



TSTR for Financial Fraud: Learning to Detect Manipulation Without Real Data

Ahmed Mahrous*

King Abdullah University of Science and Technology
Thuwal, Saudi Arabia
ahmed.mahrous@kaust.edu.sa

Roberto Di Pietro

King Abdullah University of Science and Technology
Thuwal, Saudi Arabia
roberto.dipietro@kaust.edu.sa

Abstract

One of the main challenges in financial data analysis using supervised machine learning techniques is the scarcity of labeled real-world data. For instance, this issue is quite compelling in the context of detecting market manipulation, such as pump-and-dump schemes. In this paper, we address the cited issue by introducing the train-on-synthetic, test-on-real (TSTR) paradigm. We generate synthetic pump-and-dump time series using Gaussian Sampling, Generative Adversarial Networks (GANs), Variational Autoencoders (VAEs), and Diffusion Models, and train a Rocket classifier exclusively on this data. Our results show that training on synthetic data alone achieves a detection accuracy of 79.75%, and adding just 20% real data raises it to 88.1%—close to the performance achieved by training on real data alone. Notably, Gaussian sampling outperforms all deep-learning generators, challenging the assumption that model complexity correlates with utility. Furthermore, we demonstrate that TSTR exposes a mismatch between conventional similarity metrics and the practical utility of synthetic data. The proposed framework is novel and, as shown by our extensive experimental campaign, viable. Finally, it is applicable to a broad class of data-scarce financial anomaly detection problems.

CCS Concepts

• **Applied computing** → *Economics*; • **Mathematics of computing** → *time series analysis*; • **Computing methodologies** → *Supervised learning by classification*; Neural networks; *Anomaly detection*.

Keywords

market manipulation, financial time series, synthetic data, generative artificial intelligence, anomaly detection

ACM Reference Format:

Ahmed Mahrous and Roberto Di Pietro. 2025. TSTR for Financial Fraud: Learning to Detect Manipulation Without Real Data. In *6th ACM International Conference on AI in Finance (ICAIF '25)*, November 15–18, 2025, Singapore, Singapore. ACM, New York, NY, USA, 9 pages. <https://doi.org/10.1145/3768292.3770393>

*Corresponding author.



This work is licensed under a Creative Commons Attribution 4.0 International License. ICAIF '25, Singapore, Singapore
© 2025 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-2220-2/25/11
<https://doi.org/10.1145/3768292.3770393>

1 Introduction

Market manipulation, such as pump-and-dump schemes, poses a significant threat to financial market integrity. Pump-and-dump schemes involve coordinated buying actions designed to artificially inflate prices, followed by rapid selling to secure profits, causing substantial financial losses to uninformed investors [31]. Such schemes are particularly prevalent in cryptocurrency markets due to weaker regulation.

Detecting market manipulations remains challenging, partially due to the scarcity of labeled real-world data instances [16, 28]. Market manipulations are rarely publicly documented or labeled, thus creating difficulties in building and validating detection models. Synthetic data generation provides a promising tool to address this challenge.

Recently, various synthetic generation methods, including Generative Adversarial Networks (GANs) and Variational Autoencoders (VAEs), have shown potential in replicating financial time series data [4, 5]. However, the extent to which they can successfully generate useful anomalous financial events, specifically pump-and-dump manipulations, remains underexplored. Furthermore, evaluating the quality of synthetic financial time series remains an open research problem [10, 27]. Traditional metrics, such as distance-based and distributional metrics, typically assess statistical similarity but could be misleading [27]. This paper addresses these critical gaps by evaluating the utility of synthetic financial time series data in detecting real-world pump-and-dump schemes. Specifically, we generate synthetic time series using Gaussian Sampling, GANs, Variational Autoencoders (VAEs), and Diffusion Models. Additionally, we introduce and empirically validate a classifier, trained using 466 real cryptocurrency pump-and-dump cases, designed to detect such schemes. To evaluate and compare the generation quality of the different models, we apply the train-on-synthetic, test-on-real (TSTR) methodology. To the best of our knowledge, this paper is the first to apply this methodology in the context of market manipulation detection. Our results demonstrate that training on synthetic data alone achieves strong detection performance on real manipulation events.

Specifically, this study explores the following key research questions:

- RQ1 Can synthetic time series data effectively train classifiers for detecting real-world pump-and-dump schemes?
- RQ2 How do simpler synthetic data generation methods, such as Gaussian Sampling, compare to deep learning-based models in generating realistic pump-and-dump time series?
- RQ3 How does a hybrid approach (mixing synthetic and real data) impact classifier performance for pump-and-dump detection?

Contribution

While addressing the above-mentioned research questions, this paper contributes the following:

- (1) We demonstrate the value of generating synthetic data to train market manipulation detection algorithms. This approach can be useful for training detection algorithms of various market manipulation activities, particularly when labeled data can be even more scarce than cryptocurrency pump-and-dump schemes.
- (2) We are the first, to our knowledge, to apply the “train-on-synthetic, test-on-real” (TSTR) methodology to assess the quality of synthetic time series data in financial anomaly detection. Our findings using TSTR reveal a key disconnect between traditional similarity metrics and the practical utility of synthetic data. While the GAN model produced time series that are more statistically similar to real pump-and-dump events (based on distance, correlation, and distributional statistics), their effectiveness in training the detection model is significantly lower compared to Gaussian Sampling, which had worse conventional statistical similarity scores but yielded the highest classification accuracy. This highlights the limitations of conventional similarity measures in evaluating synthetic financial time series and underscores the value of TSTR as a more meaningful framework for assessing the real-world applicability of synthetic data.
- (3) We are the first, to the best of our knowledge, to model the type of financial time series analyzed in this paper: crypto pump-and-dump manipulations. We demonstrate that financial anomalous time series can be effectively modeled not only with deep learning techniques but also with Gaussian-based methods. Notably, Gaussian sampling offers a simple, interpretable, and surprisingly effective approach.

