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

Generative Adversarial Networks (GANs) have emerged as a powerful tool for synthesizing realistic financial time-series data, addressing challenges such as data scarcity, privacy, and cost. This paper explores the integration of attention mechanisms, convolutional neural networks (CNNs), and long short-term memory (LSTM) layers within a GAN framework to generate synthetic financial price data. We evaluate the performance of these architectures using three financial instruments—JPMorgan Chase (JPM), Apple Inc. (AAPL), and the MSCI ACWI Index (ACWI)—spanning over two decades of market data. Our results demonstrate that CNN-enhanced GANs significantly improve the modeling of temporal dependencies and volatility clustering compared to Attention- and LSTM-based variants. The synthetic data exhibits statistical properties closely aligned with real-world financial time series, as validated by rigorous tests for distributional similarity and moments. These findings provide actionable insights for financial institutions seeking to augment datasets for backtesting, risk management, and machine learning applications.

Keywords: Generative Adversatial Networks, Financial Time-Series, Attention Layers

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. In particular, we aim to synthesize new price time series data that can be highly useful for various applications. These include backtesting trading strategies, training machine learning models, and conducting stress tests and scenario analyses. By generating realistic synthetic data, we can provide financial institutions with a valuable resource to enhance their decision-making processes, improve model robustness, and better prepare for adverse market conditions.

Given the great impact and power of generative AI models, we decided to exploit one of the main pillars of Generative AI: the Generative Adversarial Network develope by Goodfellow et al. (2014). GANs have been extensively used in the computer vision field, particularly in image generation. Iglesias, Talavera,

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

and Díaz-Álvarez (2023) show the ability to generate realistic data samples through adversarial training makes them a powerful tool for synthesizing financial time-series data as well. This groundbreaking model has profoundly influenced subsequent research by demonstrating that adversarial training can generate remarkably realistic data samples. Eckerli and Osterrieder (2021) mentions different applications of GANs, including market manipulation, credit card fraud, portfolio optimization and market prediction.

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. In simple terms, we aim to find out the distribution by which the prices have been generated. When looking for new data, among the most well-known techniques, we recall different and exotic sampling techniques that tend to be very complex and computationally expensive. As highlighted by Cont (2001), understanding the empirical properties of asset returns, such as heavy tails and volatility clustering, is essential for developing realistic financial models. These properties must be carefully considered when designing synthetic data generation frameworks to ensure that the generated data accurately reflects the behavior observed in real financial markets.

We designed the neural networks for the GAN models by implementing three different types of layers: Attention, CNN, and LSTM. We decided to implement these three different types of layers because each offers unique advantages for capturing the complex dynamics of financial time series data. In accordance with the literature, Attention layers allow the model to focus on significant temporal patterns, enhancing its ability to prioritize key segments of the data. Convolutional Neural Networks (CNNs) are effective at detecting localized patterns and extracting spatial features, which are crucial for modeling short-term trends and volatility clusters. Long Short-Term Memory (LSTM) networks excel at capturing long-range dependencies and the sequential nature of financial data, making them ideal for modeling the temporal evolution of asset prices. By integrating these diverse neural network components, we aim to improve the overall performance and robustness of our GAN models in generating realistic synthetic financial time series data.

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.

The project code, datasets, and additional resources are available at our GitHub repository: `https://github.com/mariodematteis/ml-ii-final-project`.

2 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 ACWI 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.

3 Model

3.1 Neural Network Architectures

In this section, we describe the neural network architectures employed in our GAN framework, focusing on the integration of Attention layers, Convolutional Neural Networks (CNNs), and Long Short-Term Memory (LSTM) networks. Each of these components offers unique advantages for capturing the complex dynamics of financial time series data.

3.1.1 Attention Layers

Attention mechanisms have revolutionized various fields, particularly natural language processing, by allowing models to dynamically focus on different parts of the input sequence Vaswani et al. (2023). In our generator, we incorporate a Self-Attention layer inspired by this work. The attention mechanism computes a weighted sum of feature representations across time steps, enabling the model to prioritize significant temporal patterns. This approach is particularly beneficial for capturing long-range dependencies and subtle patterns in financial data. In quantitative finance, attention-based models have garnered significant interest; for example, Qin et al. (2017) introduced a dual-stage attention-based recurrent neural network for time series prediction, while Chen and Ge (2019) leveraged attention mechanisms for enhanced stock price forecasting. Our implementation integrates a series of deconvolutional layers followed by a Self-Attention module, thereby enhancing the model's capability to generate realistic and coherent time series data.

