samples are less distinguishable from real data, confirming better privacy preservation compared to the discriminator-only and both-networks configurations.

Increasing the privacy loss coefficient (λ) is generally expected to improve privacy by moving RSD closer to 50%. In practice, Figures 3a and 3b show that this trend holds for smaller λ values (up to approximately 0.5) but reverses at higher values. Excessive privacy pressure ($\lambda_G > 0.5$) degrades the quality of the generated samples, making



(a) Adult



(b) Linuxprocess

Fig. 3: RSD results on synthetic data generated using different GAN-based models for (a) the Adult dataset and (b) the Linuxprocess dataset. The left panel shows the baseline RSD scores for standard GAN and DPGAN models under RMSProp and ADAM optimizers. The right panel illustrates RSD scores for DL-DPGAN under varying generator privacy loss intensities (λ_G) with three configurations: applying the privacy loss to the generator only, discriminator only, and both.

them more distinguishable from real data and increasing RSD. Therefore, $\lambda_G = 0.5$ represents the best trade-off in these experiments, achieving improved privacy while maintaining acceptable data utility.

Finally, the left panels of Figures 3a and 3b (no privacy loss) show relatively high RSD values, around 65–70% for the Adult dataset and 73–76% for the Linuxprocess dataset, indicating that both GAN and DPGAN models are more vulnerable to information leakage when no privacy mechanism is applied.

In contrast, the right panels (privacy-aware scenarios), particularly the generator-only configuration with $\lambda_G = 0.5$, show lower RSD values for both datasets, indicating improved protection. Overall, DL-DPGAN with generator-only privacy loss achieves slightly higher classification accuracy than other configurations, confirming that this approach provides an effective balance between privacy and utility.

Figure 4 depicts the privacy–utility relationship using predictive performance (classification accuracy as utility) and RSD accuracy (an empirical proxy for real–synthetic

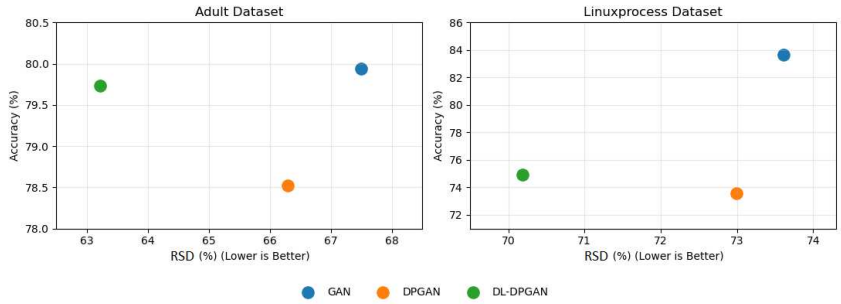


Fig. 4: Privacy–utility trade-off for GAN, DPGAN ($\varepsilon = 1$), and DL-DPGAN ($\varepsilon = 1$, $\lambda_G = 0.5$) under RMSProp on the Adult and Linuxprocess datasets. Utility is measured by classification accuracy, and empirical privacy is proxied by RSD accuracy (lower indicates reduced real–synthetic separability).

separability, where lower values indicate reduced separability). Across both datasets, introducing differential privacy (DPGAN) is associated with lower RSD values compared to standard GAN, and adding the proposed correlation-based penalty term (DL-DPGAN) leads to a further decrease. This pattern suggests that the penalty term reduces empirical similarity between real and synthetic samples under the considered settings.

Under identical $\varepsilon = 1$ and RMSProp configurations, DL-DPGAN achieves classification accuracy comparable to (and slightly higher than) DPGAN on both datasets. This indicates that, in the generator-penalized setting with $\lambda_G = 0.5$, reduced empirical separability (lower RSD), as a proxy for privacy leakage risk can be obtained without reducing predictive performance relative to the DPGAN baseline. This observation is consistent with the Figures 3 and 2 results, where $\lambda_G = 0.5$ provides a favorable operating point in terms of reduced separability and predictive performance.