Roadmap

The remainder of this paper is structured as follows: Section 2 reviews related literature on market manipulation detection and synthetic financial time series generation, highlighting existing methodologies and evaluation challenges. Section 3 details our dataset, the methods used for generating synthetic time series (Gaussian Sampling, GAN, VAE, Diffusion models), and the Rocket classifier utilized for evaluating detection performance. Section 4 presents experimental results, comparing traditional similarity metrics with the effectiveness of synthetic data using the TSTR evaluation framework. It includes an analysis of similarity metrics, an assessment of classifier accuracy trained on real versus synthetic data, and an evaluation of classifier performance across varying ratios of synthetic and real training data. Section 5 discusses the implications, limitations, and potential future research directions. Finally, Section 6 wraps up findings and concluding remarks.

2 Background and Related Work

This section provides an overview of existing research on the detection of market manipulation, synthetic financial time series generation methods, and challenges in evaluating the quality of synthetic time series, which we propose could potentially be addressed by the TSTR methodology.

2.1 Market Manipulation Detection

Market manipulation encompasses a range of practices aimed at artificially influencing market prices or trading volumes to deceive investors or profit unfairly [16]. Such practices broadly fall into categories including price-based, volume-based, and information-based manipulation. Price-based manipulation directly affects asset prices through actions such as spoofing, layering, and pump-and-dump schemes [25]. Volume-based manipulation includes techniques like wash trading, designed to create misleading signals about market liquidity and trading activity [2]. Information-based manipulation spreads false or misleading information to manipulate market sentiment and influence investor decisions [25].

Among these, pump-and-dump schemes have emerged as pervasive and problematic, particularly within cryptocurrency markets. Pump-and-dump manipulation involves coordinated buying actions to artificially inflate the price of an asset, followed by rapid selling to secure profits at the expense of unsuspecting investors [9, 31]. Due to their pseudonymity, weak regulatory frameworks, and appeal to speculative retail investors, cryptocurrency markets have become fertile ground for such schemes, leading to significant economic consequences [9]. These schemes often leverage social media platforms to orchestrate and amplify their impact, affecting millions of investors globally and at times moving millions of dollars in a matter of minutes [9, 14]. Studies have documented significant damage, including rapid price surges of up to nineteen-fold within minutes, and financial losses of millions of dollars over a few months [6, 9].

Given the scale and negative impacts of market manipulation in general, and pump-and-dump schemes in particular, detection has become a topic of interest in financial research. However, progress in detection methodologies has been significantly hindered by a scarcity of publicly available labeled data [16, 28]. Market manipulation events are often covert, with limited public disclosure, complicating efforts to train and evaluate machine learning models.

Initial efforts at pump-and-dump detection employed rule-based detection systems designed to identify abrupt price and volume anomalies indicative of manipulation [15]. While straightforward to implement, such systems frequently exhibit high false-positive rates, particularly given cryptocurrency markets’ inherent volatility [15]. To enhance detection precision, unsupervised learning techniques have emerged, such as Long Short-Term Memory (LSTM) autoencoders, capable of learning normal market behaviors and identifying deviations indicative of manipulation events [3]. These approaches show promising performance but rely heavily on expert knowledge and are difficult to evaluate objectively.

To address these limitations, supervised machine learning models trained on labeled data from publicly announced pump-and-dump events have gained popularity. For example, Victor and Hagemann [29] utilized XGBoost classifiers trained on 149 confirmed pump events sourced from Telegram groups. Similarly, Xu and Livshits [31] deployed Random Forest models trained on 412 pump events collected from multiple Telegram channels. Moreover, some methodologies have incorporated data from Twitter, Reddit, and GitHub, as signals for market manipulation [21]. Nevertheless, supervised-learning approaches remain constrained by their reliance on publicly disclosed manipulations, limiting their generalization capabilities.

Notably, many forms of market manipulation, such as insider trading, spoofing, and wash trading, rarely yield publicly labeled datasets, making detection inherently more challenging [16]. Synthetic financial time series generation methods present promising solutions to address these data limitations. Synthetic data generation can simulate diverse manipulation patterns, providing scalable labeled datasets essential for training supervised machine learning models [4, 5]. By doing so, they can enable training and evaluation without extensive real-world data.

2.2 Generating Synthetic Time Series in Finance

Addressing the scarcity of labeled market manipulation data, the generation of synthetic financial time series has emerged as a significant research area. For that, various methodologies have been proposed. Generative Adversarial Networks have been extensively explored for financial data generation. Wiese et al. [30] introduced Quant GANs, employing Temporal Convolutional Networks (TCNs) to effectively model multivariate time series. Quant GANs are designed to capture long-range dependencies and volatility clustering that are common in financial markets. Their evaluation utilized distance metrics such as Earth Mover Distance, assessing distributional similarity as well as the Diebold–Yilmaz spillover index, Autocorrelation Functions (ACF), and Leverage Effect scores to measure temporal dependency accuracy. Nonetheless, the authors acknowledge challenges in developing metrics capable of fully capturing synthetic data quality. Building on GAN architectures, Samuel et al. [24] proposed the Wasserstein GAN with Gradient Penalty (WGAN-GP), integrating Long Short-Term Memory (LSTM) networks to capture temporal dependencies, and testing on Bitcoin price data. Despite visually convincing generated series, statistical evaluations—using skewness, kurtosis, QQ plots, and autocorrelation—highlighted discrepancies from real data, highlighting ongoing challenges in generating and evaluating synthetic financial time series.

