New Jersey Institute of Technology
Digital Commons @ NJIT

Dissertations | Electronic Theses and Dissertations

8-31-2025

Robust AI solutions for financial markets through generative modeling, dynamic graph learning, and reinforcement-based portfolio optimization

Jingyi Gu
New Jersey Institute of Technology

Follow this and additional works at: https://digitalcommons.njit.edu/dissertations

 Part of the Artificial Intelligence and Robotics Commons, and the Finance Commons

Recommended Citation

Gu, Jingyi, "Robust AI solutions for financial markets through generative modeling, dynamic graph learning, and reinforcement-based portfolio optimization" (2025). *Dissertations*. 1852.
https://digitalcommons.njit.edu/dissertations/1852

This Dissertation is brought to you for free and open access by the Electronic Theses and Dissertations at Digital Commons @ NJIT. It has been accepted for inclusion in Dissertations by an authorized administrator of Digital Commons @ NJIT. For more information, please contact digitalcommons@njit.edu.

Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be "used for any purpose other than private study, scholarship, or research." If a, user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of "fair use" that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select "Pages from: first page # to: last page #" on the print dialog screen

NJIT New Jersey's Science & Technology University

The Van Houten library has removed some of the personal information and all signatures from the approval page and biographical sketches of theses and dissertations in order to protect the identity of NJIT graduates and faculty.

ABSTRACT

ROBUST AI SOLUTIONS FOR FINANCIAL MARKETS THROUGH GENERATIVE MODELING, DYNAMIC GRAPH LEARNING, AND REINFORCEMENT-BASED PORTFOLIO OPTIMIZATION

by
Jingyi Gu

Financial markets are inherently uncertain and dynamic, driven by complex factors such as macroeconomic signals, investor sentiment, and evolving inter-asset relationships. While machine learning has advanced financial modeling, existing approaches often fall short in addressing the real-world intricacies of finance. This dissertation confronts two critical challenges, human-driven stochasticity and risk-intensive decision-making under real-world trading constraints, while seizing a pivotal opportunity, the structural dynamics of evolving financial systems. These elements are foundational to advancing robust and practical financial intelligence.

To this end, this dissertation develops a unified framework for robust financial modeling and decision-making. The framework is architected as a progressive, modular pipeline that mirrors the real-world flow of financial intelligence, starting from macro-level index forecasting, advancing to individual asset pricing, and culminating in actionable trading decisions under real-world constraints. (1) At the macro level, this framework captures stochasticity driven by human behaviors. Exploratory studies investigate the weekend effect of news on asset prices and generative modeling for market behavior, providing foundational insights into stochastic market dynamics. Building on this, *RAGIC*, a novel hybrid framework that fuses generative models with statistical inference, is established to construct risk-aware intervals that quantify forecast uncertainty and risk exposure under volatility regimes. (2) At the asset level, this framework leverages structural dynamics in financial markets and develops *DySTAGE*, a dynamic graph learning framework that models assets as nodes in a time-varying market graph, accommodating changing asset composition and evolving

asset relationships. By incorporating spatio-temporal attention and graph encodings, *DySTAGE* enables structure-aware asset pricing and reveals hidden financial patterns. (3) Lastly, to support actionable trading under realistic constraints, the framework formulates *Margin Trader*, the first public reinforcement learning framework that incorporates realistic margin requirements and leverage rules to learn long-short trading strategies. It is further extended with large language models to integrate external signals and guide explainable and adaptive portfolio reallocation decisions.

Together, these innovations deliver a robust, scalable, and deployable financial AI framework with broad impact: it empowers individual investors with accessible, open-source tools for uncertainty-aware forecasting; supports academic researchers with a testbed for hybrid AI-finance exploration; and offers institutional stakeholders with data-driven, risk-sensitive systems for compliant and adaptive portfolio management in high-stakes, fast-evolving markets.

ROBUST AI SOLUTIONS FOR FINANCIAL MARKETS THROUGH GENERATIVE MODELING, DYNAMIC GRAPH LEARNING, AND REINFORCEMENT-BASED PORTFOLIO OPTIMIZATION

by
Jingyi Gu

A Dissertation
Submitted to the Faculty of
New Jersey Institute of Technology
in Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Computer Science

Department of Computer Science

August 2025

Copyright © 2025 by Jingyi Gu
ALL RIGHTS RESERVED

APPROVAL PAGE

ROBUST AI SOLUTIONS FOR FINANCIAL MARKETS THROUGH GENERATIVE MODELING, DYNAMIC GRAPH LEARNING, AND REINFORCEMENT-BASED PORTFOLIO OPTIMIZATION

Jingyi Gu

Dr. Guiling Wang, Dissertation Advisor — Date
Distinguished Professor, Department of Computer Science, NJIT

Dr. Zhi Wei, Committee Member — Date
Distinguished Professor, Department of Computer Science, NJIT

Dr. Ioannis Koutis, Committee Member — Date
Associate Professor, Department of Computer Science, NJIT

Dr. Tucker Balch, Committee Member — Date
Professor, Practice and Research of Finance, Emory University, Atlanta, GA, USA

Dr. Zhaodong Zhong, Committee Member — Date
Associate Professor, Department of Finance and Economics
Rutgers University, Business School, Newark, NJ, USA

BIOGRAPHICAL SKETCH

Author: Jingyi Gu

Degree: Doctor of Philosophy

Date: August 2025

Undergraduate and Graduate Education:

- Doctor of Philosophy in Computer Science
  New Jersey Institute of Technology, Newark, NJ, 2025
- Master of Science in Data Science, Statistics Track
  Rutgers, the State University of New Jersey, Piscataway, NJ, 2019
- Bachelor of Science in Economics and Statistics
  Xiamen University, China, 2017

Major: Computer Science

Presentations and Publications:

J. Gu*, X. Zhang*, and G. Wang, "Beyond the Norm: A Survey of Synthetic Data Generation for Rare Events," *ACM Computing Surveys*, 2025. *Equal Contribution. (Under Review)

J. Gu, W. Du, X. Zhao, A. M. M. Rahman, and G. Wang, "Incorporating Realistic Margin Constraints: A Data-Driven Deep Reinforcement Learning Framework for Advanced Portfolio Management," *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 2025. (Under Review)

A. Dash, **J. Gu**, and G. Wang, "Eye-See-You: Reverse Pass-Through VR and Head Avatars," in *International Joint Conference on Artificial Intelligence (IJCAI) Special Track on AI4Tech*, 2025.

J. Gu, W. Du, and G. Wang, "RAGIC: Risk-Aware Generative Adversarial Model for Stock Interval Construction," *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 2025.

J. Ye, **J. Gu**, X. Zhao, W. Yin, and G. Wang, "Assessing the Creativity of LLMs in Mathematical Problem Solving," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025.

J. Gu*, J. Ye*, G. Wang, and W. Yin, "Adaptive and Explainable Margin Trading via Large Language Models in Portfolio Management," in *Proceedings of the 5th ACM International Conference on AI in Finance*, 2024. *Equal Contribution.

J. Gu*, J. Ye*, A. Uddin, and G. Wang, "DySTAGE: Dynamic Graph Representation Learning on Asset Pricing via Spatio-Temporal Attention and Graph Encodings," *Proceedings of the 5th ACM International Conference on AI in Finance*, 2024. *Equal Contribution.

A. Dash, **J. Gu**, and G. Wang, "HierGAN: GAN-Based Hierarchical Model for Combined RGB and Depth Inpainting," in *The IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshop on Computer Vision in the Wild (CVinW)*, 2024.

W. Yao, W. Du, **J. Gu**, J. Ye, F. P. Deek, and G. Wang, "Establishing a Baseline for Evaluating Blockchain-Based Self-Sovereign Identity Systems: A Systematic Approach to Assess Capability, Compatibility, and Interoperability," in *6th Blockchain and Internet of Things Conference*, 2024.

J. Ye*, B. Goswami*, **J. Gu***, A. Uddin, and G. Wang, "From Factor Model to AI: Machine Learning Paradigm Reshaping Empirical Asset Pricing," *arXiv preprint arXiv:2403.06779*, 2024. *Equal Contribution.

A. Dash, **J. Gu**, and G. Wang, N. Ansari, "Self-Supervised Learning for User Localization," in *International Conference on Computing, Networking and Communications*, 2024.

J. Gu, W. Du, A. M. M. Rahman, and G. Wang, "Margin Trader: A Reinforcement Learning Framework for Portfolio Management with Margin and Constraints," in *Proceedings of the Fourth ACM International Conference on AI in Finance,*, 2023.

J. Gu, S. Shukla, J. Ye, A. Uddin, and G. Wang. "Deep Learning Model with Sentiment Score and Weekend Effect in Stock Price Prediction," *SN Business and Economics 3(7): 119*, 2023.

J. Gu, F. P. Deek, and G. Wang. "Stock Broad-Index Trend Patterns Learning via Domain Knowledge Informed Generative Network," *International Journal of Artificial Intelligence and Applications 14(2):11-28*, 2023.

J. Ye, **J. Gu**, A. Dash, F. P. Deek, and G. Wang. "Prediction with Time-Series Mixer for the S&P500 Index," in *IEEE 39th International Conference on Data Engineering (ICDE) Workshops on Big Data Analytics in Finance and Commerce (BDAFC)*, 2023.

W. Du, J. Ye, **J. Gu**, J. Li, H. Wei, and G. Wang. "SafeLight: A Reinforcement Learning Method toward Collision-free Traffic Signal Control," in *Proceedings of the 37th AAAI Conference on Artificial Intelligence*, 2023.

W. Yao*, **J. Gu***, W. Du*, F. P. Deek, and G. Wang. "ADPP: A Novel Anomaly Detection and Privacy-Preserving Framework in Tokenomics," *International Journal of Artificial Intelligence and Applications 13(6):17-32*, 2022. *Equal Contribution.

To my family.

ACKNOWLEDGEMENTS

First and foremost, I am profoundly thankful to my advisor, Professor Guiling (Grace) Wang, for her steady support, insightful guidance, and mentorship throughout my Ph.D journey. Her vision, expertise, and high standards have been instrumental in shaping both my academic work and my professional development. I deeply appreciate the intellectual freedom she granted me to explore ambitious ideas, while always providing the critical feedback needed to refine and strengthen my work. Her encouragement during the more difficult periods of my doctoral studies was especially meaningful. Beyond research, she has been a role model for leadership and dedication, and I feel incredibly fortunate to have had her as both an advisor and mentor.

I would also like to thank Prof. Zhi Wei, Prof. Ioannis Koutis, Prof. Tucker Balch, and Prof. Zhaodong (Ken) Zhong for serving on my dissertation committee. Their time, thoughtful feedback, and expertise have been invaluable in strengthening the quality and rigor of this work.

In addition, I am grateful to Prof. Ajim Uddin, Prof. Fadi P. Deek, and Prof. Wenpeng Yin for their scientific support, academic guidance, and valuable collaboration in our research. Their insights and support have shaped the direction and depth of several key research projects included in this dissertation.

I would like to express my appreciation to my advisor, Prof. Guiling (Grace) Wang, and New Jersey Institute of Technology for providing financial support through research and teaching assistantships during my doctoral studies, respectively. This support not only enabled the continuation of my research but also enriched my academic experience by allowing me to engage in teaching and contribute to the department's research mission.

I am also thankful to my peers in the research group, Dr. Wenlu Du, Dr. Wei Yao, Dr. A M Muntasir Rahman, Dr. Junyi Ye, Dr. Ankan Dash, and Xuan Zhang,

for our effective collaborations. Their diverse perspectives and generous sharing of knowledge have broadened my understanding and improved the scope of my research.

Finally, I am deeply grateful to my family, whose unwavering belief in me made it possible to pursue this path. Their encouragement was a constant source of strength through moments of doubt and distance. And to my two cats, thank you for the comforting companionship, which brought warmth and calm during many long hours of research.

TABLE OF CONTENTS

Chapter **Page**

1 INTRODUCTION 1

1.1 Technical Contributions 4

1.2 Impact and Broader Applicability 7

1.3 Future Directions 8

2 RELATED WORK 10

2.1 Stock Market Prediction 10

2.2 Interval Prediction on Time Series 12

2.3 Portfolio Management 13

2.4 Natural Language Processing for Financial Prediction 14

2.5 GANs on Time Series 15

2.6 Graph Models for Financial Time Series 16

2.7 LLMs in Finance 17

3 SENTIMENT IMPACT AND WEEKEND EFFECT ON STOCK MARKET PREDICTION 19

3.1 Introduction 19

3.2 Methodologies 21

3.2.1 Problem formulation 21

3.2.2 Sentiment mapping using VADER 22

3.2.3 Temporal modeling using GRU 25

3.3 Experiments 26

3.3.1 Dataset 26

3.3.2 Baselines 28

3.3.3 Hyperparameter settings 29

3.3.4 Results and discussion 29

3.3.5 Ablation study on weekend effect 32

TABLE OF CONTENTS
(Continued)

Chapter **Page**

3.4 Conclusions 34

4 STOCK BROAD-INDEX PREDICTION BY GENERATIVE ADVERSARIAL NETWORKS 36

4.1 Introduction 36

4.2 Preliminary on Generative Adversarial Nets 38

4.3 Problem Formulation 39

4.4 Methodologies 40

4.4.1 Generator 40

4.4.2 Critic 44

4.4.3 Optimization 44

4.4.4 Training process of *IndexGAN* 46

4.4.5 Deployment 46

4.5 Model Evaluation 48

4.5.1 Dataset 48

4.5.2 Baselines 48

4.5.3 Parameter settings 49

4.5.4 Evaluation metrics 50

4.5.5 Experiment results 50

4.6 Conclusion 55

5 RAGIC: RISK-AWARE GENERATIVE FRAMEWORK FOR STOCK INTERVAL PREDICTION 57

5.1 Introduction 57

5.2 Problem Formulation 61

5.3 Sequence Generation 61

5.3.1 Generator 63

5.3.2 Critic 65

5.3.3 Optimization 66

TABLE OF CONTENTS
(Continued)

Chapter **Page**

5.4 Interval Construction 67

5.4.1 Sequence simulation 67

5.4.2 Risk-sensitive interval 68

5.5 Model Evaluation 70

5.5.1 Experiment 1: interval prediction 70

5.5.2 Experiment 2: ablation study 77

5.5.3 Experiment 3: point prediction 80

5.6 Discussion on Downstream Applications 82

5.7 Conclusions and Future Work 83

6 DYSTAGE: GRAPH REPRESENTATION LEARNING FOR DYNAMIC ASSET PRICING 84

6.1 Introduction 84

6.2 Problem Formulation 87

6.3 Methodology 87

6.3.1 Dynamic graph construction 88

6.3.2 Topological module 89

6.3.3 Temporal module 93

6.3.4 Node-level prediction task 94

6.4 Experiment 95

6.4.1 Dataset 96

6.4.2 Baselines 98

6.4.3 Implementation details 98

6.4.4 Comparison of baselines (RQ1) 99

6.4.5 Portfolio management (RQ2) 101

6.4.6 Graph learning (RQ3) 102

6.4.7 Ablation study (RQ4) 103

TABLE OF CONTENTS
(Continued)

Chapter **Page**

6.5 Conclusion 104

7 MARGIN TRADER: A REINFORCEMENT LEARNING FRAMEWORK FOR PORTFOLIO MANAGEMENT WITH MARGIN AND CONSTRAINTS 106

7.1 Introduction 106

7.2 Preliminary 109

7.2.1 Trading using a margin account with long and short positions 109

7.2.2 Assumptions 110

7.3 Margin Trader 110

7.3.1 Customizable settings and notations 111

7.3.2 Reinforcement learning framework 111

7.3.3 Optimization and margin handling 116

7.4 Experiments 121

7.4.1 Portfolio pool 121

7.4.2 Baselines 121

7.4.3 Evaluation metrics 122

7.4.4 Implementation 122

7.4.5 Results comparison 123

7.4.6 Rolling window evaluation 126

7.5 Conclusions 129

8 ENHANCED MARGIN TRADING VIA LLMS 130

8.1 Introduction 130

8.2 Methodology 132

8.2.1 External datasets 132

8.2.2 Explainable market forecasting/reasoning 134

8.2.3 Position reallocation 138

8.3 Experiments 140

TABLE OF CONTENTS
(Continued)

Chapter **Page**

8.3.1 Data source 140

8.3.2 Baseline 140

8.3.3 Implementation details 141

8.3.4 Results and analysis 143

8.4 Conclusion 146

9 CONCLUSION AND FUTURE DIRECTIONS 147

9.1 Summary 147

9.2 Potential Research Directions 148

REFERENCES 151

LIST OF TABLES

Table		Page
3.1	Statistics of Stock and News Datasets	28
3.2	Performance Comparison of Proposal Framework and Benchmarks	30
3.3	Ablation Study on the Weekend Effect of Sentiment Features	33
4.1	Performance Comparison of *IndexGAN* and Baselines	51
5.1	Statistics of Stock Indices	72
5.2	Numerical Results for Interval Prediction	74
5.3	Comparison of Interval Width Adjustment Methods on DJI	80
5.4	Numerical Results Regarding MAPE (%) for Point Prediction	82
6.1	Summary of Statistics for Dynamic Asset Graph Datasets	96
6.2	Descriptive Statistics for Selected Asset Pricing Features	97
6.3	Comparison Results for Asset Pricing	99
6.4	Portfolio Management Results	100
6.5	Ablation Study Results	103
7.1	Hyperparameters for Margin Trader and FinRL	123
7.2	Numerical Results on Portfolio Management	124
7.3	Time Span for Rolling Window Analysis	126
7.4	Numerical Results of Three Agents in Rolling Window Analysis	128
8.1	Mapping from Sentiment to Margin Changes	134
8.2	Portfolio Management Results on Moderate Strategy	142
8.3	Portfolio Management Results on Radical Strategy	143

LIST OF FIGURES

Figure		Page
1.1	The unified framework for robust financial intelligence.	5
3.1	Proposed framework with sentiment analysis and deep learning.	21
3.2	Actual closing price and predicted prices from LSTM and GRU with averaged sentiment feature 2 on DJIA.	32
4.1	The architecture of *IndexGAN*.	40
4.2	Horizon-wise strategy.	48
4.3	Forecasting horizon analysis.	53
4.4	Ablation study on variants of *IndexGAN*.	54
5.1	Illustration of the relationship between SPX and VIX in 1997-2021. The red areas in SPX show the price drop, and the corresponding red areas in VIX show the peaks.	59
5.2	The overall architecture of GAN in *RAGIC* during the training process.	62
5.3	Sequence simulation (left) and *risk-sensitive interval* construction (right).	66
5.4	Comparison of predicted intervals on SPX.	73
5.5	Ablation study of *IndexGAN*.	78
5.6	Impact of the prediction horizon.	78
5.7	Visualized risk score for VIX and VIX return.	79
6.1	An example of dynamic graphs in asset pricing.	85
6.2	The framework of *DySTAGE*.	88
6.3	Similarity comparison between node embeddings from *DySTAGE* and ground truth.	102
7.1	The interaction process between agent and environment in the *Margin Trader*.	110
7.2	The equity changes of DJI, FinRL and *Margin Trader*.	125
7.3	The equity changes of DJI, FinRL, and *Margin Trader* under four rolling windows.	127
8.1	The framework for Margin Trading with RL and LLM.	132

LIST OF FIGURES
(Continued)

Figure **Page**

8.2 Sample prompt and response for macroeconomics indicator. 135

8.3 Sample prompt and response for new summarization. 136

8.4 Sample prompt and response for news. 137

8.5 Comparison of GPT-4o trend predictions using micro firm-specific news as input with DJIA movements. 145

8.6 Equity changes on various adjustment frequencies. 146

CHAPTER 1

INTRODUCTION

Financial markets are inherently volatile and uncertain, driven by a complex interplay of economic conditions, political developments, corporate actions, and investor behavior. This uncertainty collectively results in nonlinear price behavior and often unpredictable asset price movements, posing significant challenges for modeling and decision-making. To address these complexities, a wide range of machine learning techniques have been increasingly applied across finance applications, including generative models for simulating plausible market scenarios, graph neural networks for capturing inter-asset relationships, and reinforcement learning for optimizing sequential trading decisions. While these approaches have shown promise in advancing financial intelligence, critical gaps remain in their ability to reflect real-world market uncertainty, structural evolution, and operational constraints. Based on a thorough review of the literature and practical considerations in finance, this dissertation identifies two critical challenges that must be addressed and one structural characteristic that can be strategically leverage to advance robust financial modeling.

Challenge 1: human-driven stochasticity in financial markets. Unlike natural systems governed by physical laws, the stochastic nature of financial markets is driven by human behavior, socio-economic structures, and collective decision-making. For instance, the 2008 subprime mortgage crisis was not a natural inevitability but a human-driven catastrophe fueled by speculative housing investments, excessive leverage, and insufficient regulatory oversight. Market trends can shift abruptly due to psychology factors such as greed, fear, herd behavior, and rumor propagation, resulting in volatile dynamics that are inherently difficult

to model. Asset prices, influenced by these behavioral factors, exhibit random fluctuations and uncertainty, making risk management a persistent concern for investors.

Despite this complexity, many existing works simplify it by single-point predictions that provide only a single expected outcome. Such forecasts are insufficient in capturing the range of plausible market behaviors or quantifying risk. In practice, even small deviations from expected outcomes can lead to substantial financial consequences, especially during volatile or illiquid periods.

To support risk-aware decision-making, it is essential to move beyond point forecasts and adopt interval prediction methods that reflect the full spectrum of market scenarios [1]. Despite their importance, such approaches remain underexplored in financial modeling. Accurate uncertainty estimation is critical for early risk detection, effective hedging against black swan events, and preserving liquidity during extreme market conditions. Therefore, a risk-aware model that reflects multiple plausible futures and produces reliable interval predictions is necessary under noisy, incomplete, or shifting market information.

Challenge 2: risk-intensive decision-making under real-world constraints. Financial decision-making in practice is shaped not only by market dynamics but also by operational constraints and regulatory mechanisms. Investors periodically reallocate funds across financial assets to optimize returns while mitigating risk. One of the most impactful and risk-intensive elements is margin trading, which allows investors to borrow funds to for larger positions. It can amplify potential returns, e.g., larger funds may aim for 50% gains through margin trading compared to 5% from cash-only retail investing. However, this amplification comes at a cost: leverage significantly increases downside exposure, where even minor adverse market movements can rapidly trigger margin calls, forced liquidations, and cascading portfolio losses.

Margin accounts enforce constraints to prevent excessive losses, such as triggering position closures when equity falls below a threshold. Additionally, it allows short selling to hedge risks during market downturns, making dynamic long-short position allocation crucial in evolving markets. These mechanisms highlight the necessity of proactive risk control, as mismanaging leveraged positions can quickly escalate into significant financial losses. These dynamics make margin trading a double-edged sword that magnifies both opportunity and vulnerability.

Despite its central role in practice, most existing reinforcement learning approaches assume simplified environment with cash-only trading, neglecting the complex dynamics introduced by leverage, margin constraints, and position risk. To build robust and deployable trading agents, it is essential to develop constraint-aware decision-making frameworks that can dynamically manage long-short positions, enforce margin requirements, and respond to leverage-induced risk in evolving market conditions.

