Journal of Soft Computing Paradigm (ISSN: 2582-2640)
www.irojournals.com/jscp/

Differentially Private Time Series Wasserstein Generative Adversarial Network for Private and Utilizable Synthetic Time Series Data Generation

Sathiyapriya K.[1], Mridula M.[2], Kumaresh S.[3], Sravya Vankadara[4]

Department of Computer Science and Engineering, PSG College of Technology, Coimbatore, India

E-mail: [1]Spk.cse@psgtech.ac.in, [2]mridul.murali05@gmail.com, [3]skumaresh34@gmail.com, [4]shravyav20@gmail.com

Abstract

The humongous volumes of data utilized to train the machine learning models are vulnerable to leakage by model inversion attacks and membership inference attacks. These days, massive amounts of research are being conducted to leverage differential privacy to safeguard the privacy of users. Tabular data generation from differentially private generative adversarial networks is still an untapped area. This work suggests a framework to enhance privacy protection in generating synthetic data by utilizing Wasserstein distance. The developed architecture generated synthetic data that replicated the time series relations of real-world data without compromising identifiable features of members of the input data. Results obtained from the architecture were compared with two other current GAN frameworks, DP-WGAN, and Time GAN. The privacy vs. utility tradeoff was found to be improved in the case of the architecture under discussion, as can be seen from the RMSE scores and Overall Quality Report.

Keywords: Generative Adversarial Networks; Synthetic Data; Differential Privacy; Privacy-Utility Trade-Off; Time-Series Data; Tabular Data; Wasserstein Distance.

1. Introduction

Machine Learning (ML) has become a part of many industries, from business and medicine to government administration [1]. This is due to ongoing development and ever-

DOI: https://doi.org/10.36548/jscp.2025.3.002
Received: 13.06.2025, received in revised form: 15.07.2025, accepted: 31.07.2025, published: 12.08.2025

growing flexibility. Yet, generating top-performing models not only needs vast amounts of data but also high computational capacities. Recent developments in hardware, like high-performance CPUs and GPUs, along with cloud-based operations have eased many computational limitations. Conversely, the presence of data, though plentiful, poses a different problem: the safe, ethical, and privacy-respecting manipulation and utilization of said data is still an unresolved issue [2]. This constraint is a significant bottleneck to the wider and safer use of ML and DL technologies. Today, model exploitation can be avoided by redesigning the architectures of the models to be more secure [6] or by training the models on synthetic data [7]. The former represents a much more labor-intensive process. This is due to the fact that moving from an old architecture to a new one is difficult. Alternatively, the latter may not lead to as much model utility.

Synthetic Data Generation (SDG) is the creation and tagging of machine generated data that is very much like actual world data. SDG is generally applied in order to fill gaps in datasets or to substitute highly sensitive data. Businesses use synthetic data to train their models to safeguard privacy and avoid data leaks. For instance, to balance patient data privacy and facilitate AI research, NHS England partnered with researchers to create synthetic medical records through GAN-based models under differential privacy. The aim was to permit hospitals and researchers to study patient trends without revealing sensitive health data. By substituting actual patient data with privacy-protecting synthetic data, the models retained high utility for predicting disease progression and the risk of readmission for patients. Such an approach maintained GDPR compliance while facilitating innovation. The suggested LSTM-augmented DP-WGAN model can be applied across privacy-sensitive industries that process time-series tabular information, including healthcare, finance, and energy systems.

For instance, in a healthcare setting, synthetic patient records with actual real-time vital signs or medication history can be created and made available to researchers without invading the privacy of patients. The system may be plugged into the data pipeline of an organization, receiving actual data, adding differential privacy mechanisms, and producing synthetic data. In banking environments, the system may produce realistic transaction records for fraud detection models without the customer identities being disclosed. In energy applications, synthetic replication of smart meter's time-series data is possible for load forecasting or infrastructure planning. The synthetic data features may be very similar to real-world data but not exactly the same due to privacy-utility trade-off [8]. As the synthetic and original data become more

similar, the privacy measure of the synthetic data decreases, but the utility improves. Increasing the privacy measure of the synthetic data makes the similarities between the generated and original data sparse, leading to reduced usefulness of the synthetic data. A good balance in the privacy-utility trade-off needs to be determined depending on the requirements of the company.

2. Literature Review

The revolutionary advances made in deep learning have raised the use of deep learning models in a diverse range of sectors, but this has also exposed the models to different privacy challenges. Research conducted by Ximeng Liu et al. [3] and Hui Sun et al. [4] classifies and describes a broad variety of Deep Neural Network (DNN) and Generative Adversarial Network (GAN) vulnerabilities, specifically model inversion attacks, which are capable of reversing-sensitive training data based on model parameters [5]. All these issues underline the necessity for strong privacy-preserving mechanisms within synthetic data generation. Synthetic Data Generation (SDG) tries to strike a balance between data utility and privacy by generating artificial data that replicates real-world datasets while retaining the utility and privacy scores specified by the application needs. Yet, as synthetic data becomes increasingly realistic, privacy threats grow as well, with the perennial privacy-utility trade-off [7]. Differential Privacy (DP) offers an exact mathematical system to address this issue. The paper by Martin Abadi et al. [8] proposed DP-SGD, a base training algorithm that uses noise and gradient clipping to ensure privacy at the cost of little utility loss.

GANs have been widely used for DP data generation because of their better generative performance. In contrast to Variational Autoencoders (VAEs), GANs generally require simpler training configurations and provide high-quality outputs. The Wasserstein GAN (WGAN), introduced by Arjovsky et al. [11], substituted the Jensen-Shannon divergence with the Wasserstein distance, enhancing convergence and stability. To solve the privacy issues of WGANs, Dingfan Chen et al. [10] proposed the Gradient Sanitized WGAN (GS-WGAN) that selectively applies gradient clipping, providing robust privacy guarantees. Subsequently, Liyang Xie et al. [15] investigated noise addition and weight clipping mechanisms for differentially private WGANs, whereas Jinsung Yoon et al. [16] generalized GANs to the time-series domain with the TimeGAN model that employs supervised loss to learn temporal patterns better.

In addition, research like Pepijn te Marvelde [13] and Valtteri Nieminen [14] modified GS-WGAN to produce time-series and tabular data, respectively. Te Marvelde proved that image-based architectures can be used for time-series synthesis, and Nieminen utilized subsampling for privacy amplification in training. Xie et al. [17] proposed a DP-GAN framework that incorporates representation learning, making it more efficient to synthesize complicated tabular data. Though their model improves utility, it is computationally expensive and does not perform well on sparse, high-dimensional inputs. Chen et al. [18] introduced PrivSyn, which trades off privacy and utility with sophisticated GAN training methods; however, it has not been subjected to thorough evaluation on real-world downstream tasks and thus has limited practical understanding. In healthcare applications, Esteban et. al.[19] introduced Recurrent Conditional GANs (RCGANs) to synthesize realistic medical time-series data with excellent temporal coherence. However, their approach doesn't include formal differential privacy mechanisms, potentially introducing gaps in privacy guarantees. Feature-label protection. Torkzadehmahani et al. [20] presented DP-CGAN, enabling supervised learning on privacy-protected synthetic data with robust privacy guarantees; however, it doesn't naturally extend to multivariate or long-range sequential data. Zhang et al. [21] designed a two-stage GAN that enhances temporal pattern generation with privacy constraints but has a complicated structure requiring considerable training effort and hyper parameter adjustment. Lastly, Mohamed et al. [22] introduced Secure GAN, which scales DP-GANs for enterprise-level deployment and high-dimensional data. Even so, the model suffers from limitations in preserving sample diversity while scaling to extremely large sets.

