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 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) defined by the use of different neutral network structures. Eckerli and Osterrieder (2021) mentions that

2 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.

*gsawyer@andrew.cmu.edu
†zhongfay@andrew.cmu.edu
‡ktallam@andrew.cmu.edu
§mdematte@andrew.cmu.edu
¶troemer@andrew.cmu.edu

- **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:

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

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

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.

3 Data and Variable of Interest

4 Results and Evaluations

5 Conclusion

6 References

References

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