Opportunity: leveraging structural dynamics in evolving financial systems. Besides the previous two challenges, we also identify a critical structural characteristic of financial markets that can be strategically leveraged: their inherently dynamic and interconnected nature. Financial markets are not composed of isolated assets but function as interconnected systems, where relationships among assets continuously evolve over time. These relationships may arise from sector affiliation, return correlations, supply chain dependencies, co-movement patterns, or shared exposure to macroeconomic factors. Market structures shift dynamically in response to corporate actions, economic cycles, geopolitical events, and investor behavior—making the financial system a constantly evolving graph. Traditional models often rely on static representations of asset relationships, assuming fixed structures over time. Such simplifications fail to capture the temporal dynamics of asset interdependencies, limiting the ability to adapt to changing market conditions.

In contrast, the dynamic nature of financial systems offers a rich source of information that can be leveraged for more responsive and context-aware modeling.

Dynamic graph learning provides a powerful framework to model these evolving structures, allowing for the simultaneous encoding of both topological changes and temporal patterns. While widely adopted in domains like traffic forecasting and social network analysis, this approach remains relatively underexplored in finance. By capturing structural shifts in asset relationships, dynamic graph models can improve forecasting accuracy, enhance market understanding, and support more adaptive portfolio strategies.

Therefore, harnessing structural dynamics not only deepens the understanding of market complexity but also enables AI systems to align more closely with the reality of financial environments—where adaptation to structural change is key to long-term resilience and performance.

1.1 Technical Contributions

Motivated by the above challenges and opportunity, this dissertation designs a unified and integrated framework for robust financial intelligence, as shown in Figure 1.1. The framework is designed as a progressive pipeline that mirrors the practical flow of financial decision-making—from high-level market forecasting to fine-grained asset pricing and actionable portfolio execution. The pipeline begins with uncertainty-aware forecasting on macro-level financial indices, capturing the stochastic nature of market dynamics driven by investor sentiment and external shocks. It then transitions to micro-level asset pricing, incorporating dynamic structural representations of inter-asset relationships to reflect the evolving nature of financial systems. Finally, it advances to constraint-aware portfolio management, where the system takes real trading actions—allocating capital across long and short positions—under real-world constraints such as such as margin requirements,

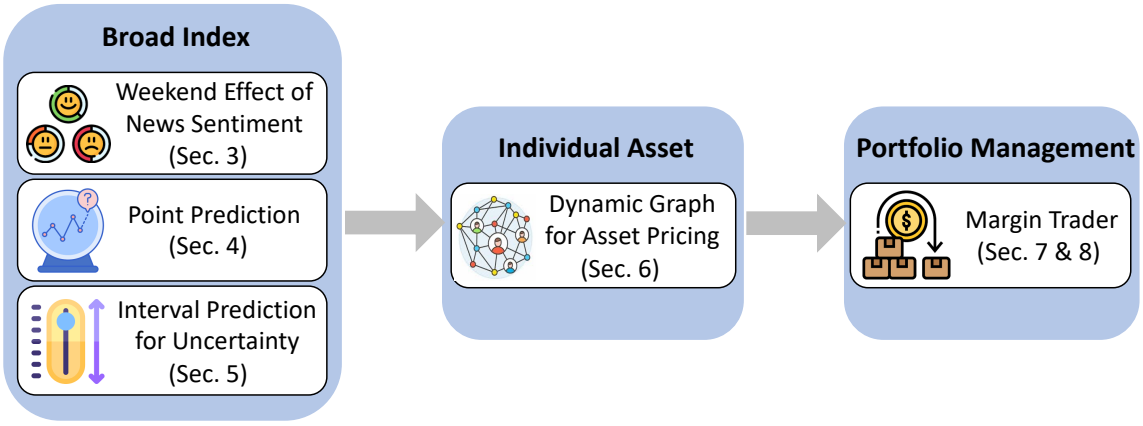


Figure 1.1 The unified framework for robust financial intelligence.

leverage limits, and regulatory rules, enabling risk-managed execution of investment strategies. Each module in the framework is modular yet inter-operable, allowing for adaptive integration and end-to-end optimization. Together, they form a cohesive system capable of navigating the complexity of modern financial markets with both robustness and flexibility.

Uncertainty-aware forecasting and interval prediction on macro indices: To capture the stochastic nature of financial markets and support risk-sensitive decision-making, the framework first introduces models for interval prediction that explicitly quantify uncertainty. Initial work in Chapter 3 explores the weekend effect of sentiment accumulated over weekends and holidays using temporal deep learning models. Chapter 4 presents *IndexGAN*, a Generative Adversarial Network (GAN) for simulating multi-step stock index trajectories under stochastic volatility. These early contributions demonstrate the potential of combining sequential modeling and generative simulation to capture diverse market behaviors. Building on these insights, Chapter 5 introduces *RAGIC*, a hybrid generative-inference framework for risk-aware interval prediction. *RAGIC* employs sequence generation and statistical inference to build intervals for asset returns, explicitly quantifying uncertainty in varying volatility regimes. These intervals also adapt to underlying risk conditions, offering a robust basis for financial decisions. It

significantly advances uncertainty modeling by shifting the focus from point prediction to confidence-aware output, supporting potential downstream applications such as risk management and position sizing.

Dynamic graph learning for individual asset pricing: To model the evolving relationships among assets, Chapter 6 presents *DySTAGE*, a dynamic graph learning framework for asset pricing from two contributions: the first universal formulation from asset time series to dynamic graphs accommodating changes in asset composition and relationships, and a graph learning model enriched with spatio-temporal attention and graph encodings revealing financial insights to capture the evolving nature of financial networks over time. *DySTAGE* improves multi-asset prediction accuracy and supports informed investment strategies in a rapidly changing market.

Adaptive and explainable margin trading with realistic constraints: Robust intelligence must extend beyond prediction to decision-making that is both feasible and interpretable under real-world constraints. Therefore, the framework includes a decision-making module on constraint-aware portfolio management via reinforcement learning. Chapter 7 introduces *Margin Trader*, the first RL framework designed for margin trading that allows traders to leverage their positions in both long and short directions, aiming to balance profit maximization and risk management. It integrates margin accounts and realistic constraints, such as margin requirements, leverage ratios, and buying power, into the trading environment. *Margin Trader* supports various Deep Reinforcement Learning (DRL) algorithms and offers traders the flexibility to customize critical settings, such as equity allocation, margin ratios, and maintenance requirements, to suit diverse market conditions, individual preferences, and risk tolerance. *Margin Trader* is versatile and can be extended to encompass various financial markets and instruments, such as futures and cryptocurrency markets. Chapter 8 further enhances its adaptivity and explainability

by integrating large language models (LLMs) to guide portfolio reallocation in response to changing market conditions. LLMs learns market trends from diverse external data sources and determine optimal long-short ratios with a clear reasoning path, then interact with *Margin Trader* to adjust positions and improve trading strategies. This module is flexible to accommodate various data types, including time series and text, along with multiple LLMs.

1.2 Impact and Broader Applicability

This unified framework offers both methodological innovation and practical relevance across multiple stakeholder groups in the financial ecosystem.

For individual investors: this robust financial intelligence framework is lightweight, modular, and publicly available, making it accessible for hands-on implementation using open-source tools and datasets. These components can be easily integrated into personal trading workflows, providing everyday investors with enhanced forecasting, uncertainty quantification, and risk-aware decision support that are typically only available to professional traders.

For researchers in the academic area: this framework explores the frontiers of combining various techniques that are shaping the future of financial technology, such as generative modeling, dynamic graph learning, reinforcement learning, and large language models. It offers a research testbed for experimenting with hybrid learning paradigms, explainable AI in finance, and real-time adaptive systems, contributing to the growing body of work at the intersection of AI and quantitative finance.

For institutional investors: for mutual funds, hedge funds, and asset managers, the framework serves as a scalable and extensible solution for real-time portfolio optimization, risk-sensitive forecasting, and market structure analysis. Its ability to adapt to new information, integrate external signals, and operate

under trading constraints makes it suitable for enterprise-level deployment in active investment strategies, stress testing, and regulatory compliance.

Together, this dissertation contributes a comprehensive framework for advancing robust financial intelligence, which tackles the core challenges of stochastic uncertainty, dynamic structural modeling, and real-world trading constraints. This robust financial intelligence framework not only enhances forecasting accuracy and interpretability but also supports actionable, risk-aware decision-making in volatile and evolving markets.

1.3 Future Directions

Building on the foundational contributions, Chapter 9 identifies several promising avenues for extending the framework:

Asset-level uncertainty forecasting: *RAGIC* can be expanded to individual stock-level interval prediction by incorporating firm-specific events and implied volatility signals derived from call options, to improve asset-specific risk quantification.

Scalable graph modeling: *DySTAGE* can be extended through subgraph decomposition, using sectoral or correlation-based clusters to improve computational efficiency and reveal more granular structural insights.

Multi-agent RL for long-short adjustment: *Margin Trader* can evolve into a multi-agent RL system, where separate agents coordinate portfolio allocation and position rebalancing, creating a layered decision-making architecture responsive to both market conditions and constraint sets.

Event-triggered adjustment mechanisms: To avoid missing sudden regime shifts or volatility spikes, future work can monitor key market signals and trigger immediate position reallocation. This enhances the framework's ability to respond rapidly to urgent market needs and avoid delayed reactions in turbulent environments.

Real-time model adaptation with evolving LLMs: The automatic integration of new or updated LLMs into the decision-making pipeline enables the system to avoid retrospective data leakage and continuously stay on the cutting edge of techniques to improve performance on financial trend extraction, external signals parsing, and strategy generation.

These directions further strengthen the framework's granularity, adaptability, and responsiveness, positioning it as a foundation for intelligent financial systems capable of operating under real-time constraints, regime shifts, and high-stakes uncertainty.

CHAPTER 2

RELATED WORK

Advances in machine learning have led to a growing body of research on intelligent financial modeling, spanning prediction, generative modeling, and decision-making systems. This section reviews relevant literature grouped by both task-specific objectives and methodological developments. We begin with core financial modeling tasks, including stock market prediction in Section 2.1, interval-based forecasting on time series in Section 2.2, and portfolio management in Section 2.3. We then examine methodological literature that support these tasks, including natural language processing for extracting market signals in Section 2.4, GANs for time series in Section 2.5, and graph-based models for capturing inter-asset dependencies in Section 2.6. Finally, we highlight recent work on applying large language models (LLMs) to financial applications in Section 2.7. Together, these threads form the foundation for this dissertation's integrated approach to uncertainty-aware forecasting, structurally informed asset pricing, and constraint-compliant portfolio management.

2.1 Stock Market Prediction

Early studies on the stock market, like the efficient market hypothesis (EMH), suggest that prices reflect all available information, and the future price movement is driven by new information, e.g., news on the same day [2]. Hence, it is impossible to generate consistent alpha and beat the market by predicting future returns since news is unpredictable [3]. However, many researchers dispute EMH by showing predictable patterns from emerging and mature markets both empirically and theoretically [4], from various perspectives of statistical inference and behavioral finance. Besides early research on the effectiveness of the market, the conventional approaches range

from traditional models like ARIMA [5] to machine learning models. They are often presented as baselines for comparison with deep learning models now.

Ballings et al. [6] run experiments on European companies to benchmark ensemble methods (Random Forest, AdaBoost, Kernel Factory) against single classifier models (Neural Networks, Logistic Regression, K-Nearest Neighbors, and Support Vector Machine (SVM)). The results show that Random Forest is the top algorithm, followed by SVM and other ensemble methods, which clearly suggests that ensembles in algorithms are a contributing factor. Other than performing as a prediction model, machine learning architectures are also developed as tools for data preprocessing. Wang and Wang [7] conduct principal component analysis (PCA) on various index data to project input features to a lower dimensional space, for the purpose of eliminating redundancies and correlation between original features.

Meanwhile, with the power of exploring nonlinear relationships and dealing with big data, deep learning has become a useful tool. Gudelek et al. [8] adopt a sliding window on financial time series data to generate 2-dimensional images, and implement Convolutional Neural Networks (CNN) to classify the movement of prices on the next day and predict the trading signal. DARNN [9] proposes a dual-stage attention-based RNN model to predict the stock trend. The input attention mechanism adaptively extracts input features; the temporal attention mechanism selects relevant hidden states across time spans. AdvLSTM [10] employs the concept of adversarial training on Long Short-Term Memory (LSTM) [11] to predict stock market price. The perturbations are added as the inputs to simulate the stochasticity of prices. HMG-TF [12] designs an enhanced version of the Transformer by incorporating a multi-scale Gaussian prior, orthogonal regularization, and the trading gap splitter into the stock movement prediction. FactorVAE [13] integrates factor as the latent variables into the Variational Autoencoder (VAE) as a prior-posterior learning model to predict stock returns.

There have been few works on predicting future price ranges. Among them, linear regression with ordinary least square [14] is adopted to forecast the interval for price. The Bootstrap method [15] is applied to build a confidence interval for the mean price. Even though some works have attempted to learn from noisy data, they still have neglected risk, even when facing market uncertainty.

2.2 Interval Prediction on Time Series

Uncertainty quantification is often formulated into predicting intervals. Previous work related to neural networks can be categorized into four types for dynamic time series. (1) Bayesian methods [16] apply prior and posterior probability distributions on the parameters of neural networks to quantify uncertainty [17]. The approximated Bayesian inference with dropout training [18] is used in practice. (2) Quantile methods extend quantile regression [19] to neural networks. They model the cumulative distribution of the target and predict the interval boundary directly [20, 21], while being limited by generalizability to new tasks and datasets. (3) Ensemble methods combine a set of models trained with bootstrapped samples[22], different configurations of parameters or model structures [23], but such methods have a prohibitive computational cost on time and space. (4) Conformal prediction methods [24] use a nonconformity measure for calibration and derive the prediction interval from a set of predictions with residual errors [25]. They assume the exchangeability of data. However, real-world time series hardly satisfy such an assumption due to temporal dependency. It is necessary to develop a method that is efficient in time and space to quantify the uncertainty in the stock market, and is able to capture temporal relations.

2.3 Portfolio Management

Two main lines of study have emerged in this field. The first line focuses on model structures specifically designed for the task of portfolio management. Notably, the study [26] presented an RL framework using an Ensemble of Identical Independent Evaluators (EIIE) topology, considering transaction costs and previous period weights. Alpha Stock [27] is proposed as an RL-based model with Cross-Asset Attention Network to model the interrelationship among assets and a portfolio generator, and is optimized towards the Sharpe Ratio in a return-risk-balanced way. HRPM [28] addressed commission fee and non-zero price slippage using a deep hierarchical RL method. The high-level policy learns to generate portfolio weights to maximize profit, while the low-level policy minimizes the trading cost. Lately, Smart Trader [29] integrates an encoder-decoder into the geometric brownian motion process to model unobservable and complicated stochastic processes of price time series, and utilizes graph attention networks (GATs) for portfolio weight calculation. Unlike the studies above, which consider only the long position, several studies have incorporated both long and short positions in their research. iRDPG [30] introduced the recurrent deterministic policy gradient with imitative learning to address noisy high-frequency financial data and balance exploration and exploitation. Although they include short positions in their trading actions, there is a lack of detailed explanation regarding the short setting, which is crucial in real-world trading. Similarly, other studies, such as Deep Trader and Meta Trader [31, 32], also involve both long and short operations. However, a major limitation of these studies is the strong assumption of a zero market, where the long market value is considered equal to the short market value. These studies typically involve selling stocks in a long position, borrowing stocks initially, mortgaging stocks based on the short position, and making purchases based on the long position. Additionally, RAT [33] explored margin trading using a transformer-based framework, but it failed to incorporate realistic constraints. In

reality, margin trading is only allowed when investors possess sufficient equity as a deposit.

Another line of research aims to develop a generalized framework, such as the FinRL and FinRL-Meta library [34], offering multiple market environments, reproducing popular benchmarks, and cloud deployment for visualization and assessment. With data APIs from different market platforms, it enables training agents for various financial tasks. However, it primarily focuses on long positions, motivating our work to extend it by incorporating margin and constraints to mimic real-world trading procedures.

2.4 Natural Language Processing for Financial Prediction

With the growing digital and social media enabling real-time streaming of financial news, their impact on stock market behavior is explored by utilizing NLP techniques. Existing work for textual stock analysis can be classified into sentiment analysis and context-aware approaches.

Sentiment analysis is leveraged to extract emotions from text and generate sentiment scores as features [35], although this often ignores useful textual information. Zhang, Fuehres, and Gloor [36] use mood words as emotional tags to classify the sentiment of tweets as hope, happiness, fear, worry, and so on, and analyze the correlation between emotional percentage and stock market indices such as Dow Jones, NASDAQ, and SP500. De Fortuny et al. [37] select sentiment words from the news by TF-IDF and calculate sentiment polarity and subjectivity as sentiment features for SVM in making predictions. FinBERT [38] is proposed as a language model based on Bidirectional Encoder Representations from Transformers (BERT) to tackle the NLP task in the financial area and evaluated on two financial sentiment analysis datasets.

On the other hand, context-aware approaches create embeddings from news and feed them into complicated structures to explore context as much as possible [39, 40, 41]. The early attempts at context-aware approaches in the stock market employed the bag-of-words (BoW) model [42] to represent the semantics of words by calculating the frequency of word occurrence and fed word vectors together with price into SVM or RNN to predict movements. Recently, word embedding has been developed to consider the context of words, exploring the semantic similarity of words. Lee and Soo [43] process financial news with word2vec to create input for recurrent CNN. Minh et al. [44] implement Stock2Vec and two-stream Gated Recurrent Unit (GRU) models to create embeddings from financial news for stock price classification. Li et al. [45] propose using Graph Convolutional Networks to represent connections given news embeddings and predict overnight movement. Other than word2vec, Global Vectors for Word Representation (GloVe) [46] and BERT [47] are also widely used to generate word embeddings.

2.5 GANs on Time Series

More recently, there are emerging efforts in using GANs on time series data in the domain of image generation [48, 49], video generation [50], medical [51], text generation [52], biology [53], finance [54] and music [55]. These GANs aim to generate a synthetic sequence that retains the characteristics of the original sequence. For instance, Yu et al. [56] introduce SeqGAN, which extends GANs with reinforcement learning to generate sequences of discrete tokens. Yoon et al. [57] propose TimeGAN for realistic time-series generation, combining an unsupervised paradigm with supervised autoregressive models. These methods are used for tasks of enriching limited real-world datasets [58], data imputation [59], and machine translation [60]. However, the stock market predictions are attempts to discover future patterns instead

of simulating original time series and thus cannot employ most existing efforts on time series GANs.

Some recent approaches implement GANs to predict stock trends. Zhang et al. [61] employ LSTM as a generator and Multi-Layer Perceptron (MLP) as a discriminator to forecast stock pricing. Faraz and Khaloozadeh [62] enhance GAN with wavelet transformation and z-score method in the data preprocessing phase. Sonkiya et al. [63] generate binary sentiment scores by FinBERT and employ GAN for stock price predictions. Such methods adapt Vanilla GAN, which often suffers from mode collapse, and neglect possible driving factors such as complex public emotions, financial policies, and events, which limits the ability to achieve superior performance.

2.6 Graph Models for Financial Time Series

Graph neural networks (GNNs) facilitate the mapping of spatial connections between financial assets. Static graphs, with invariant structures and variable node attributes, serve as the foundation for this analysis. GNN approaches like GCN [64], GraphSAGE [65], GAT [66], and UniMP [67] implement distinct aggregation strategies to interpret complex graph structures. Studies have explored various relational frameworks through shareholding ratios, ownership structures, sectoral similarities, and stock correlations [68, 69, 70, 71]. While they fall short in capturing dynamic markets, where firms constantly emerge and dissolve. Dynamic graphs track evolving system dynamics by accommodating changes in graph structure and node characteristics. Most works, such as T-GCN [72], GCLSTM [73], and DY-GAP [70], maintain a fixed node count despite dynamic linkages and node changes. Conversely, studies like DySAT [74] and EvolveGCN [75] address scenarios with variable node counts, utilizing attention mechanisms and GCN regularization to manage node embeddings in these complex environments. Many of these models are designed for different domains and require significant adaptation for financial time series. Additionally,

most dynamic graph analyses, such as DyTed [76] and DGIB [77], primarily target link prediction and struggle to extend to regression tasks. Complex asset pricing models require handling large spatio-temporal datasets with thousands of nodes and time steps, underscoring the need for innovative approaches tailored to financial markets dynamics.

2.7 LLMs in Finance

The increasing attention on LLMs in the finance sector has led to significant advancements. Unlike traditional machine learning (ML) models trained on specific datasets, LLMs leverage extremely large training datasets, enabling them to perform instruction-following on various tasks with zero-shot capabilities. LLMs have diverse applications in finance, such as automating report/workflow generation [78, 79], predicting market trends [80], analyzing Named Entity Recognition (NER) and sentiment analysis [81, 82, 83, 84, 85], and providing financial advice and question answering [86, 82].

Compared to traditional ML models [87, 88], LLMs offer significant advantages in finance through enhanced reasoning [78], interactivity, and integration capabilities. They provide clear explanations for predictions, improving transparency and trust. Their interactive nature allows dynamic user engagement for refined queries and responses. Moreover, LLMs can seamlessly integrate with various tools and data sources using techniques like Retrieval-Augmented Generation (RAG) [83], delivering comprehensive analyses and decision-making [89]. However, deploying LLMs in finance presents significant challenges. They require immense computational power, leading to environmental concerns. Additionally, they can inherit biases [90] from training data, resulting in unethical outcomes. Their "black-box" nature makes it difficult to understand their conclusions, raising accountability and trust issues. Additionally, small-parameter LLMs often struggle with generalization to new tasks.

Currently, they are suited for simpler tasks instead of complex ones like portfolio management. Their integration into advanced financial strategies is limited, needing significant improvements in model capabilities, fine-tuning techniques, and pipeline design. Addressing these issues is crucial for fully harnessing LLMs' potential in finance and ensuring innovative and ethical applications.

CHAPTER 3

SENTIMENT IMPACT AND WEEKEND EFFECT ON STOCK MARKET PREDICTION

3.1 Introduction

Social media has emerged as a powerful source to derive market sentiments, such as Twitter, Reddit, and Facebook. The majority of existing research extracts the textual information on trading days only, neglecting the potential impact of news or events that occur over weekends and holidays on the stock market. It's proven that the sentiment from news accumulated over weekends significantly affects stock returns [91]. This is also one of the factors that cause the weekend effect [92]. It is a well-known financial anomaly that the average return on Mondays is much lower than that on the other days of the week [93]. The reason behind this is that the firms tend to delay the announcement of bad news until the weekend, allowing investors more time to digest it and avoid market fluctuation. Therefore, it is crucial to consider the news on weekends and holidays when predicting prices. Despite the few efforts that have been made to explore the impact of weekend news on the stock market, they mostly implement a single machine learning or deep learning model and lack comparison on the improvement of news or sentiment on various types of models [94].