3.1.2 Convolutional Neural Networks

CNNs are well-suited for detecting localized patterns and extracting spatial features from time series data. By applying convolutional filters along the temporal dimension, CNNs can capture short-term trends and volatility clusters, which are crucial for modeling the non-stationary characteristics of financial markets. Our generator architecture includes deconvolutional layers that progressively upsample the latent representation, allowing the model to generate detailed and realistic time series data. The effectiveness of CNNs in time series forecasting and anomaly detection has been demonstrated in various studies, including LeCun, Bengio, and Hinton (2015). Moreover, a growing body of literature has shown that CNNs excel in quantitative finance applications. For instance, Dixon (2017) illustrated their utility in market trend forecasting, while Liu and Wu (2023) and Zhang, Zohren, and Roberts (2019) demonstrated their capabilities in option pricing, risk management, volatility prediction, and anomaly detection in high-frequency trading data. These studies underscore the versatility and robustness of CNN architectures in capturing complex, localized patterns within financial datasets.

3.1.3 Long Short-Term Memory Networks

LSTM networks are designed to capture long-term dependencies in sequential data, making them ideal for modeling the temporal evolution of financial time series. The gating mechanisms in LSTMs allow the network to retain relevant information over extended periods, addressing issues such as vanishing gradients. Our LSTM-based generator includes fully connected layers to transform the latent space into

a suitable input for the LSTM layers, followed by an output layer that generates the final time series. The use of LSTMs in financial modeling has been extensively studied, as highlighted by Hochreiter and Schmidhuber (1997).

While LSTMs have historically been a cornerstone in Natural Language Processing (NLP) for tasks such as language modeling and machine translation (Fischer and Krauss (2018) and Sutskever, Vinyals, and Le (2014)), the emergence of attention-based mechanisms—introduced by Bahdanau, Cho, and Bengio (2016) and further advanced by Vaswani et al. (2023)—has largely supplanted them in that field. In contrast, within quantitative finance, LSTMs remain a robust tool. Their ability to model sequential dependencies has proven valuable in a range of applications, including market trend forecasting (Fischer and Krauss (2018)), risk management (Bao, Yue, and Rao (2017)), and volatility prediction, where capturing temporal nuances is critical.

By integrating these advanced neural network components, our GAN framework aims to improve the generation of synthetic financial time series data. The comparative analysis of these architectures will provide insights into their respective strengths and limitations, ultimately guiding the development of more robust and accurate generative models for financial applications.

3.2 Generative Adversarial Network

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:

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

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.

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**. 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 $p_{\text{data}}(x)$ 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 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:

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

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.

5 Results and Evaluations

By taking a look at the appendix, it is possible to observe that among the three different implementation, the one based on the convolutional performs better in terms of returns distribution. They look pretty normal and similar to the real ones. It is possible to analyze this result also from a more statitical perspective like the Jensen-Shannon divergence. The model based on the convolutional layers showed better results with respect to Apple and the MSCI ACWI Index. Differently, with respect to the JP Morgan stock, the results tends not to be very consistent with the reality, the distribution of returns obtained by sampling time-series from our model and the returns obtained on the entire financial time-series data of the stock don't look great. Returns tend to be more volatile compared to the ones coming from the historical data.

As we can see, the distributions for AAPL and ACWI actually fit very well to the historical distributions. Looking at the Jensen-Shannon Divergence (JSD, min. of 0 and max. of $ln(2) = 0.693$) we get a score in the order of 10^-1 which is very good. JPM on the otherhand is quite bad (both visually and numerically). We see that the generated distribution has considerably fatter tails, especially a fatter left tail which would explain why all the generated time series plotted above for JPM have a drastic negative spike at one location. This may have been caused by a large drawdown in the historical data that the model overemphasized during training. The LSTM model performs worst according to JSD and produces a much too narrow distribution, which may indicate that there was some sort of structural mistake in applying that architecture to our data. Perhaps after tuning the architecture for our data better and retraining we would achieve better results. The attention model manages to produce better distributions than the LSTM, however, they are also not very strongly aligned with historical data. Overall, both visually and numerically, the CNN GAN strongly outcompetes the other specific architectures used in this paper in generating synthetic time series.