In summary, the literature review offers a solid foundation for applying GANs to synthetic data generation with privacy preservation. WGAN and GS-WGAN were the two frameworks of choice because they strike a balance between utility and convergence. Mechanisms based on differential privacy such as DP-SGD and gradient sanitization enhance privacy guarantees. Although attempts are ongoing to extend GANs to tabular and time-series data, there is scope to enhance model flexibility, efficiency, and evaluation methods.

The following were inferred from the literature survey:

- Methods such as DP-SGD and gradient sanitization make it possible for GANs to offer privacy assurances without a substantial loss of utility.
- Wasserstein distance improves training stability in GANs over previous adversarial losses.

- Time GAN and GS-WGAN extensions were demonstrated to work efficiently in producing time-series and tabular data.
- Prior image-based GANs can be employed for tabular and time-series data generation with high utility and privacy.

2.1 Research Gaps

- Methods such as gradient sanitization become computationally costly when applied to large scale data.
- Memory-based models such as LSTMs can inadvertently hold sensitive patterns present in real data, which necessitates the precise tuning of privacy parameters.
- Several GAN architectures are tailored towards particular types of data and might not be effective for structured data formats such as tabular time-series.
- Quantitative evaluation of both privacy and data utility is still challenging and without standard metrics

3. Background

3.1. Differential Privacy

The foundation of many algorithms that use privacy protections is Differential Privacy (DP) [9]. DP uses a mathematical definition of privacy that blends machine learning constraints with statistical thresholds. Regardless of whether a person's private information is included in the differentially private analysis, DP mathematically guarantees that the results will be the same. (ε,δ)-DP is a randomized mechanism M with range R, if holds for any subset O and for adjacent datasets S and S^', where both differs by one training sample as shown in formula (1).

$$\Pr[\mathcal{M}(S) \in \mathcal{O}] \leq e^{\varepsilon} . \Pr[\mathcal{M}(S') \in \mathcal{O}] + \delta \quad (1)$$

M is the training algorithm and ε corresponds to the upper bound of privacy loss whereas δ corresponds to the probability of breaching DP constraints.

3.2 Wasserstein Distance

Wasserstein Distance (WD) is similar to a cost function since it is calculated via the minimum amount of work [8]. represents the quantity of work needed to discover similarities among distributions. As WD is a distance function, it can be applied to a wide range of machine learning issues that can be formulated in metric space. The main advantage of WD compared

to other distance methods is that it can be employed with any type of data and distribution. The Wasserstein loss function is an integral component of Wasserstein GANs (WGANs) and plays a crucial role in stabilizing the GAN training process.

Classical GANs employ Jensen-Shannon (JS) divergence, which tends to produce vanishing gradients and unstable training. Wasserstein distance gives smoother gradients, enabling the generator to learn better. In contrast to JS divergence, Wasserstein loss gives a continuous and interpretable measure of how close the generated distribution is to the real distribution. A lower Wasserstein loss value indicates that generated samples are closer to real data. WGANs are less likely to collapse the mode of the generated data, where the generator produces more similar output. Wasserstein distance aids GANs in converging more stably during training, particularly when used together with methods such as gradient penalty or gradient sanitization. Within the proposed architecture, the discriminator (critic) calculates Wasserstein loss between fake and real time-series samples. This loss helps the generator enhance its outputs, generating more realistic and temporally coherent synthetic data.

3.3 Generative Adversarial Networks

Generative Adversarial Networks (GANs) are models utilized in the generation of novel data from a given input dataset, mimicking patterns or regularities in the initial dataset. The GAN identifies these patterns by breaking up the problem of unsupervised learning into a problem of supervised learning. This is done with the aid of two sub-models known as the generator and the discriminator.

The generator model is trained to produce new data, whereas the discriminator model is created to label data as fake or real. The generator trains to deceive the discriminator, into thinking the data is original. The discriminator on the other hand, trains to label more precisely whether the data is original or generated. The discriminator predicts the class label of the data as real or fake according to examples provided by the domain. The training dataset consists of actual examples along with synthetic examples from the generator model. This way, the two models engage in an adversarial game that culminates in the generator being capable of producing synthetic data that can deceive the discriminator into believing it is real.

4. Proposed Methodology

This research work achieves the generation of artificial tabular time series data through the use of the GS-WGAN architecture to impart differential privacy to the training data. The utility versus privacy trade-off the generated data is observed and noted against two benchmarked GAN models mentioned. The ADANI GREEN dataset is utilized as the real-world input to the used GAN architectures. The performance of the resulting synthetic data is measured by training an LSTM model on actual data and evaluating it on the generated data from each of the models. For privacy monitoring, data point similarity between the real and synthetic datasets is monitored.

4.1 DP-TWGAN Architecture

The DP-TWGAN architecture, depicted in Figure 1, builds upon the core DP-WGAN framework through the addition of two significant changes: (i) replacing weight clipping with gradient sanitization, and (ii) adding Long Short-Term Memory (LSTM) units to both the generator and discriminator. Gradient sanitization, a mechanism introduced in GS-WGAN [10], provides enhanced convergence for differentially private Wasserstein GANs through the selective use of gradient clipping on a set number of parameters. This bound clipping limits oscillation near local minima, allowing for quicker convergence than standard weight clipping but at increased computational expense.

In the architecture presented, in Figure 1, LSTM (Long Short-Term Memory) is utilized prominently in tabular time-series data generation in the generator and discriminator operations. The LSTM component in the generator learns the temporal relationship within the input noise vector and assists in generating sequential data that replicates the patterns and structures of real-world datasets. As LSTM can maintain memory over long-time steps, it keeps generated samples consistent over time which is important in time-series synthesis. The LSTM in the discriminator assesses the temporal coherence of real and artificial sequences, helping to detect temporal patterns and anomalies in the synthesized sequences. This enhances the ability of the discriminator to differentiate between actual and artificial data. It also assists in the computation of the Wasserstein loss more accurately by taking into account not only feature distribution but also time-dependent relationships.

The addition of LSTM units addresses the temporal dependencies inherent in sequential data, taking advantage of LSTMs' ability to store memory in order to capture long-range patterns better. Yet, this temporal memory also poses the risk of privacy, since LSTMs might inadvertently store and forward sensitive features from the actual data. Therefore, the privacy parameters should be well-adjusted in order to achieve a proper privacy-utility tradeoff.