To fill this gap, we present a deep learning framework that incorporates the sentiment from weekend news on social media with price features to predict the stock price. Our model first identifies the market sentiment from the news and then uses an RNN model to learn temporal patterns in conjunction with other trading characteristics for predicting future prices. Specifically, given a number of news headlines from Reddit WorldNews Channel, we utilize a sentiment analysis tool, Valence Aware Dictionary and Sentiment Reasoner (VADER) [95], to generate three sentiment scores representing three categories (positive, negative, and neutral) for

each day. The scores on weekends or holidays are combined and averaged with those on the next trading day. Then we design two types of sentiment measures that account for positive and negative attitudes from news on social media. Next, we use RNN to learn the temporal dynamics and potential correlation between sentiment measures and historical stock prices to predict the closing price of the next day. To evaluate the proposed model, we compare performance against a comprehensive set of benchmark models, including traditional econometrics-based models (e.g., Moving Average (MA) and ARIMA), machine learning models (e.g., Support Vector Regression (SVR), Random Forest, and XGBoost), and neural networks (e.g., LSTM and GRU). Additionally, we evaluate the performance of various definitions of sentiment measures as well as the sentiment scores on trading days only to fully investigate the effect of the news during weekends and holidays on the stock market.

Moreover, the selected news and stock data are among the most popular benchmark datasets in the stock prediction domain. The news data comprises daily headlines crawled from Reddit WorldNews Channels, ranked by user votes. The stock indices include DJIA and S&P 500. Both datasets are in a period of high volatility, covering both the financial crisis in 2009 and two dramatic drops in 2015. To enable the model to learn the intricacies of dynamics in real-world stock market data and retain stability, the former is split into the training set, and the latter is selected into the test set. We summarize our contribution as follows.

1. We provide a comprehensive study on various types of models, different sentiment measures definitions, and the impact of averaging weekend news. Our study reveals that incorporating news on weekends and holidays with stock prices can enhance the performance of the price prediction task.

2. The proposed framework, with designed sentiment measures and a deep learning model, effectively learns temporal patterns and potential correlation between stock prices and news on social media. We demonstrate that our framework consistently achieves significant improvements over popular benchmarks for predicting the stock market.

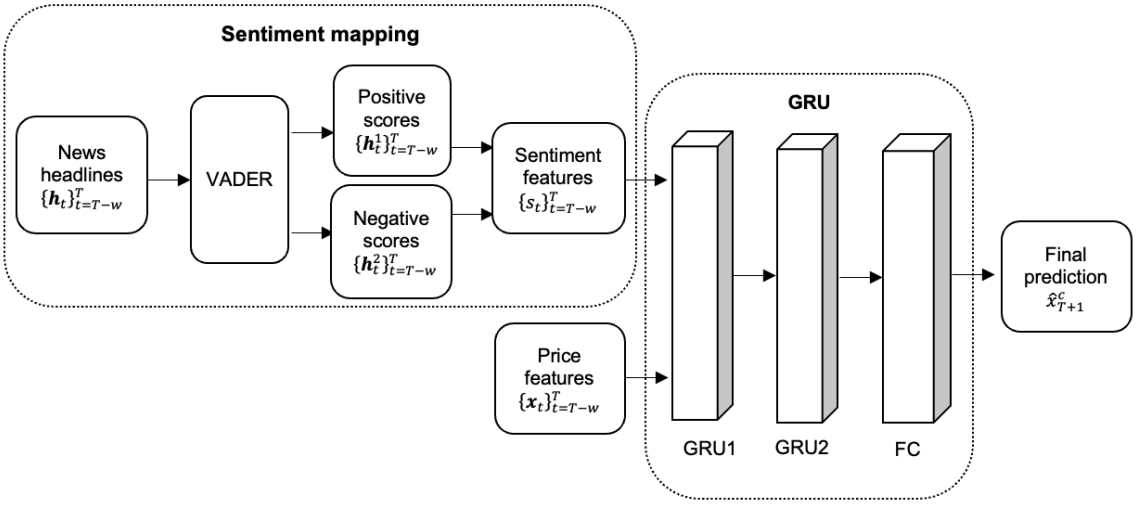


Figure 3.1 Proposed framework with sentiment analysis and deep learning.

3. Our model is trained and tested on the Reddit news dataset and stock indices (DJIA and S&P 500) in a period of high volatility to improve model robustness.

3.2 Methodologies

The proposed model, as depicted in Figure 3.1, consists of three key components: VADER-based sentiment mapping, time-series temporal modeling using GRU, and a dense layer employed for future price prediction. In this section, we first delineate the problem statement, followed by an introduction to VADER and its application in generating sentiment scores from raw text. We also present the definition of sentiment measure that represents the emotional opinion of the financial market and how to incorporate news on weekends and holidays. In the end, we show the process of RNN learning on the temporal patterns of the stock market.

3.2.1 Problem formulation

The typical datasets of the stock market prediction with news include a stock dataset with price features (open, close, high, low, volume) and a textual dataset with raw news headlines in the same period.

Let $\mathcal{X} = \{\mathbf{x}_t\}_{t=1}^{n} \in \mathbb{R}^{n \times p}$ be a matrix which represents the stock dataset with total n samples, where $\mathbf{x}_t$ is a vector with p price-based numerical features. Let $\mathcal{H} = \{\mathbf{h}_t\}_{t=1}^{n} \in \mathbb{R}^{n \times k}$ be a matrix which denotes a textual dataset, we assume each vector $\mathbf{h}_t$ contains a number of k raw news headlines at the time step t. By concatenating stock features and headlines together, we obtain the training data $\mathcal{D} = \{(\mathbf{x}_t, \mathbf{h}_t)\}_{t=1}^{n}$. To investigate the temporal dependency of sequential data, we define the length of the time window as w, shift the window across $\mathcal{D}$, and generate s sets of sequence features $\{(\mathbf{x}_t, \mathbf{h}_t)\}_{t=T-w}^{T}$ within the time window w. The target is the stock price in the future one-time step, where all price features can be used as target features. In this study, we choose the close price x_{T+1}^{c}, as most investors monitor the close price to make informed decisions. The proposed model can be generalized to predict other prices as well. Our goal is to use training data $\mathcal{D}$ to learn a feature map F:

$$\tilde{\Theta} = \arg\min_{\Theta} L(x_{T+1}^{c}, F_{\Theta}(\{(\mathbf{x}_t, \mathbf{h}_t)\}_{t=T-w}^{T})) \tag{3.1}$$

$$\hat{x}_{T+1}^{c} = F_{\Theta}(\{(\mathbf{x}_t, \mathbf{h}_t)\}_{t=T-w}^{T}) \tag{3.2}$$

where $\hat{x}_{T+1}^{c}$ is the predicted price at time $T+1$, Θ is the set of parameters in the mapping function, and it can be estimated by minimizing the loss function L. The loss function can take any form commonly used for regression problems, such as Mean Square Error (MSE), Mean Absolute Error (MAE), and Root Mean Square Error (RMSE). In our study, we employ MAE as the objective function since it is robust to outliers.

3.2.2 Sentiment mapping using VADER

We apply VADER as a lexicon and rule-based sentiment analysis tool attuned to sentiment specifically, which shows great success in various applications. Instead

of determining whether a paragraph is objective or subjective, we are interested in what emotion the text expresses. The words in the news headlines are called lexical features in VADER. The lexicon of VADER contains over 7,500 lexical features with a corresponding valence score on a scale from -4 to 4, indicating extremely negative to extremely positive sentiment.

Given a paragraph $\mathbf{h}_t$ at time step t consisting of k news headlines, VADER maps all lexical features to valence scores. These valence scores are aggregated and normalized to produce the compound score and generate proportions of text that fall into three categories, which are represented by positive score $\mathbf{h}_t^1$, negative score $\mathbf{h}_t^2$, and neutral score $\mathbf{h}_t^3$ in a scale from 0 to 1, such that these three scores all add up to 1. The score in each category $\mathbf{h}_t^c = \{h_{t,j}^c\}_{j=1}^k$ is a row vector with k dimension, where $h_{t,j}^c$ denotes the score value for j^{th} news headline. The scores capture both polarity and intensity of emotion and provide insight into the context of how sentiment is conveyed or embedded in rhetoric for a given text.

We design a sentiment measure from VADER sentiment scores, inspired by the bullishness signal proposed by Antweiler and Frank [96]. First we define

$$m_t^c = \frac{\sum_{j=1}^k w_j h_{t,j}^c}{\sum_{j=1}^k w_j} \tag{3.3}$$

where m_t^c represent the weighted average of sentiment score of type $c \in \{1, 2\}$ over k news headlines at time step t, and $\mathbf{w} = \{w_j\}_{j=1}^k$ is the weight of the score. There are many ways to define weight. For instance, w_j can be the number of words in j^{th} headline, which gives higher importance to longer paragraphs. Besides, considering that news with higher media exposure influences more investors and draws more attention, it's reasonable to set the weight w_j as the number of comments, votes, or views of the post. In this study, we set all weights to 1, so that m_t^c is the simple

averaged sentiment score of type c over k headlines, and $m_t = m_t^1 + m_t^2$ represents the total proportion of averaged positive and negative sentiment scores taken in three categories.

Let $r_t = m_t^1/m_t^2$ be the ratio of positive to negative emotion. The first sentiment measure s_t is defined as follows:

$$s_t^1 = \frac{m_t^1 - m_t^2}{m_t^1 + m_t^2} = \frac{r_t - 1}{r_t + 1} \tag{3.4}$$

As $h_{t,j}^1 \in [0, 1]$, therefore, the sentiment measure s_t^1 is in the scale of -1 to 1. Besides, s_t^1 is invariant if m_t^1 and m_t^2 are scaled by the same constant at the same time. To involve m_t in the sentiment measure, we introduce the second definition:

$$s_t^2 = ln\left(\frac{1 + m_t^1}{1 + m_t^2}\right) = ln\left(\frac{2 + m_t(1 + s_t)}{2 + m_t(1 - s_t)}\right) \approx s_t^1 ln(1 + m_t) \tag{3.5}$$

Compared with s_t^1, s_t^2 will change when both weighted average sentiment scores increase. The range of s_t^2 is approximately between -0.5 to 0.5. The latter sentiment measure s_t^2 always achieves better results in real-world applications.

Considering that the news headlines are not only from trading days but also weekends and holidays, for the averaged sentiment features s^1 and s^2, the sentiment from weekends and holidays is averaged to the next opening day of the market. The key difference between s^1 and s^2 is that the former only considers the ratio of positive and negative sentiment, whereas the latter considers not only their ratio but also their sum.

In addition, to evaluate the presence of the weekend effect, we created two alternative sentiment scores that exclude news from weekends and holidays. In line with our two averaged sentiment measures, we created two non-averaged news

sentiment features s^{1*} and s^{2*}. In the non-averaged news sentiment feature, only the news sentiment on the market opening days is considered, and news on the weekends and holidays is dropped. These two non-averaged sentiment measures allow us to directly compare the contribution of the weekend and holiday effect on the stock price.

In Section 3.2.3 for neural networks, all kinds of sentiment features are denoted as s_t for notational convenience. The detailed comparison for each sentiment measure and its capability to capture emotions is presented in Section 3.3.4.

3.2.3 Temporal modeling using GRU

To obtain the time series patterns of stock price and sentiment from the news, we employ GRU as the RNN model. It is one of the common approaches to solve vanishing and exploding gradient issues of vanilla RNN. Our training input $\{X_t\}_{t=T-w}^{T}$ combines sentiment features s_t and stock prices $\mathbf{x}_t$ in the past w time steps. Let X_t denote the input feature at time step t.

The key design of GRU consists of only two gates. Reset gate R_t takes account of short-term memory, which determines how much the previous state will be kept. Update gate Z_t is responsible for long-term memory, which controls how much new information will be passed along to the future.

$$\begin{aligned} R_t &= \sigma(X_t W_{xr} + H_{t-1} W_{hr} + b_r) \\ Z_t &= \sigma(X_t W_{xz} + H_{t-1} W_{hz} + b_z) \\ \tilde{H}_t &= tanh(X_t W_{xh} + (R_t \odot H_{t-1}) W_{hh} + b_h) \\ H_t &= Z_t \odot H_{t-1} + (1 - Z_t) \odot \tilde{H}_t \end{aligned} \tag{3.6}$$

where H_{t-1} is the hidden state at time step $t-1$, W_{xr}, W_{hr}, W_{xz}, W_{hz}, and W_{xh} are weight parameters, b_r, b_z, and b_h are the bias parameter. Sigmoid function $\sigma(\cdot)$ maps

the value to the interval of $(0, 1)$. To incorporate the effect of reset gate R_t with the regular latent state updating mechanism, candidate hidden state $\tilde{H}_t$ is constrained by the activation function in the interval $(-1, 1)$. The Hadamard product operator $\odot$ controls which matrix to remove from the previous time step. If entries in R_t are close to 0, then it ignores the previous hidden state H_{t-1} and focuses on the current state X_t only. If entries in R_t are close to 1, then it is simply vanilla RNNs. Finally, the update gate Z_t determines how much information from H_{t-1} and $\tilde{H}_t$ is included in H_t. Suppose the update gates in all-time steps are 1, then the hidden state of the beginning will be kept and passed to the final output, no matter how long the period is. That is the way GRU deals with the vanishing gradient problem.

In our study, two layers of GRU are applied to temporally map training input $\{X_t\}_{t=T-w}^{T}$ into hidden representations. The deeper input gate is shown to be beneficial for modeling temporal structures in RNN in previous work [97]. Then the last hidden state is passed into a fully connected layer to generate the final prediction $\hat{x}_{T+1}^{c}$.

3.3 Experiments

3.3.1 Dataset

We evaluate our proposed model on two stock indices and one news dataset. The data is publicly available and downloaded from Kaggle.[1] The stock dataset contains historical daily stock prices in Dow Jones Industrial Average (DJIA) and S&P 500 from 2008/08/08 to 2016/07/01, downloaded from Yahoo Finance[2]. DJIA is one of the oldest and most commonly followed equity indices, tracking 30 of the most highly capitalized and influential companies in major sectors of the U.S. market. S&P 500 is composed of the 500 largest publicly traded companies. Both indices are used as benchmarks of the economy. Stock indices in the chosen period cover the financial

[1]https://www.kaggle.com/aaron7sun/stocknews
[2]https://finance.yahoo.com

crisis and recovery with a large number of ups and downs, which provides stability to our proposed method. The features involved in the model include open, close, high, low, and volume; the target is the close price. For the purpose of efficient computing, all the features from the stock dataset are rescaled by min-max normalization:

$$\mathbf{x}_t^* = \frac{\mathbf{x}_t - min(\mathbf{x}_t)}{max(\mathbf{x}_t) - min(\mathbf{x}_t)} \tag{3.7}$$

where $\mathbf{x}_t^*$ is the features after normalization. After obtaining predicted price data, de-normalization is performed in the opposite way.

News data is crawled from Reddit WorldNews Channel [3], which is one of the largest social news aggregation websites. Content uploaded by users to the site, such as text posts, images, links, and videos, is voted up or down by other users. The news and posts are not only about the financial market in the U.S. but also related to popular events in various industry sectors in the world, which presents an overall attitude with a high potential to influence the market. Our dataset contains the top 25 headlines ranked by users' votes for a single day within the same period as the stock dataset. The headlines are cleaned by removing stop words, punctuation, and digits from the raw text, then processed to extract the sentiment scores by VADER.

To tune the parameters when training the model and evaluate the performance, both stock and news data are split into training, validation, and testing approximately as the ratio of 4:1:1, as shown in Table 3.1. To capture the dependency of time series, a lag window with a size of 5 time steps is moved along both datasets to construct the sequence.

[3]https://www.reddit.com/r/worldnews/?hl=

Table 3.1 Statistics of Stock and News Datasets

Dataset	Time spans	No. of instances
Training	2008/08-2014/02	1384
Validation	2014/02-2015/06	346
Testing	2015/06-2016/06	250

3.3.2 Baselines

We compare our proposed framework with multiple traditional time series and machine learning based. First, we compare our model with traditional technical analysis tools: Simple Moving Average (SMA), Exponential Moving Average (EMA), and ARIMA. SMA is calculated by taking the arithmetic average of the previous days. EMA is calculated by the weighted average of prices in previous time steps, where recent prices are assigned with higher weight. ARIMA is one of the most popular linear models in financial time series prediction. It describes the future value of a variable as a linear combination of lagged values and random errors.

Second, we choose three machine learning models for comparison: SVR [98], RF [99], and XGBoost [100]. Since they cannot deal with temporal dependency in time series, we take the stock features and sentiment features in the previous five days as individual features for the following models.

Finally, we implement LSTM [11] as a variation of RNN. The key design of LSTM is the memory cell, which records short-term information. Compared with the gate mechanism in GRU, it includes an additional input gate to control what information to record in the new memory cell. It shares many properties of GRU and addresses gradient problems as well. In general, the structure in GRU is simpler and contains fewer parameters to tune, and thus the training process is faster.

The performance of the proposed model and all baselines is evaluated with three metrics: Mean absolute error (MAE), Mean absolute percentage error (MAPE), and Root mean square error (RMSE).

3.3.3 Hyperparameter settings

The neural networks (LSTM, GRU) are implemented in Python using Keras. Hyperparameters are tuned based on the validation set and chosen according to the smallest MAE. We use Adam as the optimizer with a batch size of 32 for 250 epochs, and employ early stopping with 100 epochs and dropout to prevent overfitting. The hidden dimensions in two layers of GRU are set to 300 and 100, respectively.

For baselines, we set the period as 2 days, 3 days, 5 days, and 10 days to calculate both SMA and EMA. ARIMA is implemented in Python with the package 'pmdarima'. The optimal parameters are determined automatically regarding the lowest Akaike's Information Criteria (AIC). We tune hyperparameters based on the validation set and report the result from the best model.

3.3.4 Results and discussion

To demonstrate the effectiveness of the proposed model, we conduct a set of experiments. The comparison of results from the proposed model and different baselines is presented in Table 3.2. The first panel of Table 3.2 includes traditional baselines where the forecasting is modeled on closing price only. The middle panel reports the results from machine learning models, and the last panel reports neural networks. Each of them considers different sentiment features following the corresponding definition of sentiment measures to predict stock price. Both features consider news on weekends and holidays and average the VADER score of sentiment from news to the next opening day of the market.

From the tables, it is obvious that GRU with averaged sentiment feature 2 achieves the best results with 0.746% of MAPE on DJIA and 0.795% on S&P.

Table 3.2 Performance Comparison of Proposal Framework and Benchmarks

Model	DJIA			S&P		
	MAE	MAPE(%)	RMSE	MAE	MAPE(%)	RMSE
SMA(2)	143.749	0.845	192.942	17.402	0.870	23.661
SMA(3)	155.843	0.917	213.205	19.025	0.952	26.173
SMA(5)	184.972	1.088	250.672	22.405	1.122	30.622
SMA(10)	244.148	1.438	320.744	29.284	1.468	38.874
EMA(2)	**135.866**	**0.800**	**181.657**	**16.411**	**0.821**	**23.908**
EMA(3)	145.939	0.859	195.625	17.678	0.884	23.908
EMA(5)	166.618	0.981	224.468	20.122	1.008	27.333
EMA(10)	218.391	1.286	288.044	25.761	1.291	34.374
ARIMA	626.68	3.698	761.511	61.666	3.14	85.542
SVR+s^1	156.964	0.921	199.99	17.99	0.901	23.623
SVR+s^2	164.310	0.960	203.173	19.408	0.966	24.258
RF+s^1	153.610	0.905	213.820	18.435	0.923	25.473
RF+s^2	149.865	0.883	208.016	17.634	0.883	24.186
XGBoost+s^1	135.311	0.798	183.709	16.432	0.823	22.54
XGBoost+s^2	**133.764**	**0.789**	**183.230**	**16.35**	**0.818**	**22.11**
LSTM+s^1	145.867	0.855	182.5	16.427	0.821	21.355
LSTM+s^2	139.708	0.82	178.63	15.938	0.798	21.191
GRU+s^1	131.96	0.778	176.332	16.254	0.813	21.107
GRU+s^2	**126.85**	**0.746**	**165.385**	**15.847**	**0.795**	**21.558**

Note: The numbers next to SMA and EMA indicate the number of lags, e.g., SMA(2) is calculated simple average price of the previous two days. s^1 and s^2 indicate averaged sentiment features 1 and 2, respectively.

Compared with traditional MA, ARIMA, and machine learning, as well as neural networks considering price features only, models with sentiment features have lower errors. This indicates the predictive power of financial news on the stock price forecasting task.

Additionally, neural networks (LSTM, GRU) perform better than machine learning models (SVR, RF, XGBoost) in general. The reason is that machine learning models flatten the features across temporal dimensions into single individual features. The temporal dependency is largely ignored by machine learning models, while considered by recurrent neural networks. However, one exception is that XGBoost outperforms LSTM on DJIA due to its highly scalable and powerful implementation of gradient boosting trees. Besides, XGBoost achieves lower error regarding all metrics and all indices than EMA(2), which is the best model in traditional baselines.

Moreover, among the traditional and statistical baselines, EMA(2) outperforms all other models. It is the most basic technical indicator, but it's effective for predicting price, even better than some neural networks. In addition, as the period of the moving average decreases from 10 to 2, prediction becomes closer to the actual price, and MAPE changes from 1.286% to 0.800% on DJIA and from 1.291% to 0.821% on S&P. Besides, ARIMA has the highest error. The main reason is that ARIMA deals with short-term patterns only, the prediction converges to a straight line when it comes to long-term dependency.

Furthermore, Figure 3.2 exhibits the predicted prices from LSTM and GRU with the best performance on DJIA, i.e., LSTM and GRU with averaged sentiment feature 2, as well as the actual closing price during the testing period. This particular period covers two significant price drops in 2015. Thus, the market in the testing period can be seen as high risk. Compared with LSTM on the same sentiment features, GRU gives a better prediction on DJIA in this period due to its simpler structure with only two gates and competitive ability, even the best performance of 0.820% of MAE

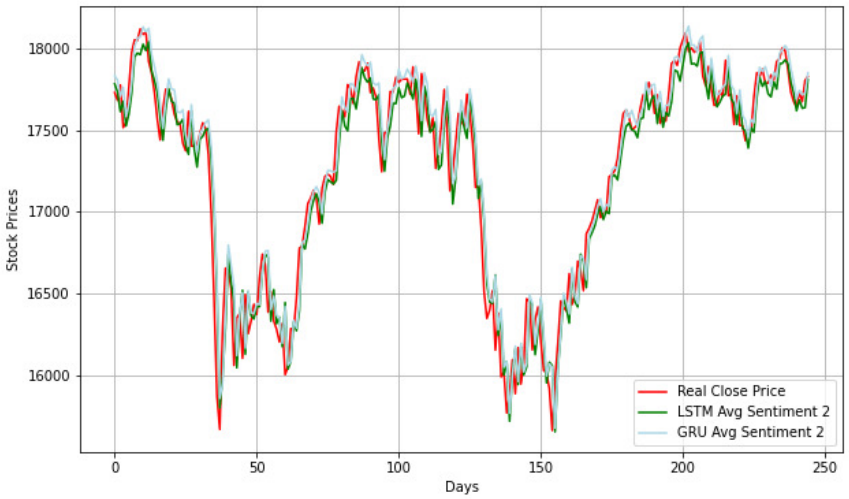


Figure 3.2 Actual closing price and predicted prices from LSTM and GRU with averaged sentiment feature 2 on DJIA.

from LSTM on s^2 is worse than GRUs with both sentiment features. Since LSTM and GRU generally outperform most machine learning and traditional models, the proposed framework, including RNNs and sentiment features, is demonstrated to be more appropriate for the stock price prediction task.