Compared to non-private GAN, both DPGAN and DL-DPGAN exhibit lower accuracy, which aligns with the expected privacy–utility trade-off introduced by differential privacy. Notably, the utility degradation is more pronounced on the Linuxprocess dataset, highlighting that the privacy–utility trade-off is dataset-dependent and that stronger privacy regularization may incur higher utility costs in more complex, high-dimensional tabular settings.

6.2 Privacy Budget Effects

So far, analyses focused on varying privacy loss strength with a fixed privacy budget ($\varepsilon = 1$), reporting average accuracy across five classifiers. In this part, we retain the best configuration ($\lambda_G = 0.5$) from previous section and assess DL-DPGAN under varying privacy budgets, evaluating each classifier individually. This allows a finer understanding of DP's impact on synthetic data quality and classifier-specific performance. DL-DPGAN is trained using both ADAM and RMSProp across $\varepsilon \in \{0.5, 1, 2, 10\}$ on both datasets. For comparison, standard GANs without privacy loss are trained under the same conditions.

Table 1: Accuracy of five classifiers for DL-DPGAN under different privacy budgets, with generators trained using a privacy loss function with $\lambda_G = 0.5$. Results are reported for both datasets under RMSProp and ADAM optimizers. The performance of the original GAN (without privacy loss function) is also included for comparison.

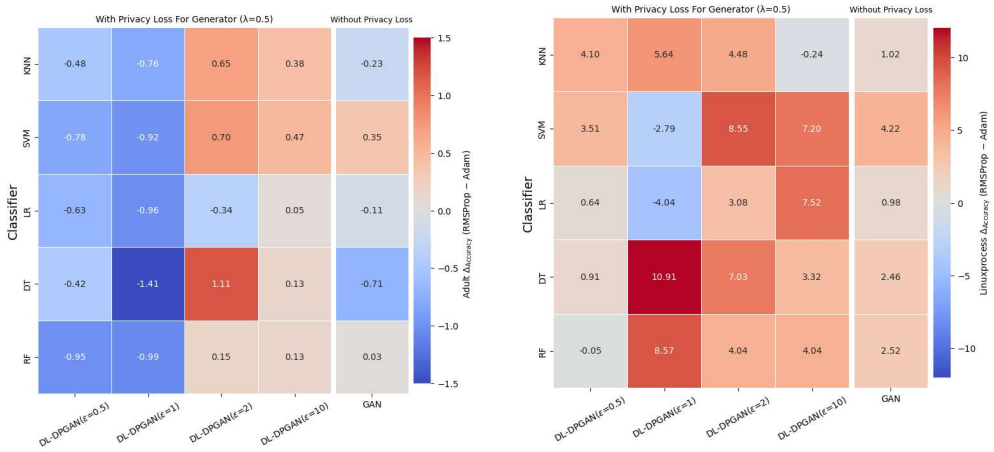
Dataset	Optimizer	Classifier	Privacy Budget ($\lambda = 0.5$ for Generator Privacy Loss)				GAN
			$\varepsilon = 0.5$	$\varepsilon = 1$	$\varepsilon = 2$	$\varepsilon = 10$	
Adult	RMSProp	KNN Accuracy (%)	78.25	78.49	79.76	80.32	79.99
		SVM Accuracy (%)	78.99	79.34	80.70	81.44	81.28
		LR Accuracy (%)	78.66	79.22	80.19	80.52	80.60
		DT Accuracy (%)	75.20	75.35	77.49	77.34	76.76
		RF Accuracy (%)	79.54	80.35	81.30	81.98	81.91
	ADAM	KNN Accuracy (%)	78.72	79.25	79.12	79.93	80.21
		SVM Accuracy (%)	79.77	80.26	80.00	80.97	80.93
		LR Accuracy (%)	79.28	80.18	80.54	80.48	80.71
		DT Accuracy (%)	75.62	76.76	76.38	77.21	77.47
		RF Accuracy (%)	80.49	81.34	81.15	81.85	81.88
Linuxprocess	RMSProp	KNN Accuracy (%)	66.57	78.39	78.29	80.95	82.99
		SVM Accuracy (%)	58.81	69.74	75.38	81.09	85.55
		LR Accuracy (%)	54.96	61.37	62.78	69.32	73.46
		DT Accuracy (%)	55.83	71.74	74.95	76.62	88.37
		RF Accuracy (%)	60.73	86.60	86.08	93.89	96.28
	ADAM	KNN Accuracy (%)	62.47	72.74	73.81	81.19	81.97
		SVM Accuracy (%)	55.30	72.53	66.82	73.89	81.32
		LR Accuracy (%)	54.32	65.41	59.71	61.80	72.47
		DT Accuracy (%)	54.92	60.82	67.91	73.30	85.90
		RF Accuracy (%)	60.78	78.03	82.04	89.85	93.77

