Early Detection of Mode Collapse in GANs Through Loss Monitoring

Farhat Lamia Barsha, William Eberle
Department of Computer Science, Tennessee Technological University
fbarsha42@tntech.edu, weberle@tntech.edu

***Abstract*—Generative Adversarial Networks (GANs) have become a powerful tool for generating synthetic data, which is valuable in domains like credit card fraud detection, where the availability of real data may be limited. However, GANs are prone to mode collapse, a phenomenon where the model produces repetitive samples with limited diversity, reducing the quality and utility of the generated data. Early detection and mitigation of mode collapse is crucial to maintaining sample diversity, conserving computational resources, and improving model robustness. Despite various strategies for addressing this issue, identifying the collapse as it begins remains a significant challenge. Very few studies have focused on detecting mode collapse at the earliest possible stage, though increased attention in this area is essential for improving GANs' performance. This paper proposes a novel approach to detect mode collapse early by monitoring fluctuations in the generator and discriminator loss values throughout training. Our approach initiates mode collapse detection dynamically after the model stabilizes during training, making it adaptable to any GAN architecture. By identifying collapse at its onset, our method allows for prompt intervention through targeted mitigation strategies, reducing wasted computational effort. As a result, this approach enhances GAN stability, improving its applicability and reliability across various real-world scenarios, from image synthesis to fraud detection.**

***Keywords*—Generative Adversarial Networks, Mode Collapse, Generator, Discriminator, Unsupervised Learning.**

I. Introduction

Data privacy has become a critical issue in today's technologically advanced world. Although new technologies offer several advantages, they also pose concerns, particularly with the increasing incidence of personal data theft. In the domain of credit card fraud, Experian's vice president emphasizes that fraud has expanded beyond basic credit card theft and now includes many aspects of an individual's financial and identification information stealing, such as social security numbers [1]. The financial consequences of fraud are significant: worldwide card losses reached $246 million in 2023, with the United States accounting for around 38.83% of these losses (Nilson Report) [2]. The ongoing risk highlights the necessity for advanced machine learning (ML) algorithms that can effectively detect fraudulent transactions. Nonetheless, restricted availability to real-world data limits this advancement, prompting companies to consider synthetic data as a feasible substitute. Generative Adversarial Networks (GANs) have come up as a viable option, facilitating the creation of high-quality synthetic data for the training and enhancement of fraud detection models, thus proficiently tackling the obstacle of data accessibility.

A GAN is a machine learning model that produces synthetic data through a unique adversarial training procedure involving two neural networks [3]. GANs have been extensively utilized in diverse fields, such as image generation, music composition, and text synthesis. The first network in a GAN, referred to as the generator, is designed to produce high-quality synthetic data that closely mimics real data. Simultaneously, the second network, the discriminator, functions to differentiate between real and generated data. During training, the generator initiates the production of synthetic data from random noise and progressively enhances its outputs depending on feedback from the discriminator, which assesses its ability to differentiate between real and generated data. Initially, the generator's output is readily identifiable as fake; but, as the discriminator improves its detection capabilities, the generator adjusts, fine-tuning its parameters to produce more realistic data. This adversarial process enhances both networks, ultimately resulting in synthetic data that is almost identical to real data.

In GANs, two principal loss functions are employed: the generator loss and the discriminator loss. The generator loss is determined by the accuracy with which the discriminator classifies the generated samples as real. The loss is minimized when the discriminator increasingly misclassifies generated samples as real, signifying that the generator is effectively fooling the discriminator by generating good-quality synthetic data. On the other hand, The discriminator loss evaluates the discriminator's capacity to distinguish between real and synthetic data. A lower discriminator loss indicates that the discriminator can effectively differentiate between real samples and generated samples [4].

In the GAN design, the losses for both the generator and discriminator are determined from a common metric that measures the divergence between the real and generated data distributions. Throughout the training process, the discriminator is trained initially to enhance its capacity to distinguish between real and fake data; thereafter, the generator is trained to create more realistic synthetic data capable of misleading the discriminator. This repeated training loop persists, gradually enhancing the performance of both networks via adversarial learning. The minimax game between the generator G and the discriminator D is expressed by the goal function [5]:

$$\begin{aligned}\min_G \max_D V(D,G) =& \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log D(x)] \\ &+ \mathbb{E}_{z\sim p_z(z)}[\log(1-D(G(z)))]\end{aligned} \tag{1}$$

In this function, $D(x)$ represents the probability that input x is real, and $G(z)$ represents the generator's output for a noise vector z. Here, $p_{\text{data}}(x)$ is the real data distribution, where $p_z(z)$ is the noise distribution. The first section of the equation prompts the discriminator to identify real data, whereas the next section drives the generator to generate samples considered real by the discriminator.