Finally, we investigate the potential difference between the two definitions of sentiment measure. Most machine learning models and neural networks with the lowest MAE are trained on the second kind of sentiment feature s^2. It indicates that the sentiment features involved with the total proportion of averaged positive and negative sentiment scores can significantly improve the quality of prediction.

3.3.5 Ablation study on weekend effect

To further investigate the impact of news and its weekend effect on the stock market, we implement several variants on both indices: machine learning models and neural networks without sentiment features, and those with sentiment features but not considering news on any weekend and holidays s^{1*} and s^{2*}.

Table 3.3 Ablation Study on the Weekend Effect of Sentiment Features

Model	DJIA			S&P		
	MAE	MAPE(%)	RMSE	MAE	MAPE(%)	RMSE
SVR	184.148	1.080	242.273	21.489	1.068	27.607
SVR+s^{1*}	146.672	0.862	195.692	17.466	0.873	23.024
SVR+s^{2*}	140.59	0.827	186.82	16.713	0.837	22.482
RF	170.324	1.003	236.07	19.915	0.998	27.697
RF+s^{1*}	149.989	0.884	208.411	18.666	0.934	26.033
RF+s^{2*}	151.473	0.893	211.052	17.905	0.897	24.69
XGBoost	145.634	0.857	196.164	17.357	0.868	22.695
XGBoost+s^{1*}	136.237	0.803	185.93	16.758	0.838	22.791
XGBoost+s^{2*}	**135.026**	**0.796**	**183.110**	**16.42**	**0.822**	**22.483**
LSTM	172.492	1.005	205.718	16.666	0.835	22.093
LSTM+s^{1*}	146.489	0.858	182.743	16.581	0.832	22.462
LSTM+s^{2*}	140.197	0.825	185.132	**16.103**	**0.809**	**22.18**
GRU	142.591	0.836	181.198	16.725	0.833	21.388
GRU+s^{1*}	130.065	0.768	171.129	16.592	0.831	21.929
GRU+s^{2*}	**128.435**	**0.755**	**170.883**	16.326	0.819	22.039

Note: s^1* and s^2* indicate non-averaged sentiment feature 1 and non-averaged sentiment feature 2, respectively. The models without sentiment features are implemented on price-based features only. The best-performing models are marked in bold.

As reported in Table 3.3, the numerical results suggest that machine learning models and neural networks trained on non-averaged sentiment features (s^{1*} and s^{2*}) have worse performance generally, compared with averaged sentiment features s^1 and s^2 in Table 3.2. This observation reaffirms the presence of the weekend factor in asset pricing. It also signifies that taking news on holidays and weekends into consideration can further boost the performance of the neural network on stock forecasting.

For each type of model, the one without sentiment features performs worst on the price prediction task; the model with s^{2*} achieves better performance than the one with s^{1*} in general. This phenomenon is consistent with the average sentiment features. It validates the importance of sentiment features from news to the stock market, and the superior performance of the sentiment feature considering the total proportion of positive and negative scores.

3.4 Conclusions

Motivated by fully exploring how emotion influences stock price and its weekend effect, we propose a deep learning framework with VADER and two sentiment measures, and then conduct a comprehensive study on various types of models. Specifically, we use VADER and design two definitions of sentiment measures to process text from news headlines, then generate averaged and non-averaged sentiment features to investigate the impact of news from holidays and weekends. The sentiment features and stock price are concatenated and fed into GRU layers to capture temporal dependency as well as the correlation between news and future stock price. To evaluate the proposed framework, we conduct a set of popular benchmarks, including traditional and statistical models, machine learning models, and RNNs on stock indices (DJIA and S&P 500) and Reddit news datasets in a testing period with high market risk for comparison. Among the baselines, the numerical results regarding MAE, MAPE, and RMSE validate that neural networks achieve the best performance due to

their powerful ability to handle temporal dependencies, followed by machine learning models (e.g., XGBoost). Moreover, the news on holidays and weekends is also proven to be influential on the stock market price.

CHAPTER 4

STOCK BROAD-INDEX PREDICTION BY GENERATIVE ADVERSARIAL NETWORKS

4.1 Introduction

Since existing efforts predominantly rely on historical prices and neglect the inherent characteristics of the stock market, there is a need to better address the issues and yield higher prediction accuracy. Unlike many other tasks, the prediction accuracy of broad-index return cannot keep increasing, in theory, because market arbitrage opportunities will degrade accuracy over time. Naturally, an effective trading strategy that is likely to generate profit will be adopted by an increasing number of investors; eventually, such a strategy may no longer perform as well, considering that the market is composed of its collective participants. Thus, in practice, if a certain method is capable of achieving over 60% accuracy using a dataset of a lengthy period, encompassing a financial crisis, it is then considered powerful enough. Therefore, we aim to provide a framework that performs with accuracy, over time, and in different market conditions.

In this chapter, we propose the following strategies which employ domain knowledge in financial systems: (1) formulating stock movement to a multiple-step prediction problem; (2) exploiting Wasserstein GAN to avoid mode collapse that tends to predict an up-market, and improve stability by prior noise; (3) leveraging volatility index to capture the market sentiment of seasoned investors; (4) designing news context learning to fully explore textual information from public news and events; and (5) adopting a rolling strategy for deployment to mitigate noise from uncertainty in the financial market, and generate the prediction with horizon-wise information.

Specifically, our proposed *IndexGAN* contains both a generator and a critic to predict the stock movement. Inside the generator, a news context learning

module extracts latent word representations from the GloVe embedding matrix of news for two purposes. Firstly, it fully explores context information from the news. Secondly, it strategically shrinks the size of the embedding matrix to a comparable extent with price features, considering the size of GloVe embedding matrix is several orders of magnitude larger than market-based features and prior noise, and simply concatenating them together will let the following seq2seq learning entirely concentrate on news and mask the impact of other features. An attentive seq2seq learning module captures temporal dependence from historical prices, volatility index, various technical indicators, prior noise, and latent word representations. The generator with both modules predicts the future sequence. The critic (discriminator in GAN) is designed to approximate the Wasserstein distance between real and generated sequences, which is incorporated into an adversarial loss for optimization. During the training process, besides adversarial loss, two additional losses are added to improve the similarity between real and fake distributions efficiently and avoid biased movement.

Accurate predictions of an index return enable hedging against black swan events and thus can significantly reduce risk and help maintain market liquidity when facing extreme circumstances, thereby contributing to financial market stabilization. *IndexGAN* is implemented on broad indices. To understand the complexity and dynamics of the real-world market index, we select a period of high volatility that covers the 2009 financial crisis and the subsequent recovery for training and testing of *IndexGAN*. Experiment results show that the model boosts the accuracy from around 58% to over 60% for both indices (Dow Jones Industrial Average (DJIA) and S&P 500) compared to the previous state-of-the-art model. This improvement demonstrates the effectiveness of the proposed method. The major contributions of this work are summarized as follows:

- To the best of our knowledge, ours is a first attempt at formulating stock movement predictions into a Wasserstein generative adversarial networks framework for multi-step prediction. The new formulation enhances the robustness by prior noise and provides reasonable guidance on market timing to investors through the use of multi-step prediction.
- *IndexGAN* consists of particularly crafted designs that consider the inherent characteristics of the stock market. It includes a news context learning module, features of the volatility index and multiple empirically valid technical indicators, and a rolling deployment strategy to mitigate uncertainty and generate predictions with horizon-wise information.
- We conduct extensive experiments on two stock indices that attract the most global attention. *IndexGAN* achieves consistent and significant improvements over multiple popular benchmarks.

4.2 Preliminary on Generative Adversarial Nets

The general form of GAN consists of two networks. The generator $\mathcal{G}$ learns the distribution of the real sequence $\boldsymbol{y}_t$. Specifically, a random noise sample z, together with stock features $\boldsymbol{x}_t$ as the condition to control the mode of generated sequence, are fed into generator $\mathcal{G}$ to build a synthetic sequence $\hat{\boldsymbol{y}}_t$. The critic $\mathcal{C}$, also known as the discriminator, is an appropriate measure of distance. In this paper, we choose the Wasserstein distance from WGAN [101] to measure the difference between the real and generated data distribution from the perspective of the optimal transportation problem.

Intuitively, Wasserstein distance indicates the cost or mass caused by transporting $\boldsymbol{y}$ to $\hat{\boldsymbol{y}}$ [102]. As the infimum in Wasserstein distance $W(P_r, P_g)$ is highly intractable, Kantorovich-Rubinstein duality is used to reconstruct the WGAN value function in critic and solve the optimization problem in practice:

$$\min_{\mathcal{G}} \max_{\mathcal{C}\in\mathcal{F}} \mathbb{E}_{\boldsymbol{y}\sim P_r}[\mathcal{C}(\boldsymbol{y})] - \mathbb{E}_{\hat{\boldsymbol{y}}\sim P_g}[\mathcal{C}(\hat{\boldsymbol{y}})] \tag{4.1}$$

where P_g and P_r are implicitly defined by $\hat{\boldsymbol{y}}$ and $\boldsymbol{y}$ respectively, $\mathcal{F}$ is the set of 1-Lipschitz functions. During the training process, the generator $\mathcal{G}$ and critic $\mathcal{C}$ are optimized simultaneously. The generator is responsible for minimizing the value function in critic, which is equivalent to minimizing $W(P_r, P_g)$, so that the generated sequence can be enforced to be as close as possible to the real sequence. The critic aims to find the optimal value function to better approximate $W(P_r, P_g)$. In the inference phase, generator $\mathcal{G}$ produces a fake sequence from the random noise $\boldsymbol{z}$ and condition $\boldsymbol{x}$.

4.3 Problem Formulation

In this section, we aim to develop a model that accurately predicts whether a stock index is up or down the next day. If the close price of the next day is higher than that of the current day, we have an up day and vice versa. We select two different types of features: market-based features read derived from the stock market data, and news-based features derived from raw news headlines. For market-based features, we retrieve the open, high, low, and close prices of the stock index of trading days, deliberately select empirically useful technical indicators, including RSI, MACD DIFF, Bollinger band, EMA, and SMA, and calculate their values based on the price data. In addition, we employ the market volatility data, i.e., VIX, if we predict S&P 500 index and VXD if we predict Dow Jones Index.

The formulation of input features is similar to that one in Chapter 3. Beside $\mathcal{S} = \{\mathbf{s}_t\}_{t=1}^{n} \in \mathbb{R}^{n \times p}$ for the stock dataset with p market-based features and $\mathcal{H} = \{\mathbf{h}_t\}_{t=1}^{n} \in \mathbb{R}^{n \times k}$ for textual dataset, we add the prior noise z as one of the inputs which is assumed to be drawn from Gaussian distribution. Suppose the current time is day T, given the training data $\mathcal{D} = \{(\mathbf{s}_t, \mathbf{h}_t, z)\}_{t=1}^{n}$, we aim to learn a generative model that takes $X = \{(\mathbf{s}_t, \mathbf{h}_t, z)\}_{t=T-w+1}^{T}$ as input and generates $\hat{\mathbf{c}} = \{\hat{c}_t\}_{t=T+1}^{T+H}$,

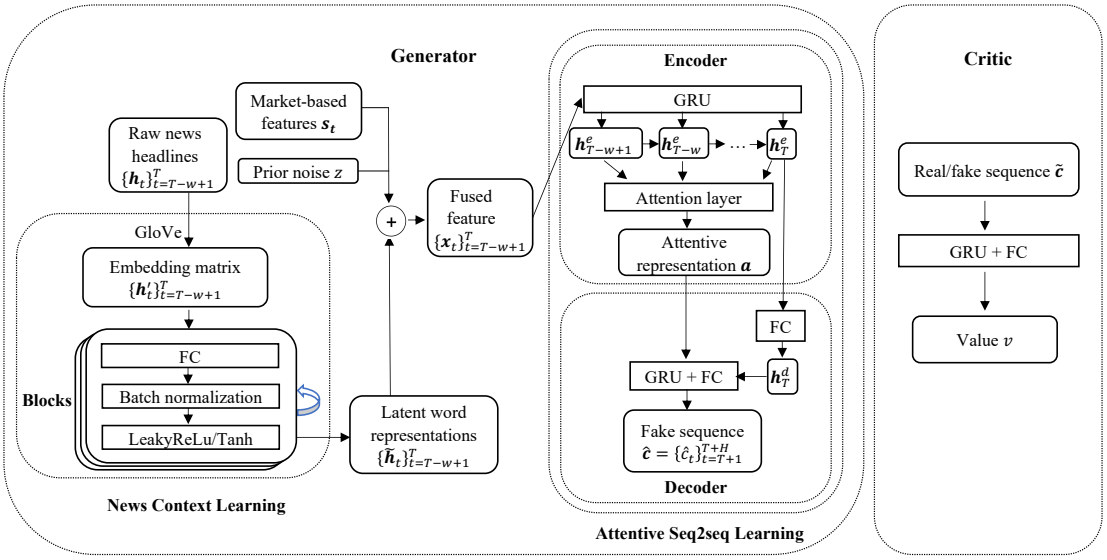


Figure 4.1 The architecture of *IndexGAN*.

where $\hat{c}_t$ is the predicted return based on close price in the future H steps. From the generated sequence, we can decide whether day $T+1$ is up or down.

4.4 Methodologies

4.4.1 Generator

Generator $\mathcal{G}$ with parameter set $\Theta_{\mathcal{G}}$ takes input X and aims to build a fake sequence $\hat{\mathbf{c}}$ whose distribution is as close as possible to the real sequence $\mathbf{c} = \{c_t\}_{t=T+1}^{T+H}$. Our generator has two parts: news context learning and attentive seq2seq learning, as shown in Figure 4.1.

News Context Learning To convert the text from news into word vectors, word2vec is applied to encode the textual input as a continuous distributed representation. GloVe, one of the most popular word representation techniques in word2vec, is chosen to be employed in our model. Compared with general word2vec models, GloVe not only considers words and related local context information but also incorporates aggregated global word co-occurrence statistics from the corpus to

create an embedding matrix. Given a paragraph $\mathbf{h}_t$ at time step t consisting of k news headlines, each word in the headlines is mapped to a vector in m dimensions. The process is formulated as:

$$\mathbf{h}^{'}{}_t = f_{GloVe}(\mathbf{h}_t) \tag{4.2}$$

where f_{GloVe} is the mapping function from GloVe to word vectors, $\mathbf{h}^{'}{}_t \in \mathbb{R}^{k \times l \times m}$ represents the word embedding matrix, and l is the maximum length of headlines in the textual dataset $\mathcal{H}$. Since the length of words in different headlines may be different, we add zero paddings to the end of word vectors from GloVe if the length is shorter than l.

Then, a block of nonlinear layers for the embedding matrix is designed. The first component in a block is a fully connected layer. Next, batch normalization is applied to accelerate the training process by normalizing the features in the embedding matrix. In addition, since batch normalization is shown to be effective in preventing the mode collapse problem, we apply it to all blocks except the last one. The third layer is LeakyReLU activation to avoid saturation in the learning process and dying neurons, except that the last block uses tanh activation function for the output layer. We feed the embedding matrix $\mathbf{h}^{'}{}_t$ into a number of m successive blocks formulated as follows:

$$\begin{aligned}
\mathbf{h}_t^1 &= \phi(f_{BN}(\mathbf{W}_{b1}\mathbf{h}^{'}{}_t + \mathbf{b}_{b1})) \\
\mathbf{h}_t^2 &= \phi(f_{BN}(\mathbf{W}_{b2}\mathbf{h}_t^1 + \mathbf{b}_{b2})) \\
&\ldots \\
\tilde{\mathbf{h}}_t &= tanh(\mathbf{W}_{bm}\mathbf{h}_t^{m-1} + \mathbf{b}_{bm})
\end{aligned} \tag{4.3}$$

where $\mathbf{h}_t^i, i = 1...m$ is the output from i^{th} block and input for $(i+1)^{th}$ block, $\tilde{\mathbf{h}}_t \in \mathbb{R}^g$ is the latent word representation with g dimensions, ϕ is the LeakyReLU activation function, $\mathbf{W}_{bi}$ and $\mathbf{b}_{bi}$ are parameters in a fully connected layer.

Attentive Seq2seq Learning Stock data shows seasonality. For example, a high open price on Monday morning frequently ends up with an up day on Friday. To adaptively capture the temporal relationship and generate sequences, we employ an encoder-decoder with attentive mechanisms as sequence-to-sequence learning. To improve the robustness of prediction, we add prior noise as an input feature to simulate the stochasticity nature of stock prices. In addition, among market-based features, Exponential Moving Average (EMA) and Simple Moving Average (SMA) over different periods of time (5, 13, 21, 50, and 200 days) are used to capture temporal information across multi-scale time spans.

The encoder extracts the attentive variable $\mathbf{a}$. Its input is the sequence of a concatenation of features $\mathbf{x}_t$ containing prior noise z, market-based feature $\mathbf{s}_t$, and latent word representation $\tilde{\mathbf{h}}_t$. We adopt the simple yet effective GRU to obtain dynamic changes from historical sequence $\{\mathbf{x}_t\}_{t=T-w+1}^{T}$, where f_e represents a GRU layer, and $\mathbf{h}_t^e$ denotes the hidden state at each time step.

$$\mathbf{x}_t = [z, \mathbf{s}_t, \tilde{\mathbf{h}}_t], \quad \mathbf{h}_t^e = f_e(\mathbf{x}_t, \mathbf{h}_{t-1}^e) \tag{4.4}$$

To adaptively learn the different contributions of input features in the different historical time steps, we implement an attention layer to aggregate information of the

hidden state from the GRU:

$$a_t = \mathbf{v}^\top tanh(\mathbf{W}_a \mathbf{h}_t^e + \mathbf{b}_a) \tag{4.5}$$

$$\tilde{a}_t = \frac{exp(a_t)}{\sum_{t=T-w+1}^{T} exp(a_t)} \tag{4.6}$$

$$\mathbf{a} = \sum_{t=T-w+1}^{T} \tilde{a}_t \mathbf{h}_t^e \tag{4.7}$$

where $\tilde{a}_t$ denotes contribution score for hidden state $\mathbf{h}_t^e$ at time t, $\mathbf{v}$, $\mathbf{W}_a$ and $\mathbf{b}_a$ are parameters, and $\mathbf{a}$ is the attentive representation which encloses periodical information of the entire sequence.

The decoder aims to extract patterns from the attentive representation $\mathbf{a}$ and generate the fake sequence $\hat{\mathbf{c}} \in \mathbb{R}^q$ in the future q time steps. We also adopt GRU as the decoder. The hidden state in the decoder $\mathbf{h}_T^d$ is transformed from the last hidden state in the encoder $\mathbf{h}_T^e$ by a fully connected layer. Then the attentive representation $\mathbf{a}$ is presented as input to the GRU. The process is computed as follows:

$$\begin{aligned} \mathbf{h}_T^d &= tanh(\mathbf{W}_d \mathbf{h}_T^e + \mathbf{b}_d) \\ \hat{\mathbf{c}} &= f_{fc}(f_d(\mathbf{a}, \mathbf{h}_T^d)) \end{aligned} \tag{4.8}$$

where f_d denotes the GRU layer in decoder, $\mathbf{W}_d$ and $\mathbf{b}_d$ are weight and bias parameters, and f_{fc} represents the prediction layer to generate fake sequence $\hat{\mathbf{c}}$.

For training purposes, the next step is to feed both fake and real sequences into the critic for further optimization.

4.4.2 Critic

The critic is designed to approximate Wasserstein distance $W(P_r, P_g)$ between real data distribution P_r and generated data distribution P_g:

$$W(P_r, P_g) = \inf_{\gamma \in \Pi(P_r, P_g)} \mathbb{E}_{(x,y)\sim\gamma}[\|x - y\|] \tag{4.9}$$

where $\Pi(P_r, P_g)$ denotes the set of all joint distribution $\gamma(x, y)$ whose marginals are P_r and P_g, respectively. The joint distribution $\gamma(x, y)$ represents the mass caused by transporting x to y when transforming P_r to P_g. Therefore, Wasserstein distance indicates the cost of such an optimal transportation problem in intuition [102, 101].

Let $\tilde{\mathbf{c}}$ denote the input sequence for either real or fake data. We use GRU to temporally capture the dependency and a fully connected layer to pass the last hidden state, which is presented as:

$$v = \mathcal{C}(\tilde{\mathbf{c}}; \Theta_{\mathcal{C}}) = f_v(f_{GRU}(\tilde{\mathbf{c}})) \tag{4.10}$$

where $\mathcal{C}$ denotes the critic consisting of the GRU layer f_{GRU} and the fully connected layer f_v, $\Theta_{\mathcal{C}}$ represents the set of parameters in $\mathcal{C}$, and the output v is the approximated distance value.

4.4.3 Optimization

Relying solely on adversarial feedback is insufficient for the generator to capture the conditional distribution of data. To improve the similarity between two distributions more efficiently, we introduce two additional losses on the generator to discipline the learning. Firstly, the supervised loss $\mathcal{L}_S$ is to directly describe the $L1$ distance

between $\mathbf{c}$ and $\hat{\mathbf{c}}$:

$$\mathcal{L}_S = \mathbb{E}_{\hat{\mathbf{c}}\sim P_g, \mathbf{c}\sim P_r}[\| \mathbf{c} - \hat{\mathbf{c}} \|] \tag{4.11}$$

Secondly, from the perspective of stock movement, the long-term stock market shows an increasing trend; hence, the generator tends to construct sequences of higher return even with the detection of short-term fluctuations, which leads to the abnormally high false positive ratio in the classification performance. To address this problem, we design a weighted loss $\mathcal{L}_W$ which represents the error rate with more focus on False Positive (FP) samples than False Negative (FN) samples:

$$\begin{gathered}\mathcal{L}_W = \frac{1}{q}(\lambda_1 \sum_{t=1}^{q} I(y_t < 0, \hat{y}_t > 0) + \lambda_2 \sum_{i=1}^{q} I(y_t > 0, \hat{y}_t < 0)), \\ \lambda_1 > \lambda_2, \lambda_1 + \lambda_2 = 1, \lambda_1 \in (0,1), \lambda_2 \in (0,1)\end{gathered} \tag{4.12}$$

where $I(\cdot)$ is the indicator function, the movement $y_t = sign(c_t)$ is derived from the return, similar to the predicted movement $\hat{y}_t = sign(\hat{c}_t)$, λ_1 and λ_2 are hyper-parameters to balance the error, the constraint $\lambda_1 > \lambda_2$ guarantees more penalty on FP ratio.

By integrating with the additional losses in adversarial feedback, the generator and critic networks are trained simultaneously as follows:

$$\mathcal{L}_\mathcal{G} = -\mathbb{E}_{\hat{\mathbf{c}}\sim P_g}[\mathcal{C}(\hat{\mathbf{c}})] + \gamma_1 \mathcal{L}_S + \gamma_2 \mathcal{L}_W \tag{4.13}$$

$$\mathcal{L}_\mathcal{C} = -\mathbb{E}_{\mathbf{c}\sim P_r}[\mathcal{C}(\mathbf{c})] + \mathbb{E}_{\hat{\mathbf{c}}\sim P_g}[\mathcal{C}(\hat{\mathbf{c}})] \tag{4.14}$$

where $\mathcal{L}_\mathcal{G}$ and $\mathcal{L}_\mathcal{C}$ are the objective functions for $\mathcal{G}$ and $\mathcal{C}$ respectively, $\gamma_1 \geq 0$ and $\gamma_2 \geq 0$ are penalty parameters on supervised loss and weighted loss, respectively.