Ni et al. [22] presented the SigWGAN model that integrates signature-based methods with Wasserstein GANs to efficiently generate high-fidelity time series data, evaluating the generation quality using metrics such as their newly proposed Signature-W1 metric, marginal distribution metric, and correlation metric to assess the fidelity of the generated time series in capturing the statistical properties and dependencies of the original data. Fu et al. [13] explored the application of Conditional Generative Adversarial Networks (CGANs) for simulating time series data. To evaluate the generation of time series, they statistically compare the distributions of the generated data with the actual data. This involves analyzing the distribution properties (such as mean, variance, skewness, and kurtosis), QQ-plots, and autocorrelation. They also evaluate the generated time series performance in predicting stock prices. The complexity and inconsistency of the metrics' results underscore the difficulty of assessing financial time series generation quality.

Researchers have also experimented with generative models other than GANs. Buehler et al. [5] introduced a market simulation model that uses variational autoencoders and signatures of financial time series to generate realistic market scenarios, particularly effective in environments with limited data availability. The authors discuss limitations of traditional similarity metrics such as distributional metrics, divergences, two-sample test, etc. Forestier et al. [12]

introduce new data augmentation techniques for time series classification, which leverage Dynamic Time Warping (DTW) to create synthetic examples from a set of original time series. The main innovation is the use of a weighted version of DTW Barycentric Averaging (DBA) to calculate weights that determine how much each real time series will contribute to the creation of synthetic ones. They demonstrate that their time series generation method increases classifier accuracy through data augmentation. The time series used in this paper were non-financial. Moreover, they did not test classifier accuracy on real data when trained *purely* on generated data. Coletta et al. [7] propose a framework based on guided diffusion models and constrained optimization to generate synthetic time series that adhere to user-specified constraints while maintaining realism. The evaluation metrics for synthetic time series generation models assess realism through discriminative (calculated by a neural network) and t-SNE scores, utility via predictive performance, adherence to constraints through percentage error distance, and satisfaction rate.

2.3 Measurement of Time Series Generation Quality

While progress has been made in time series generation, the evaluation of generation quality remains a complex and open challenge. Several papers highlight this challenge ([4, 5, 10, 24, 27, 30]). Dogariu et al. [10] state that when it comes to the generation of financial time series, they “stress the need for finding a metric or validation framework that can harness both objective and subjective properties under a single quantifiable value”. Stenger et al. [27] say that “evaluating synthetic time series is a complex task and still considered an open problem”. Brophy et al. [4] state that “contrary to evaluating image-based GANs, it is difficult to evaluate time series data from human psycho-perceptual sense qualitatively”.

One approach to evaluating the usefulness of synthetic data in training predictive models is the train-on-synthetic, test-on-real (TSTR) methodology. First formalized in 2017 by Esteban et al. [11], this approach involves training a model exclusively on synthetic data and subsequently testing its performance on real-world data. It can serve as a measure of how well the synthetic dataset captures meaningful patterns and structures present in real data. While still a relatively recent and understudied approach, TSTR has been applied by a few papers in different domains [11, 17, 19, 26]. Esteban et al. [11] applied the train on synthetic, test on real (TSTR) evaluation to assess the utility of time series data generated by an LSTM-based GAN. Through serialized MNIST digit classification and ICU patient health indicator prediction, they found that a random forest classifier trained on synthetic data could generalize to real test data. However, performance varied by task (e.g., SpO₂ vs. heart rate prediction), with some tasks experiencing significant accuracy degradation when trained on synthetic data. Leznik et al. [17] propose another LSTM-based GAN model for generating synthetic multivariate time series data related to cloud computing workloads. Their experiments show that an LSTM forecasting model trained on synthetic data performed comparably or slightly better than one trained on real data. This advantage may arise from the greater diversity and volume of synthetic data, which helps mitigate overfitting. Extending this approach to broader time series

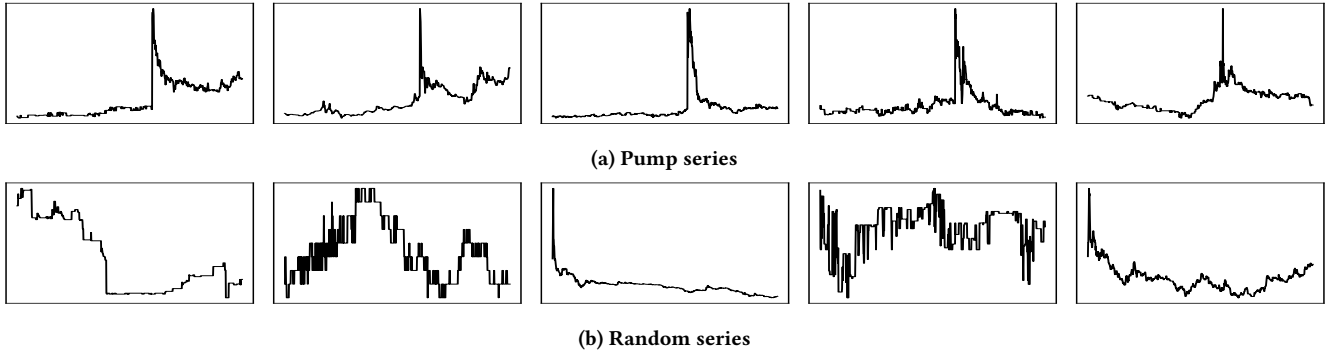


Figure 1: Comparison of pump and random series.

applications, Smith and Smith [26] introduced time series GAN (TSGAN), a model designed to generate synthetic one-dimensional time series data, and evaluated it using TSTR on 70 datasets from the University of California Riverside (UCR) time series data collection. Similar to Esteban et al. [11], the quality of generated data widely varies depending on the dataset and classification task. In the context of finance, the TSTR methodology has mainly been used by a few papers to train reinforcement learning agents to trade in real markets after being trained on synthetic data [19, 20].