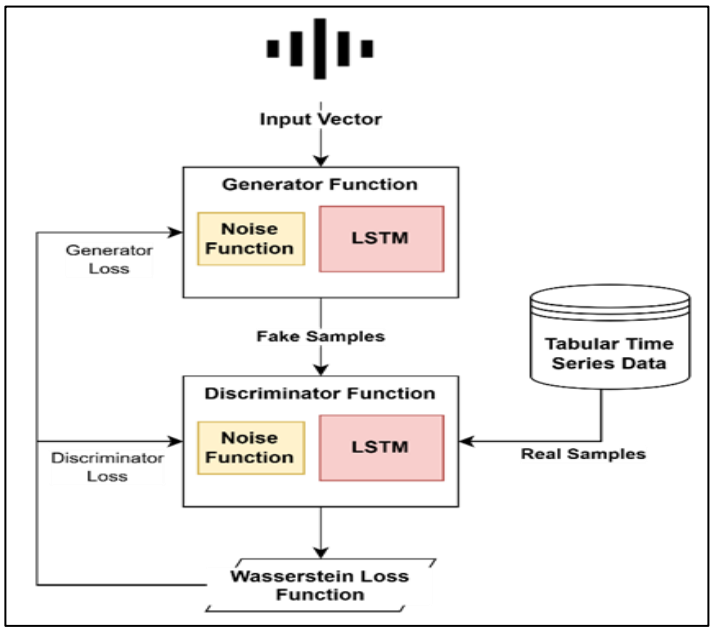

Figure 1. Overall Architecture of DP-TWGAN

- **Utility**

The utility of the proposed model is compared with pre-existing models as shown in Fig 2. Each model's usefulness is evaluated by making a price prediction using Long Short-Term Memory (LSTM). The original ADANI GREEN dataset is used to create synthetic data for each model. The generated data is statistically compared to the original dataset following Synthetic Data Generation (SDG). The LSTM model is trained using the data that each model synthesizes following the statistical comparison. The original dataset is used for testing in order to simulate real-world situations. The performance of the GAN models is evaluated based on the RMSE scores obtained for each model. To guarantee an objective investigation, the privacy bound of each model is set to be the same during the performance comparison.

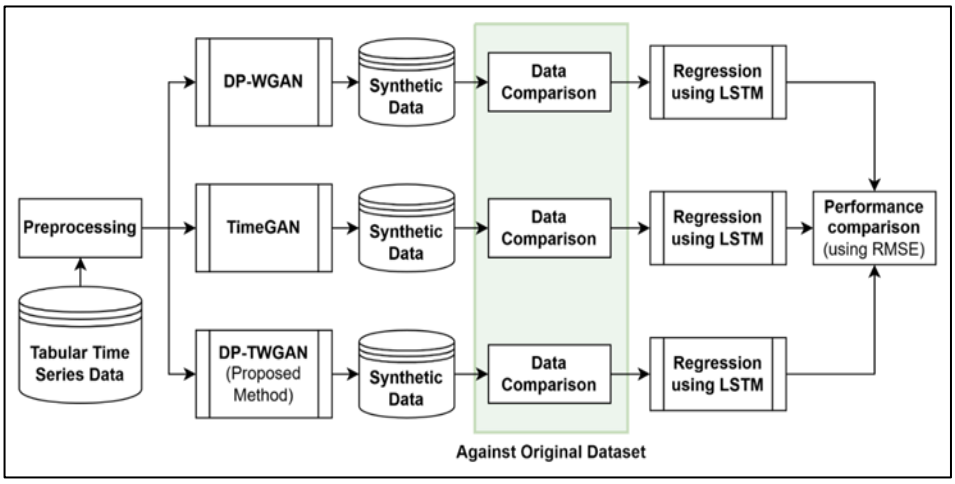

Figure 2. Utility Comparison

- **Privacy**

As illustrated in Figure 3, the privacy of the suggested model is contrasted with that of existing models. The purpose of the privacy comparison is to assess the models' vulnerability and privacy bounds. The datasets produced by each model are used to efficiently calculate the privacy scores. This facilitates a clear understanding of each model's limitations and shortcomings. Based on various privacy metrics and an overall quality report, the original dataset and the synthetic time-series dataset produced by each model are compared. For every model, the privacy-utility ratio is monitored and evaluated.

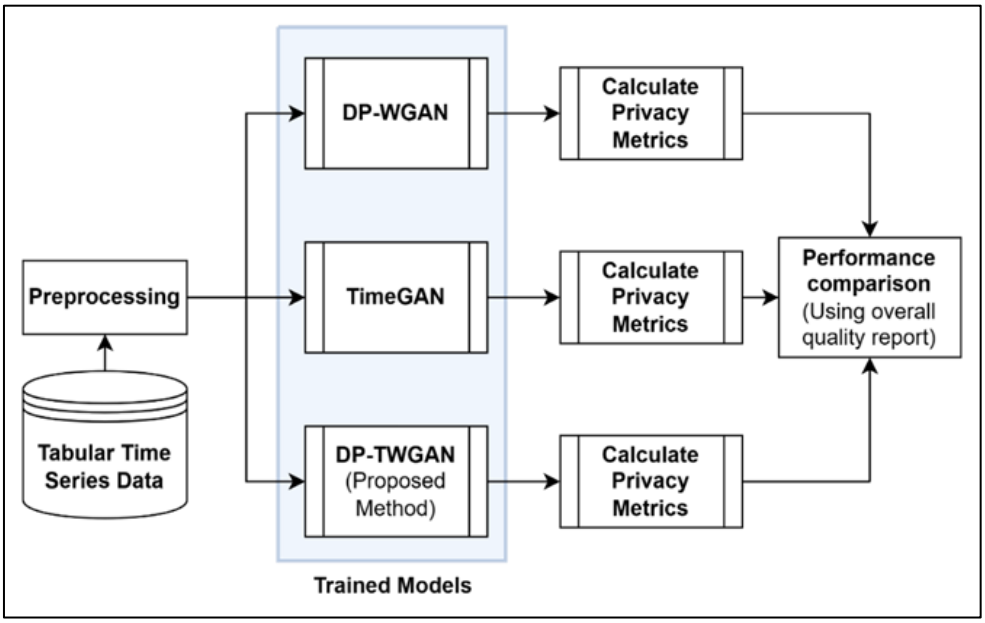

Figure 3. Privacy Comparison

5. Implementation

5.1 DPWGAN

Here, the objective function is the loss function of the Wasserstein GAN. DP-WGAN applies randomized response in order to add random noise to the output of the discriminator, making it harder for the generator to learn about the sensitive data in the training dataset. The privacy budget parameter determines the level of noise addition. DP-WGAN applies weight clipping and Wasserstein distance to achieve convergence of parameters.

A random permutation of the dataset is created with a random permutation function, and a batch of samples is extracted from the data by slicing. Fake samples are created with the generator network. The loss of the discriminator is computed as the negative of the difference between the mean scores of real and fake samples and is backpropagated through the discriminator network when the discriminator weights are updated with the optimizer. The weights are clipped for privacy. The loss of the generator is thereafter computed using the discriminator's score, aiming to maximize the discriminator's score for synthetic samples. The gradients are then backpropagated through the generator, and a step in the gradient's opposite direction is taken by the generator's optimizer to update the weights of the generator.

5.2 TimeGAN

TimeGAN [16] is a GAN-based model that has the capability to generate precise time-series data across diverse domains. TimeGAN is able to learn the stepwise conditional distributions in the data through the use of original data for supervision and by including a stepwise supervised loss. To offer a reversible transformation between features and representations, an embedding network is employed to reduce the high dimensionality of the adversarial learning space. On this basis, the recovery networks, the discriminator, and the generator are implemented as recurrent neural networks, each providing their own respective features. The discriminator serves to capture stepwise conditional distributions, whereas the network is useful for offering a reverse mapping of features.

5.3 DP-TWGAN

Differentially Private-Time based Wasserstein GAN (DP-TWGAN) is employed to create differentially private synthetic time series data. The DP-TWGAN consists of two primary

parts: a generator used to create new time series data similar to actual data, and a discriminator that attempts to identify real and synthetic data. The generator is designed as an LSTM neural network with one or more layers of hidden units. It accepts a random noise vector as input and produces a sequence of values that are supposed to approximate a time series. The discriminator is similarly an LSTM neural network but with one output node that determines if the input sequence is real or fake. The GAN model is then trained by cycling through training the generator to create realistic data and training the discriminator on how to identify real and fake data correctly.