Training GANs pose several challenges, including mode collapse, maintaining a dynamic equilibrium between the networks, the non-convex nature of the optimization process, and overall instability during training [6]. Mode collapse occurs when the generator becomes overly focused on generating a narrow set of data patterns that successfully deceive the discriminator, neglecting the diversity of the data distribution [7]. For example, when tasked with generating images of digits from 0 to 9, a GAN experiencing mode collapse might only produce images of the digit "9," ignoring the rest [8]. Various factors may contribute to the occurrence of mode collapse, such as:

- **Inconsistent Learning Rates:** When the discriminator's learning rate drastically outperforms that of the generator, the generator finds it challenging to efficiently utilize gradients for learning. This imbalance limits the diversity of generated samples [9].
- **Limited Input Data Variability:** When the input data offers minimal variances, the generator may struggle to generate a wide array of outputs, hence limiting the diversity of the generated content [10].
- **Convergence to a Local Minimum:** If the generator's training convergence attains a local minimum instead of a global one, it may generate a limited range of outputs, failing to capture the full data distribution accurately [11].

This phenomenon severely limits the model's performance and compromises the quality of the generated outputs. The repetitive and narrow samples produced fail to represent the full complexity of the data, reducing their utility for practical applications. Moreover, mode collapse hinders the training process, as the generator struggles to benefit from the discriminator's feedback, leading to stagnation and slow convergence. Several underlying factors contribute to this issue, including catastrophic forgetting, discriminator overfitting [12], and generator instability [13].

Identifying and addressing mode collapse is essential for the effective training of GANs. Although extensive research has concentrated on resolving this issue, there has been limited work on early detection approaches for mode collapse. Immediate identification is essential for sustaining efficient training and generating better results. Mode collapse restricts the diversity of generated data, causing the generator to consistently provide similar outputs, which significantly undermines model performance. Early identification of this issue facilitates prompt corrective measures, resulting in more effective training and faster convergence. Furthermore, early diagnosis preserves computational resources by preventing the ongoing training of an insufficient model.

In this paper, we propose an early detection method for mode collapse, aimed at promptly identifying the issue by continuous monitoring of generator and discriminator losses. Given that most GAN models encounter initial instability, our method first permits the model to achieve stabilization. Upon achieving a stable balance between the generator and discriminator losses, the detection procedure begins. Indicators of mode collapse are identified through inconsistencies in losses, signifying that the generator is producing limited or repetitive outputs. Upon observing such variations, the algorithm indicates a potential mode collapse and pauses training, allowing for prompt intervention.

This work makes the following contribution:

- **Early Detection Framework**: We propose an innovative approach for the early identification of mode collapse in GANs, employing continuous observation of generator and discriminator losses.
- **Stabilization-Aware Detection**: Our approach addresses early training instability by permitting the model to stabilize prior to initiating the detection procedure, hence ensuring enhanced accuracy.

We organize the rest of the paper as follows: Section II provides an overview of prior research. Section III presents experimental datasets. Section IV breaks down the experimental setting, while Section V highlights our findings. The study concludes with a summary and future work in Section VI.

II. Related works

Although relatively little work has focused on the early detection of mode collapse, researchers have developed various metrics to identify the issue and proposed approaches to address the issue over the years. In this section, we will review some prior research works that have explored methods for detecting mode collapse in GANs.

In 2018, Sayeri et al. [14] assessed various GAN architectures, including AdaGAN, VEEGAN, Wasserstein GAN, and Unrolled GAN, with an emphasis on their vulnerability to mode collapse. AdaGAN had superior performance across the majority of datasets, but Wasserstein GAN encountered difficulties. The research indicates that no singular metric can accurately measure mode collapse, as different metrics produce inconsistent outcomes. The authors recommend the utilization of multiple metrics, such as the Adjusted Coverage metric, to achieve a more precise evaluation of mode collapse.

Adiga et al. [15] introduce two innovative evaluation metrics, Mode Collapse Divergence (MCD) and Generative Quality Score (GQS), to quantitatively evaluate the efficacy of Generative Adversarial Networks (GANs). These measures are explicitly formulated to detect mode collapse and assess the quality of synthetic samples. The study used these metrics to evaluate several GAN designs, including vanilla GAN, WGAN, LSGAN, PacGAN, and CGAN, across diverse datasets. The findings demonstrate that the efficacy of each GAN architecture is heavily influenced by the underlying dataset, with MCD and GQS correlating closely with human visual evaluations. These measurements provide an efficient way to select the optimal GAN framework by evaluating the balance between sample quality and mode collapse.

In 2019, Jia et al [16] proposed the Siamese Score, an innovative metric utilizing a Siamese network to identify intra-class mode collapse in GANs. In contrast to conventional measures such as the Inception Score (IS) or Fréchet Inception Distance (FID), the Siamese Score uses an embedding space to detect mode collapse with greater efficiency. Experiments indicate that it is not only more efficient but also considerably quicker, decreasing computing expenses by roughly 59 times relative to the Inception Score.

In [17], the authors introduce an approach for detecting and alleviating mode collapse through the examination of the spectral distribution of the weight matrices within the discriminator network. Mode collapse results in a substantial reduction of numerous unique values inside the discriminator's spectral distribution, a phenomenon referred to as spectral collapse. This failure diminishes the discriminator's capacity to differentiate between real and fake samples, resulting in the generator producing outputs that are substantially similar or identical. The paper proposes the monitoring of spectral values as a metric for mode collapse and the integration of spectral regularization to address the problem.

In 2021, Wu et al. [18] examined intra-mode collapse, defined by a deficiency of diversity across generated samples within specific data modes. The authors offer a black-box approach for detecting and diagnosing intra-mode collapse through statistical methods and Monte Carlo sampling without requiring access to the training data or model parameters. The Monte Carlo-based Collapse Score (MCCS) is introduced as a novel metric for evaluating collapse by pinpointing areas where the generated data exhibits excessive similarity.