To comprehensively evaluate the impact of privacy budgets, optimizer selection, and classifier choice, Table 1 summarizes the classification accuracy of five classifiers trained on synthetic data generated by DL-DPGAN under varying privacy budgets, alongside a standard GAN model without privacy constraints. The models were optimized using either RMSProp or ADAM. These results provide insights into how privacy settings and optimizer strategies affect the utility of synthetic data across different classifiers.

For both datasets and both optimizers, increasing the privacy budget (ε), which corresponds to reducing the strength of DP noise, generally leads to higher classification accuracy. This trend is expected, as lower noise allows the generator to produce higher-quality synthetic samples. Conversely, at lower ε values ($\varepsilon = 0.5$), stronger noise degrades data quality and reduces accuracy. As anticipated, models trained with standard GANs, which do not incorporate any privacy mechanism, achieve higher accuracy than DL-DPGAN models due to the absence of privacy-induced noise. Across both datasets, RF consistently achieves the highest accuracy, reflecting its ensemble structure that reduces variance by aggregating multiple decision trees. In the Adult dataset, DT perform the worst, likely due to overfitting on noisy synthetic data, while LR and SVM achieve moderate accuracy. For the Linuxprocess dataset, however, DT performs substantially better and, in some cases, even surpasses SVM and KNN, suggesting that tree-based models are particularly well-suited to this type of data. Conversely, LR exhibits the weakest performance on Linuxprocess, highlighting its sensitivity to complex synthetic distributions.

6.3 Optimizer Effects (RMSProp vs. ADAM)

Figure 5 illustrates the difference in classification accuracy between RMSProp and ADAM optimizers across different privacy budgets and classifiers for both datasets.



(a) Adult (b) Linuxprocess

Fig. 5: Difference in classification accuracy between RMSProp and ADAM optimizers for five classifiers trained on synthetic data generated by DL-DPGAN (with $\lambda_G = 0.5$) under different privacy budgets (ε) on both datasets. Negative values (black) indicate better performance with ADAM, while positive values (red) reflect higher accuracy with RMSProp. The standard GAN without privacy loss is also included for comparison. (a) Adult dataset, (b) Linuxprocess dataset. All values are in percentages.

Negative values (black) indicate that ADAM outperforms RMSProp, while positive values (red) show the opposite.

- For the Adult dataset (Figure 5a), the heatmap reveals that at lower privacy budgets ($\varepsilon = 0.5$ and $\varepsilon = 1$), most classifiers exhibit negative differences in accuracy, indicating that ADAM consistently outperforms RMSProp. This performance gain is especially noticeable for the DT classifier at $\varepsilon = 1$ (diff = -1.41%) and for SVM and RF across both $\varepsilon = 0.5$ and $\varepsilon = 1$. The advantage of ADAM in these settings can be attributed to its adaptive learning rate, which leverages historical first- and second-order moments, enabling more stable and effective training under high-noise (low-ε) conditions. Conversely, at higher privacy budgets ($\varepsilon = 2$ and $\varepsilon = 10$), the trend partially reverses. Several classifiers, such as KNN, SVM, and DT, show positive difference accuracy, suggesting that RMSProp starts to outperform ADAM when the privacy constraints are relaxed, and less noise is injected during training. For instance, DT achieves a notably higher accuracy with RMSProp at $\varepsilon = 2$ (diff = +1.11%), and SVM and KNN also see improvements. For the standard GAN without privacy loss, the results are mixed. ADAM slightly outperforms RMSProp for SVM and LR, while RMSProp does better with DT. This suggests that the choice between ADAM and RMSProp in non-private settings may depend more heavily on the classifier architecture.