3 Data and Methods

This section describes the dataset of cryptocurrency pump-and-dump events used in this study, the synthetic data generation methods applied, and the Rocket classifier implemented to evaluate the synthetic data’s utility.

3.1 Data

The data that will be used to test generation quality is composed of 466 time series that correspond to instances of manipulations that occur through coordinated buying and selling on the social media platform Telegram. Typically, group administrators announce that there will be a manipulation at a specified future time, keeping the identity of the target coin hidden. When the pre-specified time arrives, the target coin is announced, creating an artificial buying spree by group members. Often, the group also spreads misinformation and hype around the manipulated coin on different social media platforms, attempting to lure unaware outsiders into buying the coin, sending the coin price even higher. Shortly after, the manipulating group announces that the manipulation is over, and group members start selling the coin rapidly, resulting in sharp price drops—leaving unwitting buyers with significant losses. This type of coordinated activity is a recurrent one in cryptocurrency pump-and-dump schemes [6, 9, 18] and is also the one we will focus on in this paper.

The data we will rely on has been collected by mining Telegram posts to extract the time of pump announcements, target coin, and target exchange. Telegram messages were manually reviewed and annotated to detect the pump announcements. All the manipulations identified in this study have targeted the Binance exchange from July 2017 to January 2022. Binance was selected due to the accessibility and reliability of its historical minute-level price

data. For each manipulation event, we retrieved per-minute closing prices from the Binance API, covering a window spanning from 600 minutes before to 400 minutes after the moment of the pump announcement.

For analysis, each pump event is aligned so the spike-and-crash occurs at a fixed reference position (the 600th timestep); this alignment is realistic in practice because a rolling detection window will eventually position a pump at that reference point, enabling the same localized pattern to be observed.

3.2 Methods

3.2.1 Gaussian Sampling. With this method, we generate synthetic time series by sampling price changes from a series of Gaussian distributions. For each of the 1000 timesteps, we calculate the mean and standard deviation of price changes from our set of 466 real pump-and-dump time series. This process creates a unique Gaussian distribution for each timestep, ensuring that the synthetic data mirrors the statistical properties (central tendency and variability) of the real data at every point in time. Starting from an initial price, we iteratively apply a sampled price change from the corresponding timestep’s Gaussian distribution to generate a full 1000-point time series.

3.2.2 GAN. The Generative Adversarial Network (GAN) model used in this study consists of a simple feedforward architecture. The generator maps a 100-dimensional noise vector to a 1000-dimensional output using fully connected layers with ReLU activations. A *Tanh* activation normalizes the output. The discriminator receives either real or generated time series, and its task is to learn how to accurately differentiate between them. It processes data through fully connected layers with ReLU activations. The process concludes with a sigmoid output that classifies the input as real or fake. Both the generator and the discriminator are trained using binary cross-entropy loss and optimized with separate Adam optimizers over 10,000 training epochs.

3.2.3 VAE. The Variational Autoencoder (VAE) used in this study consists of a standard encoder–decoder architecture. The encoder maps the 1000-dimensional input time series (representing 1000 timesteps) into a compressed 256-representation using a fully connected layer. This step is followed by a ReLU activation function and a dropout layer, keeping the dimensionality unchanged. The

encoder then produces two 64-dimensional vectors representing the mean and log-variance of the latent distribution. These vectors are used to define a Gaussian distribution from which we can sample. The sampled latent code is then processed by the decoder, which essentially reverses the encoder’s steps, reconstructing the original 1000-dimensional time series. The model uses the Adam optimizer with gradient clipping for stability. The loss function combines mean squared error (to penalize the reconstruction error) and Kullback–Leibler (KL) divergence (to regularize the latent space towards a unit Gaussian).

3.2.4 Diffusion. We employ a diffusion-based generative model based on denoising diffusion probabilistic models (DDPMs). The implementation follows an open-source library¹, which uses a 1D U-Net architecture with Gaussian noise scheduling. This approach is well-suited for our task due to the one-dimensional structure of the time series. The model is trained on pump-and-dump price sequences and learns to iteratively denoise random noise into realistic time series through a reverse diffusion process.

3.2.5 Classifier. The downstream classification task involves distinguishing between two classes of time series: pump (label 1) and non-pump (label 0); see Figure 1. The non-pump time series are sampled from the same cryptocurrencies that were manipulated, but from randomly chosen intervals unrelated to any pump event. All pump time series are aligned such that the pump occurs at the timestep 600. It is highly unlikely that the non-pump time series would have a pump at that exact timestep, making them reasonable negative examples. To validate this assumption, 50 randomly selected non-pump samples were manually reviewed, and none appeared to resemble a pump-and-dump pattern.

The sampling process maintains a one-to-one correspondence between pump and non-pump time series for each cryptocurrency. This ensures that the classifier learns to identify manipulation patterns rather than asset-specific characteristics. In other words, negative examples are drawn from the same assets as the positive examples, which prevents the classifier from relying on confounding cues such as differences in price behavior between Bitcoin and the typically smaller altcoins targeted in pump-and-dump schemes. It is worth noting that this detail has been overlooked in some state-of-the-art pump-and-dump related contributions (e.g., [3]).