In 2022, Saad et al. [19] assessed mode collapse across various datasets, concluding that AdaGAN outperforms other models while the Wasserstein GAN exhibits poor performance. Their findings demonstrate that no singular metric adequately assesses mode collapse in GANs. They proposed Adaptive Instance Normalization Initialization (AIIN) for DCGAN to resolve this issue. DCGAN combined with AIIN generates a greater variety of X-ray pictures, enhancing MS-SSIM and FID metrics. AIIN preprocessing outperforms conventional methods such as Gaussian and median filtering, with augmented X-ray pictures improving the efficacy of machine learning classifiers.

In 2024, Farhat et al. [20] emphasized identifying mode collapse in GANs employed for generating numerical data, specifically with credit card fraud detection. The authors presented an early detection technique that considers multiple metrics, including generator and discriminator losses, Wasserstein distance, t-SNE visualization, and precision and recall, to provide a comprehensive evaluation. As it's difficult to detect mode collapse efficiently using a single metric, their findings indicate that this multi-metric method facilitates the early detection of mode collapse, resulting in enhanced stability and performance in GAN-generated data for fraud detection tasks.

While the previously listed papers mostly concentrate on comparing different GAN approaches regarding mode collapse or proposing metrics for its identification, this paper introduces an *automated mode collapse detection approach* at its earliest stages, applicable to any GAN model. Our methodology prioritizes the observation of generator and discriminator losses to assess GAN performance. By dynamically identifying the stabilization phase and initiating mode collapse detection afterward, the method becomes adaptable to any GAN architecture. By detecting mode collapse early, we can enhance efficiency and significantly reduce computing time and expenses.

III. Dataset

In this paper, we utilize two datasets from Kaggle: a synthetic dataset [21] and a real-world dataset [22]. The synthetic dataset consists of machine-generated data simulating ”Card Not Present” fraud, with 151,112 transactions and a 9.36% fraud rate, featuring variables like purchase value, device ID, browser, and IP address. The real-world dataset contains 284,807 transactions from European cardholders over two days in September 2013, with only 0.17% fraudulent transactions, representing actual human transactions. Due to privacy concerns, most features in the real dataset are PCA-transformed, with 'Time' and 'Amount' being the only unaltered features.

IV. Experimental setup

This section outlines the key steps in developing the early mode collapse detection approach. The following sections demonstrate the data preprocessing, model architecture, training methodology, and the development of the mode collapse detection method in detail.

A. Hardware and Software Configuration

1) Software: Following Python libraries have been utilized to develop the approach: PyTorch for model training and data management, scikit-learn for dataset partitioning and assessment, Pandas for data manipulation, and Matplotlib for visual representation.

2) Hardware: The models were trained on a system utilizing either a CUDA-compatible GPU or a CPU, depending upon hardware availability. The training environment automatically identified and employed the GPU when accessible, as indicated by torch.device('cuda' if torch.cuda.is_available() else 'cpu').

B. Data Preprocessing

The dataset was preprocessed initially by eliminating redundant attribute columns, with a primary emphasis on transaction time, amount, and class attributes, as shown in Table I. After extracting essential features, the dataset was balanced by undersampling the non-fraud class. The dataset was then divided into features (X) and the target label (Y), followed by an 80/20 train-test partition with a fixed random seed for reproducibility. The training data was subsequently transformed into PyTorch tensors and imported into a DataLoader for batch processing, utilizing a batch size of 64.

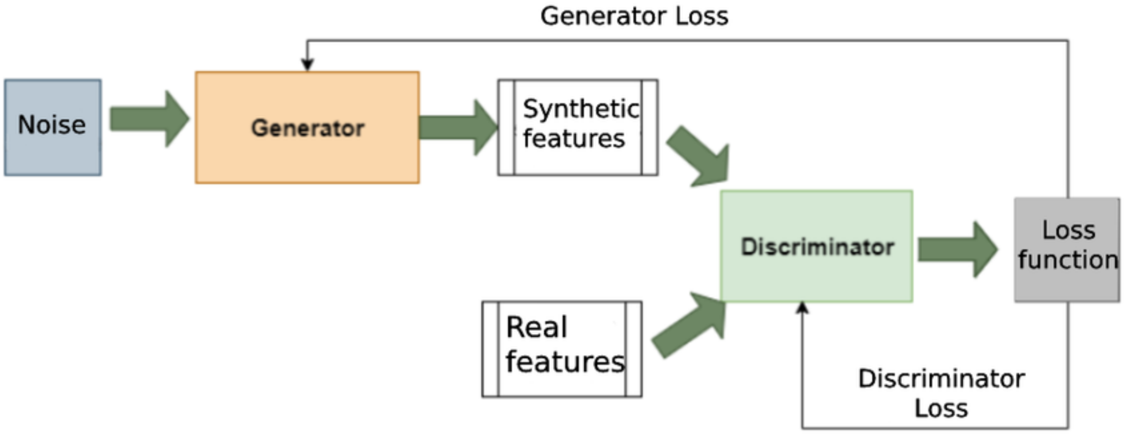


Fig. 1: WGAN-GP architecture [23]