The DP-TWGAN specifies a personalized loss function involving the Wasserstein distance and the gradient penalty to ensure that the generator generates realistic data close to the real data. Furthermore, a PyTorch DataLoader is employed to feed the actual data into the DP-TWGAN model for training. Differential privacy is applied by defining hook functions at last. A hook function is an in-between function that is attached to a neural layer. It manipulates intermediate outputs or gradients of the neural layers during the forward and backward propagations. In order to defend against attacks from adversaries, the hook function injects noise into the gradients (generator and discriminator parameters) during training. DP-TWGAN training is illustrated in Algorithm 1.

Algorithm 1 The training of DP-TWGAN

Input: Number of epochs *num_epochs*, DataLoader*train_loader*, Batch size *batch_size*, Input dimensions *input_dim*, Output dimensions *output_dim*, Discriminator function *discriminator*, Generator function *generator*, Wasserstein loss with gradient penalty w*asserstein_loss_gp*
Output: Trained discriminator module *discriminator*, Trained generator module *generator*
1. Initialize n=0
2. for each *epoch* in range(num_epochs):
   a. for each *batch* in train_loader:
      i. Extract the input feature from the batch and take only the first *output_dim* features as *real_x*
      ii. Create a tensor of ones with shape (*batch_size, 1*) as *real_y*
      iii. Generate a fake feature set by passing the input feature through the generator and take only the first output_dim features as *fake_x*
      iv. Create a tensor of zeros with shape (*batch_size, 1*) as *fake_y*
      v. Pass *real_x* and *fake_x* through the *discriminator* and obtain the outputs *dis_real* and *dis_fake* respectively
      vi. Compute the Wasserstein loss with gradient penalty using *dis_real*, *dis_fake*, *real_x* and *fake_x* as input to the *wasserstein_loss_gp* function and store the result in *loss_d*
      vii. Zero out the gradients of the discriminator optimizer

viii. Compute the gradients of the *loss_d* with respect to the parameters of the *discriminator*
ix. Update the discriminator parameters using the optimizer_d
x. If *n* is divisible by 5, then:
a. Generate a fake feature set by passing the input feature through the *generator* and take only the first output_dim features as *fake_x*
b. Pass *fake_x* through the *discriminator* and obtain the output *dis_fake*
c. Compute the *loss_g* as the negative mean of *dis_fake*
d. Zero out the gradients of the generator optimizer
e. Compute the gradients of the *loss_g* with respect to the parameters of the *generator*
f. Update the generator parameters using the optimizer_g
g. Increment *n* by 1

5.3.1 Generator

The generator structure of the suggested DP-TWGAN is made up of fully connected units and an LSTM unit to produce data. The random noise is obtained sequentially to preserve the nature of temporal spatiality. Synthetic data is produced using the random noise in every iteration. The neurons are stimulated using the noise and hidden states as input. The output of the LSTM layer is fed into fully connected layers to generate individual attributes. To apply differential privacy to the model's architecture, the generator receives a certain level of noise according to the privacy budget. To improve the capture of the time dependency among each datum, LSTM is added to the fully connected layers. This improves the capacity of the model to generate data with temporal locality. While it diminishes the privacy budget since additional memory is being added through the use of the LSTM, it enhances the utility of the synthetic data. Despite introducing noise, the gradients of the fully connected layer parameters, hidden layer parameters, and LSTM model parameters are clipped to avoid parameter convergence. Weight clipping is not implemented, and clipping of gradients is employed instead to enhance the rate of convergence of the model over its ancestor.

5.3.2 Discriminator

The architecture of the Discriminator contains fully connected layers and an LSTM layer to detect synthesized and real data. Every feature is associated with the LSTM layer, which indicates a sequence of time. The result of the LSTM layer is fed to a fully connected layer in order to provide scores according to the Wasserstein loss function. The whole array of features is mapped to numbers indicating the latent space. The Wasserstein loss measures how close the latent space distribution of the provided data is to the original dataset's latent space.

Discriminator scores more accurately with increased iterations. At the same time, the generator also produces higher-quality synthetic data based on more knowledge of the initial latent space. Similar to the generator model, object perturbation is implemented in the Discriminator's training phase. Gradient clipping is also implemented to attain parameter convergence despite noise injection. The Discriminator goes through five cycles of training for every training iteration of the generator to avoid mode collapse and to prevent the generator from employing recurring patterns to deceive the Discriminator.

6. Evaluation and Results

6.1 Observing Utility

Utility measurement is done to evaluate the synthetic dataset created in the real world. Utility measurement has been done for downstream model utility. Downstream model utility is applied to measure how efficient the created dataset is for downstream operations within a system or pipeline. The downstream utility highlights the significance of the real-world models that will be using the synthetic time-series dataset created. To this end, an LSTM model is trained to carry out stock price prediction for Adani Green stock. The closing price of the subsequent minute is calculated based on the 6-dimensional input using the LSTM model. The synthetic time series dataset is employed to train the LSTM model and the learned LSTM model is validated on the original dataset.

The usefulness of the created dataset is assessed through the Root Mean Square Error (RMSE), which indicates how dense the data is around the best-fit line. For verifying experimental data, root mean square error is typically utilized in forecasting as well as regression analysis. The value of RMSE is calculated in the following way:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(Predicted_i - Actual_i)^2}{N}} \quad (2)$$

where Predictedi stands for the predicted ith value, Actuali stands for the actual ith value, and N stands for the total number of samples. Figure 4 displays the results of training and testing the model on the original dataset. High levels of overlap between the actual and predicted values demonstrate the model's exceptional performance during the training and testing stages. The RMSE scores for the test and train sets are 0.19 and 0.17, respectively.

However, the sensitive data is extremely vulnerable to leakage because the model uses it directly.

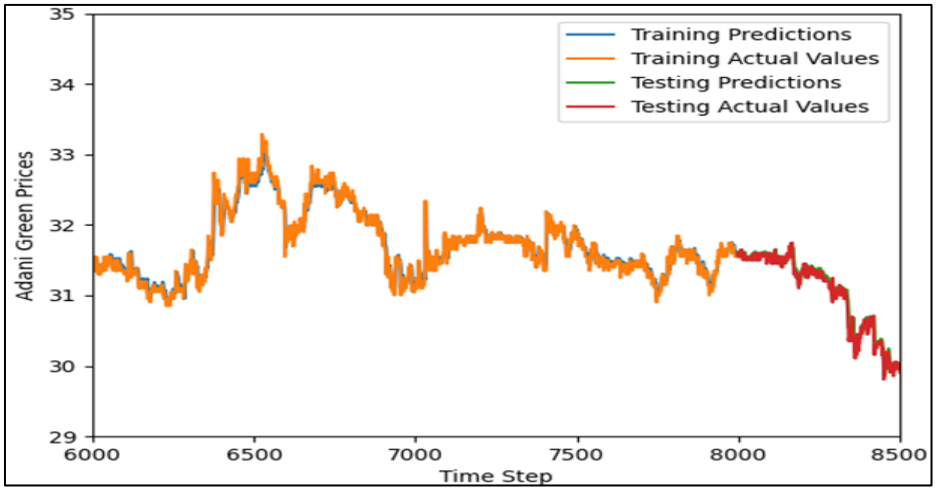

Figure 4. Original Dataset based Model Performance

6.1.1 DP-WGAN

The DP-WGAN model was tested by fine tuning hyper parameters such as epochs, batch size. number of critics, hidden dimensions, noise dimensions, and privacy budget (sigma value) as shown in Table 1.