To guarantee the Lipschitz constraint on the critic, it's necessary to force the weights of the critic to lie in a compact space. In the training process, we clip the weights to a fixed interval $[-\alpha, \alpha]$ after each gradient update.

4.4.4 Training process of *IndexGAN*

Algorithm 1 details the learning process for the generator $\mathcal{G}$ and critic $\mathcal{C}$ in the *IndexGAN*. Denote $\mathcal{D}_{train}$ as the training data containing input features $\mathbf{X}$ and real sequence $\mathbf{c}$, and denote $\mathcal{D}_{batch}$ as a subset of $\mathcal{D}_{train}$ for the training in each batch. We first initialize $\mathcal{G}$ and $\mathcal{C}$. In each training iteration, the prior noise z sampled from Gaussian distribution and input features are fed into $\mathcal{G}$ to generate a predicted sequence $\hat{\mathbf{c}}$, then the critic estimates the Wasserstein distance between real and fake sequences. We compute the gradient from critic loss $\mathcal{L}_{\mathcal{C}}$ and update parameters $\Theta_{\mathcal{C}}$ of $\mathcal{C}$ (line 8-9), also clip all weights to accelerate them till optimality. We keep training the critic for n_{critic} times, then we focus on generator training. The reason for this iterative process is that the gradient of Wasserstein becomes reliable as the number of iterations for the critic increases. Usually, n_{critic} is set to 5. For generator training, predicted sequences are generated by $\mathcal{G}$ and fed into $\mathcal{C}$. Then $\mathcal{L}_{\mathcal{G}}$ is calculated by Equations (4.12)-(4.14) and the parameter $\Theta_{\mathcal{G}}$ of the generator is updated (line 15-16). The number of epochs is set to 80. After the training in an adversarial manner, both $\mathcal{G}$ and $\mathcal{C}$ with well-trained parameters are converged and used for deployment to real-world data.

4.4.5 Deployment

Once the training is completed, the model is ready to be deployed. The generated sequence $\hat{\boldsymbol{y}}$ from our proposed architecture covers H steps ahead, i.e., we have H future prices predicted at different times. To fully utilize this horizon information, we propose a *horizon-wise strategy* shown in Figure 4.2. Specifically, to predict the target at time t, the GAN model is executed for H rounds. For each round $j \in$

Algorithm 1 *IndexGAN* Training Process

Input: training data $\mathcal{D}_{train}$, Generator $\mathcal{G}$, Critic $\mathcal{C}$
Output: well-trained models $\mathcal{G}$ and $\mathcal{C}$ with optimal parameters
Parameter: model parameters: $\Theta_{\mathcal{G}}$, $\Theta_{\mathcal{C}}$, penalty γ_1, γ_2, balancing parameters λ_1, λ_2, clipping threshold α, the number of epochs K, the number of iterations of critic per generator iteration n_{critic}

Initialize $\mathcal{G}$, $\mathcal{C}$, $i \leftarrow 0$.
for $epoch \in \{1, \ldots, n_{epochs}\}$ **do**
  for $\mathcal{D}_{batch} \leftarrow \mathcal{D}_{train}$ **do**
    $\mathbf{X}, \mathbf{c} \leftarrow \mathcal{D}_{batch}$.
    Sample a batch of z from Gaussian.
    $\hat{\mathbf{c}} = \mathcal{G}(\mathbf{X}, z; \Theta_{\mathcal{G}})$.
    Compute $\mathcal{C}(\hat{\mathbf{c}}; \Theta_{\mathcal{C}})$ and $\mathcal{C}(\mathbf{c}; \Theta_{\mathcal{C}})$.
    Compute $\mathcal{L}_{\mathcal{C}}$ by Equation 4.14.
    Update $\Theta_{\mathcal{C}}$ in $\mathcal{C}$ by minimizing $\mathcal{L}_{\mathcal{C}}$.
    Clip weights of $\Theta_{\mathcal{C}}$ into $[-\alpha, \alpha]$.
    $i \leftarrow i + 1$.
    if $i\%n_{critic} == 0$ **then**
      $\hat{\mathbf{c}} = \mathcal{G}(\mathbf{X}, z; \Theta_{\mathcal{C}})$.
      Compute critic value $\mathcal{C}(\hat{\mathbf{c}}; \Theta_{\mathcal{C}})$.
      Compute $\mathcal{L}_{\mathcal{G}}$ by Equation 4.12-14.
      Update $\Theta_{\mathcal{G}}$ in $\mathcal{G}$ by minimizing $\mathcal{L}_{\mathcal{G}}$.
    end if
  end for
end for

$(1, ..., H)$, the noise vector z and historical features $\boldsymbol{x}_{t-j} = (\boldsymbol{x}_{t-j-w+1}, ..., \boldsymbol{x}_{t-j})$ are fed into generator to generate sequence $\hat{\boldsymbol{y}}_{t-j} = (\hat{y}_{t-j+1}, ..., \hat{y}_{t-j+H}) \in \mathbb{R}^H$. We denote $\hat{y}_t^j = \hat{y}_t$ as prediction at j^{th} position. In this way, we obtain a prediction vector $\tilde{\boldsymbol{y}}_t = (\hat{y}_t^1, ..., \hat{y}_t^H) \in \mathbb{R}^H$ for time step t from H sequences. An example of this prediction vector is marked by the dotted blue line in Figure 4.2. We take the average of the $\tilde{\boldsymbol{y}}_t$ to obtain the final predicted return $\hat{c}_t$ whose direction is the final predicted movement $\hat{y}_t$ at time t. The rolling strategy considers the dynamic change of prediction since the trend of the stock market can be rapidly influenced by the various factors discussed earlier. Meanwhile, the average predicted return mitigates the noise caused by the uncertainty of the financial market. We expect the predicted horizon H to be 5 which can generate the best result considering every week has five trading days, and our model evaluation verifies this conjecture.

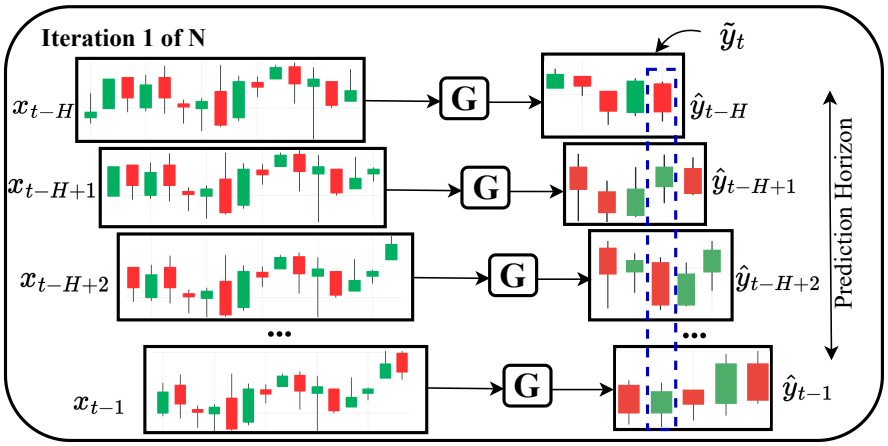


Figure 4.2 Horizon-wise strategy.

4.5 Model Evaluation

To evaluate the performance of the proposed model, extensive experiments are carried out with real-world stock markets and news data. We will first introduce implementation details including data description, baseline models, evaluation metrics, and parameter settings. Then we will show the empirical results of compared models, forecasting horizon analysis, and ablation study to demonstrate the effectiveness of the proposed model, *IndexGAN*. The code is publicly available [1].

4.5.1 Dataset

We choose the same datasets for stock and news as in Chapter 3. Besides, volatility indices of the same period, VIX for S&P 500 and VXD for DJIA, are retrieved from Yahoo Finance. Both price and volatility features s_t are converted to return calculated by $s_t/s_{t-1} - 1$. Both datasets are split into training, validation and testing sets, with same period as in Chapter 3

4.5.2 Baselines

We compare proposed *IndexGAN* with following methods:

- **MA** [103] is one of the most popular indicators used by momentum traders. 5-day EMA is chosen.

[1] https://github.com/JingyiGu/IndexGAN

- **ARIMA** [104] is a widely used statistical model for time series prediction.
- **DA-RNN** [9] integrates the attention mechanism with LSTM to extract input features and relevant hidden states.
- **StockNet** [105] employs Variational Auto-Encoder (VAE) on tweets and historical prices to encode stock movements as probabilistic vectors.
- **AdvLSTM** [10] leverages adversarial training to add simulated perturbations on stock data to improve the robustness of stock prediction.
- **Trans** [12] is the previous state-of-the-art model enhancing Transformer with a trading gap splitter and gaussian prior to predicting stock movement.
- **DCGAN** [106] is a powerful GAN that uses convolutional-transpose layers in the generator and convolutional layers in the discriminator.
- **GAN-FD** [54] uses LSTM and convolutional layers to design a vanilla GAN to predict stock price one step ahead.
- **S-GAN** [63] employs FinBERT on sentiment score and uses vanilla GAN with convolutional layers and GRU to predict price.

Note that MA and ARIMA use close prices only. DA-RNN, Adv-LSTM, Trans, DCGAN, and GAN-FD use market-based prices only, following their architecture designs. StockNet and S-GAN use both market-based features and news data.

4.5.3 Parameter settings

IndexGAN is implemented in PyTorch, with hyperparameters tuned using the validation set. The length of the historical input sequence w is set to 35, and the prediction horizon H is set to 5. RMSprop is used as the optimizer. The remaining hyperparameters are largely consistent across the DJI and SPX datasets. A batch size of 32 is used, with generator and critic learning rates set to 0.0001 and 0.00005, respectively. The GloVe embedding size is fixed at 50, and the weight clipping threshold is set to 0.01. Trade-off parameters λ_1, λ_2, γ_1, γ_2 are set to 0.8, 0.2, 10, 3, respectively. For DJI, training runs for 100 epochs with a latent word vector size of 6 and decoder hidden size of 200, while SPX uses 80 epochs, a latent vector size of 3,

and a decoder hidden size of 50. Both datasets use an encoder hidden size of 100. For other baselines, we follow the default hyperparameter settings for implementation.

4.5.4 Evaluation metrics

We evaluate the performance of all methods by three measures: Accuracy, F1 score, and Matthews Correlation Coefficient (MCC). Accuracy is the proportion of correctly predicted samples in all samples. F1 score incorporates both recall and precision. MCC takes all components of the confusion matrix into consideration, while the true negative is ignored in the F1 score. Therefore, MCC is in a more balanced manner than the F1 score. A low value of MCC close to 0 indicates that classes in the prediction are highly imbalanced.

4.5.5 Experiment results

Comparison Results with Baselines Table 4.1 presents the numerical performance of our *IndexGAN* and baselines on both datasets for trend prediction based on the average and standard deviation of 10 runs, from which we make the following observations. (1) The proposed *IndexGAN* achieves the best results on both indices regarding accuracy and MCC. (2) On both data sets, *IndexGAN* achieves an accuracy of over 60% and MCC of around 0.200, which justifies the predictive power of our model. The previous state-of-the-art model, Trans, achieves the second-best performance but does not outperform *IndexGAN* on MCC. Hence, *IndexGAN* is proven to surpass all baselines, including Trans, on stock movement prediction tasks. In addition, the improvement in MCC by *IndexGAN* is more significant than accuracy, which illustrates that prediction from *IndexGAN* is in a more balanced manner. (3) Among the baselines, there exists an obvious gap between traditional time series models (MA, ARIMA) modeling on close price only and neural networks (DA-RNN, StockNet Adv-LSTM, Trans) modeling on all market-based features, demonstrating the impact of additional features and the

Table 4.1 Performance Comparison of *IndexGAN* and Baselines

Model	DJIA		
	Acc (%)	F1	MCC
EMA	51.83	0.483	0.058
ARIMA	53.15	0.479	0.091
DA-RNN	56.23 ±1.12	0.642 ±0.007	0.104 ±0.021
StockNet	57.90 ±0.84	0.632 ±0.008	0.111 ±0.031
Adv-LSTM	58.12 ±1.05	0.674 ±0.010	0.145 ±0.018
Trans	58.87 ±1.01	0.676 ±0.009	0.179 ±0.027
DCGAN	56.79 ±0.09	0.622 ±0.003	0.082 ±0.014
GAN-FD	57.02 ±1.09	0.607 ±0.011	0.094 ±0.023
S-GAN	57.93 ±1.08	0.632 ±0.010	0.099 ±0.020
IndexGAN	**60.85 ±0.95**	**0.713 ±0.006**	**0.208 ±0.024**

Model	S&P500		
	Acc (%)	F1	MCC
EMA	53.93	0.511	0.087
ARIMA	50.53	0.475	0.085
DA-RNN	56.71 ±1.53	0.513 ±0.010	0.092 ±0.013
StockNet	55.37 ±1.35	0.585 ±0.009	0.091 ±0.019
Adv-LSTM	57.13 ±1.20	0.596 ±0.011	0.143 ±0.022
Trans	58.30 ±1.34	0.610 ±0.011	0.176 ±0.031
DCGAN	55.85 ±1.01	0.495 ±0.008	0.121 ±0.010
GAN-FD	56.93 ±1.15	0.591 ±0.012	0.105 ±0.024
S-GAN	57.42 ±1.07	0.601 ±0.012	0.112 ±0.018
IndexGAN	**60.00 ±1.37**	**0.616 ±0.014**	**0.199 ±0.027**

superior ability of neural networks in capturing complex temporal dynamics for financial data. StockNet uses VAE, a generative model as well, which maps features from high-dimensional space to low-dimensional space and infers the posterior distribution. Its performance is worse than *IndexGAN*. We postulate that the reason is that it cannot learn the posterior distribution well due to the large noise of stock prices. It reveals that GAN performs better in stock prediction tasks. (4) *IndexGAN* surpasses all GAN-based approaches. DCGAN is designed for image generation and exploits convolution layers. Since patterns in time series are not similar to those in images, this architecture fails to capture the temporal dependencies of stock features. Hence, GANs for image generation are not suitable for time series data. Moreover, fake and real sequences fed into the discriminator in GAN-FD are the same except for the last element, which is challenging for the discriminator to differentiate. The low values of MCC on both indices show that GAN-FD does not perform well on movement prediction. Furthermore, S-GAN outperforms other GAN-based baselines due to its sentiment analysis, but the sentiment score from FinBERT in S-GAN falls into three categories (positive, neutral, and negative). The complex pattern of public emotions is hard to capture by such a simple sentiment score. The results of MCC illustrate that Vanilla GAN implemented in S-GAN prevents the high diversity of prediction due to the mode collapse problem. Therefore, S-GAN falls behind *IndexGAN*. The proposed structure of *IndexGAN* achieves twice of MCC increase, in general, because it implements the concept of WGAN and multi-step prediction with horizon-wise information.

Forecasting Horizon Analysis We investigate the effects of the prediction horizon H on the proposed *IndexGAN*. Figure 4.3 shows the change of accuracy and MCC on DJIA and SPX. As H goes up from 2 to 5, the performance of the model shows a rising trend, since the critic can differentiate real and fake sequences as their length becomes

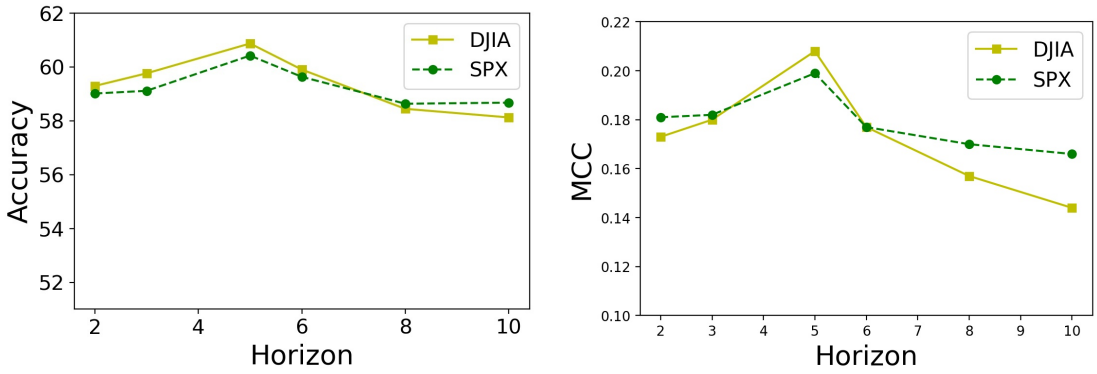


Figure 4.3 Forecasting horizon analysis.

longer. Both accuracy and MCC achieve the highest value when H is 5. But as the length of the predicted sequence exceeds 5 days and continues increasing to 10 days, the quality of the model degrades. This reveals that it becomes harder to capture movement patterns if the forecasting length is too long. Hence, there exists a trade-off between the uncertainty of the stock market and the ability of GAN. Choosing the appropriate prediction horizon is critical to achieving significant performance. The experiment conducted on various lengths justifies our conjecture that 5 days forward is most effective because it is a normal and popular short trading period.

Ablation Study We conduct experiments on ablation variants of *IndexGAN* to fully examine the predictive power gained from each component:

- ***IndexGAN*-Block**: Blocks in news context learning are removed. The GloVe embedding matrix for each headline is flattened and concatenated with price features as input to be fed into seq2seq learning.
- ***IndexGAN*-News**: News headlines are removed from input features. The noise and market-based features are concatenated together and fed into attentive seq2seq learning directly. Thus, news context learning is removed as well.
- ***IndexGAN*-FC**: The blocks are replaced by a fully connected layer. In the news context learning, the news is converted into embeddings by GloVe, then the latent word representation $\tilde{\mathbf{h}}_t$ with dimension g is generated by the fully connected layer.
- ***IndexGAN*-Vol**: Volatility index is removed from input features.

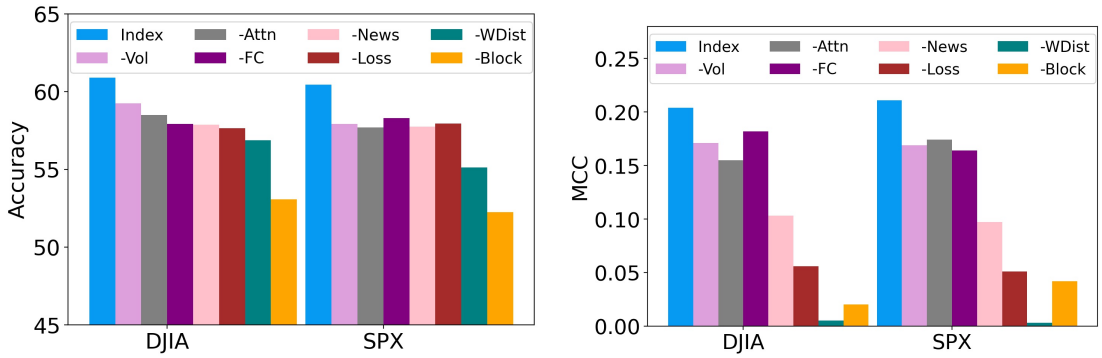


Figure 4.4 Ablation study results on accuracy (top) and MCC (bottom) for comparing variants of *IndexGAN*. For example, -Attn represents *IndexGAN* without attention in seq2seq learning.

- ***IndexGAN*-WDist**: Wasserstain distance is removed and vanilla GAN is implemented.
- ***IndexGAN*-Attn**: The attention layer in seq2seq learning is removed. The output from GRU in the encoder is directly presented as input to the decoder.
- ***IndexGAN*-Loss**: $\mathcal{L}_S$ and $\mathcal{L}_W$ are removed during the training and only adversarial loss is used.

Figure 4.4 exhibits the performance of variants. Firstly, *IndexGAN*-News, *IndexGAN*-Block, and *IndexGAN*-FC are employed to test the importance of news context learning. With market-based features only, the performance of *IndexGAN*-News decreases significantly. It validates that news reflects the information in the stock market and can efficiently boost performance. On the other hand, even with news, the performance of *IndexGAN*-Block is not ideal. It illustrates our notion that directly passing the embedding matrix into the seq2seq model is likely to mask the impact of price. When the embedding size is thousands of times greater than the price, the encoder and decoder concentrate on embedding entirely. This phenomenon reveals that merely depending on the news is challenging, and both price and news features are necessary for stock movement prediction. Compared with a single fully connected layer, the blocks containing nonlinear layers bring approximately a 3% accuracy increase on the S&P 500 and DJIA, and almost 0.05 of MCC increase on the S&P 500. This phenomenon justifies that nonlinear layers in the designed

blocks can boost the predictive power efficiently. With the help of blocks that fully investigate the news context and shrink the size of the embedding matrix gradually, *IndexGAN* achieves the highest results.

Secondly, *IndexGAN*-WDist does not perform well, especially MCC is close to 0 which is even worse than EMA, revealing low diversity of predicted movements and the mode collapse problem of *IndexGAN*-WDist. Simply leveraging Vanilla GAN cannot capture dynamics for movement prediction. It's necessary to employ the concept of Wasserstein distance to measure the similarity of prediction and ground truth, which improves the quality of the model significantly by accelerating the speed of convergence and avoiding mode collapse.

Thirdly, without the volatility index as a feature, the accuracy of Index-Vol on the S&P 500 decreases by 2.5% which is 1% more than the drop in accuracy on DJIA. The results indicate the necessity of a volatility index, especially for VIX which depends on more options than VDX. Moreover, by comparing the *IndexGAN*-Attn and *IndexGAN*-Loss with *IndexGAN*, the attention mechanism and two additional losses on the generator enhance the predictive power by over a 2% accuracy increase.

4.6 Conclusion

In this chapter, we present a new formulation for predicting stock movement. This work is the first to propose the use of Wasserstein Generative Adversarial Networks for multi-step stock movement prediction, which includes particularly crafted designs for the inherent characteristics of the stock market. Specifically, in the generator, *IndexGAN* first exploits prior noise to improve the robustness of the model, implements news context learning to explore embeddings from emerging news, and employs the volatility index as one of the impacting factors. Then, an attentive encoder-decoder architecture is designed for seq2seq learning to capture temporal dynamics and interactions between market-based features and word representations.

The critic is used for approximating the Wasserstein distance between the actual and predicted sequences. Moreover, *IndexGAN* adds two additional losses for the generator to constrain similarity efficiently and avoid biased movement during optimization. Also, to mitigate uncertainty, a rolling strategy for deployment is employed. Experiments conducted on DJIA and S&P 500 show the significant improvement of *IndexGAN* and the effectiveness of multi-step prediction over other state-of-the-art methods. The ablation study verifies the significant advantage of each component in the proposed *IndexGAN* as well.