TABLE I: Feature Categories

Feature Name	Feature Type	Description
Transaction Time	Numerical	Purchase time (synthetic dataset), number of seconds since first transaction (real dataset)
Transaction amount	Numerical	Transaction amount
Class	Numerical	Indicates whether a transaction is fraudulent or real

C. Model Architecture

In these experiments, a Wasserstein Generative Adversarial Network with Gradient Penalty (WGAN-GP) [24] was employed to generate synthetic data. Figure 1 illustrates the basic WGAN-GP architecture. The Generator network has been designed to accept a random noise vector as input, commonly known as the latent vector (latent_dim = 100), and then produce synthetic samples that replicate the original dataset's feature distribution. The generator network consists of four fully linked layers, each utilizing ReLU activations to maintain non-linearity and enhance learning. The generator's output corresponds to the dimensions of the original dataset's feature space. The Discriminator network comprises several fully linked layers and employs LeakyReLU activations to facilitate the learning of more intricate patterns, especially for small negative values. The discriminator produces a singular score that determines the authenticity of the input data as either real or fake. A gradient penalty is implemented during the discriminator's training to enforce the Lipschitz requirement. Both networks were initialized utilizing the weights_init function, which assigns weights to all layers based on a normal distribution, thereby enhancing the training stability of GANs.

D. Training Process

The WGAN-GP model was trained for 500 epochs, following a defined training protocol to ensure the convergence and stability of the model. For each generator update, the discriminator was updated five times, following the conventional WGAN-GP methodology, wherein the discriminator is crucial in ensuring that the generated data is indistinguishable from real data. This experiment utilized loss functions such as the Wasserstein loss, wherein the discriminator's loss was determined by the difference between the average scores attributed to false and real data, supplemented by a gradient penalty term to uphold the 1-Lipschitz constraint. The generator's loss was the negative of the fake data score, motivating it to generate real samples. A feature matching loss was also implemented to enhance training stability. The loss was calculated using the features recovered by the discriminator from both real and fake data, aiming to align their feature distributions. Both networks were optimized with the Adam optimizer, featuring a learning rate of 0.0002 and beta values of (0.5, 0.999), which are standard in GAN training to facilitate convergence and mitigate oscillations in loss values. To maintain equilibrium between the generator and discriminator, careful observation of their individual losses was conducted during the training process, and the discriminator was regulated using a gradient penalty to avoid overfitting to real data. Table II lists all the hyperparameters used in the training process, along with their values and descriptions.

E. Mode Collapse Detection Approach

During the early phases of GAN training, the generator loss exhibits considerable fluctuations due to the dynamic and adversarial nature of the learning process. Initially, the generator has a limited knowledge of the data distribution and generally produces outputs that seem random or unrealistic. This results in the discriminator, which is initially more robust and efficient at differentiating between real and generated data, delivering extremely varying feedback. Therefore, the generator obtains inconsistent gradients, resulting in rapid fluctuations in its loss. Furthermore, the discriminator typically acquires knowledge more rapidly initially, as its objective of differentiating between real and fake is comparatively uncomplicated, resulting in the generator ”pursuing” the discriminator's feedback, hence

TABLE II: Hyperparameters Used in Training

Hyperparameter	**Value**	**Description**
latent_dim	100	Dimension of the random noise vector input to the generator.
lr	0.0002	Learning rate for both generator and discriminator optimizers.
n_epochs	500	Number of training epochs.
n_critic	5	Number of discriminator updates per generator update.
lambda_gp	10	Weight for the gradient penalty in the discriminator loss function.
batch_size	64	Number of samples per batch during training.
random_state	99	Seed for reproducibility in train-test split.
betas	(0.5, 0.999)	Coefficients for computing running averages in Adam optimizer.
stabilization_window	10	Window size for dynamic stabilization detection in loss values.
stability_threshold	0.5	Threshold for detecting stability in loss values (sensitivity).
feature_matching_weight	10	Weight applied to the feature matching loss for the generator.

increasing loss variations. The fluctuations are impacted by the generator's initial output, which only partially reflects the actual data distribution and frequently fails to include important details. As time progresses, the generator enhances its representations and generates outputs that more accurately resemble real data, resulting in less unpredictable feedback from the discriminator and a stabilization of the generator loss. The instability at the beginning is a primary reason for activating mode collapse detection only after the model has had sufficient time to stabilize. Stability was considered achieved when the relative variation in losses across subsequent epochs dropped below a threshold of 0.5, signifying that, on average, the alteration in loss values within the stabilization window was under 50%. Establishing the threshold at this level enabled the stability detection process to be sufficiently sensitive to identify significant stability while limiting false positives from small fluctuations.

As training progressed, the variations in generator and discriminator loss decreased, resulting in a more stable differential between the two losses, indicating that the generator was producing better outputs. Upon detecting stability, mode collapse detection was initiated by observing the losses subsequent to stabilization: if either the generator or discriminator loss exceeded 50% of their respective peak recorded values, it was identified as a potential sign of mode collapse. An early stopping procedure was initiated to terminate training and avert a further decline in model performance.