Table 1. Hyper Parameter Tuning for DP-WGAN

Batch Size	Epochs	Number of Critics	Hidden Dimensions	Noise Dimensions	Sigma	Utility (RMSE Score)	Privacy
64	100	3	8	10	1	17.04	76.3%
64	400	5	10	15	3	**13.82**	79.8%
128	100	5	10	15	1	21.07	**90.5%**
128	200	8	15	10	3	20.47	89.1%
128	400	10	20	20	5	17.62	88.4%
256	100	1000	10	20	1	11.71	87.9%

Table 1 shows that smaller batch sizes and fewer critics result in more useful and private generated data. When batch sizes were smaller than larger batches, the introduction of noise,

which Sigma controlled, had an impact on the privacy scores. Lower epochs were tested more for efficiency because the number of epochs did not seem to have a significant impact on privacy and utility.

The LSTM model is trained using the dataset produced by DP-WGAN. The model's performance with the dataset produced by DP-WGAN is shown in Figure 5. The generated dataset is used to train the model, and the original dataset is used for testing. With minimal overlap between actual and predicted values, the LSTM model performs poorly in both training and testing. This is due to the DP-WGAN model's inability to create datasets while maintaining temporal locality and a lack of memory states. The model's usefulness is rather low, even though the dataset has high privacy protection levels. The model will not be useful in the real world due to its training RMSE score of 11.97 and test RMSE score of 21.07.

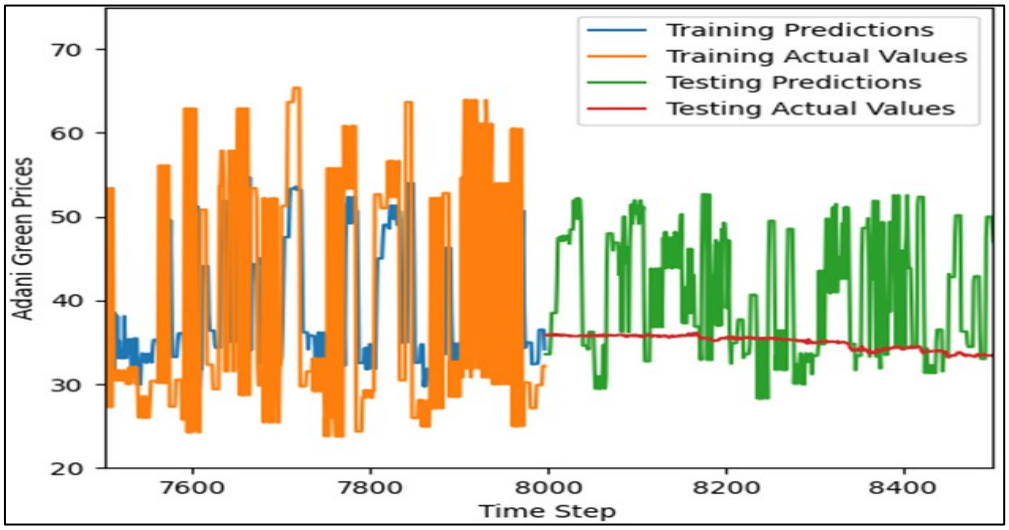

Figure 5. DP-WGAN Synthesized Dataset based Model Performance

6.1.2 Time GAN

The TimeGAN model was tested by fine tuning hyper parameters such as epochs, batch size, sequence length, hidden dimensions, and noise dimensions as shown in Table 2.

Table 2. Hyper Parameter Tuning for Time GAN

Batch Size	Epochs	Sequence Length	Hidden Dimensions	Noise Dimensions	Utility (RMSE Score)	Privacy
80	1000	10	20	20	1.53	66%

80	2000	12	25	20	2.09	68.2%
100	2000	14	24	32	1.01	75%
100	3000	20	30	30	1.87	73%
125	2000	15	20	25	1.24	70%

As the sequence length decreased, Time GAN's data became more useful. The privacy score rose as a result of the increased noise dimensions. For better utility and privacy trade-offs, smaller batch sizes are typically more ideal. The utility and privacy scores did not appear to be significantly impacted by the quantity of hidden dimensions or epochs. The LSTM model is trained using the Time GAN-generated dataset. The model's performance with the Time GAN-generated dataset is shown in Figure 6. Prior to testing on the original dataset, the model is trained on the generated dataset. With exceptionally high levels of overlap between actual and predicted values, the LSTM model exhibits strong performance in both training and testing. This is because of the TimeGAN model's architecture, which aims to identify temporal relationships in the dataset and use those correlations to create a synthetic dataset. The test RMSE score is 1.01 and the train RMSE score is 0.29. The model's privacy scores are low, despite the dataset's high model utility, which makes it perfect for real-world situations.

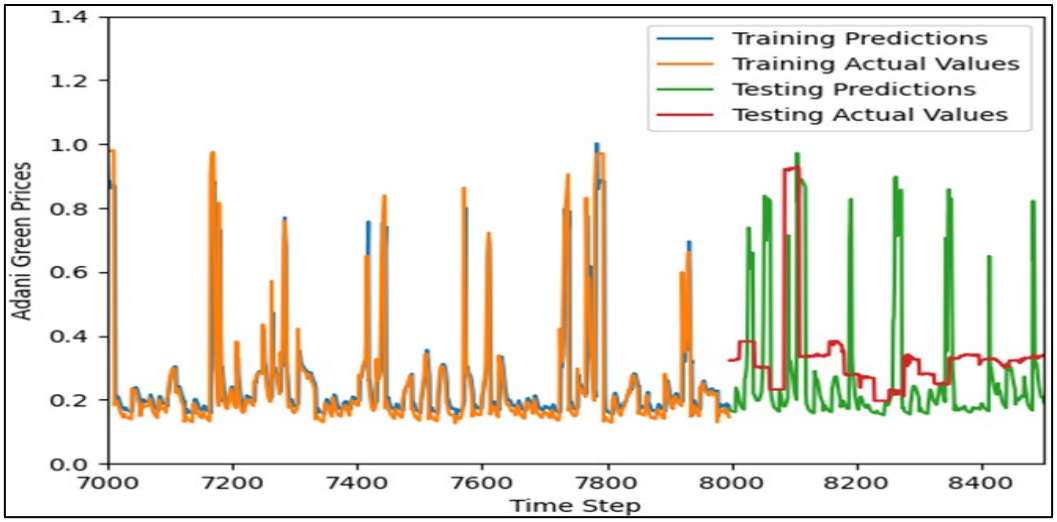

Figure 6. Time GAN Synthesized Dataset based Model Performance

6.1.3 DP-TWGAN

As shown in Table 3, the DP-TWGAN model was tested by fine-tuning hyperparameters like epochs, batch size, number of critics, hidden dimensions, and privacy budget parameters (lambda and epsilon).