6 Conclusion

In conclusion, our investigation into enhanced GAN architectures for synthesizing financial time series data demonstrates that incorporating convolutional layers yields superior performance in capturing the distributional characteristics of asset returns compared to LSTM and attention-based models. While the CNN-based model successfully replicates the historical return distributions for instruments such as AAPL and the MSCI ACWI Index, discrepancies observed for JPM highlight areas for further refinement. Future work should explore additional tuning of the architectures and consider hybrid approaches that combine the strengths of attention, CNNs, and LSTMs to achieve even closer alignment with real market data. These improvements could provide financial institutions with more robust synthetic datasets for applications in backtesting, risk management, and machine learning research.

Acknowledgments

We would like to thank CodeWilling for providing the computational resources that enabled us to train and evaluate our models effectively.

References

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio (May 19, 2016). *Neural Machine Translation by Jointly Learning to Align and Translate*. DOI: 10.48550/arXiv.1409.0473. arXiv: 1409.0473[cs]. URL: http://arxiv.org/abs/1409.0473 (visited on 02/25/2025).

Bao, Wei, Jun Yue, and Yulei Rao (July 14, 2017). "A deep learning framework for financial time series using stacked autoencoders and long-short term memory". In: *PLOS ONE* 12.7. Publisher: Public Library of Science, e0180944. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0180944. URL: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0180944 (visited on 02/25/2025).

Chen, Shun and Lei Ge (Sept. 2, 2019). "Exploring the attention mechanism in LSTM-based Hong Kong stock price movement prediction". In: *Quantitative Finance*. Publisher: Routledge. ISSN: 1469-7688. URL: https://www.tandfonline.com/doi/abs/10.1080/14697688.2019.1622287 (visited on 02/25/2025).

Cont, R. (2001). "Empirical properties of asset returns: stylized facts and statistical issues". In: *Quantitative Finance* 1.2. Publisher: Taylor & Francis Journals, pp. 223–236. ISSN: 1469-7688. URL: https://econpapers.repec.org/article/tafquantf/v_3a1_3ay_3a2001_3ai_3a2_3ap_3a223-236.htm (visited on 05/29/2024).

Dixon, Matthew F. (Dec. 5, 2017). *A High Frequency Trade Execution Model for Supervised Learning*. DOI: 10.48550/arXiv.1710.03870. arXiv: 1710.03870[q-fin]. URL: http://arxiv.org/abs/1710.03870 (visited on 02/25/2025).

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

Fischer, Thomas and Christopher Krauss (Oct. 16, 2018). "Deep learning with long short-term memory networks for financial market predictions". In: *European Journal of Operational Research* 270.2, pp. 654–669. ISSN: 0377-2217. DOI: 10.1016/j.ejor.2017.11.054. URL: https://www.sciencedirect.com/science/article/pii/S0377221717310652 (visited on 02/25/2025).

Goodfellow, Ian J. et al. (June 10, 2014). *Generative Adversarial Networks*. DOI: 10.48550/arXiv.1406.2661. arXiv: 1406.2661[stat]. URL: http://arxiv.org/abs/1406.2661 (visited on 02/25/2025).

Hochreiter, Sepp and Jürgen Schmidhuber (Nov. 15, 1997). "Long Short-Term Memory". In: *Neural Computation* 9.8, pp. 1735–1780. ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. URL: https://doi.org/10.1162/neco.1997.9.8.1735 (visited on 02/25/2025).

Iglesias, Guillermo, Edgar Talavera, and Alberto Díaz-Álvarez (May 2023). "A survey on GANs for computer vision: Recent research, analysis and taxonomy". In: *Computer Science Review* 48, p. 100553. ISSN: 15740137. DOI: 10.1016/j.cosrev.2023.100553. arXiv: 2203.11242[cs]. URL: http://arxiv.org/abs/2203.11242 (visited on 02/25/2025).

LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton (May 2015). "Deep learning". In: *Nature* 521.7553. Publisher: Nature Publishing Group, pp. 436–444. ISSN: 1476-4687. DOI: 10.1038/nature14539. URL: https://www.nature.com/articles/nature14539 (visited on 02/25/2025).

Liu, David and Yu Wu (Aug. 2023). "Option pricing using deep convolutional neural networks enhanced by technical indicators". In: *2023 IEEE 9th International Conference on Cloud Computing and Intelligent Systems (CCIS)*. 2023 IEEE 9th International Conference on Cloud Computing and Intelligent Systems (CCIS). ISSN: 2376-595X, pp. 143–147. DOI: 10.1109/CCIS59572.2023.10262865. URL: https://ieeexplore.ieee.org/document/10262865 (visited on 02/25/2025).

