Generating Synthetic Financial Time-Series Data Through Generative Adversatial Network

Methodologies And Comparison Among Different Deep Learning Structures

Grant Sawyer*[1], Harold Yuan†[1], Kaushik Tallam‡[1], Mario Nicolo' De Matteis§[1], and Tristan Roemer¶[1]

[1]Carnegie Mellon University, MSCF, New York, USA

February 2025

Abstract

This is a brief abstract of your paper, summarizing the key points and findings.

Keywords: Generative Adversatial Networks, Financial Time-Series, Deep Learning

1 Introduction

Hedge Funds, Proprietary Trading Firms, and other financial institutions rely on large amounts of data to make informed decisions. However, obtaining high-quality financial data can be challenging due to privacy concerns, data access restrictions, and the high cost of data acquisition. To address these challenges, researchers have developed generative models that can synthesize realistic financial time-series data. These models can be used to augment existing datasets, generate new data for backtesting trading strategies, and simulate market conditions for risk management. Over the last few years, many different developments have been conducted in the field of synthethic financial time-series data generation. In this paper, we provide a comprehensive overview of the methodologies and comparison among different deep learning structures used to generate synthetic financial time-series data. Our main focus is on the Generative Adversatial Networks (GANs) constructed over different neutral network structures. Eckerli and Osterrieder (2021) mentions that

One of the seminal contributions to the field of generative models is the 2014 paper by Ian Goodfellow et al., which introduced the concept of Generative Adversarial Networks (GANs). This groundbreaking work has profoundly influenced subsequent research by demonstrating that adversarial training can generate remarkably realistic data samples. Its impact is evident in numerous applications, including the synthesis of financial time series data, where GANs help address issues related to data scarcity, privacy concerns, and the high cost of data acquisition.

In the realm of synthetic financial data, several other aspects are equally important. For example, robust data augmentation techniques can enhance model reliability, and careful evaluation of synthetic data against real-world financial indicators is crucial for ensuring its practical utility. Moreover, addressing the temporal dependencies and non-linear dynamics inherent in financial markets requires advanced modeling techniques that go beyond the original GAN framework.

Building on these insights, our paper aims to explore and compare the effectiveness of various neural network architectures within the GAN framework. Specifically, we focus on:

Attention Layers: To capture and prioritize significant patterns in the data, enhancing the model's ability to focus on key temporal segments. Convolutional Neural Networks (CNNs): To detect localized patterns and extract spatial features from the financial time series. Long Short-Term Memory (LSTM)

*gsawyer@andrew.cmu.edu

†zhongfay@andrew.cmu.edu

‡ktallam@andrew.cmu.edu

§mdematte@andrew.cmu.edu

¶troemer@andrew.cmu.edu

Networks: To model long-range dependencies and effectively capture the sequential nature of financial data. Through a detailed comparative analysis, our study will assess the performance of these architectures in generating synthetic financial time series data. We aim to demonstrate how integrating Attention mechanisms can potentially improve the synthesis process by better capturing the intricate dynamics of financial markets.

2 Introduction

Hedge funds, proprietary trading firms, and other financial institutions rely heavily on vast amounts of data to drive informed decision-making. However, acquiring high-quality financial data presents significant challenges due to privacy concerns, access restrictions, and high acquisition costs. In response, researchers have developed generative models capable of synthesizing realistic financial time series data. Such synthetic data not only augments existing datasets but also provides valuable resources for backtesting trading strategies and simulating market conditions for risk management.

A seminal contribution to this field is the 2014 paper by Ian Goodfellow et al., which introduced the concept of Generative Adversarial Networks (GANs) **Goodfellow2014GAN**. This groundbreaking work demonstrated that adversarial training could produce remarkably realistic data samples, thereby influencing a wide array of applications, including synthetic financial data generation. As Eckerli and Osterrieder (2021) points out, the architecture of GANs plays a critical role in capturing the complex, non-linear dynamics inherent in financial markets.

In addition to the foundational aspects of GANs, several other factors are pivotal for the successful synthesis of financial data. Robust data augmentation methods and stringent evaluation against real-world financial indicators are essential to ensure the practical utility of the generated data. Moreover, capturing the temporal dependencies and intricate market dynamics requires advanced modeling techniques that extend beyond the original GAN framework.