We use the ROCKET classifier [8], which applies a large number of random, untrained convolutional kernels to transform each time series into a fixed set of features. These features are then passed to a linear classifier for binary classification. ROCKET consistently ranks at or near state-of-the-art on diverse time series classification benchmarks with minimal tuning and very low training cost because only the final linear layer is learned; the random convolutional transforms remain fixed, which also reduces the risk of overfitting [8, 23].

Although some of the fixed random kernels are uninformative, many act as localized detectors that respond to the aligned pump-and-crash pattern: those covering the spike-and-crash region fire strongly on pump series and weakly on non-pumps, making the

Table 1: Aggregated moments of time series samples. Showing mean, variance, skewness, and kurtosis for real pump, real non-pump, and generated pump samples. Values are means (standard deviations in brackets).

Data Category	Mean	Variance	Skewness	Kurtosis
Real Pump	0.20 (0.10)	0.02 (0.02)	2.30 (3.27)	24.03 (76.83)
Real Non-Pump	0.41 (0.13)	0.05 (0.02)	0.40 (1.00)	1.22 (7.45)
GAN Generated Pump	0.15 (0.07)	0.02 (0.02)	2.46 (2.01)	17.55 (32.38)
VAE Generated Pump	0.18 (0.07)	0.02 (0.02)	1.76 (1.63)	10.83 (18.74)
Gaussian-Sampling Generated Pump	0.38 (0.14)	0.10 (0.06)	0.35 (0.99)	0.76 (12.04)
Diffusion Generated Pump	0.17 (0.04)	0.06 (0.02)	1.26 (0.30)	0.37 (0.92)

manipulation signal effectively linearly separable. With few hyperparameters and low need for labeled data to learn complex representations, ROCKET is a natural fit for low-data settings.

4 Results

This section presents the empirical results, including the evaluation of synthetic data quality using similarity metrics, and classifier performance comparisons between different proportions of synthetic and real training data.

4.1 Time-Series Statistical Similarity Analysis

We evaluate the effectiveness of commonly used time series similarity metrics in two settings. First, the metrics are used to compare real positive (pump) versus real negative (non-pump) samples. Second, we assess how closely the pump time series generated by each synthetic method resembles the real pump time series.

Table 1 reports the first four moments—mean, variance, skewness, and kurtosis—of each dataset. These moments offer insight into the distributional properties of the time series.

The following considerations can be drawn:

- Real pump time series exhibit high skewness and kurtosis compared to the real non-pump time series, reflecting sharp price spikes and heavy-tailed behavior typical of pump-and-dump events.
- GAN- and VAE-generated pump series provide a better match to real pump characteristics, approximating the statistical shape of real pumps more closely, especially in terms of skewness and kurtosis, though they still underestimate the extreme tails.
- Gaussian Sampling performs poorly in reproducing real pump statistics, with substantially lower kurtosis and higher variance.
- The diffusion model also underperforms, producing sequences with comparatively low kurtosis and skewness.

¹<https://github.com/lucidrains/denoising-diffusion-pytorch>

Table 2: Classification accuracy (mean and standard deviation over 10 repetitions) using real, synthetic, and hybrid training data. Evaluation is performed on 200 real time series (100 pump, 100 non-pump). The train dataset in the “real” experiment constitutes 366 real pump and 366 real non-pump time series. The train dataset in the “generated” experiment constitutes 366 generated pumps and the same 366 real non-pump time series. The train dataset in the “real + generated” experiment constitutes 183 generated pumps, 183 real pumps, and 366 real non-pump time series.

Model	Real	Generated	Real + Generated
Gaussian Sampling	90.3 (0.6)	81.4 (1.0)	90.4 (0.6)
GAN	90.3 (0.6)	61.8 (1.1)	90.0 (1.3)
VAE	90.3 (0.6)	79.8 (1.0)	90.2 (1.1)
Diffusion	90.3 (0.6)	50.85 (0.5)	90.0 (1.1)

Overall, moment-based comparisons suggest that the GAN and VAE models more closely approximate the distributional shape of real pump-and-dump time series than Gaussian or diffusion-based methods.

Table 3 reports the average values of various metrics comparing each dataset to the real pump time series. *Euclidean Distance* measures the straight-line distance between two points in multi-dimensional space. *Manhattan Distance* is the sum of the absolute differences between points across dimensions. *Earth Mover’s Distance* measures the minimal amount of work required to transform one distribution into another. For these distance-based metrics, lower values indicate greater similarity [1].

Cosine Similarity and *Pearson Correlation* measure alignment of vectors and linear correlation, respectively, with values closer to 1 indicating stronger similarity. The *Kolmogorov-Smirnov (KS) Statistic* quantifies the maximum difference between cumulative distributions; lower values indicate more similarity.

Real non-pump time series are markedly different from real pump time series across all six metrics. Distance-based metrics (Euclidean, Manhattan, Earth Mover’s) show large values, indicating high dissimilarity. Cosine similarity is moderate, suggesting that the general direction of changes has some resemblance. Pearson correlation is near zero or negative, showing no linear relationship. The KS statistic is high, reflecting large differences in distributions.

GAN- and VAE-generated pump time series show the highest similarity to real pump series across all metrics, particularly in distance-based metrics. They also achieve slightly higher values in Cosine Similarity and Pearson Correlation. In contrast, pump series generated via diffusion or Gaussian sampling are generally less similar to the real pump samples across all metrics. The similarity metric results in Table 3 align closely with the moment-based comparison in Table 1. These results reinforce the conclusion that GAN and VAE models better capture the distributional and structural characteristics of real pump-and-dump dynamics.

The above metrics provide a baseline understanding of how statistically similar each generated dataset is to the real time series. However, as shown in the next sections, statistical similarity does not always translate to utility in downstream learning tasks.