CHAPTER 5

RAGIC: RISK-AWARE GENERATIVE FRAMEWORK FOR STOCK INTERVAL PREDICTION

5.1 Introduction

As described in previous chapters, existing works mostly focus on *point* prediction, i.e., return one single scalar for each time step. This approach is not informative enough to reflect market conditions driven by complex factors, as it overlooks the uncertainty and randomness inherent in prices. To support risk assessment and informed decision-making, it is essential to provide *interval* prediction [25], i.e., a price interval with lower and upper bounds for each future time step. Both the width of the interval and the position of the bounds themselves imply critical risk information. (1) The interval width reflects the degree of uncertainty: wider intervals suggest higher volatility or less model confidence, while narrower intervals indicate more stable or predictable conditions. (2) These bounds themselves quantify expected price range with a certain confidence level, offering directional insights for risk management. For investors in long positions, the lower bound highlights potential losses in the event of a market downturn, helping to assess downside risk. For short positions, the upper bound captures the risk of price increases that could lead to losses. These position-aware interpretations make interval predictions a powerful tool for anticipating adverse scenarios and enabling more robust, risk-sensitive decisions. Together, they provide a more comprehensive view of potential risk exposure and support prudent capital allocation under uncertainty [1].

Despite its importance, interval prediction in the financial market, and even in other domains, remains significantly underexplored, making it the central focus of this research. Existing methods for interval prediction in time series often rely on strong assumptions of independence, lack generalization capabilities, or have

high computational requirements, making them impractical for the stock market. Moreover, it is challenging to balance accuracy and informativeness, since wider intervals tend to increase coverage accuracy but reduce informativeness [107, 108]. Therefore, developing an interval prediction framework tailored to the unique complexities of financial markets with a balanced trade-off is necessary yet ignored.

Another issue is that most previous works largely neglect market risk, failing to account for the volatility and unpredictable shifts. Accurate detection and early warning of risk enables hedging against black swan events, helps maintain market liquidity facing extreme circumstances, and stabilizes the financial market. The volatility index, derived from the prices of index options with near-term expiration dates, is often seen as a way to gauge market sentiment and evaluate the fluctuation risk.

One illustrative piece of evidence is presented in Figure 5.1. The CBOE Volatility Index (VIX) is a real-time indicator reflecting the market's outlook on the expected magnitude of near-term price fluctuations in the S&P 500 index (SPX). Notably, the substantial decline in SPX during the financial crisis of 2009 and the COVID-19 pandemic aligns with the peak values observed in the VIX during the same periods. This correlation highlights the crucial role of the volatility index in detecting potential risks associated with market fluctuations.

To address the two major issues stemming from the stochastic nature of financial markets, we propose *RAGIC*, a Risk-Aware Generative model for Interval Construction. Unlike the common task of point prediction, this novel framework is tailored for interval prediction. By predicting a price range (lower and upper bounds), we provide a more informative forecast that accounts for the uncertainty and randomness inherent in financial markets while also facilitating point prediction. Meanwhile, it incorporates the volatility index to detect market risk and adaptively adjust the interval width, making the framework more responsive to changing

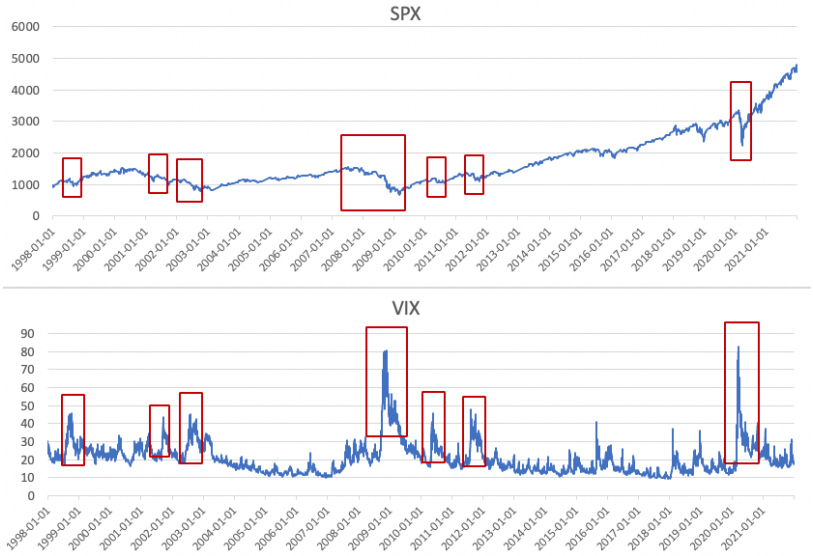


Figure 5.1 Illustration of the relationship between SPX and VIX in 1997-2021. The red areas in SPX show the price drop, and the corresponding red areas in VIX show the peaks.

market conditions and better equipped to handle risks in real time. We choose the generative model because it learns the entire data distribution, allowing it to generate diverse market scenarios, including extreme events, which is crucial for robust stock prediction. The randomness introduced through the latent space influences every layer of the model, ensuring coherent variability. Generative Adversarial Networks (GANs), in particular, are well-suited for this task as they produce sequences with lower diversity, resulting in narrower and more informative intervals—ideal for precise financial forecasting compared to other generative models, such as VAE [13] and diffusion models [109, 110].

Specifically, *RAGIC* consists of two phases: sequence generation and interval construction. In the first phase, we employ a GAN to learn the historical stock features and simulate the future price sequences. The generator incorporates a risk module, which captures risk using a *risk attention score* derived from the volatility index [1], and a temporal module, which captures the multi-scale trend expressed

[1]Please note that the volatility index differs from volatility. Volatility is normally calculated as the square root of variance. But the stock volatility index is derived from multiple option prices of the underlying stock index. For example, VIX is the volatility index of SPX, and VXD is the volatility index of DOW.

by historical prices locally and globally. The well-trained generator can produce an adequate set of future price sequences with artificial randomness learned from the financial market. In the second phase, a *horizon-wise strategy* is designed to gather simulated sequences with different prediction horizons, and statistical inference is utilized to construct a *risk-sensitive interval* to reflect uncertainty, with the interval width adaptively determined by the volatility index. In practice, *RAGIC* is lightweight for its low computational cost and data availability. The GAN model is trained only once and can be repeatedly deployed to produce multiple sequences. Its reliance on public data only further enhances its accessibility. This method not only offers improved prediction reliability in the financial domain but also presents a flexible framework adaptable to other fields involving stochastic variability. Experiment results on multiple broad-based indices validate that our model achieves a balanced accuracy-informativeness manner, consistently providing 95% coverage while maintaining a low width for intervals.

The major contributions are summarized as follows:

- To the best of our knowledge, we make the first attempt to employ a sequence generative model for interval prediction in stock markets, which quantifies uncertainty and provides much richer information regarding future market trends and risks.
- We incorporate the volatility index to capture the risk perceived by smart money and design a *risk-sensitive interval* which adjusts the width of the predicted interval automatically. A wider interval indicates higher market volatility and a potentially bigger need for hedging.
- Extensive experiments on multiple stock indices worldwide illustrate that *RAGIC* achieves consistently over 95% coverage with reasonable interval width in a balanced manner, significantly outperforming multiple popular benchmarks on interval prediction.

5.2 Problem Formulation

The study aims to provide stock interval prediction. The problem is formulated into two phases: sequence generation and interval construction.[2] (1) In the first phase, we aim to develop a model to generate sequences of future stock prices. Let $\boldsymbol{X} = (\boldsymbol{x}_1, ...\boldsymbol{x}_W) \in \mathbb{R}^{W \times K}$ denote the historical stock characteristics with W time steps and K features. The stock features consist of price-based features $\boldsymbol{p}$ and volatility-based features $\boldsymbol{v}$. Let $\boldsymbol{z} \in \mathbb{R}^W$ denote a random noise sample which is assumed to be drawn from a uniform distribution. Given $\boldsymbol{X}$ and $\boldsymbol{z}$, our task is to learn a generative adversarial model f with parameter Θ to predict the sequence of close price $\boldsymbol{y} = (y_1, ..., y_H) \in \mathbb{R}^H$ in the future H steps. (2) In the second phase, the well-trained generative model can produce a set of future price sequences $\hat{\boldsymbol{Y}} = (\hat{\boldsymbol{y}}_1, ..., \hat{\boldsymbol{y}}_N) \in \mathbb{R}^{N \times H}$ by repeatedly feeding stock features and different noise for N iterations:

$$\hat{\boldsymbol{y}}_n = f(\boldsymbol{X}, \boldsymbol{z}_n; \Theta), n \in 1, ..., N \tag{5.1}$$

where $\boldsymbol{z}_n$ is drawn from distribution randomly and independently for n^{th} iteration. Given the generated sequences $\hat{\boldsymbol{Y}}$, the goal is to build a prediction interval in the form of $[Y_t^L, Y_t^U]$ such that it accurately covers future actual price y_t, where Y_t^L and Y_t^U are lower and upper bounds, respectively. A prediction interval with a narrow width $Y_t^U - Y_t^L$ and high accuracy in covering the actual price is desired.

5.3 Sequence Generation

The general form of GAN consists of two networks. The generator $\mathcal{G}$ learns real sequence $\boldsymbol{y}_t$. Specifically, a random noise sample $\boldsymbol{z}$, together with stock features $\boldsymbol{X}_t$ as the condition to control the mode of generated sequence, are fed into generator $\mathcal{G}$

[2]For notations, we use bold letters for vectors or matrices and plain letters for scalars, if not specified otherwise.

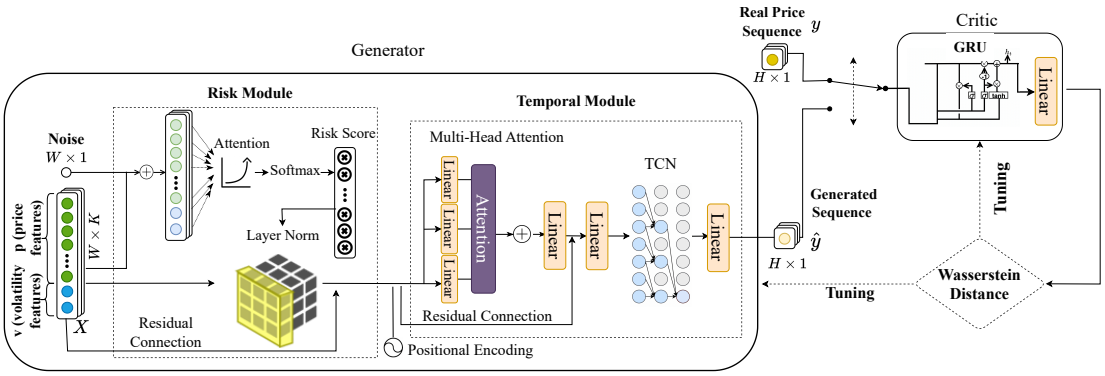


Figure 5.2 The overall architecture of GAN in *RAGIC* during the training process.

to build a synthetic sequence $\hat{\boldsymbol{y}}_t$. The critic $\mathcal{C}$, also known as the discriminator, is an appropriate measure of distance. In this study, we choose Wasserstein distance from WGAN [101] to measure the difference between the real and generated data from the perspective of the optimal transportation problem [101]. Intuitively, it indicates the cost or mass caused by transporting $\boldsymbol{y}$ to $\hat{\boldsymbol{y}}$ [102]. In practice, Kantorovich-Rubinstein duality is used to reconstruct the WGAN value function in critic and solve the optimization problem:

$$\min_{\mathcal{G}} \max_{\mathcal{C} \in \mathcal{F}} \mathbb{E}_{\boldsymbol{y} \sim P_r}[\mathcal{C}(\boldsymbol{y})] - \mathbb{E}_{\hat{\boldsymbol{y}} \sim P_g}[\mathcal{C}(\hat{\boldsymbol{y}})] \tag{5.2}$$

where P_g and P_r are implicitly defined by $\hat{\boldsymbol{y}}$ and $\boldsymbol{y}$ respectively, $\mathcal{F}$ is the set of 1-Lipschitz functions. The generator $\mathcal{G}$ and critic $\mathcal{C}$ are trained simultaneously. The generated sequence from the generator is enforced to be as close as possible to the real sequence. The critic aims to find the optimal value function to better approximate the Wasserstein Distance. In the inference phase, generator $\mathcal{G}$ produces a fake sequence from the random noise $\boldsymbol{z}$ and condition $\boldsymbol{x}$.

5.3.1 Generator

Risk Module Market volatility implies investment risk. A higher volatility index indicates a declining market with a higher probability, while a lower one may imply a rising market. To capture the risk perceived by smart money, we design a risk score to enhance the features. For each feature $\boldsymbol{x}_k \in \mathbb{R}^W, k \in (1, ...K)$, a *risk attention score* $\boldsymbol{\beta}_k \in \mathbb{R}^W$ is computed:

$$\boldsymbol{\beta}_k = \begin{cases} \exp\left(\boldsymbol{\lambda}_k \max(0, \boldsymbol{x}_k - \boldsymbol{\delta}_k)\right) & \boldsymbol{x}_k \in \boldsymbol{v} \\ 1 & \boldsymbol{x}_k \notin \boldsymbol{v} \end{cases} \tag{5.3}$$

For volatility-based features $\boldsymbol{v}$, if its value exceeds the risk threshold $\boldsymbol{\delta}_k \in \mathbb{R}^W$, then the current market can be seen as with higher risk and the corresponding *risk attention score* $\boldsymbol{\beta_k}$ will grow exponentially. The coefficient $\boldsymbol{\lambda}_k \in \mathbb{R}^W$ controls the growth rate. Otherwise, $\boldsymbol{\beta}_k$ is set to 1, the same as those of price-based features $\boldsymbol{p}$. Then we normalize the *risk attention score* by $\boldsymbol{r}_k = softmax(\boldsymbol{\beta}_k)$, assign it to corresponding feature $\boldsymbol{x}_k$ for combination. Additionally, we adopt layer normalization [111] for faster training, followed by residual connections to produce risk-enhanced features $\tilde{\boldsymbol{x}_k} \in \mathbb{R}^W$:

$$\tilde{\boldsymbol{x}}_k = \boldsymbol{x}_k + LayerNorm(\boldsymbol{r}_k \odot \boldsymbol{x}_k) \tag{5.4}$$

where $\odot$ is element-wise multiplication. The output of risk module is $\tilde{\boldsymbol{X}} = (\tilde{\boldsymbol{x}}_1, ..., \tilde{\boldsymbol{x}}_k) \in \mathbb{R}^{W \times K}$, which is obtained by concatenating all enhanced features together.

Temporal Module To adaptively capture the temporal dependency and seasonal patterns inherent in stock market data, the Temporal Module integrates two comple-

mentary techniques: multi-head attention and Temporal Convolutional Networks (TCNs). Multi-head attention is employed to model long-term, global dependencies across time steps, while TCNs focus on capturing local, short-term, and periodic patterns. This combination ensures robust and comprehensive temporal representation.

Global and Long-Term Interrelationships: We employ multi-head attention to learn the interrelationship between different historical time steps at a global scale. By dynamically weighting the impact of each time step, this mechanism enables the model to identify the most influential key patterns that influence future stock movements. The risk-enhanced features $\tilde{\boldsymbol{X}}$ is firstly added by positional encoding [112] $\tilde{\boldsymbol{X}} = \tilde{\boldsymbol{X}} + PositionalEncoding(\tilde{\boldsymbol{X}})$, then it is linearly mapped into query $\boldsymbol{Q}_d = \tilde{\boldsymbol{X}}\boldsymbol{W}_d^q \in \mathbb{R}^{W\times d_k}$, key $\boldsymbol{K}_d = \tilde{\boldsymbol{X}}\boldsymbol{W}_d^k \in \mathbb{R}^{W\times d_k}$ and value $\boldsymbol{V}_d = \tilde{\boldsymbol{X}}\boldsymbol{W}_d^v \in \mathbb{R}^{W\times d_k}$ in each head, where $d \in (1, .., D)$ is the head index, d_k is hidden dimension, $\boldsymbol{W}_d^q$, $\boldsymbol{W}_d^k$, and $\boldsymbol{W}_d^v$ are weight parameters. Attention score $\boldsymbol{\phi}_h \in \mathbb{R}^{W\times W}$ and the output $\boldsymbol{O}_d \in \mathbb{R}^{W\times d_k}$ from d^{th} head can be calculated:

$$\boldsymbol{O}_d = softmax(\boldsymbol{\phi}_d)\boldsymbol{V}_d, \quad \boldsymbol{\phi}_d = \frac{\boldsymbol{Q}_d\boldsymbol{K}_d^\top}{\sqrt{d_k}} \tag{5.5}$$

Each element of the attention score $\boldsymbol{\phi}_{d,(i,j)}$ represents the interrelationship of features between time step w_i and w_j. We concatenate the outputs from all the heads together, then employ a linear mapping layer and a residual connection to generate final output $\boldsymbol{S} = \tilde{\boldsymbol{X}} + [\boldsymbol{O}_1, ..., \boldsymbol{O}_d]\boldsymbol{W} \in \mathbb{R}^{W\times K}$.

Local and Periodical Dependencies: Moreover, stock data exhibits seasonal patterns across various timeframes, such as weekly, monthly, and quarterly cycles. For instance, historical studies have shown that weekly data reveals lower S&P 500 returns on Mondays compared to Wednesdays [92]. Similarly, monthly trends further

reveal increased volatility and volume around the expiration of options contracts on the third Friday [113]. Additionally, quarterly earnings seasons and U.S. Federal Reserve Federal Open Market Committee (FOMC) meetings contribute significantly to increased market volatility [114]. To capture these seasonal and local trends, we adopt dilated convolution layers in TCN [115] which employs a 1D fully convolutional network and causal convolutions to capture fine-grained local dependencies, ensuring that the model is sensitive to the shorter-term cyclical behavior inherent in financial data, thus complementing the global focus provided by the multi-head attention mechanism. Specifically, the dilated convolution representation $\tilde{\boldsymbol{s}}_w^l$ at the w^{th} time step and l^{th} layer is calculated from the input $\boldsymbol{s}$:

$$\tilde{\boldsymbol{s}}_w^{(l)} = \boldsymbol{f} \circledast (\tilde{\boldsymbol{s}}_w^{(l-1)}, ..., \tilde{\boldsymbol{s}}_{w-\eta p}^{(l-1)}) \tag{5.6}$$

$$\tilde{\boldsymbol{s}}_w^{(1)} = \boldsymbol{f} \circledast (\boldsymbol{s}_w, ..., \boldsymbol{s}_{w-\eta p}) \tag{5.7}$$

where $l \in (1, ..., L)$ is the layer index, $\boldsymbol{f} \in \mathbb{R}^p$ is the kernel with size p, $\circledast$ is the convolution operation between kernel and sequence element, $\eta = 2^{l-1}$ is the dilation rate. When $\eta = 1$, it is equivalent to a regular convolution layer. As the dilation rate grows in the following layer exponentially, the hidden output at the top layers receives a wider range of historical information. This scheme is able to learn longer periodical patterns without information loss. After obtaining the hidden output from the top layer, we adopt a fully connected layer to generate a fake sequence $\hat{\boldsymbol{y}} \in \mathbb{R}^H$.

5.3.2 Critic

The critic is designed to approximate the Wasserstein distance between real and generated sequences, which is used for optimization during the training. Let $\boldsymbol{y}'$ denote either a real or fake sequence as the input to the critic. We use Gated Recurrent

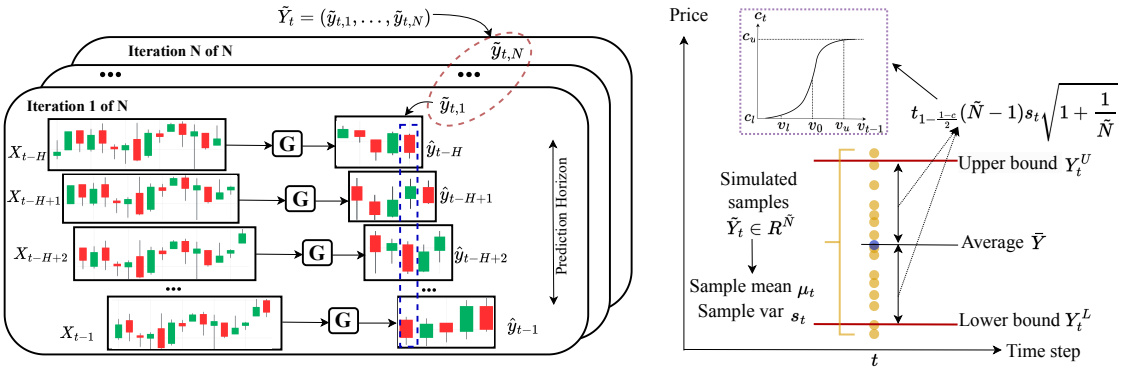


Figure 5.3 Sequence simulation (left) and *risk-sensitive interval* construction (right).

Units (GRU), which is a simple yet effective structure, to temporally capture the dependency and a fully connected layer to pass the last hidden state, and obtain the approximated distance value $c = \mathcal{C}(\boldsymbol{y}'; \Theta_c) = FC(GRU(\boldsymbol{y}'))$. Θ_c represents the set of parameters in $\mathcal{C}$.

5.3.3 Optimization

Solely relying on adversarial feedback is insufficient for the generator to capture the conditional distribution of data. To improve the similarity between two distributions more efficiently, we introduce an additional loss on the generator to discipline the learning. Specifically, the supervised loss $\mathcal{L}_S$ is to directly describe the $L1$ distance between $\boldsymbol{y}$ and $\hat{\boldsymbol{y}}$. By integrating the supervised loss with adversarial feedback, the objective functions for the generator and critic networks are presented:

$$\mathcal{L}_\mathcal{G} = -\mathbb{E}_{\hat{\boldsymbol{y}} \sim P_g}[\mathcal{C}(\hat{\boldsymbol{y}})] + \gamma \mathcal{L}_S \tag{5.8}$$

$$\mathcal{L}_S = \mathbb{E}_{\hat{\boldsymbol{y}} \sim P_g, \boldsymbol{y} \sim P_r}[\| \boldsymbol{y} - \hat{\boldsymbol{y}} \|] \tag{5.9}$$

$$\mathcal{L}_\mathcal{C} = -\mathbb{E}_{\boldsymbol{y} \sim P_r}[\mathcal{C}(\boldsymbol{y})] + \mathbb{E}_{\hat{\boldsymbol{y}} \sim P_g}[\mathcal{C}(\hat{\boldsymbol{y}})] \tag{5.10}$$

where $\gamma \geq 0$ is the penalty parameter on supervised loss.

To guarantee the Lipschitz constraint on Critic, it's necessary to force the weights of the critic to lie in a compact space. In the training process, we follow the WGAN implementation [101] to clip the weights to a fixed interval $[-\xi, \xi]$ after each gradient update, which effectively avoids vanishing or exploding gradients and ensures stable training.

5.4 Interval Construction

The interval construction phase leverages the power of GANs to produce diverse price trends and account for uncertainty in financial markets. Once trained, the GAN model can be deployed repeatedly to generate multiple sequences that can be utilized to construct informative intervals. This phase consists of two key innovations, *horizon-wise* sequence simulation and *risk-sensitive interval* construction. This method not only enhances predictive accuracy in finance but also offers broader implications for other domains where uncertainty and stochastic variability play a critical role.