- For the Linuxprocess dataset (Figure 5b), the observed trends differ substantially from those in the Adult dataset. Here, RMSProp consistently achieves higher accuracy than ADAM across most privacy budgets, with particularly large positive differences at $\varepsilon = 1$ and $\varepsilon = 2$. For example, DT and RF classifiers benefit greatly from RMSProp, showing accuracy improvements of +10.91% and +8.57% at $\varepsilon = 1$, respectively. These strong gains suggest that RMSProp is more effective in handling the structural complexity and higher variability of the Linuxprocess data. Interestingly, even at lower privacy budgets (e.g., $\varepsilon = 0.5$), where ADAM had an advantage in the Adult dataset, RMSProp already outperforms ADAM for classifiers such as KNN and DT. This indicates that the adaptive learning rate of ADAM is less beneficial here, while RMSProp's simpler update rule provides more stable convergence under the noise introduced by DP. For the standard GAN without privacy loss, the performance gap narrows, with RMSProp still slightly outperforming ADAM for tree-based models (DT and RF).
- Overall, the results show that optimizer performance depends on dataset complexity, classifier type, and privacy budget. ADAM tends to perform better at lower privacy budgets in the Adult dataset, while RMSProp consistently outperforms ADAM across most settings in the Linuxprocess dataset, especially for tree-based classifiers.

6.4 Computational Cost

The training times reported in Table 2 reflect the computational cost of 100 epochs for each model (GAN, DPGAN, and DL-DPGAN) when trained using the RMSProp optimizer for both datasets. As shown in Table 2, training DL-DPGAN (by applying privacy loss function to the generator) takes longer than both the standard GAN and DPGAN architectures. This increase in training time is primarily due to the additional privacy loss component introduced in the generator (or discriminator) training objectives, which adds extra computations per iteration. Furthermore, the noise injection mechanism required by DP adds to the computational cost. Overall, the results indicate a trade-off between privacy protection and computational efficiency: achieving stronger privacy guarantees requires more training time.

6.5 Entropy Analysis

Table 3 shows the entropy values of synthetic data generated by DL-DPGAN under different generator privacy loss intensities (λ_G), compared with the entropy of the original real data on both datasets. As observed, all DL-DPGAN configurations yield higher entropy values than the real dataset, suggesting an increase in data diversity and a potential improvement in privacy preservation. According to Table 3, the average entropy of synthetic data is consistently higher than that of real data across both datasets, indicating that both the GAN and DL-DPGAN models increase data variability and uncertainty, which is beneficial for preserving privacy. Moreover, the synthetic data generated by DL-DPGAN exhibits higher entropy than the data generated by the standard GAN model, suggesting that the integration of a generator privacy loss contributes to enhanced privacy protection.

Table 2: Training time of the networks on the datasets for 100 epochs using GAN, DPGAN, and DL-DPGAN architectures.

Dataset		Network architectures		
		GAN	DPGAN	DL-DPGAN
Adult	Time (seconds)	1760.9	1798.6	2036.9
Linuxprocess	Time (seconds)	445.3	453.4	550.2

Table 3: The entropy values of synthetic datasets generated by the DL-DPGAN model and entropy values of real datasets under different generator privacy loss intensities (λ_G) settings across different datasets (using $\varepsilon = 1$ and the RMSProp optimizer).