F. Algorithm

Figure 2 represents the algorithm of our earlier mode collapse detection approach. The algorithm initiates with three primary inputs: the number of epochs (n_epochs), the stabilization window (stabilization_window), and the stability threshold (stability_threshold). It also initializes variables for maximum losses, counters for stable epochs, and indicators for stabilization. The primary training loop runs through each epoch and batch and calculates the generating (loss_G) and discriminator (loss_D) losses while updating the maximum observed losses when current values surpass the recorded maxima. If stabilization has not been seen, the algorithm assesses the stability of the generator and discriminator losses. If the generator's losses remain inside the threshold within the designated stabilization window, the stable_gen_epochs counter is increased; if not, it is reset. A similar stability assessment and counter-updating occurs for the discriminator losses. Once both stable_gen_epochs and stable_disc_epochs reach the stabilization window, the is_stabilized flag is set to True. If either loss exceeds fifty percent of its maximum documented loss, mode collapse is indicated, and a warning message is generated along with the epoch number. Upon detection of mode collapse, the loop terminates quickly. If no mode collapse is seen during training, the model runs through all epochs, and a confirmation message is displayed at the conclusion that no mode collapse has been detected. This approach facilitates the prompt identification of mode collapse, hence enhancing the effective training of GANs, saving time, cost, and resources.

G. Evaluation and Visualization

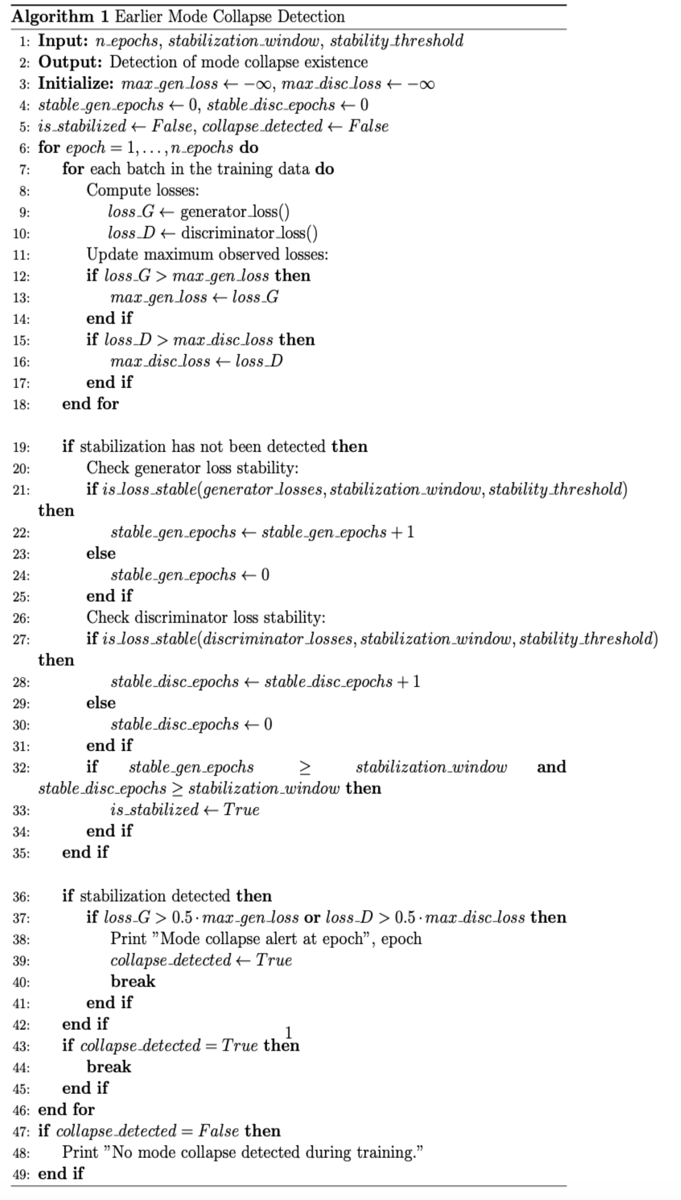
Both quantitative and qualitative metrics were utilized to assess the performance of the WGAN-GP model. Throughout the training process, the losses of the generator and discriminator were documented for each epoch, and these losses were subsequently graphed over time to evaluate the model's convergence. A well-functioning GAN exhibits a decline in generator loss and a stabilization in discriminator loss. Furthermore, the losses of both the generator and discriminator were visually compared through plots to verify that the networks did not overfit and maintained stability during the training phase.

V. Experimental Result

This section presents outcomes from experiments conducted using both real and synthetic datasets across four separate scenarios: the entire real-world dataset, a subset of the real-world dataset, the entire synthetic dataset, and a subset of the synthetic dataset. Each scenario was examined utilizing a mode collapse detection methodology to observe and assess the results. This configuration enables the comparison of the efficacy and dependability of mode collapse detection across various dataset configurations, providing insights into the behavior and performance of synthetic data in fraud detection algorithms.