Building on these insights, our paper aims to explore and compare the effectiveness of various neural network architectures within the GAN framework. In particular, we focus on:

- **Attention Layers**: To capture and prioritize significant temporal patterns, thereby enabling the model to focus on key segments of the time series.
- **Convolutional Neural Networks (CNNs)**: To detect localized patterns and extract spatial features from financial time series data.
- **Long Short-Term Memory (LSTM) Networks**: To model long-range dependencies and effectively capture the sequential characteristics of financial data.

Through a detailed comparative analysis, this study evaluates the performance of these architectures in generating synthetic financial time series data. In particular, we investigate how the integration of attention mechanisms can enhance the synthesis process by more accurately reflecting the intricate dynamics of financial markets.

3 Data and Variables of Interest

In this study, we focus on synthesizing financial time series data using a Generative Adversarial Network (GAN) framework. The selection of financial instruments is a critical component in ensuring that our model is both robust and generalizable. To this end, we have chosen three key instruments: JPMorgan Chase (JPM), Apple Inc. (AAPL), and the MSCI All Country World Index (ACWI). These selections enable us to assess our model's performance across both individual equities and a broad market index, which together represent a wide array of market dynamics.

Data for these financial instruments was sourced from the Wharton Research Data Services (WRDS) platform, which provides extensive access to high-quality daily historical data. For JPM and AAPL, the dataset spans from January 1, 2000, to January 31, 2024. This period covers multiple market cycles and includes significant economic events, thereby offering a rich context for capturing the complex dynamics inherent in financial markets. Such an extensive historical record is invaluable for training deep learning models, as it encompasses both periods of high volatility and relative stability.

In contrast, the historical data for the MSCI ACWI index is available from March 28, 2008 onward. Although this dataset has a shorter temporal coverage compared to the individual stock data, the ACWI index represents a diversified portfolio of global equities, including both developed and emerging markets.

This diversification is instrumental in evaluating the generalizability of the synthetic data generated by our model across different market conditions and geographic regions.

The integration of these datasets ensures a comprehensive evaluation of our GAN-based approach. In subsequent sections, we describe the preprocessing steps applied to the raw data, including cleaning, normalization, and the transformation procedures that were necessary to render the data suitable for model training. This careful curation and preparation of the dataset lay the groundwork for the subsequent analysis of model performance, particularly with respect to the incorporation of Attention layers, Convolutional Neural Networks, and Long Short-Term Memory (LSTM) networks.

4 Model

Generative Adversarial Networks (GANs) provide a framework to learn complex data distributions through a two-player minimax game involving two neural networks: a **generator** and a **discriminator**.

1. Setup and Notation

- **Real Data Distribution:** Let $p_{\text{data}}(x)$ denote the probability distribution of real data samples $x \in \mathcal{X}$.
- **Latent Space and Prior:** Define a latent space $\mathcal{Z}$ with a simple prior distribution $p_z(z)$ (e.g., a Gaussian or uniform distribution). A latent variable $z \sim p_z(z)$ is sampled and then transformed into the data space.
- **Generator:** The generator is a function $G : \mathcal{Z} \to \mathcal{X}$ parameterized by θ_G. It maps a latent variable z to a synthetic sample $G(z)$, thereby inducing an implicit distribution $p_g(x)$ over the data space.
- **Discriminator:** The discriminator is a function $D : \mathcal{X} \to [0, 1]$ parameterized by θ_D. It outputs a scalar representing the probability that a given sample x originates from the real data distribution $p_{\text{data}}(x)$ rather than from $p_g(x)$.