Dataset	Real Entropy	Fake Entropy (DL-DPGAN)						Fake Entropy Average
		$\lambda_G = 0.1$	$\lambda_G = 0.5$	$\lambda_G = 1$	$\lambda_G = 2$	$\lambda_G = 10$	GAN	
Adult	9.2969	10.3711	10.3713	10.3735	10.3741	10.3633	10.3355	10.3648
Linuxprocess	6.3809	8.8783	8.8906	8.8719	8.8845	8.7735	8.8687	8.8613

Across different λ_G values, the entropy of DL-DPGAN-generated data shows a relatively stable or slightly increasing trend on both datasets, implying that moderate increases in the privacy loss intensity can promote more diverse and uncertain synthetic outputs. While some fluctuations are observed (particularly in the Linuxprocess dataset) this may be partly due to entropy being better suited for numerical data. Given the mixed-type nature of the dataset (i.e., containing both numerical and categorical features), the metric may not fully reflect all aspects of variability, though the overall trend still indicates increased diversity in the synthetic data. However, in both datasets, it is also observed that beyond a certain λ_G threshold (e.g., $\lambda_G = 10$), the entropy begins to decrease, which may indicate a degradation in data quality or mode collapse due to excessive emphasis on privacy. Therefore, tuning λ_G needs to be carefully done to achieve an optimal balance between privacy preservation and data utility.

7 Conclusion

This study introduced DL-DPGAN, an enhanced differentially private generative framework designed to improve the privacy and utility of synthetic data for machine learning applications. By incorporating a correlation-based privacy loss alongside the Wasserstein distance, the proposed model effectively reduces information leakage while maintaining training stability. Extensive experiments on diverse tabular datasets demonstrate that the proposed method effectively reduces empirical privacy leakage while maintaining competitive utility. The results indicate that applying the privacy loss primarily to the generator, with a moderate weighting factor, leads to reduced empirical privacy leakage while maintaining competitive analytical performance.

Future work will explore stronger privacy auditing techniques, including advanced membership inference attacks and improved mechanisms to better align empirical privacy behavior with formal differential privacy guarantees.

References

[1] Siam, S.I., Ahn, H., Liu, L., Alam, S., Shen, H., Cao, Z., Shroff, N., Krishnamachari, B., Srivastava, M., Zhang, M.: Artificial intelligence of things: A survey. ACM Trans. Sen. Netw. **21**(1) (2025) https://doi.org/10.1145/3690639

[2] Yaghoubi, E., Yaghoubi, E., Khamees, A., Razmi, D., Lu, T.: A systematic review and meta-analysis of machine learning, deep learning, and ensemble learning approaches in predicting ev charging behavior. Engineering Applications of Artificial Intelligence **135**, 108789 (2024)

[3] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Advances in Neural Information Processing Systems, pp. 2672–2680 (2014). http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf

[4] Xie, L., Lin, K., Wang, S., Wang, F., Zhou, J.: Differentially Private Generative Adversarial Network. arXiv (2018)

[5] Stadler, T., Oprisanu, B., Troncoso, C.: Synthetic data–anonymisation groundhog day. In: 31st USENIX Security Symposium (USENIX Security 22), pp. 1451–1468 (2022)

[6] Melzi, P., Rathgeb, C., Tolosana, R., Vera-Rodriguez, R., Busch, C.: An overview of privacy-enhancing technologies in biometric recognition. ACM Comput. Surv. **56**(12) (2024)

[7] Fathalizadeh, A., Moghtadaiee, V., Alishahi, M.: Indoor geo-indistinguishability: Adopting differential privacy for indoor location data protection. IEEE Transactions on Emerging Topics in Computing **12**(1), 293–306 (2024) https://doi.org/10.1109/TETC.2023.3242166

[8] Ghosheh, G.O., Li, J., Zhu, T.: A survey of generative adversarial networks for synthesizing structured electronic health records. ACM Comput. Surv. **56**(6) (2024)

[9] Moghtadaiee, V., Alishahi, M., Rabiei, M.: Differentially private gans for generating synthetic indoor location data. Int. J. Inf. Secur. **24**(3) (2025) https://doi.org/10.1007/s10207-025-01020-4

[10] Zhang, L., Shen, B., Barnawi, A., Xi, S., Kumar, N., Wu, Y.: Feddpgan: federated differentially private generative adversarial networks framework for the detection of covid-19 pneumonia. Information Systems Frontiers **23**(6), 1403–1415 (2021)