Qin, Yao et al. (Aug. 14, 2017). *A Dual-Stage Attention-Based Recurrent Neural Network for Time Series Prediction*. DOI: 10.48550/arXiv.1704.02971. arXiv: 1704.02971[cs]. URL: http://arxiv.org/abs/1704.02971 (visited on 02/25/2025).

Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le (Dec. 14, 2014). *Sequence to Sequence Learning with Neural Networks*. DOI: 10.48550/arXiv.1409.3215. arXiv: 1409.3215[cs]. URL: http://arxiv.org/abs/1409.3215 (visited on 02/25/2025).

Vaswani, Ashish et al. (Aug. 2, 2023). *Attention Is All You Need*. DOI: 10.48550/arXiv.1706.03762. arXiv: 1706.03762[cs]. URL: http://arxiv.org/abs/1706.03762 (visited on 02/25/2025).

Zhang, Zihao, Stefan Zohren, and Stephen Roberts (June 1, 2019). "DeepLOB: Deep Convolutional Neural Networks for Limit Order Books". In: *IEEE Transactions on Signal Processing* 67.11, pp. 3001–3012. ISSN: 1053-587X, 1941-0476. DOI: 10.1109/TSP.2019.2907260. arXiv: 1808.03668[q-fin]. URL: http://arxiv.org/abs/1808.03668 (visited on 02/25/2025).

A Appendix

Metric	Original	Generated
Mean	...	...
Std	...	...
Skewness	...	...
Kurtosis	...	...
Ljung-Box t-statistic	...	...
Ljung-Box p-value	...	...

Table 1: AAPL Moments Comparison for the Attention-Based GAN. Jensen–Shannon Divergence: ...

Metric	Original	Generated
Mean	...	...
Std	...	...
Skewness	...	...
Kurtosis	...	...
Ljung-Box t-statistic	...	...
Ljung-Box p-value	...	...

Table 2: ACWI Moments Comparison for the Attention-Based GAN. Jensen–Shannon Divergence: ...

Metric	Original	Generated
Mean	...	...
Std	...	...
Skewness	...	...
Kurtosis	...	...
Ljung-Box t-statistic	...	...
Ljung-Box p-value	...	...

Table 3: JPM Moments Comparison for the Attention-Based GAN. Jensen–Shannon Divergence: ...

Figure 1: Comparison of Real vs. Generated AAPL, ACWI and JPM Returns using Attention-Based GAN.

Metric	Original	Generated
Mean	...	...
Std	...	...
Skewness	...	...
Kurtosis	...	...
Ljung-Box t-statistic	...	...
Ljung-Box p-value	...	...

Table 4: AAPL Moments Comparison for the CNN-Based GAN. Jensen–Shannon Divergence: ...

Metric	Original	Generated
Mean	...	...
Std	...	...
Skewness	...	...
Kurtosis	...	...
Ljung-Box t-statistic	...	...
Ljung-Box p-value	...	...

Table 5: ACWI Moments Comparison for the CNN-Based GAN. Jensen–Shannon Divergence: ...

Metric	Original	Generated
Mean	...	...
Std	...	...
Skewness	...	...
Kurtosis	...	...
Ljung-Box t-statistic	...	...
Ljung-Box p-value	...	...

Table 6: JPM Moments Comparison for the CNN-Based GAN. Jensen–Shannon Divergence: ...

Metric	Original	Generated
Mean	...	...
Std	...	...
Skewness	...	...
Kurtosis	...	...
Ljung-Box t-statistic	...	...
Ljung-Box p-value	...	...

Table 7: AAPL Moments Comparison for the LSTM-Based GAN. Jensen–Shannon Divergence: ...

Metric	Original	Generated
Mean	...	...
Std	...	...
Skewness	...	...
Kurtosis	...	...
Ljung-Box t-statistic	...	...
Ljung-Box p-value	...	...

Table 8: ACWI Moments Comparison for the LSTM-Based GAN. Jensen–Shannon Divergence: ...

Figure 2: Comparison of Real vs. Generated AAPL, ACWI and JPM Returns using CNN-Based GAN.

Figure 3: Comparison of Real vs. Generated AAPL, ACWI and JPM Returns using LSTM-Based GAN.

Metric	Original	Generated
Mean	...	...
Std	...	...
Skewness	...	...
Kurtosis	...	...
Ljung-Box t-statistic	...	...
Ljung-Box p-value	...	...

Table 9: JPM Moments Comparison for the LSTM-Based GAN. Jensen–Shannon Divergence: ...