4.2 Classifier Accuracy with Real and Synthetic Data

Table 2 summarizes the accuracy results from experiments testing the utility of the time series generated by the different models in training a downstream task. In the ideal scenario where the synthetic time series generation matches the generation process of the real time series, training the downstream model on synthetic or real time series should make no difference when tested on the same real data. Therefore, it could be possible to use this difference to evaluate how close the important characteristics of synthetic data are to real data. If a model trained on synthetic data performs almost as well as one trained on real data, it means the synthetic time series captures the important features of the real data for that task. The smaller the accuracy gap, the higher the similarity between synthetic and real data.

We see that the generation methods providing the most utility are Gaussian sampling and VAE. It is noteworthy that Gaussian Sampling achieves the highest utility despite performing poorly on statistical similarity metrics, as shown in Section 4.1. This may be because the Gaussian sampling method introduces pronounced shifts in means and standard deviations around the pump phase—arguably the most critical feature for detecting pump events. The VAE also shows good utility, likely due to the presence of sharp peaks at the pump timestep and flatter behavior elsewhere, making the pump signal more distinguishable. In contrast, the GAN-generated series exhibit higher volatility outside the pump phase, which may reduce their effectiveness for training, as these fluctuations introduce noise that dilutes the pump signal. The diffusion model performs the worst, likely because it fails to consistently reproduce the critical spike around the 600th timestep.

Table 2 also shows that combining real and generated data does not significantly affect classifier performance compared to using real data alone. Across all generation methods, the accuracy remains stable when augmenting the real training set with synthetic data. On the one hand, this indicates that only a limited marginal benefit comes from such an augmentation in this setup. On the other hand, this is also an indication that synthetic data replicates some idiosyncratic features of real data, paving the way to use synthetic data when real data are scarce, as detailed here.

Table 4 reports the accuracy achieved when varying the proportion of synthetic to real pump samples using the VAE generator. As the ratio of synthetic to real pump time series decreases, the classifier’s accuracy increases, peaking at a ratio of 0.5 with a mean accuracy of 90.15%. This suggests that even a small number of real, documented financial anomaly cases can significantly enhance downstream model performance. Notably, even at a high ratio of synthetic data (ratio = 1.0), the classifier achieves reasonable accuracy, underscoring the potential utility of synthetic data when real data is unavailable—particularly when the underlying generation process, such as the Gaussian process used for pump-and-dump time series, is understood.

5 Discussion

This section interprets the key findings, discusses their implications, highlights limitations, and suggests potential directions for future research.

Table 3: Similarity metrics comparing each data category to real pump time series. Values reported are means (standard deviations in brackets).

Metrics with Real Pump	Euclidean Distance	Manhattan Distance	Cosine Similarity	Pearson Correlation	Earth Mover’s Distance	Kolmogorov Smirnov Statistic
Real Non-Pump	11.50 (3.57)	304.77 (108.56)	0.68 (0.16)	-0.03 (0.41)	0.24 (0.13)	0.60 (0.21)
GAN Generated Pump	6.34 (2.85)	159.54 (79.59)	0.72 (0.16)	0.35 (0.38)	0.12 (0.08)	0.45 (0.18)
VAE Generated Pump	6.18 (2.82)	158.52 (78.23)	0.76 (0.15)	0.36 (0.44)	0.11 (0.07)	0.44 (0.17)
Gaussian-Sampling Generated Pump	12.17 (5.49)	322.65 (161.51)	0.66 (0.23)	0.08 (0.60)	0.24 (0.13)	0.52 (0.15)
Diffusion Generated Pump	9.33 (2.26)	229.60 (70.12)	0.46 (0.16)	0.00 (0.29)	0.14 (0.06)	0.48 (0.11)

Table 4: Classification accuracy of the VAE model with varying ratios of synthetic to real pump time series in training. Accuracy is evaluated on 200 real time series (100 pump, 100 non-pump); reported as mean (standard deviation) over 10 repetitions.

Ratio	1.00	0.95	0.80	0.70	0.60	0.50
Mean Accuracy	79.75	84.60	88.10	90.05	89.75	90.15
(σ)	(1.05)	(0.86)	(0.70)	(0.65)	(1.23)	(0.81)

5.1 Results Discussion and Implications

Our study provides insights into the feasibility of training financial anomaly detection models using synthetic data. The results confirm that synthetic data, even when generated using relatively simple methods, can provide useful training material for classifiers designed to detect financial anomalies. This finding has significant implications for market surveillance, particularly in domains where real, labeled data is scarce.

One notable result is that training on synthetic data alone achieved a reasonable accuracy of 79.75% on real market manipulation events. This demonstrates that synthetic data can encode meaningful patterns that generalize to real-world anomalies. Among the generation methods, Gaussian Sampling—despite being the simplest—achieved the highest utility, outperforming more complex models like GANs, VAEs, and Diffusion in downstream performance.

Combining real and synthetic data yielded accuracy similar to using only real data (90.3% vs. 90.4% with Gaussian Sampling), suggesting that hybrid training strategies may help mitigate data scarcity without sacrificing performance. Notably, however, adding even a small number of real samples significantly improved accuracy, indicating that a few well-documented real cases can go a long way in training effective models.

One key takeaway is the disconnect between traditional statistical similarity metrics and the TSTR-based evaluation. While GAN-generated time series were more similar to real pump events in terms of similarity metrics (Table 1 and Table 3), they performed poorly in downstream classification (61.8% accuracy). Conversely, Gaussian Sampling scored poorly on similarity metrics yet yielded the highest classification accuracy in the TSTR setting (81.4%). This highlights that standard similarity metrics may not reliably predict the downstream effectiveness of synthetic financial time series, and

that task-specific evaluation frameworks like TSTR offer a more practical benchmark.