[11] Wakili, A., Bakkali, S.: Privacy-preserving security of iot networks: A comparative analysis of methods and applications. Cyber Security and Applications **3**, 100084 (2025)

[12] Dhinakaran, D., Sankar, S., Selvaraj, D., Raja, S.: Privacy-preserving data in iot-based cloud systems: A comprehensive survey with ai integration. arXiv preprint arXiv:2401.00794 (2024)

[13] Alabdulwahab, S., Kim, Y.-T., Son, Y.: Privacy-preserving synthetic data generation method for iot-sensor network ids using ctgan. Sensors **24**(22), 7389 (2024)

[14] Han, C., Yang, T., Sun, X., Cui, Z.: Secure hierarchical federated learning for large-scale ai models: Poisoning attack defense and privacy preservation in aiot. Electronics **14**(8), 1611 (2025)

[15] Lopuhaä-Zwakenberg, M., Sheikhalishahi, M., Kivits, J., Klarenbeek, J., Velde, G.-J., Zannone, N.: Comparing classifier's performance under differential privacy. In: 18th International Conference on Security and Cryptography, SECRYPT 2021, pp. 50–61 (2021)

[16] Emadi, M., Moghtadaiee, V., Alishahi, M.: From real to synthetic: Gan and dpgan for privacy preserving. In: The 22nd International Conference on Security and Cryptography (SECRYPT 2025), pp. 711–716 (2025)

[17] Alishahi, M., Zannone, N.: Not a free lunch, but a cheap one: On classifiers performance on anonymized datasets. In: Data and Applications Security and Privacy. Lecture Notes in Computer Science, vol. 12840, pp. 237–258. Springer, ??? (2021)

[18] Dat, P.T., Dutt, A., Pellerin, D., Quénot, G.: Classifier training from a generative model. In: International Conference on Content-Based Multimedia Indexing (CBMI), pp. 1–6 (2019). IEEE

[19] Rashid, H., Tanveer, M.A., Khan, H.A.: Skin lesion classification using gan based data augmentation. In: 2019 41St Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), pp. 916–919 (2019). IEEE

[20] Alishahi, M., Moghtadaiee, V., Fathalizadeh, A., Rabiei, M.: Mutual impact of feature selection and privacy-preserving mechanisms. International Journal of Machine Learning and Cybernetics **17**(3), 106 (2026)

[21] Azadmanesh, M., Ghahfarokhi, B.S., Talouki, M.A.: Adam-dpgan: A differential private mechanism for generative adversarial network. Applied Intelligence **53**(9), 11142–11161 (2023)

[22] Dwork, C., Kenthapadi, K., McSherry, F., Mironov, I., Naor, M.: Our data, ourselves: Privacy via distributed noise generation. In: Advances in Cryptology–EUROCRYPT 2006, pp. 486–503 (2006). Springer

[23] Dwork, C., Roth, A.: The algorithmic foundations of differential privacy. Found.

Trends Theor. Comput. Sci. **9**(3–4), 211–407 (2013)

[24] Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein generative adversarial networks. In: Precup, D., Teh, Y.W. (eds.) International Conference on Machine Learning. Machine Learning Research, pp. 214–223. PMLR, ??? (2017)

[25] Abadi, M., Chu, A., Goodfellow, I., McMahan, H.B., Mironov, I., Talwar, K., Zhang, L.: Deep learning with differential privacy. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. CCS '16, pp. 308–318. Association for Computing Machinery, ??? (2016)

[26] Torkzadehmahani, R., Kairouz, P., Paten, B.: Dp-cgan: Differentially private synthetic data and label generation. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, pp. 1–10 (2019)

[27] London, B., Getoor, L., Aggarwal, C.C.: Data Classification: Algorithms and Applications. Chapman and Hall/CRC, ??? (2014)

[28] Kim, S.-H., Jung, C., Lee, Y.-J.: An entropy-based analytic model for the privacy-preserving in open data. In: 2016 IEEE International Conference on Big Data (Big Data), pp. 3676–3684. IEEE, ??? (2016)