A. Mode collapse exists scenarios

1) Entire synthetic dataset used: In this experiment, we used the entire synthetic dataset, including 151,112 samples. As shown in Figure 3, the generator loss exhibits continuous



Algorithm 1 Earlier Mode Collapse Detection

```
Input: $n\_epochs$, $stabilization\_window$, $stability\_threshold$
Output: Detection of mode collapse existence
Initialize: $max\_gen\_loss \leftarrow -\infty$, $max\_disc\_loss \leftarrow -\infty$
$stable\_gen\_epochs \leftarrow 0$, $stable\_disc\_epochs \leftarrow 0$
$is\_stabilized \leftarrow False$, $collapse\_detected \leftarrow False$
for $epoch = 1, \ldots, n\_epochs$ do
    for each batch in the training data do
        Compute losses:
            $loss\_G \leftarrow$ generator_loss()
            $loss\_D \leftarrow$ discriminator_loss()
        Update maximum observed losses:
        if $loss\_G > max\_gen\_loss$ then
            $max\_gen\_loss \leftarrow loss\_G$
        end if
        if $loss\_D > max\_disc\_loss$ then
            $max\_disc\_loss \leftarrow loss\_D$
        end if
    end for

    if stabilization has not been detected then
        Check generator loss stability:
        if $is\_loss\_stable(generator\_losses, stabilization\_window, stability\_threshold)$ then
            $stable\_gen\_epochs \leftarrow stable\_gen\_epochs + 1$
        else
            $stable\_gen\_epochs \leftarrow 0$
        end if
        Check discriminator loss stability:
        if $is\_loss\_stable(discriminator\_losses, stabilization\_window, stability\_threshold)$ then
            $stable\_disc\_epochs \leftarrow stable\_disc\_epochs + 1$
        else
            $stable\_disc\_epochs \leftarrow 0$
        end if
        if $stable\_gen\_epochs \geq stabilization\_window$ and $stable\_disc\_epochs \geq stabilization\_window$ then
            $is\_stabilized \leftarrow True$
        end if
    end if

    if stabilization detected then
        if $loss\_G > 0.5 \cdot max\_gen\_loss$ or $loss\_D > 0.5 \cdot max\_disc\_loss$ then
            Print "Mode collapse alert at epoch", epoch
            $collapse\_detected \leftarrow True$
            break
        end if
    end if
    if $collapse\_detected = True$ then
        break
    end if
end for
if $collapse\_detected = False$ then
    Print "No mode collapse detected during training."
end if
```

Fig. 2: Earlier Mode Collapse Detection Algorithm

fluctuations, signifying the model's difficulty in precisely reproducing the synthetic data. However, around epoch 30, it starts to stabilize, indicating a temporary enhancement in the model's capacity to resemble the data distribution. As training approaches epoch 140, fluctuations in generator loss increase, indicating a possible mode collapse. Mode collapse, characterized by the model generating repetitive or constrained data patterns, is evident around epoch 148, highlighting difficulties in preserving data diversity within the synthetic dataset.

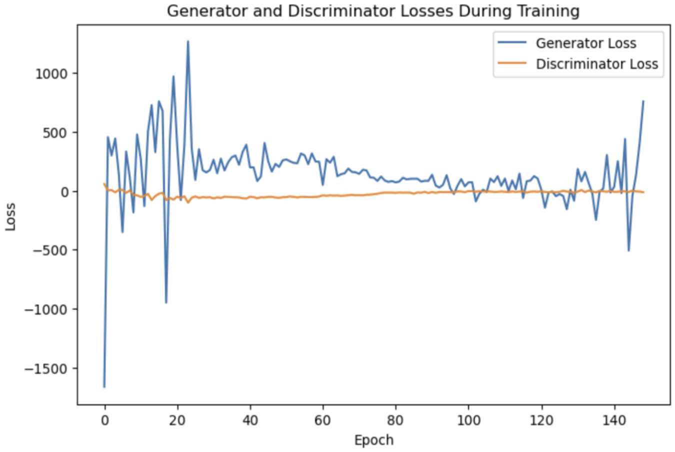


Fig. 3: Entire synthetic dataset used

2) *Entire real dataset used:* In this experiment, we utilized the complete real-world dataset, including 284,807 samples. Figure 4 illustrates that the generator loss demonstrates significant fluctuation during the training phase, signifying difficulties in precisely modeling the data distribution. Although the discriminator loss remains rather consistent, the fluctuations continue until approximately epoch 27, at which point a significant pattern emerges, indicating a possible mode collapse.

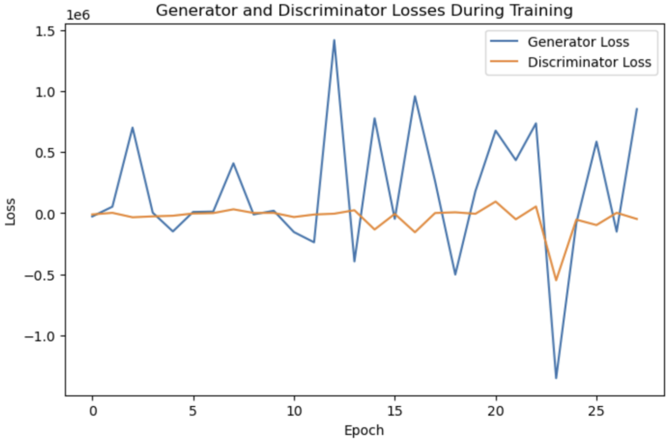


Fig. 4: Entire real dataset used

3) *Subset of the real dataset (10000 samples) used:* This experiment utilized a fraction of the real dataset, comprising 10,000 samples, to train the model. Figure 5 illustrates that the generator loss exhibits initial fluctuations, whereas the discriminator loss remains comparatively stable over the epochs, signifying a consistent assessment of the generator's efficacy. Significantly, during epoch 22, the generator loss experiences a substantial jump, followed by a steep decline, indicating possible indications of mode collapse.