5.4.1 Sequence simulation

The generated sequence $\hat{\boldsymbol{y}}$ from the generator covers H steps ahead, providing a forward-looking perspective on price movements across different time horizons. To fully utilize this horizon information, we propose a *horizon-wise strategy*. Specifically, to predict the target at time step t, our generator is executed for H rounds. For each round $j \in (1, ..., H)$, the noise vector $\boldsymbol{z}$ and historical features $\boldsymbol{X}_{t-j} = (bmx_{t-j-W+1}, ..., \boldsymbol{x}_{t-j})$ are fed into generator to generate the sequence $\hat{\boldsymbol{y}}_{t-j} = (\hat{y}_{t-j+1}, ..., \hat{y}_{t-j+H}) \in \mathbb{R}^H$. We denote $\hat{y}_t^j = \hat{y}_t$ as prediction at j^{th} position, as marked by dotted blue lines in Figure 5.3. To predict the interval for a particular day t, we obtain a prediction vector $\tilde{\boldsymbol{y}}_t = (\hat{y}_t^1, ..., \hat{y}_t^H) \in \mathbb{R}^H$ from H sequences.

We repeat the above operation for N times to capture the market randomness by feeding different noises and stock features into the model, producing $\tilde{N} = N \times H$

different predictions for time step t. This diverse set of predictions $\tilde{\boldsymbol{Y}}_t = (\tilde{\boldsymbol{y}}_{t,1}, ..., \tilde{\boldsymbol{y}}_{t,N})$ effectively captures market uncertainty from multiple perspectives and across varying horizons. The horizon-wise strategy not only enables more granular control over future predictions but also generalizes to other fields where multi-step forecasts are crucial.

5.4.2 Risk-sensitive interval

We aim to build a *risk-sensitive prediction interval* to accurately capture the possible price range with a dynamic width determined by the perceived market risk and fluctuation.

First, we utilize statistical inference to estimate the prediction interval. The stock price at an individual time step is assumed to be a random variable Y_t which follows a Gaussian distribution with the unknown population mean μ_t and variance σ_t^2 at this time step, denoted as $Y_t \sim \mathcal{N}(\mu_t, \sigma_t^2)$. Given the simulated set $\tilde{\boldsymbol{Y}}_t$ for time step t, the sample mean $\bar{Y}_t$ and sample variance s_t^2 are calculated as:

$$\bar{Y}_t = \frac{1}{\tilde{N}} \sum_{i=1}^{\tilde{N}} \tilde{y}_{t,i}, \quad s_t^2 = \frac{1}{\tilde{N}-1} \sum_{i=1}^{\tilde{N}} (\tilde{y}_{t,i} - \bar{Y}_t)^2 \tag{5.11}$$

where $\tilde{y}_{t,i}$ is i-th sample in the set, $\tilde{N} - 1$ of s_t^2 is for unbias correction.

Thus, we can infer that $\frac{Y_t - \bar{Y}_t}{s_t \sqrt{1+\frac{1}{\tilde{N}}}} \sim t_{(\tilde{N}-1)}$ empirically. [3] Accordingly, the prediction interval $[Y_t^L,\ Y_t^U]$ with Y_t^L and Y_t^U as lower and upper bound, can be

[3] Note that t in $t_{(\tilde{N}-1)}$ represents the t-distribution; t as subscript such as s_t represents the time step.

calculated:

$$Y_t^L = \bar{Y}_t - t_{1-\frac{(1-c)}{2}}(\tilde{N}-1)s_t\sqrt{1+\frac{1}{\tilde{N}}} \tag{5.12}$$

$$Y_t^U = \bar{Y}_t + t_{1-\frac{(1-c)}{2}}(\tilde{N}-1)s_t\sqrt{1+\frac{1}{\tilde{N}}} \tag{5.13}$$

The probability of the interval containing the true value equals confidence $c \in [0,1]$:

$$P(Y_t^L \leq Y_t \leq Y_t^U) = c \tag{5.14}$$

where $t_{1-\frac{(1-c)}{2}}(\tilde{N}-1)$ is the $(1-\frac{(1-c)}{2})^{th}$ percentile of t-distribution with $\tilde{N}-1$ degrees of freedom. The prediction interval gets wider with increasing c. The most commonly used values for confidence c are 90%, 95%, and 99.9%. The procedure is illustrated in Figure 5.3 (left). More details of the statistical inference are publicly available[4].

Traditional approaches rely on fixed confidence levels, which fail to adequately respond to market fluctuations. In contrast, our method adapts the confidence level c_t based on the market risk. Intuitively, the interval should expand in riskier markets to accommodate unpredictable fluctuations during high volatility, while narrowing in calmer markets to avoid overestimating uncertainty and enhance accuracy. Hence, we design a *risk-sensitive interval* construction, where market risk determines c_t and further controls the percentile of the t-distribution as well as the width of the predicted interval accordingly. v_l, v_u are defined as the lower and upper thresholds for the volatility features. c_t is limited in the range $[c_l, c_u]$. It shows a sigmoid curve F

[4]https://github.com/JingyiGu/RAGIC

between c_t and volatility feature v_{t-1}, as shown in Figure 5.3 (right):

$$c_t = F(v_{t-1}) = c_l + \frac{c_u - c_l}{1 + \exp\left(-k(v_{t-1} - v_0)\right)} \tag{5.15}$$

$$k = \ln\left(\frac{c_u - c_l}{\Delta\epsilon} - 1\right)\frac{2}{v_u - v_l}, \quad v_0 = \frac{v_l + v_u}{2} \tag{5.16}$$

where k is the growth rate showing the steepness of the curve, and v_0 is the middle point of the curve, as well as the middle point of the lower and upper volatility thresholds. At first, c_t stays close to c_l infinitely but not equal to c_l when $v_{t-1} \leq v_l$, with at most $\Delta\epsilon$ of difference; when v_{t-1} increases from v_l to v_u, c_t converges quickly from c_l to c_u; as v_{t-1} approaches to and exceeds v_u, c_t stay close to c_u infinitely. In this study, the range of confidence levels c_l and c_u is set to 90% and 99.9%. Hence, a wider predicted interval indicates higher market volatility and potentially bigger needs for hedging. This dynamic adjustment balances reliability and precision, better capturing the full range of potential market movements, ensuring more reliable and risk-aware predictions. This adaptability is valuable not only in finance but also in any domain with inherent risk and uncertainty.

5.5 Model Evaluation

5.5.1 Experiment 1: interval prediction

Dataset We choose five broad-based stock indices worldwide, including three indices from the U.S. market, one index from the European market, and one index from the Asian market:

- **Dow Jones Industrial Average (DJIA)** is one of the oldest equity indices, tracking 30 of the most highly capitalized and influential companies in major sectors of the U.S. market
- **S&P 500 (SPX)** is one of the most commonly followed indices and is composed of the 500 largest publicly traded companies

- **NASDAQ 100** contains 101 of the largest non-financial company stocks from the Nasdaq stock exchange; [5]
- **Deutscher Aktienindex performance index (DAX)** measures the performance of 40 major German blue chip companies
- **The Nikkei index** consists of 225 companies from a wide range of industry sectors in Japan.

The historical daily stock indices and their corresponding volatility indices are publicly available on Yahoo Finance [6] and Investing [7], and thus our model is easy to implement.

All indices are split into training, validation, and testing sets along with the temporal dimension. Even though the start date of the whole period varies in indices due to data availability, the validation period is consistently chosen as 2013/8-2015/12, and the testing period is 2016/1-2021/12. Specifically, the testing period includes over 1400 trading days for all indices. It covers the COVID pandemic in 2020 and the following recovery when the market was highly volatile. We use features from the previous 30 days to generate sequences on the future 5 days, i.e., $W = 30, H = 5$. Table 5.1 shows the detailed statistics and training period of all indices.

The input features of *RAGIC* include price-based features and volatility-based features. The price-based features are similar to those in Chapter 4, consisting of historical raw prices (open, high, low, close), and deliberately chosen technical indicators that can be derived from raw prices For feature preprocessing:

- Raw prices: converted to return calculated by $x_t/x_{t-1} - 1$.
- Volatility: two features are derived: (1) volatility return $v_t/v_{t-1} - 1$ is calculated the same as raw price features; (2) normalized volatility is min-max scaled. The risk module uses both features as inputs, and interval construction is based on the raw volatility index.

[5] Google has both GOOG, class C stock, and GOOGL, class A stock, listed in Nasdaq.
[6] `https://finance.yahoo.com`
[7] `https://www.investing.com`

Table 5.1 Statistics of Stock Indices

Index	Volatility	# days	Time Period of Training Set
DJI	VXD	6051	1997/12-2013/7
SPX	VIX	6051	1997/12-2013/7
Nasdaq	VXN	5259	2001/1-2013/7
DAX	VDAX	5237	2001/5-2013/7
Nikkei	JNIV	4791	2002/6-2013/7

In the phase of interval construction, the close return is converted back to the close price for sequence simulation, and the confidence is determined by the raw volatility index.

Baselines We select the following methods as our baselines for interval prediction:

- **Bollinger Bands (BB)** [116] is a momentum indicator in finance. The upper and lower bands provide the oversold or overbought signals in the market.
- **BayesianNN** [18] uses Monte Carlo dropout as Bayesian inference approximation.
- **MQRNN** [20] is a sequence-to-sequence framework for multi-horizon quantile forecasting.
- **CFRNN** [25] is a neural network extending inductive conformal prediction to time series forecasting.
- **D3VAE**[110] is a generative time series forecasting method incorporating diffusion, denoising, and disentanglement on top of VAE.
- **StockGAN** [61] is a popular GAN model for stock prediction, which employs LSTM as the generator and Multi-Layer Perceptron (MLP) as the discriminator.
- **FactorVAE** [13] is the previous state-of-the-art model for stock market prediction, treating effective factors as the latent random variables in VAE.

Among baselines: Bollinger Bands is a momentum indicator in finance, BayesianNN, MQRNN, and CFRNN are the three most representative models for

uncertainty quantification; D3VAE, StockGAN, and FactorVAE are three popular generative models that are able to produce sequences, the formal one is for general time series, the latter two are designed for the stock market. Considering they are point prediction methods and cannot be directly used for interval prediction, for a fair comparison, we develop D3VAE*, StockGAN*, and FactorVAE*, which simply apply our proposed interval construction phase to their output sequences from the original models and build intervals.

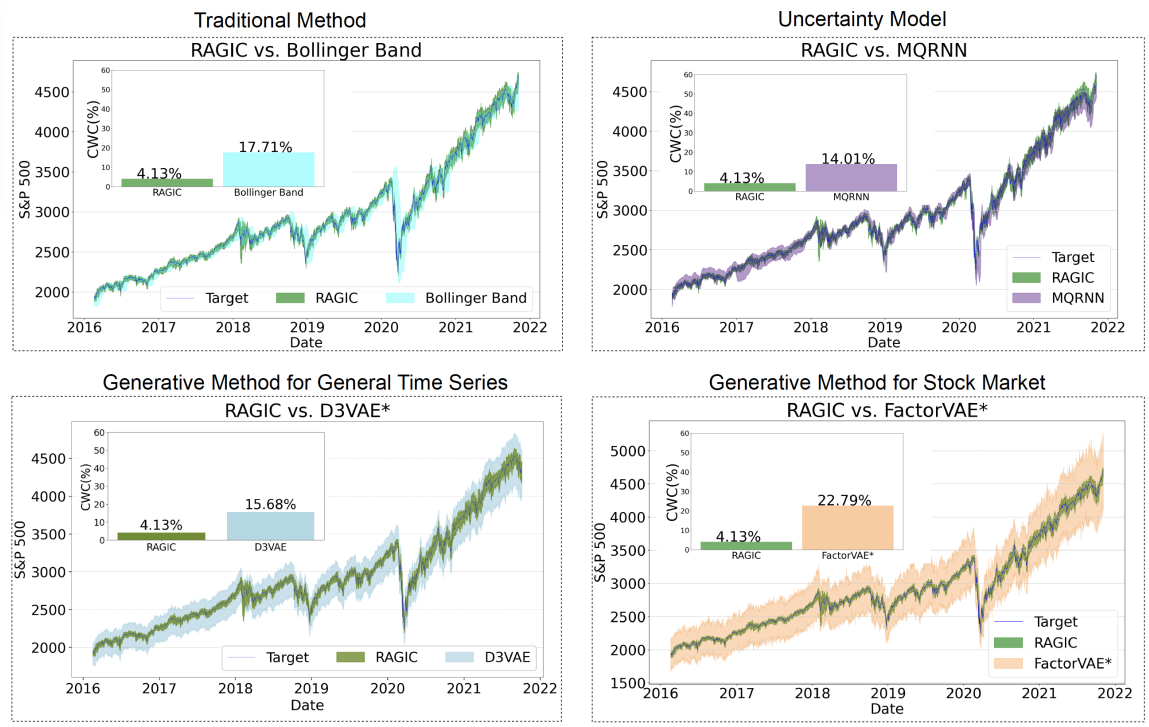


Figure 5.4 Comparison of predicted intervals on SPX.

Evaluation Metrics We follow common practice to adopt three metrics to evaluate interval prediction performance:

- **Coverage Probability (CP)** is the percentage of true values falling into the predicted interval over the test period.

$$CP = \frac{1}{T}\sum_{t=1}^{T} c_t, \quad c_t = \begin{cases} 1 & y_t \in [Y_t^L, Y_t^U] \\ 0 & y_t \notin [Y_t^L, Y_t^U] \end{cases} \tag{5.17}$$

Table 5.2 Numerical Results for Interval Prediction

Model	DJIA			SPX			Nasdaq		
	CP	NMW	CWC	CP	NMW	CWC	CP	NMW	CWC
BB	81.97	7.52	21.95	81.97	6.07	17.71	81.56	5.13	15.18
BayesianNN	86.78	13.76	34.51	88.64	12.40	29.44	91.07	12.85	28.49
MQRNN	86.37	6.11	15.52	85.60	5.39	14.01	95.00	6.63	6.63
CFRNN	99.86	56.01	56.01	100	62.51	62.51	100	25.70	25.70
D3VAE*	97.01	16.10	16.10	99.60	15.68	15.68	98.03	12.84	12.84
StockGAN*	46.65	2.99	36.53	59.77	3.40	23.19	60.24	3.68	24.60
FactorVAE*	100	27.95	27.95	100	22.79	22.79	100	17.35	17.35
RAGIC	95.56	5.42	**5.42**	95.21	4.13	**4.13**	95.81	3.99	**3.99**

Model	DAX			Nikkei		
	CP	NMW	CWC	CP	NMW	CWC
BB	83.32	10.74	30.00	82.90	10.13	28.68
BayesianNN	85.69	13.64	35.37	86.46	12.68	32.11
MQRNN	84.89	8.86	23.57	86.88	5.03	12.58
CFRNN	100	23.09	23.09	100	16.93	16.93
D3VAE*	96.94	15.13	15.13	98.95	14.11	14.11
StockGAN*	42.79	3.26	47.61	36.83	3.39	65.53
FactorVAE*	100	36.52	36.52	100	32.01	32.01
RAGIC	95.81	7.78	**7.78**	96.28	7.96	**7.96**

Note: Numerical results regarding CP (%), NMW(%), and CWC(%) for interval prediction, averaged over 5 independent runs. The improvement is significant statistically (t-test with $p < 0.01$).

- **Normalized Mean Width (NMW)** is the mean width normalized by the range of close prices in the test period, where the mean width is the mean difference between the upper and lower bounds.

$$NMW = \frac{1}{TR}\sum_{t=1}^{T}(Y_t^U - Y_t^L), \quad R = max(y) - min(y) \tag{5.18}$$

- **Coverage Width-based Criterion (CWC)** is a composite metric combining CP and NMW [21], showing the balance between accuracy and informativeness.

$$CWC = NMW(1 + \exp\left(\eta \max(0, 95\% - CP)\right)) \tag{5.19}$$

  η is a penalty parameter on the width if coverage does not achieve the expectation. A larger η leads to an extremely higher CWC if coverage falls short of the desired 95%, while a smaller value may not sufficiently highlight the difference. In our study, we set it to 5 because it provides a more effective and straightforward model performance. A perfect interval should have high coverage, low width, and a low value of CWC.

Implementation Details: *RAGIC* is implemented in PyTorch. Hyperparameters are tuned based on the validation set. We use Adam as the optimizer with 50 epochs. The learning rate of the critic is set to 0.0003. The threshold for weight clipping ξ is set to 0.01. The number of generated sequences N is 50. The penalty of supervised loss γ is 0.3. The number of heads in multi-head attention is 2. The threshold $\Delta\epsilon$ in the Sigmoid curve is 0.0001. α_l and α_u are set to 90% and 99.9%. The batch size varies in {150, 256}. The learning rate of the generator varies in {0.00005, 0.0001}. In risk module, the risk threshold $\boldsymbol{\delta}_k$ for volatility return and normalized volatility index vary in {0.05,0.1,0.2} and {0.05,0.1,0.2,0.25} respectively, the coefficient $\boldsymbol{\lambda}_k$ vary in {1.5,2.5} and {1.5, 2.5, 4} respectively. In TCN, the number of layers L varies in {2,3}, with the kernel size p in {5,8} and the dimension of hidden layer in {100,150}. The frequency of critic update varies in {3,4,5,6}. The lower and upper threshlolds for volatility features in *risk-sensitive interval* v_l, v_u vary in {10,12} and {20,22,25} respectively.

For other baselines, we fine-tune the hyperparameters based on the optimal values reported in their works and choose the best model for implementation. Besides, below are two implementation differences between *RAGIC* and baselines for fair comparisons:

- Input features: Bollinger Bands uses close price only; other models are trained on the same features with *RAGIC*
- Confidence c: the calculation of Bollinger Bands does not require it; BayesianNN, MQRNN, and CFRNN employ a target error rate which is equivalent to $1 - c$ in our work, thus c is set to 95%; D3VAE*, StockGAN*, and FactorVAE* set the lower and upper bound of c_t to 90% and 99.9%, same with *RAGIC*.

Results and Discussion Table 5.2 presents the numerical results. We have the following observations. (1) *RAGIC* significantly outperforms other baselines in terms of CWC, which shows a balanced manner between accuracy and informativeness. It consistently achieves 95% coverage on all indices with relatively narrow intervals. (2) Intervals of *RAGIC*, CFRNN, D3VAE*, and FactorVAE* accurately contain true values with over 90% of CP for all indices, while the width of *RAGIC* is the narrowest (at least 5 times narrower than the other two for the U.S. market). Hence, FactorVAE*, D3VAE, and CFRNN are not competitive in terms of CWC. FactorVAE* maintains the perfect 100% of coverage with over 15% of NMW. We postulate the reason is that it cannot learn the posterior distribution well; hence, individual point prediction is not robust and the range of the interval with noise is large. The prediction interval from D3VAE* exhibits higher coverage but is much wider, as the diffusion model generates more diverse outputs, making it better suited for creative tasks, which contrasts with the precision required for financial forecasting. CFRNN assumes exchangeable samples and adds the errors of the training set to the prediction of the test set, which does not fit the stock price time series data. For the task of stock market prediction, a narrow interval with high coverage is required,

since an infinite interval that perfectly covers all possible prices is meaningless for decision-making. (3) StockGAN* produces the narrowest interval but fails with the lowest coverage. Possibly using LSTM as the generator can not well capture the complex temporal dependencies. (4) For Bollinger Bands, BayesianNN, and MQRNN, the coverage achieves over 80% while the width is larger than *RAGIC*. Therefore, their performance regarding CWC is not ideal.

Moreover, for each type of baselines, we choose one representative model and visualize its predicted intervals alongside those generated by *RAGIC* on S&P 500, as shown in Figure 5.4. The dark green area exhibits the interval of *RAGIC*. It perfectly covers the actual price with a narrow interval, even in a very volatile market with a sharp drop, demonstrating the effectiveness of *RAGIC*.

5.5.2 Experiment 2: ablation study

Model components We implement three variants of *RAGIC* to quantify improvement from the volatility index and temporal module.

- ***RAGIC*-risk** is a variant without the risk module to detect market risk and assign *risk attention score* on stock features.
- ***RAGIC*-fixed** is a variant fixing c to 0.95 when building the interval.
- ***RAGIC*-temporal** is a variant that replaces the temporal module with a single fully connected layer.

The comparison of three variants on indices regarding CWC is visualized in Figure 5.5. We observe that all three elements consistently contribute to the model quality on all indices. The volatility index in the risk module enhances performance by detecting high market risk; *risk-sensitive interval* improves CWC by increasing coverage with an acceptable sacrifice of width; the temporal module effectively mimics the dynamics of future patterns.

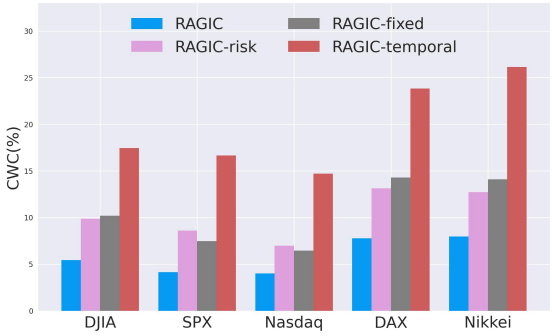


Figure 5.5 Ablation study of *IndexGAN*.

Prediction Horizon We investigate the impact of the prediction horizon H on *RAGIC*. Figure 5.6 shows the change of CWC on different H. As H goes up from 2 to 5, the performance of the predicted interval shows a rising trend since richer information from a longer horizon is incorporated in building the interval, and the critic can differentiate real and fake sequences as their length becomes longer. CWC achieves the lowest value when H is 5. However, as the length of the predicted sequence exceeds five days and continues increasing to 10 days, the interval quality goes down. It reveals that it becomes harder for GAN to simulate longer-term price patterns, and *horizon-wise* information cannot provide more simulated uncertainty. The evaluation justifies 5 days forward as the optimal value for H. Also, note that 5 is the number of trading days in a week.

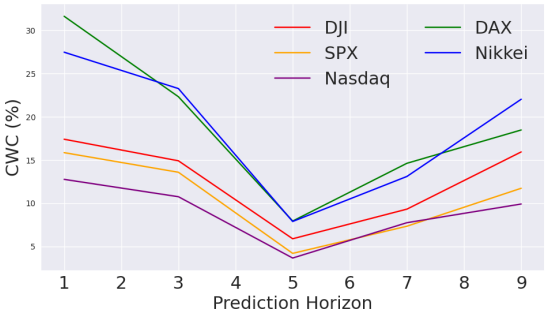


Figure 5.6 Impact of the prediction horizon.

Visualization of Risk Attention Score To verify that the risk score can capture the market risk effectively, Figure 5.7 visualizes the risk attention score for two

volatility-based features, VIX and the return of VIX, during the COVID pandemic. When the price begins to drop, the risk score captures the dramatic growth rate of VIX and alerts with an extremely high value on VIX return. During the crash period, the risk score for VIX remains high, which reveals that investors lack confidence in the market. Both scores go down with the recovery of the market.