2. The Minimax Game

The GAN framework is formulated as a two-player minimax game with the value function

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

- **Discriminator's Objective:** For a fixed generator G, the discriminator D is trained to maximize the probability of correctly classifying real data and generated data:

$$\mathbb{E}_{x \sim p_{\text{data}}(x)}\big[\log D(x)\big] + \mathbb{E}_{x \sim p_g(x)}\big[\log(1 - D(x))\big].$$

- **Generator's Objective:** Simultaneously, the generator G is trained to minimize the same objective (i.e., to "fool" D) by generating samples $G(z)$ that maximize the discriminator's misclassification:

$$\min_G \mathbb{E}_{z \sim p_z(z)}\big[\log(1 - D(G(z)))\big].$$

3. Optimal Discriminator

For any fixed generator G, the optimal discriminator $D^*_G(x)$ can be derived by maximizing the value function pointwise. For each $x \in \mathcal{X}$, consider:

$$f(D(x)) = p_{\text{data}}(x) \log D(x) + p_g(x) \log(1 - D(x)).$$

Setting the derivative with respect to $D(x)$ to zero yields:

$$\frac{p_{\text{data}}(x)}{D(x)} - \frac{p_g(x)}{1 - D(x)} = 0 \quad \Longrightarrow \quad D^*_G(x) = \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_g(x)}. \tag{1}$$

4. Connection to Jensen-Shannon Divergence

Substituting the optimal discriminator D_G^* back into the value function, we obtain:

$$V(G, D_G^*) = -\log(4) + 2\,\mathrm{JSD}\big(p_{\mathrm{data}} \,\|\, p_g\big) \tag{2}$$

where JSD denotes the Jensen-Shannon Divergence. Since $\mathrm{JSD}(p_{\mathrm{data}} \,\|\, p_g) \geq 0$ with equality if and only if $p_g = p_{\mathrm{data}}$, minimizing $V(G, D_G^*)$ with respect to G forces the generator's distribution $p_g(x)$ to converge toward the real data distribution $p_{\mathrm{data}}(x)$.

5. Training Procedure

In practice, GAN training alternates between the following steps:

1. **Discriminator Update:** Maximize $V(D, G)$ with respect to θ_D while keeping θ_G fixed.
2. **Generator Update:** Minimize $V(D, G)$ (or a modified loss, such as maximizing $\log D(G(z))$ for stronger gradients) with respect to θ_G while keeping θ_D fixed.

These updates are typically performed using stochastic gradient descent or its variants.

Application to Financial Time-Series Data

In the context of financial time-series generation, the generator G is designed to produce synthetic financial sequences (e.g., asset prices, returns) that capture the underlying statistical properties of the real data. The discriminator D evaluates these sequences, providing a feedback loop that drives the generator to model the complex dependencies and temporal structures inherent in financial markets.

This mathematically rigorous formulation underpins the GAN framework and lays the foundation for generating high-fidelity synthetic financial time-series data.

5 Results and Evaluations

6 Model

Generative Adversarial Networks (GANs) provide a framework to learn complex data distributions through a two-player minimax game involving two neural networks: a **generator** and a **discriminator**. The classical formulation of GANs is outlined below, after which we describe our enhanced generator architectures that incorporate attention, convolutional, and long short-term memory layers.

1. Setup and Notation

- **Real Data Distribution:** Let $p_{\mathrm{data}}(x)$ denote the probability distribution of real data samples $x \in \mathcal{X}$.
- **Latent Space and Prior:** Define a latent space $\mathcal{Z}$ with a simple prior distribution $p_z(z)$ (e.g., a Gaussian or uniform distribution). A latent variable $z \sim p_z(z)$ is sampled and then transformed into the data space.
- **Generator:** The generator is a function $G : \mathcal{Z} \to \mathcal{X}$ parameterized by θ_G. It maps a latent variable z to a synthetic sample $G(z)$, thereby inducing an implicit distribution $p_g(x)$ over the data space.
- **Discriminator:** The discriminator is a function $D : \mathcal{X} \to [0, 1]$ parameterized by θ_D. It outputs a scalar representing the probability that a given sample x originates from the real data distribution $p_{\mathrm{data}}(x)$ rather than from $p_g(x)$.