B. *No Mode collapse scenarios*

1) *Subset of the synthetic dataset (3000 samples) used:* In this experiment, a portion of the synthetic dataset containing 3,000 samples was utilized for training. Figure 6 illustrates that both generator and discriminator losses rapidly stabilize after initial epochs. The generator loss initially declines significantly, gradually aligning closely with the discriminator

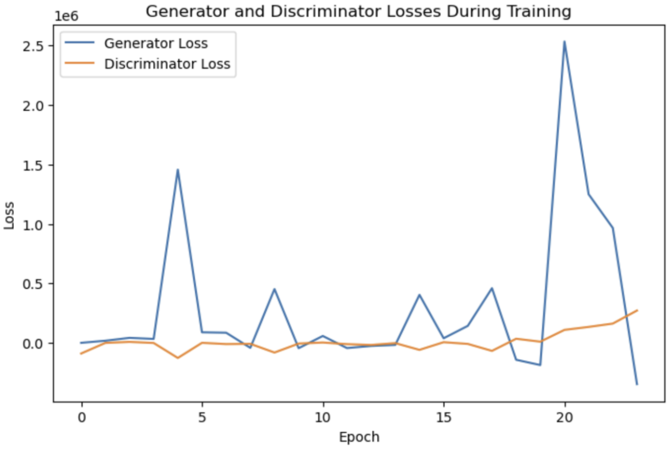


Fig. 5: Subset of the real dataset (10000 samples) used

loss and maintaining stability during the following epochs. This stability signifies the absence of mode collapse since the generator loss exhibits no major fluctuates or instability over time.

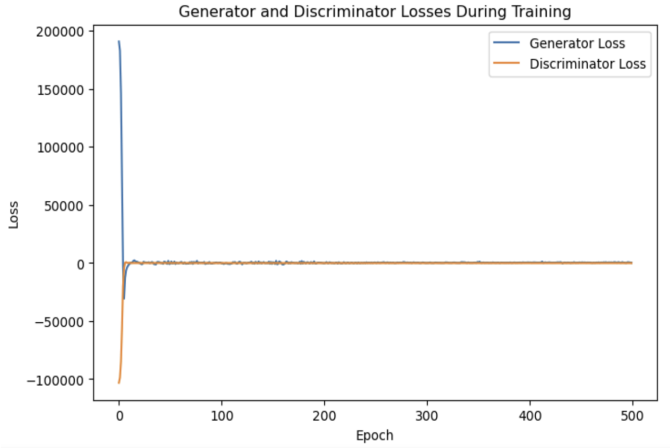


Fig. 6: Subset of the synthetic dataset (3000 samples) used

The stable loss values indicate that the model can provide diverse outputs without redundancy, showing that this synthetic data subset offers significant variability for effective learning without overfitting or collapse. This outcome diverges from other scenarios, emphasizing that a compact, well-structured synthetic dataset can facilitate consistent and reliable training in generative tasks.

Timely identification of mode collapse is essential for avoiding substantial computational resources during model training. By detecting mode collapse early, we can instantly terminate the training process, conserve computational resources, and lower the training time. Mode collapse is detected at epoch 96, reducing the training time to just 1,142.78 seconds while utilizing the entire synthetic dataset. Conversely, in the absence of early identification, we must permit the entire training process to finish in order to determine whether mode collapse occurred or not, hence extending the total training time to 4,835.83 seconds. A similar thing happens while working with both datasets. This proposed approach reduces resource consumption and accelerates experimentation cycles, facilitating quicker model enhancement and implementation.

VI. Conclusion and Future work

In this work, we present a novel approach for the early detection of mode collapse in Generative Adversarial Networks (GANs) by continuously monitoring the loss patterns of both the generator and discriminator. We considered the initial phase of instability in generator and discriminator losses and deployed our detection approach once the model started to become stable. This allows for our methodology to be adaptable across diverse GAN architectures. This strategy has been evaluated in four diverse scenarios, including with and without mode collapse, illustrating the robustness and adaptability of our model. This approach would allow the GAN model to stop the training immediately rather than waiting until the end of the entire training process, which would save computational costs and resources. The experimental results confirm the method's efficacy in promptly detecting mode collapse, facilitating timely action to avoid repetitive outputs. This approach provides a significant resource for improving GAN stability and effectiveness in critical applications, including fraud detection and synthetic data synthesis.

A limitation of our study is the challenge associated with directly comparing our technique with existing approaches because of variations in data types and evaluation metrics. Although the majority of prior research on mode collapse identification concentrates on image data, our research emphasizes numerical data, resulting in different performance metrics that limit direct comparisons. Moreover, in contrast to the majority of current research that focuses on mode collapse detection, our methodology prioritizes early detection, facilitating the proactive termination of training to optimize resource conservation.