Table 3. Hyper Parameter Tuning for DP-TWGAN

Batch Size	Epochs	Sequence Length	Hidden Dimensions	Lambda	Epsilon	Utility (RMSE Score)	Privacy
80	500	5	20	0.3	0.3	29.07	80.09%
85	750	10	25	0.7	0.4	13.56	77.81%
100	1000	5	32	0.5	0.5	**7.77**	82.11%
100	1500	10	30	0.8	0.5	14.45	67.19%
125	1000	10	27	0.5	0.3	17.9	71.34%
125	2000	15	32	0.9	0.8	7.63	79.40%

The utility is impacted inversely by the epsilon value, whereas the privacy score is directly impacted. The scores are not directly impacted by the batch size or the number of epochs. It is found that relatively small values of epsilon and lambda are associated with higher utility and privacy. Fig. 7 displays the model's performance using the dataset produced by the DP-TWGAN. The original dataset is used to test the model after it has been trained on the synthesized dataset. The LSTM model exhibits very high levels of overlap between the actual and predicted values during training, demonstrating excellent performance. The LSTM model initially performed poorly during the testing phase, exhibiting low levels of overlap, but it eventually recovered. The LSTM model cannot effectively adjust to abrupt price fluctuations because of the stock prices' volatility.

Temporal dependencies in the dataset are captured by the architecture of the DP-TWGAN model, which then uses these dependencies to create a synthetic dataset. Such

dependencies are captured by the LSTM layers, which are then used to train the discriminator and generator. The test RMSE score is 7.77, while the train RMSE score is 0.27. The dataset has sufficient levels of model utility, is practical for real-world use, and has higher privacy scores than TimeGAN and DP-WGAN.

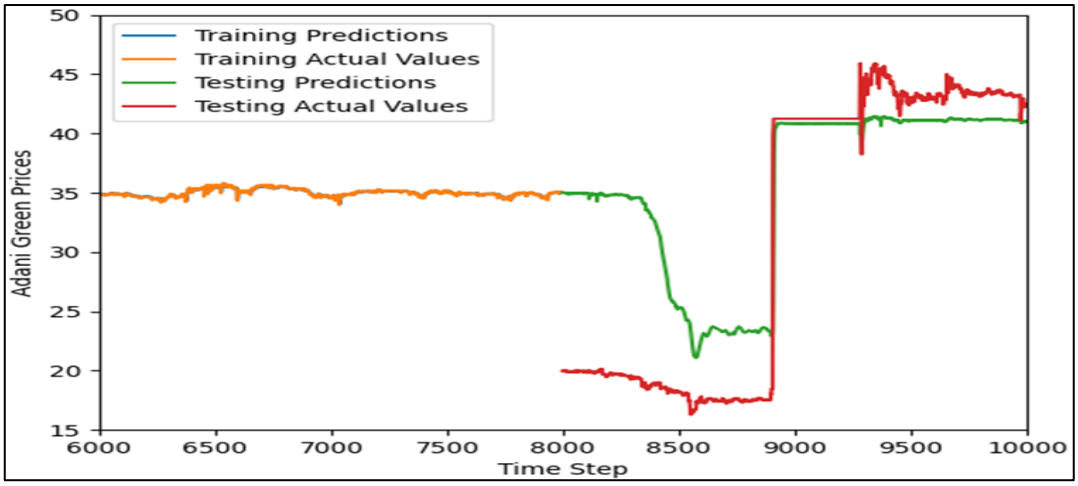

Figure 7. DP-TWGAN Synthesized Dataset based Model Performance

6.2 Observing Privacy

Four distinct metrics are combined to assess the generated datasets' privacy. Outliers' proximity to the rest of the data set's normal distribution is calculated using the Nearest Neighbour Distance Ratio (NNDR). The distance between the closest record and the next closest record is used to compute the NNDR. While synthetic data points with NNDR values close to 1 are near the original points in dense regions of the original data, those with NNDR values close to 0 are closer to the original data's sparse points. The degree of privacy increases with the extent to which the NNDR distribution of synthetic data matches that of real data. The following is the Nearest Neighbour ratio:

$$NNDR = \frac{\overline{D_o}}{\overline{D_e}} \tag{3}$$

where $\overline{D_o}$ is the observed mean distance between each feature and the corresponding nearest neighbour and $\overline{D_e}$ is the expected mean distance.

The Distance to Closest Records (DCR) method calculates the shortest distance between a record and its nearest neighbour in a dataset. DCR is useful for determining the privacy risk

associated with the disclosure of a certain dataset. A lower DCR value indicates a higher possibility of re-identification and, as a result, a higher privacy risk for the dataset. The DCR is computed in the following way:

$$DCR = \min\big(d(x,y)\big), x \neq y \quad (4)$$

where the dataset's distance metric, d(x,y), measures the separation between records x and y. All pairs of unique records in the collection are averaged to get the minimal distance. The k-anonymity measure works by dividing the dataset into distinct populations that have the same set of identifying traits. The k-anonymity measure tries to ensure that each individual in the dataset is indistinguishable from at least k-1 other individuals in the dataset in order to make it impossible to re-identify any individual in the dataset.

The Overall Quality Report module by Virtual Data Lab combines the scores obtained by DCR and K-Anonymity measures to establish an overall privacy score for the synthetic dataset. The scores assigned are based on the amount of data that will not be leaked and remain private in the case of model inversion attacks or data leaks.

6.2.1 DP-WGAN

The overall quality report for the DP-WGAN generated dataset is 82.14%. It shows that, based on the K-Anonymity and DCR, the synthetic values are closer to ideal to ensure privacy. The NNDR score for this model is greater than 0.6 for most columns which shows that for each column, privacy is maintained pretty well as shown in Figure 8.

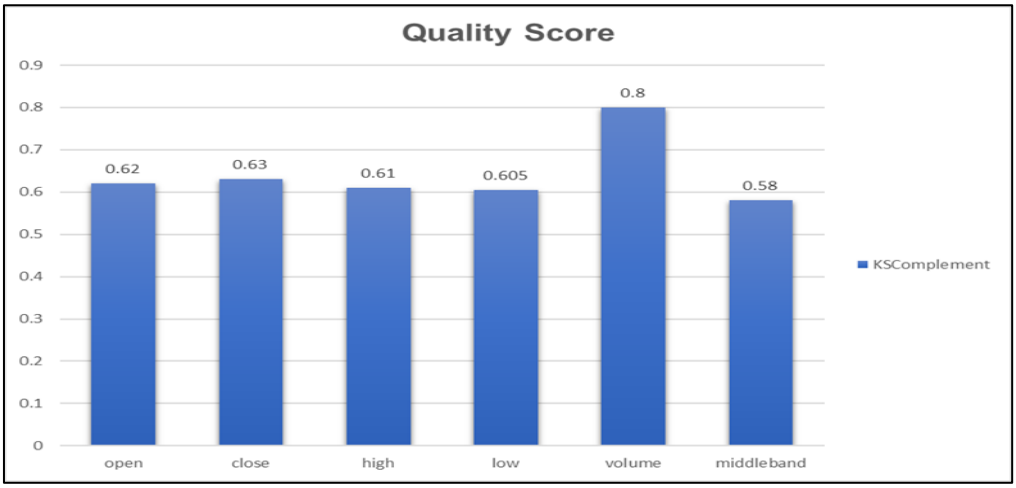

Figure 8. NNDR Scores for DP-WGAN

The dispersed and clustered data points in Figure 9 suggest poor utility, but the absence of overlap suggests good privacy. The likelihood of discovering the original values is low because the values in the synthetic and real datasets differ.