2. The Minimax Game

The GAN framework is formulated as a two-player minimax game with the value function

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

- **Discriminator's Objective:** For a fixed generator G, the discriminator D is trained to maximize the probability of correctly classifying real data and generated data:

$$\mathbb{E}_{x \sim p_{\text{data}}(x)}\big[\log D(x)\big] + \mathbb{E}_{x \sim p_g(x)}\big[\log(1 - D(x))\big].$$

- **Generator's Objective:** Simultaneously, the generator G is trained to minimize the same objective (i.e., to "fool" D) by generating samples $G(z)$ that maximize the discriminator's misclassification:

$$\min_G \mathbb{E}_{z \sim p_z(z)}\big[\log(1 - D(G(z)))\big].$$

3. Optimal Discriminator and Jensen-Shannon Divergence

For any fixed generator G, the optimal discriminator $D_G^*(x)$ is obtained by maximizing the value function pointwise:

$$f(D(x)) = p_{\text{data}}(x) \log D(x) + p_g(x) \log(1 - D(x)).$$

Taking the derivative with respect to $D(x)$ and setting it to zero leads to:

$$\frac{p_{\text{data}}(x)}{D(x)} - \frac{p_g(x)}{1 - D(x)} = 0 \quad \Longrightarrow \quad D_G^*(x) = \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_g(x)}.$$

Substituting D_G^* back into the value function yields:

$$V(G, D_G^*) = -\log(4) + 2\,\mathrm{JSD}\big(p_{\text{data}} \,\|\, p_g\big),$$

where JSD denotes the Jensen-Shannon Divergence. Minimizing $V(G, D_G^*)$ with respect to G forces $p_g(x)$ to converge to $p_{\text{data}}(x)$.

4. Training Procedure

In practice, GAN training alternates between:

1. **Discriminator Update:** Maximize $V(D, G)$ with respect to θ_D, keeping θ_G fixed.
2. **Generator Update:** Minimize $V(D, G)$ (or a modified loss, such as maximizing $\log D(G(z))$ for stronger gradients) with respect to θ_G, while keeping θ_D fixed.

These updates are typically performed using stochastic gradient descent or its variants.

5. Enhanced Generator Architectures

While the classical GAN framework provides a robust foundation for generative modeling, financial time series data pose unique challenges due to their complex temporal dependencies and non-linear structures. To address these issues, we extend the generator architecture by incorporating advanced neural network components, as detailed below.

5.1 Attention Layers

Attention mechanisms enable the model to dynamically focus on different parts of the input sequence, assigning higher weights to temporally significant features. Inspired by the Transformer architecture, our attention layer computes a weighted sum of feature representations across time steps. This mechanism is crucial for capturing long-range dependencies and subtle patterns that might be missed by standard sequential models. The attention layer is integrated into the generator to selectively emphasize critical time steps during the generation process.

5.2 Convolutional Neural Networks (CNNs)

Convolutional layers are effective at extracting localized features and detecting patterns within data. In the context of financial time series, CNNs can capture short-term trends and volatility clusters by applying convolutional filters along the temporal dimension. These filters learn hierarchical representations, which are instrumental in modeling the non-stationary characteristics of asset prices. Previous studies have shown the efficacy of CNNs in time series forecasting and anomaly detection. In our generator, convolutional layers are employed to extract spatial-temporal features before the data is processed by other sequential layers.

5.3 Long Short-Term Memory (LSTM) Networks

LSTM networks are a natural choice for modeling sequential data due to their ability to capture long-term dependencies and mitigate issues such as vanishing gradients. The gating mechanisms in LSTMs allow the network to remember relevant information over extended periods, which is essential for accurately modeling the evolution of financial time series. LSTMs have been widely used in various forecasting tasks and have proven effective in capturing both short-term fluctuations and long-term trends . In our framework, LSTM layers are used to process the sequential output from convolutional layers or attention modules, further refining the temporal dynamics captured in the generated data.

5.4 Integration within the GAN Framework

In our proposed model, these enhanced architectural components are integrated within the generator network while the discriminator maintains a conventional architecture. This integration is designed to evaluate the impact of each component on the quality of the generated synthetic financial time series. The overall training procedure remains consistent with the classical GAN paradigm, with the generator updates now encompassing additional hyperparameters related to the attention heads, convolutional filter sizes, and LSTM layer configurations.

By comparing these variants, our study aims to elucidate the advantages and limitations of incorporating attention mechanisms, CNNs, and LSTMs in the context of synthetic financial data generation. The experimental evaluation (detailed in Section) provides insights into how these components improve the generator's ability to model the intricate, non-linear, and temporal structures present in real financial time series.

7 Conclusion

8 References

References

Eckerli, Florian and Joerg Osterrieder (2021). "Generative Adversarial Networks in finance: an overview". eng. In.