In the future, we intend to modify this methodology for a broader spectrum of GAN architectures, examining its versatility and effectiveness across various model configurations. Applying this technology to further domains requiring high-quality synthetic data, including medical imaging, natural language generation, and other sensitive data areas, provides a potential research direction. These additions will enhance the approach's versatility and extend its applicability in crucial domains where data authenticity and variety are essential.

VII. Acknowledgments

The authors wish to thank the Machine Intelligence and Data Science (MInDS) Center and the Computer Science Department at Tennessee Tech for providing resources and funding to work on this project.

References

[1] "Why credit card fraud alerts are rising," https://www.cnbc.com/2024/09/12/why-credit-card-fraud-alerts-are-rising.html#:~:text=Global%20card%20losses%20attributed%20to,in%20the%20decade%20to%202032., (Accessed on 10/29/2024).

[2] "Credit card fraud statistics for 2024," https://wallethub.com/edu/cc/credit-card-fraud-statistics/25725#:~:text=Credit%20%26%20Debit%20Cards)-,Total%20Value%20of%20Credit%20Card%20Fraud%20by%20Year,rise%20from%20the%20previous%20year., (Accessed on 10/29/2024).

[3] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[4] "Understanding loss functions in gans: Gan training and impact on results — by mahzaib khalid — medium," https://shorturl.at/ojkdn, (Accessed on 10/31/2024).

[5] H. Dwivedi, "Understanding gan loss functions," https://neptune.ai/blog/gan-loss-functions, August 2023, (Accessed on 10/31/2024).

[6] "Challenges in training gan-generative adversarial network — by mohd usama — medium," https://medium.com/@usama.6832/why-its-hard-to-train-gan-generative-adversarial-network-a05a7656f26d#:~:text=Generative%20Adversarial%20Networks%20(GANs)%5B,problem%2C%20and%20instability%20while%20training., (Accessed on 10/21/2024).

[7] Y. Kossale, M. Airaj, and A. Darouichi, "Mode collapse in generative adversarial networks: An overview," in *2022 8th International Conference on Optimization and Applications (ICOA)*. IEEE, 2022, pp. 1–6.

[8] L. Metz, B. Poole, D. Pfau, and J. Sohl-Dickstein, "Unrolled generative adversarial networks," *arXiv preprint arXiv:1611.02163*, 2016.

[9] D. Saxena and J. Cao, "Generative adversarial networks (gans) challenges, solutions, and future directions," *ACM Computing Surveys (CSUR)*, vol. 54, no. 3, pp. 1–42, 2021.

[10] "Mode collapse: Understanding the challenge in gans - ak codes," https://ak-codes.com/mode-collapse/, (Accessed on 10/31/2024).

[11] "What is mode collapse in gans?. in my first article about gans, we… — by miray topal — medium," https://medium.com/@miraytopal/what-is-mode-collapse-in-gans-d3428a7bd9b8, (Accessed on 10/21/2024).

[12] "Gan mode collapse explanation. a detailed analysis of the causes of… — by ainur gainetdinov — towards ai," https://pub.towardsai.net/gan-mode-collapse-explanation-fa5f9124ee73, (Accessed on 10/31/2024).

[13] "Gans failure modes: How to identify and monitor them," https://neptune.ai/blog/gan-failure-modes, (Accessed on 10/31/2024).

[14] S. Lala, M. Shady, A. Belyaeva, and M. Liu, "Evaluation of mode collapse in generative adversarial networks," *High performance extreme computing*, 2018.

[15] S. Adiga, M. A. Attia, W.-T. Chang, and R. Tandon, "On the tradeoff between mode collapse and sample quality in generative adversarial networks," in *2018 IEEE global conference on signal and information processing (GlobalSIP)*. IEEE, 2018, pp. 1184–1188.

[16] J. Jia and Q. Zhao, "Siamese score: Detecting mode collapse for gans," in *2019 12th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*. IEEE, 2019, pp. 1–6.

[17] K. Liu, W. Tang, F. Zhou, and G. Qiu, "Spectral regularization for combating mode collapse in gans," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 6382–6390.

[18] Z. Wu, Z. Wang, Y. Yuan, J. Zhang, Z. Wang, and H. Jin, "Black-box diagnosis and calibration on gan intra-mode collapse: A pilot study," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 17, no. 3s, pp. 1–18, 2021.

[19] M. M. Saad, M. H. Rehmani, and R. O'Reilly, "Addressing the intra-class mode collapse problem using adaptive input image normalization in gan-based x-ray images," in *2022 44th annual international conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*. IEEE, 2022, pp. 2049–2052.

[20] F. L. Barsha and W. Eberle, "Mode collapse detection strategies in generative adversarial networks for credit card fraud detection," in *The International FLAIRS Conference Proceedings*, vol. 37, 2024.

[21] B. VU, "Fraud ecommerce," https://www.kaggle.com/datasets/vbinh002/fraud-ecommerce, january 2024, (Accessed on 10/31/2024).

[22] M. L. G. ULB, "Credit card fraud detection," https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud, 2024, (Accessed on 10/31/20244).

[23] A. Bousmina, M. Selmi, M. A. Ben Rhaiem, and I. R. Farah, "A hybrid approach based on gan and cnn-lstm for aerial activity recognition," *Remote Sensing*, vol. 15, no. 14, p. 3626, 2023.

[24] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *International conference on machine learning*. PMLR, 2017, pp. 214–223.