5.2 Limitations and Future Directions

While this study demonstrates the potential of synthetic time series for training financial anomaly detection models and highlights the practical value of the TSTR methodology in that context, certain limitations remain.

First, the performance of the deep generative models used—GANs, VAEs, and Diffusion Models—is sensitive to specific design choices, including architecture choices and hyperparameter configurations. Variations in these design choices can impact the quality of the generated time series. For instance, the diffusion model implemented in this paper failed to capture details about the pump-and-dump manipulation; changing the setup of this model improves its performance. In this work, we intentionally adopted relatively simple model setups to evaluate the TSTR framework and its implications, rather than to optimize generative quality. Future research could explore the performance of alternative model architectures, such as transformer-based or LSTM-based generative models (which have demonstrated effectiveness in time series generation [11]). Additionally, different architectures and configurations of GAN, VAE, or Diffusion models could be explored. This may enable the generation of higher-quality synthetic data with fewer real manipulation instances.

Second, while the TSTR methodology reduces the amount of real data required for training, it still relies on having a sufficient amount of real data for testing. Moreover, generating synthetic data that is useful for detection typically requires either domain knowledge or at least some real samples to guide the generation process. Nevertheless, the TSTR approach enables the development of detection models with a smaller amount of real data, making it valuable in data-scarce scenarios. Future research could explore the development of generation models that are not trained on any real data or test the minimum amount of real data needed for sufficient TSTR performance. This would further enhance the applicability of synthetic data approaches in scenarios where real-world labeled data is extremely scarce or unavailable.

Third, synthetic data may help control overfitting by diversifying the training distribution, yet we did not explicitly measure this effect. A fruitful future direction is to quantify how using synthetic data influences overfitting—for example, by monitoring changes in performance of datasets or models known to overfit.

Fourth, this study focuses exclusively on pump-and-dump manipulation, which is characterized by specific patterns. Other forms of financial manipulation may display more subtle or complex patterns, which could pose additional challenges for synthetic data generation and TSTR-based evaluation. However, pump-and-dump schemes represent a high-impact and well-documented manipulation type, serving as a starting point for studying synthetic time series and anomaly detection. Demonstrating strong results in this setting provides an important proof of concept. Future work can extend this methodology to other financial anomalies to assess the adaptability of synthetic data generation and the TSTR evaluation framework for financial time series.

Lastly, the TSTR methodology assumes the availability of a clearly defined downstream task, such as classification. This makes it well-suited for evaluating the utility of synthetic data when the goal is to improve task-specific model performance. However, synthetic data may also be generated for other purposes where downstream performance is not the main concern—such as exploratory data analysis, privacy-preserving data sharing, or simulation-based policy testing. In the mentioned cases, TSTR may not provide a meaningful evaluation signal. Future work could explore alternative evaluation methods tailored to these other goals. However, in contexts where a downstream task exists and performance can be measured, TSTR remains a practical framework for assessing the real-world utility of synthetic data.

6 Conclusion

This study introduces the train-on-synthetic, test-on-real (TSTR) methodology to the financial domain, focusing on market manipulation detection. We show that synthetic time series, including those generated by simple methods, can effectively train models to detect pump-and-dump schemes. For instance, combining real and synthetic data yielded accuracy similar to using only real data (90.3% vs. 90.4% with Gaussian Sampling). The results highlight the potential of synthetic financial time-series in scenarios where real labeled data is limited. We also demonstrated the value of the TSTR methodology in evaluating the quality of generated time series, offering insights beyond traditional similarity metrics. In particular, we report a counterintuitive finding: synthetic data generation methods that score highly on standard similarity metrics underperform on the assigned downstream detection task compared to methods with lower similarity scores. Finally, we have also highlighted several promising research directions.

Other than being interesting on its own, this work has the potential to contribute to the development of more robust detection systems for financial manipulation and encourages a broader use of synthetic data and TSTR-based evaluation in financial anomaly detection research.

Acknowledgments

The research reported in this publication was supported by funding from King Abdullah University of Science and Technology (KAUST), Center of Excellence for Generative AI, under award number 5940.

Artifact Availability

Code and data used to run the experiments in this paper are available at: <https://doi.org/10.5281/zenodo.17301104>.