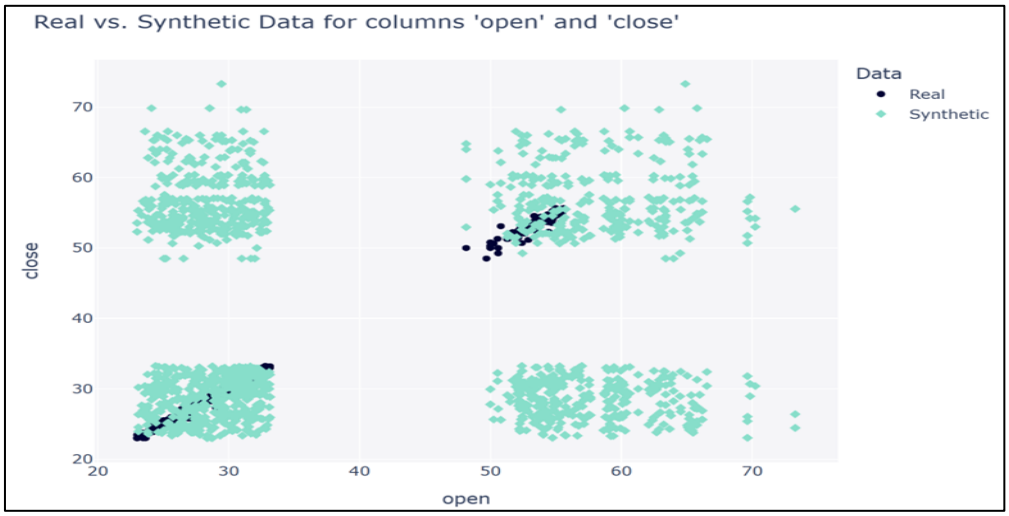

Figure 9. Scatterplot for DP-WGAN

6.2.2 TimeGAN

TimeGAN's overall quality report is 64.83%. It demonstrates that the synthetic values are not optimal, indicating lower privacy, according to the K-Anonymity and DCR scores. Figure 10 shows that each column's overall NNDR score, which averages to 0.62, is likewise low. These results are more vulnerable to attacks because the value is low due to the dissimilarity of the NNDRs.

Figure 10. NNDR Scores for TimeGAN

Figure 11's graph illustrates the visual overlap between the synthetic and real values. The similarity of exact values in real and synthetic data indicates vulnerability to leakage, even

though the overlap between the two types of data suggests great utility. Low NNDRs and the overall quality report indicate that privacy is poor.

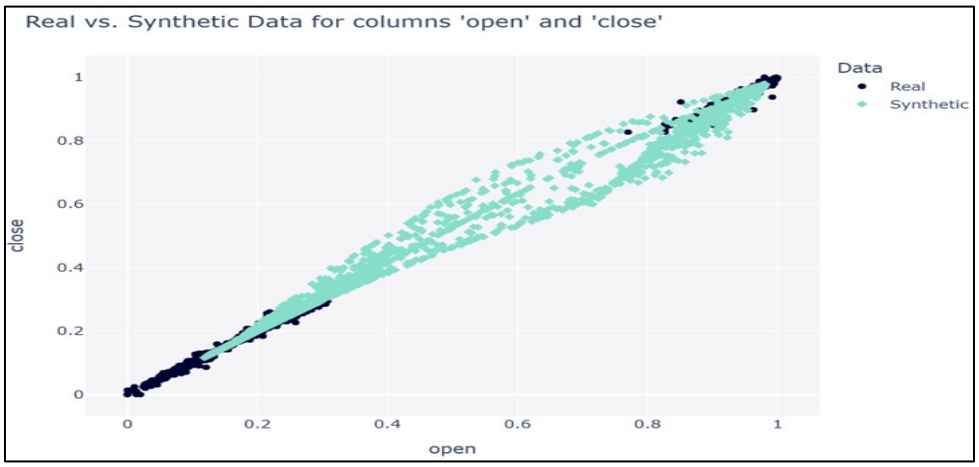

Figure 11. Scatterplot for TimeGAN

6.2.3 DP-TWGAN

DP-TWGAN has an overall quality report of 81.13%. This demonstrates that the synthetic values are optimal to guarantee privacy based on the K-Anonymity and DCR scores. Figure 12's columns display the extent to which each attribute's NNDR values match. As a result, privacy increases with the degree to which the synthetic values resemble the actual values. The DP-TWGAN offers high privacy, as evidenced by the average data quality of 0.70.

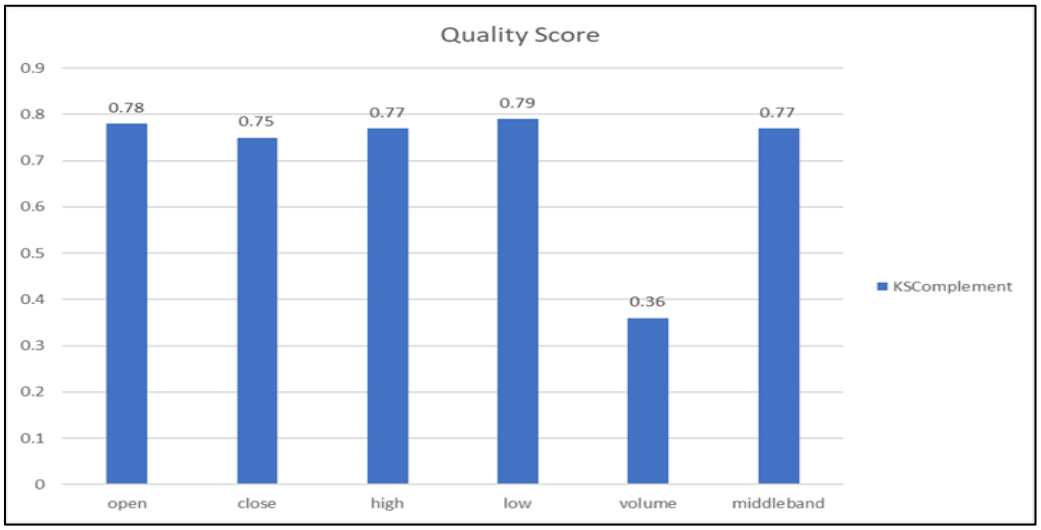

Figure 12. NNDR Scores for DP-TWGAN

The real values and the synthetic values are seen to overlap in Figure 13. In contrast to TimeGAN, the values do not exactly overlap. Therefore, it can be concluded that privacy is superior in synthetic data from DP-TWGAN, but there is a subsequent utility trade-off.