References

- [1] Charu C Aggarwal et al. 2015. *Data mining: the textbook*. Vol. 1. Springer.
- [2] Franklin Allen and Douglas Gale. 1992. Stock-price manipulation. *The Review of Financial Studies* 5, 3 (1992), 503–529.
- [3] Ahmad Sani Bello, Jens Schneider, and Roberto Di Pietro. 2023. LLD: A low latency detection solution to thwart cryptocurrency pump & dumps. In *2023 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. IEEE, 1–9.
- [4] Eoin Brophy, Zhengwei Wang, Qi She, and Tomás Ward. 2023. Generative adversarial networks in time series: A systematic literature review. *Comput. Surveys* 55, 10 (2023), 1–31.
- [5] Hans Buehler, Blanka Horvath, Terry Lyons, Imanol Perez Arribas, and Ben Wood. 2020. A data-driven market simulator for small data environments. *arXiv preprint arXiv:2006.14498* (2020).
- [6] Lanouar Charfeddine and Ahmed Mahrous. 2024. What drives cryptocurrency pump and dump schemes: Coin versus market factors? *Finance Research Letters* 67 (2024), 105861.
- [7] Andrea Coletta, Sriram Gopalakrishnan, Daniel Borrajo, and Svitlana Vyetrenko. 2024. On the constrained time-series generation problem. *Advances in Neural Information Processing Systems* 36 (2024).
- [8] Angus Dempster, François Petitjean, and Geoffrey I Webb. 2020. ROCKET: exceptionally fast and accurate time series classification using random convolutional kernels. *Data Mining and Knowledge Discovery* 34, 5 (2020), 1454–1495.
- [9] Anirudh Dhawan and Tălis J Putniņš. 2023. A new wolf in town? Pump-and-dump manipulation in cryptocurrency markets. *Review of Finance* 27, 3 (2023), 935–975.
- [10] Mihai Dogariu, Liviu-Daniel Ștefan, Bogdan Andrei Boteanu, Claudiu Lamba, Boni Kim, and Bogdan Ionescu. 2022. Generation of realistic synthetic financial time-series. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)* 18, 4 (2022), 1–27.
- [11] Cristóbal Esteban, Stephanie L Hyland, and Gunnar Rätsch. 2017. Real-valued (medical) time series generation with recurrent conditional gans. *arXiv preprint arXiv:1706.02633* (2017).
- [12] Germain Forestier, François Petitjean, Hoang Anh Dau, Geoffrey I Webb, and Eamonn Keogh. 2017. Generating synthetic time series to augment sparse datasets. In *2017 IEEE international conference on data mining (ICDM)*. IEEE, 865–870.
- [13] Rao Fu, Jie Chen, Shutian Zeng, Yiping Zhuang, and Agus Sudjianto. 2019. Time series simulation by conditional generative adversarial net. *arXiv preprint arXiv:1904.11419* (2019).
- [14] Sihao Hu, Zhen Zhang, Shengliang Lu, Bingsheng He, and Zhao Li. 2023. Sequence-based target coin prediction for cryptocurrency pump-and-dump. *Proceedings of the ACM on Management of Data* 1, 1 (2023), 1–19.
- [15] Josh Kamps and Bennett Kleinberg. 2018. To the moon: defining and detecting cryptocurrency pump-and-dumps. *Crime Science* 7, 1 (2018), 1–18.
- [16] Samira Khodabandehlou and Seyyed Alireza Hashemi Golpayegani. 2022. Market manipulation detection: A systematic literature review. *Expert Systems with Applications* 210 (2022), 118330.
- [17] Mark Leznik, Patrick Michalsky, Peter Willis, Benjamin Schanzel, Per-Olov Östberg, and Jörg Domaschka. 2021. Multivariate time series synthesis using generative adversarial networks. In *Proceedings of the ACM/SPEC International Conference on Performance Engineering*. 43–50.
- [18] Tao Li, Donghua Shin, and Baolian Wang. 2018. Cryptocurrency pump-and-dump schemes. *Journal of Financial and Quantitative Analysis* (2018), 1–59.
- [19] Chunli Liu, Carmine Ventre, and Maria Polukarov. 2022. Synthetic data augmentation for deep reinforcement learning in financial trading. In *Proceedings of the third ACM international conference on AI in finance*. 343–351.
- [20] Chung I Lu and Julian Sester. 2024. Generative model for financial time series trained with MMD using a signature kernel. *arXiv preprint arXiv:2407.19848* (2024).
- [21] Huy Nghiem, Goran Muric, Fred Morstatter, and Emilio Ferrara. 2021. Detecting cryptocurrency pump-and-dump frauds using market and social signals. *Expert Systems with Applications* 182 (2021), 115284.
- [22] Hao Ni, Lukasz Szpruch, Marc Sabate-Vidales, Baoren Xiao, Magnus Wiese, and Shujian Liao. 2021. Sig-Wasserstein GANs for time series generation. In *Proceedings of the Second ACM International Conference on AI in Finance*. 1–8.
- [23] Ali Rahimi and Benjamin Recht. 2007. Random features for large-scale kernel machines. *Advances in neural information processing systems* 20 (2007).
- [24] Rikl Samuel, Bigler Daniel Nico, Pfenninger Moritz, and Osterrieder Joerg. 2021. Wasserstein GAN: Deep Generation applied on Bitcoins financial time series. *arXiv preprint arXiv:2107.06008* (2021).
- [25] Michael Siering, Benjamin Clapham, Oliver Engel, and Peter Gomber. 2017. A taxonomy of financial market manipulations: establishing trust and market integrity in the financialized economy through automated fraud detection. *Journal of Information Technology* 32, 3 (2017), 251–269.
- [26] Kaleb E Smith and Anthony O Smith. 2020. Conditional GAN for timeseries generation. *arXiv preprint arXiv:2006.16477* (2020).

- [27] Michael Stenger, Robert Leppich, Ian Foster, Samuel Kounev, and André Bauer. 2024. Evaluation is key: a survey on evaluation measures for synthetic time series. *Journal of Big Data* 11, 1 (2024), 1–56.
- [28] Shweta Tiwari, Heri Ramampiaro, and Helge Langseth. 2021. Machine learning in financial market surveillance: A survey. *Ieee Access* 9 (2021), 159734–159754.
- [29] Friedhelm Victor and Tanja Hagemann. 2019. Cryptocurrency pump and dump schemes: Quantification and detection. In *2019 International conference on data mining workshops (ICDMW)*. IEEE, 244–251.
- [30] Magnus Wiese, Robert Knobloch, Ralf Korn, and Peter Kretschmer. 2020. Quant GANs: deep generation of financial time series. *Quantitative Finance* 20, 9 (2020), 1419–1440.
- [31] Jiahua Xu and Benjamin Livshits. 2019. The anatomy of a cryptocurrency {Pump-and-Dump} scheme. In *28th USENIX Security Symposium (USENIX Security 19)*. 1609–1625.