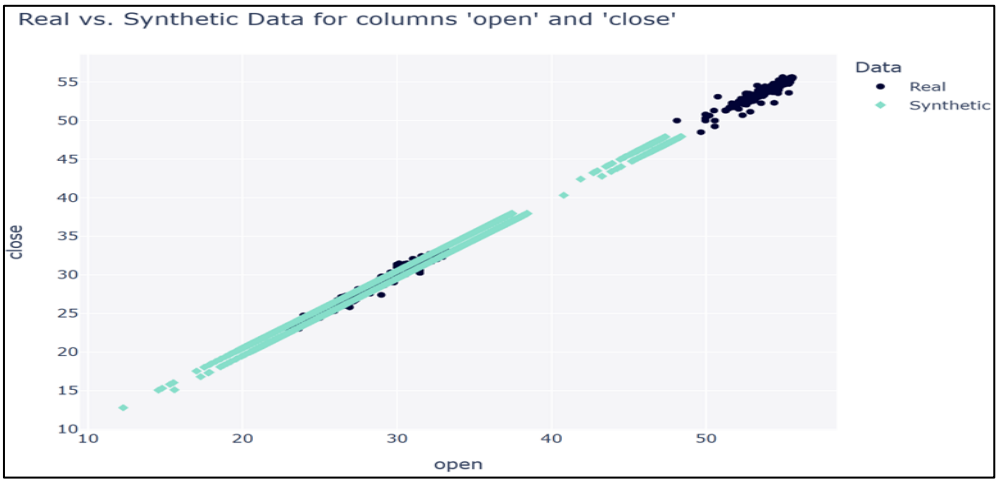

Figure 13. Scatterplot for DP-TWGAN

7. Conclusion

Machine learning and deep learning methods are also evolving and being used in increasingly real-life applications. To meet the humongous data needs of these models, significant research has been devoted to the creation of synthetic data. To leverage this information in lieu of frequently difficult-to-obtain, personally identifiable, or sensitive information in large amounts, the privacy and usefulness of synthetic data have been the central topics of this research. An architecture that leverages Wasserstein distance to enhance privacy protection and LSTM to capture temporal locality to produce synthetic time series tabular data has been used. The privacy of the created dataset was measured with metrics like DCR, NNDR, and SDV Quality Score. Post-finetuning the models and comparison with regard to various metrics, the proposed solution's privacy and utility have been determined comparatively. DP-TWGAN attained an RMSE score of 7.77 with better utility compared to DP-WGAN, which had an RMSE score of 21.07. In addition, the planned architecture produced more private data than TimeGAN since the former attained an overall quality report of 81.13% while the latter attained 64.83%. It is, therefore, noted that the proposed DP-TWGAN provides a superior utility for privacy trade off.

As additional improvements, acceptance in datasets can be validated with additional multivariate time-series datasets. Improved capture of the temporal relationships between the

input can be done through more suitable models with improved memory. Effective training strategies can also minimize computational requirements and facilitate the process of creating publishable trained generator models. In addition to improved generalization of acceptance in datasets, the conversion of time-series data into images and consequent pattern identification can significantly enhance privacy guarantees and defense against membership inference attacks.

References

[1] Madhu, M., and P. Whig. "A survey of machine learning and its applications." International Journal of Machine Learning for Sustainable Development 4, no. 1 (2022): 11-20.

[2] Kapoor, Sayash, and Arvind Narayanan. "Leakage and the reproducibility crisis in ML-based science." arXiv preprint arXiv:2207.07048 (2022).

[3] Liu, Ximeng, Lehui Xie, Yaopeng Wang, Jian Zou, Jinbo Xiong, Zuobin Ying, and Athanasios V. Vasilakos. "Privacy and security issues in deep learning: A survey." IEEe Access 9 (2020): 4566-4593.

[4] Sun, Hui, Tianqing Zhu, Zhiqiu Zhang, Dawei Jin, Ping Xiong, and Wanlei Zhou. "Adversarial attacks against deep generative models on data: A survey." IEEE Transactions on Knowledge and Data Engineering 35, no. 4 (2021): 3367-3388.

[5] Zhang, Yuheng, Ruoxi Jia, Hengzhi Pei, Wenxiao Wang, Bo Li, and Dawn Song. "The secret revealer: Generative model-inversion attacks against deep neural networks." In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, (2020): 253-261.

[6] Gupta, Rajesh, Sudeep Tanwar, Sudhanshu Tyagi, and Neeraj Kumar. "Machine learning models for secure data analytics: A taxonomy and threat model." Computer Communications 153 (2020): 406-440.

[7] Ghatak, Debolina, and Kouichi Sakurai. "A survey on privacy preserving synthetic data generation and a discussion on a privacy-utility trade-off problem." In International Conference on Science of Cyber Security, Singapore: Springer Nature Singapore, (2022): 167-180.

[8] Abadi, Martin, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. "Deep learning with differential privacy." In Proceedings of the 2016 ACM SIGSAC conference on computer and communications security, (2016): 308-318.

[9] Ha, Trung, Tran Khanh Dang, Tran Tri Dang, Tuan Anh Truong, and Manh Tuan Nguyen. "Differential privacy in deep learning: an overview." In 2019 International Conference on Advanced Computing and Applications (ACOMP), IEEE, (2019): 97-102.

[10] Mirza, Mehdi, and Simon Osindero. "Conditional generative adversarial nets." arXiv preprint arXiv:1411.1784 (2014).

[11] Arjovsky, Martin, Soumith Chintala, and Léon Bottou. "Wasserstein generative adversarial networks." In International conference on machine learning, PMLR, (2017): 214-223.

[12] Jordon, James, Jinsung Yoon, and Mihaela Van Der Schaar. "PATE-GAN: Generating synthetic data with differential privacy guarantees." In International conference on learning representations. 2018.

[13] TeMarvelde, Pepijn. "Differentially Private GAN for Time Series." CSE3000 research project, Delft University of Technology, http://resolver.tudelft.nl/uuid:8c4171d0-db68-4235-badb-6e57953162b8 (2021).

[14] Analytics, Data, and Valtteri Nieminen. "Differentially private synthetic tabular data generation with a generative adversarial network and privacy amplification by subsampling." (2022).

[15] Song, Shuang, Kamalika Chaudhuri, and Anand D. Sarwate. "Stochastic gradient descent with differentially private updates." In 2013 IEEE global conference on signal and information processing, IEEE, (2013): 245-248.

[16] Ghosheh, Ghadeer, Jin Li, and Tingting Zhu. "A review of Generative Adversarial Networks for Electronic Health Records: applications, evaluation measures and data sources." arXiv preprint arXiv:2203.07018 (2022).

[17] L. Xie, K. Lin, S. Wang, F. Wang, and J. Zhou, "Differentially Private Generative Adversarial Network with Representation Learning," IEEE Transactions on Knowledge

and Data Engineering, vol. 33, no. 11, Nov. (2021): 3784–3797. doi: 10.1109/TKDE.2020.2981333.

[18] R. Chen, M. Yu, M. Zhang, L. Yu, and L. Fan, "PrivSyn: Differentially Private Data Synthesis via Generative Adversarial Networks," in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 37, no. 7, (2023): 7542–7550. doi: 10.1609/aaai.v37i7.25973.

[19] C. Esteban, S. L. Hyland, and G. Rätsch, "Real-valued (Medical) Time Series Generation with Recurrent Conditional GANs," in Proceedings of the Machine Learning for Healthcare Conference (MLHC), PMLR 149: (2022): 325–350.

[20] R. Torkzadehmahani, P. Kairouz, and B. Paten, "DP-CGAN: Differentially Private Synthetic Data and Label Generation," IEEE Transactions on Dependable and Secure Computing, vol. 20, no. 1, Jan.–Feb. (2023): 190–204. doi: 10.1109/TDSC.2021.3119550.

[21] B. Zhang, J. Sun, J. Zhao, and Y. Zhu, "Towards Privacy-Preserving Time-Series Generation via Dual-Stage GANs," arXiv preprint arXiv:2209.09977, 2022. [Online]. Available: https://arxiv.org/abs/2209.09977

[22] A. Mohamed, U. Thakker, and B. Li, "SecureGAN: Scalable Differentially Private GANs for High-Dimensional Data," in NeurIPS 2023 Workshop on Synthetic Data Generation, New Orleans, LA, USA, 2023. Available: https://openreview.net/forum?id=secgan23