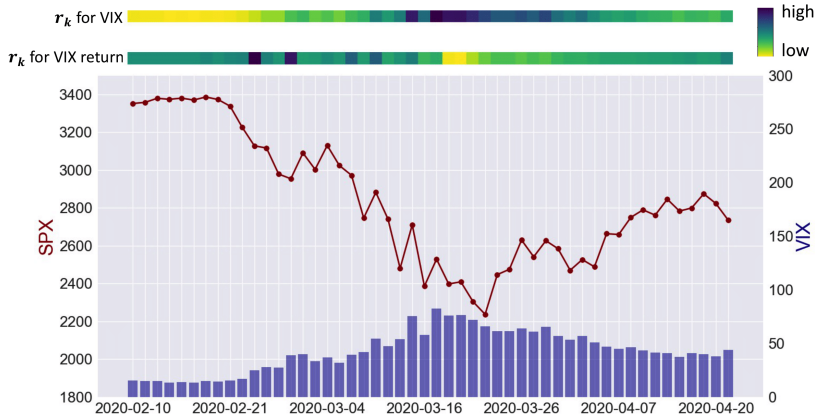


Figure 5.7 Visualized risk score for VIX and VIX return.

As H goes up from 2 to 5, the performance of the predicted interval shows a rising trend since richer information from a longer horizon is incorporated in building the interval, and the critic can differentiate real and fake sequences as their length becomes longer. CWC achieves the lowest value when H is 5.

Various Methods of Adjusting Confidence c To quantify the improvement of employing the volatility index on interval adjustment comprehensively, we implement different values on various methods to determine c_t for the interval width.

Without loss of generosity, the experiment results on DJIA are shown in Table 5.3. The value of c_l, c_u, v_l, and v_u are set to 90%, 99.9%, 10, 22, respectively. We can see that simply fixing c_t to a single higher value for all time steps in the testing period does not necessarily increase coverage, even though the interval gets wider. It is the least efficient method of increasing coverage and CWC in this table. It is worth noting that sigmoid change with a higher lower bound c (95%) covers more true close

Table 5.3 Comparison of Interval Width Adjustment Methods on DJI

Method	Parameters	CP	NMW	CWC
Fixed value	90	90.44	4.19	9.46
	95	92.68	4.80	10.19
	98	94.24	5.53	11.27
	99.9	96.54	7.51	7.51
Sigmoid change	10,22,90,99.9	95.66	5.42	**5.42**
	10,22,95,99.9	95.86	5.82	5.82

Note: Parameters for fixed value is c, parameters for sigmoid change are v_l, v_u, c_l, and c_u.

prices of DJIA over the test period, resulting in a higher coverage, while the interval gets slightly wider. When this model is used, a user can make a decision between coverage and width, given the application requirements.

5.5.3 Experiment 3: point prediction

Method Besides generating an interval, *RAGIC* can also compute the point prediction for price. Given the prediction vectors $\tilde{\boldsymbol{Y}}_t$ with element $\tilde{\boldsymbol{y}}_{t,n} = (\hat{y}^1_{t,n}, ..., \hat{y}^H_{t,n})$ from sequence simulation in the interval construction phase, we first compute weighted average value across the prediction horizon:

$$\hat{y}_{t,n} = \frac{w_h \hat{y}^h_{t,n}}{\sum_{h=1}^{H} w_h}, \quad w_h = \frac{1}{2^h} \tag{5.20}$$

where the weight w_h exponentially shrinks as the horizon step h increases, considering that the prediction from the most recent historical features is more reliable. Then we take the average over the repeated sequences as the final prediction $\hat{y}_t = \frac{1}{n} \sum_{n=1}^{N} \hat{y}_{t,n}$

for the price at time step t, which mitigates the noise caused by uncertainty in the financial market.

Baselines We identify ten baselines to compare with our work in point prediction. In addition to the eight models mentioned for interval prediction, we choose two recent popular works on stock point prediction.

- **HMG-TF** [12] enhances the Transformer with regularization and gaussian prior to predict stock.
- **AdaRNN** [117] formulates the temporal covariate shift in non-stationary time series forecasting.

Among the previously mentioned eight models, StockGAN and FactorVAE are designed for point prediction and thus need not be adjusted in this comparison. The other six baselines for interval prediction are adjusted in the following way to be used for point prediction: (1) For Bollinger Bands, the middle point of the upper and lower bounds is used as the prediction. (2) For BayesianNN, MQRNN, and CFRNN, we follow the original settings in their works [25]. (3) For D3VAE*, StockGAN* and FactorVAE*, the average of generated sequences for each time step is taken as the prediction.

Results We adopt MAPE as the evaluation metric and Table 5.4 shows the numerical results of the ten baselines and *RAGIC*. The table shows that *RAGIC* outperforms all baselines for interval prediction regarding MAPE on all indices. Though FactorVAE* and CFRNN achieve full coverage, their point predictions are not accurate. The performance of FactorVAE on point prediction regarding MAPE is even worse, despite its superior performance on ranking metrics[13]. It verifies our conjecture of large noise in individual point prediction, which is mitigated in FactorVAE*. In addition, compared with baselines for point prediction on the stock market, *RAGIC* is still competitive with the lowest MAPE.

Table 5.4 Numerical Results Regarding MAPE (%) for Point Prediction

Model	DJIA	SPX	Nasdaq	DAX	Nikkei
Bollinger Bands	1.92	1.90	2.43	2.16	2.24
BayesianNN	1.72	1.86	3.29	1.65	1.29
MQRNN	1.45	1.60	1.65	1.74	0.95
CFRNN	2.21	2.25	1.74	1.87	1.85
D3VAE*	1.62	1.93	1.71	1.85	1.88
StockGAN*	2.02	2.89	3.94	2.25	2.96
FactorVAE*	1.01	1.03	1.13	1.00	1.07
HMG-TF	0.76	0.71	0.91	0.84	0.93
AdaRNN	0.79	0.87	0.93	0.83	0.94
D3VAE	1.95	2.01	1.94	1.98	2.08
StockGAN	1.88	2.03	3.08	1.89	2.55
FactorVAE	1.31	1.25	1.40	1.36	1.33
RAGIC-risk	0.85	0.85	0.92	0.83	0.94
RAGIC	**0.71**	**0.69**	**0.89**	**0.81**	**0.89**

Note: Numerical results regarding MAPE (%) for point prediction, averaged over 5 independent runs. The improvement is significant statistically (t-test with $p < 0.01$).

5.6 Discussion on Downstream Applications

The interval prediction can offer investors valuable insights and aid them in making more informed and effective decisions. Its utility in downstream applications can be illustrated as follows: (1) Interval information can be heavily used in stock option trading. For example, if the predicted interval is [90,100], one profitable trading strategy is to sell a call option with a strike price $100 + \epsilon$, and sell a put option with a strike price $90 - \epsilon$. (2) The upper and lower bounds can serve as automatic take-profit and stop-loss levels for closing positions, which helps maximize the profit while mitigating risks effectively. (3) The width guides position sizing and risk management.

A wide interval indicates higher risk, suggesting the need to reduce the position size to limit exposure, and vice versa. (4) When employing deep reinforcement learning for portfolio management, the prediction interval can be part of the state as a valuable reference for the model to determine the trading action.

5.7 Conclusions and Future Work

This research aims to tackle the challenge of stock prediction arising from the inherently stochastic nature of stock prices. To address this, we introduce a novel framework called *RAGIC*. Our approach enhances the capabilities of the Generative Adversarial Network (GAN) in a unique manner, enabling the generation of a set of future price sequences with randomness derived from the financial market. Additionally, *RAGIC* constructs a *risk-sensitive interval*, which offers more comprehensive information compared to point predictions. Note that our approach of integrating sequence generation with interval construction is flexible and can be adapted to various model architectures across domains characterized by stochastic variability. The components within the sequence generation model exemplify our strategies for capturing temporal financial dependencies. They can be substituted with alternative mechanisms with effective prediction capabilities, highlighting the innovative potential of our work in the context of interval construction.

Looking ahead, our future endeavors focus on two key directions: (1) extending the generalization of *RAGIC* to individual stock interval prediction, and (2) exploring downstream tasks to leverage easily implemented *RAGIC* for generating more profitable trading strategies to aid decision-making.

CHAPTER 6

DYSTAGE: GRAPH REPRESENTATION LEARNING FOR DYNAMIC ASSET PRICING

6.1 Introduction

Asset pricing models are pivotal in estimating the future returns of underlying assets such as stocks, by employing a multitude of variables to model their interactions [118]. These models are essential for market participants who seek to discern potential high-performing assets within the market universe. The assets in this universe are interdependent through various dimensions such as supply chains, industry sectors, return similarities, volatility spillovers, and market conditions. These complex interrelationships significantly impact their relative prices [119, 120]. Given these intricacies, graph networks have become a preferred tool for modeling such relationships [68, 121, 122, 123]. Financial markets are inherently dynamic; assets and their interconnections evolve continuously due to factors such as new market entries, asset maturation, and corporate events like bankruptcies, mergers, and acquisitions. This ongoing evolution necessitates a dynamic representation in our models.

Figure 6.1 visually depicts the evolution of the market in terms of asset participation and their interrelationships. Conceptualized as a graph, each asset is represented as a node, and correlations are depicted as edges. Nodes dynamically appear and disappear from the asset graph, reflecting the fluid and responsive nature of the market to real-world events. For instance, Lehman Brothers was removed in October 2008 following its bankruptcy, while Tesla was added in June 2010 with its IPO. Similarly, the connections between assets—edges—also evolve, reflecting shifts in corporate strategy and market conditions. A prime example is Apple, which, upon launching Apple TV+ in 2019, pivoted from focusing solely on hardware and software

to entering the competitive streaming service market, positioning itself as a rival to Netflix.

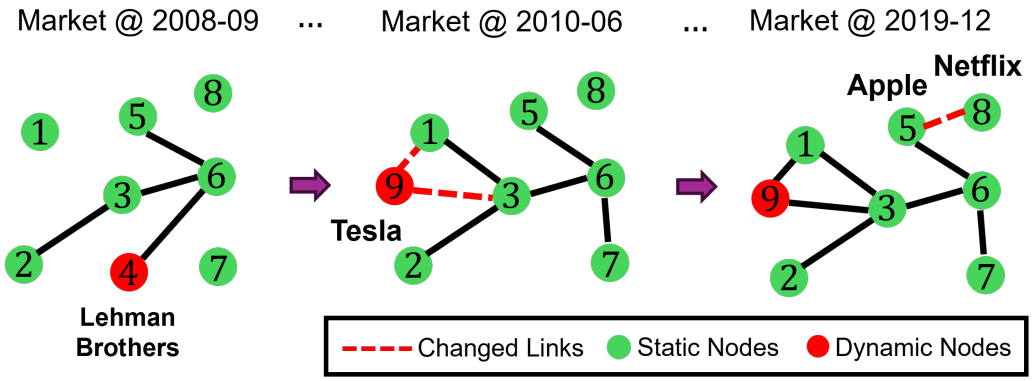


Figure 6.1 An example of dynamic graphs in asset pricing.

Despite numerous studies on network dynamics, there remains a substantial gap in developing frameworks that can effectively learn and represent the time-varying dynamics of financial markets. Recent advancements in graph-based asset pricing methodologies have leveraged static graph structures to model relationships and predict future asset prices [124, 70]. Meanwhile, dynamic graph representation learning, commonly used in domains such as traffic, social, and rating networks [73, 125, 74, 75], has been less frequently applied in financial networks. These financial networks are distinguished by their evolving interdependencies, which add complexity to their analysis.

To address this gap, we introduce a Dynamic graph representation learning model via Spatio-Temporal Attention and Graph Encodings (**DySTAGE**), an end-to-end dynamic graph learning framework designed to accurately reflect and predict market evolution. We propose a universal formulation for dynamic graphs tailored to asset pricing, constructing a sequence of dynamic graphs from asset time series data that accommodate node (asset) additions and removals. Additionally, asset correlation over different economically meaningful time scales is incorporated and reflected as multiple edge features between the same pair of nodes. Subsequently, we design a graph learning model that employs attention mechanisms and graph

encodings enriched with financial insights to predict the future excess returns of existing nodes, capturing both topological and temporal patterns. This framework enhances predictive accuracy and supports informed investment strategies in a rapidly changing market.

The framework features a *Topological Module* and a *Temporal Module*. The *Topological Module* individually processes structural information for each graph and comprises several components: *Asset Influence Attention*, which delineates the global interrelationships between assets; *Asset-wise Importance Encoding*, emphasizing the broad market impact of individual assets; *Pair-wise Spatial Encoding*, exploring intermediary assets and uncovering hidden connections; and *Edge-wise Correlation Encoding*, revealing the evolution of relationships between highly correlated assets. Following this, the *Temporal Module* delves into the historical representations of each asset across time, offering a detailed temporal analysis.

We conducted extensive experiments on three real-world datasets: Russell-3000 monthly data, MLFI monthly data, and S&P 500 daily data. The results demonstrate *DySTAGE*'s superior performance in return prediction and portfolio optimization. Our contributions can be summarized as follows[1]:

1. Introduction of *DySTAGE*, a novel dynamic graph representation learning framework for asset pricing, with formulation accommodating dynamic asset composition and evolving edge connections, and a model designed with financial insights.

2. Effective capture of both topological and temporal patterns, utilizing graph encodings that reflect asset importance, hidden connections, and their evolving impact within the financial network.

[1]Jingyi Gu and Junyi Ye contributed equally to this research, which centers on the development and evaluation of *DySTAGE*, a novel dynamic graph representation learning framework for asset pricing. Specifically, Jingyi Gu focused on the design, implementation, and subsequent evaluation of the proposed *DySTAGE* model, including its spatio-temporal attention mechanisms and graph encoding components. Junyi Ye concentrated on developing the dynamic graph construction process from asset pricing time series and implementing and evaluating the baseline models used for comparison. Both authors collaborated closely on the overall experimental design, analysis, and interpretation of the results.

3. Proven superiority of *DySTAGE* over conventional and popular benchmarks in predictive accuracy, illustrating its efficacy in balancing profit and risk for portfolio management.

6.2 Problem Formulation

The core challenge in asset pricing is to estimate the expected future value of an asset based on the information currently available. Research has consistently shown that the expected future value of an asset is influenced by a combination of factors, including asset-specific characteristics, historical performance trends, real-time news, and broader market dynamics [70, 126]. This study aims to capture both the temporal dependencies and the spatial dynamics of the market using a dynamic spatio-temporal graph model.

Our objective is to develop a model $f(\cdot)$ that can accurately predict future return for assets $\mathbf{y}^{t+1}$ based on a sequence of historical financial interconnection networks $\mathbb{G} = \{\mathcal{G}^{t-k}, ..., \mathcal{G}^t\}$ and corresponding asset characteristics $\mathbf{X} = \{\mathbf{X}^{t-k}, ..., \mathbf{X}^t\}$:

$$\mathbf{y}^{t+1} = f(\mathcal{G}^{t-k}, \cdots, \mathcal{G}^t, \mathbf{X}^{t-k}, \cdots, \mathbf{X}^{t-k}, ; \theta) \tag{6.1}$$

where k represents the window of historical data considered. This formulation enables the model to utilize temporal data from past periods and integrate it with spatial information from the asset interconnections, capturing the dynamic interplay of market variables that affect asset pricing.

6.3 Methodology

Figure 6.2 shows the architecture of *DySTAGE*. It starts with a dynamic graph construction phase. Then it contains two key modules: the topological module learns the structural information of each graph independently; the temporal module

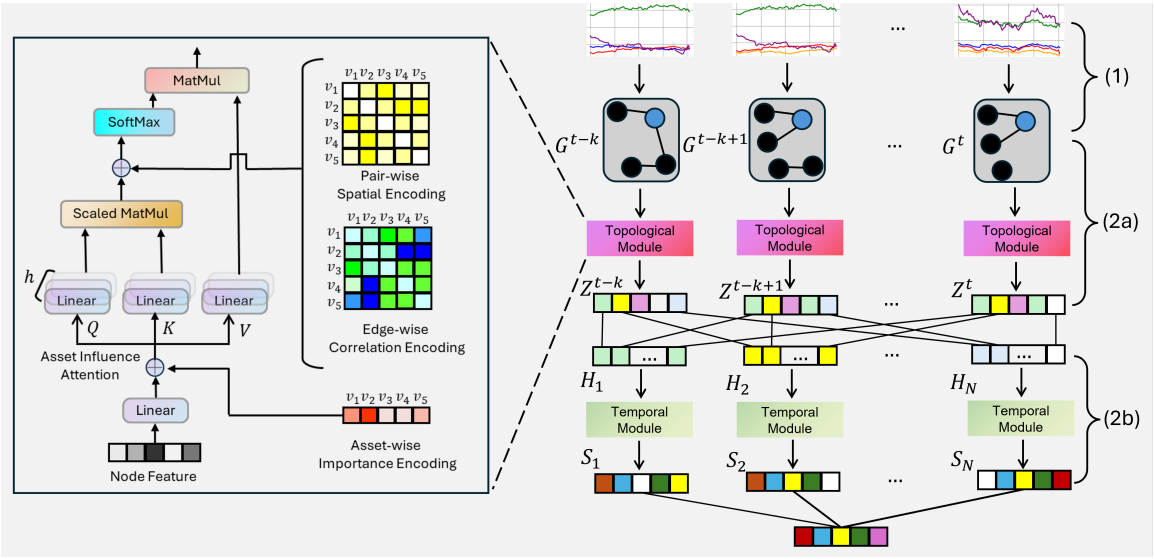


Figure 6.2 The framework of *DySTAGE*. (1) Dynamic graph construction from time series (top right). (2) Dynamic graph learning model: (a) Topological module individually processing structural information for each graph (left), (b) Temporal module capturing historical representations across time (bottom right).

generates node embeddings for each node by capturing its historical patterns along the temporal dimension.

6.3.1 Dynamic graph construction

We consider the historical dynamic graphs $\mathbb{G} = \{\mathcal{G}^{t-k}, ..., \mathcal{G}^t\}$ as a sequence, where $\mathcal{G}^t = (\mathcal{V}^t, \mathcal{E}^t)$ represents an undirected graph at time step t, with $n^t (= |\mathcal{V}^t|)$ nodes (i.e., total number of assets at time t) and $m^t (= |\mathcal{E}^t|)$ edges. Let $\mathbf{X}^t \in \mathbb{R}^{n^t \times d}$ denote node features, where n^t is the number of assets and d is the feature dimension. $\mathcal{A}^t \in \mathbb{R}^{n^t \times n^t}$ is the adjacency matrix of graph $\mathcal{G}^t$, representing the long-term interrelationship between assets. In this work, we utilize Pearson Correlation on the historical excess returns to quantify the graph connections:

$$\mathcal{A}^t_{u,v} = \begin{cases} \rho^t_{u,v} & |\rho^t_{u,v}| > \gamma \\ 0 & |\rho^t_{u,v}| \leq \gamma \end{cases} \tag{6.2}$$

For any pair of assets u and v, if their absolute correlation $\rho_{u,v}^t$ based on returns over the past w time steps is larger than the threshold $\gamma \in (0, 1)$, they are considered to have a strong long-term correlation and should be connected by an edge. Both positive and negative correlations between assets are incorporated in a single graph. $\mathcal{E}^t \in \mathbb{R}^{m^t \times p}$ denotes edge attributes containing multi-scale excess return correlations between assets, calculated from short-term to long-term perspectives, where p is the number of scales. For monthly asset data, we choose scales of 3, 6, 12, 24, and 36 to describe quarterly, semiannually, and yearly trends. For daily data, we choose scales of 5, 10, 15, and 20 for weekly patterns. The target of our problem is the excess return $\mathbf{y}^{t+1} \in \mathbb{R}^{n^{t+1}}$ of all existing assets in the future time step $t + 1$. Our primary objective is to train a network that learns historical dynamic graphs $\mathbb{G}$ and predicts the future excess return $\mathbf{y}^{t+1}$.

To streamline implementation, we unify the node set across all time steps into a shared set $\mathcal{V}$, encompassing all appearing nodes, irrespective of their addition or removal over time. Therefore, the graphs, including their adjacency matrices $\mathcal{A}^t \in \mathbb{R}^n$ and node features $\mathbf{X}^t \in \mathbb{R}^{n \times d}$ at all time steps, contain the same nodes. If node u does not exist at time step t, then there does not exist an edge between itself and any other nodes, and its feature $\mathbf{x}_u^t \in \mathbb{R}^d = \mathbf{0}$ is also set to a zero vector.

6.3.2 Topological module

To predict future asset returns, each historical graph $\mathcal{G}^{t-k}, \ldots, \mathcal{G}^t$ is individually fed into the Topological Module. $\mathcal{G}^t$ represents the asset interconnection network at that time point. Asset Influence attention is utilized to describe the interrelationship and their influences between assets based on their node features globally. To fully investigate the structural information, we incorporate three graph encodings derived from dynamic asset graphs from financial perspectives: Asset-wise Importance Encoding highlights the importance of each asset and its potential impact on the

market; Pair-wise Spatial Encoding investigates the hidden connections between assets; Edge-wise Correlation Encoding provides evolving correlation between assets. The output of this module is a set of node embeddings in the graph. The details of each of these encodings are discussed below. Note that the index of time step t is omitted in the remaining discussion in this module for simplicity.

Asset Influence Attention Given the asset features $\mathbf{X}$ in the input graph $\mathcal{G}$, we follow the standard multi-head attention mechanism [112] to design an asset influence attention and derive the attention matrix. Within each head, the input feature $\mathbf{X}$ is firstly mapped into $\mathbf{X}^* \in \mathbb{R}^{d \times d^*}$, where superscript $\cdot^*$ indicates the topological module, and d^* denote the dimension in this module. Then it is projected by three learnable weight matrices $\mathbf{W}_q^*, \mathbf{W}_k^*, \mathbf{W}_V^* \in \mathbb{R}^{d^* \times d_k^*}$, where d_k^* denotes the hidden dimension of each head. Then the attention matrix $\mathbf{A}_h \in \mathbb{R}^{n \times n}$ is computed, where $h \in \{1, ..., H\}$ and H represents the number of heads, with each attention element $A_{u,v,h}^*$ describing the influence of asset u on asset v:

$$\mathbf{A}_h^* = \mathbf{M_0}^* \odot softmax(\frac{(\mathbf{X}^*\mathbf{W}_q^*)(\mathbf{X}^*\mathbf{W}_k^*)^\top}{\sqrt{d_k^*}} + \mathbf{M_\infty}^*) \tag{6.3}$$

$$\mathbf{Z} = \mathbf{X}^* + [\mathbf{A}_1^*(\mathbf{X}^*\mathbf{W}_v^*), ..., \mathbf{A}_H^*(\mathbf{X}^*\mathbf{W}_v^*)]\mathbf{W}_o^* \tag{6.4}$$

For a non-existing asset u, it should not exert influence and relationships with other assets. To exclude the current non-existing assets, we incorporate two mask matrices before and after the softmax activation function. Softmax activation operates on the row sum of the attention matrix. For negative mask matrix $\mathbf{M_\infty}^* \in \mathbb{R}^{n \times n}$ with $M_{iu}^* = -\infty, i \in \mathcal{V}$, it ensures the column vector of asset u is masked out, yielding corresponding softmax results is 0. The zero mask matrix

$\mathbf{M_0}^* \in \mathbb{R}^{n \times n}$ with $M^*_{uj} = 0, j \in \mathcal{V}$ guarantees the row vector of asset u is masked out as a zero vector, $\odot$ denotes the element-wise multiplication.

Utilizing multiple heads facilitates the aggregation of diverse embedding spaces. The embeddings from multiple attention heads are concatenated together and mapped by $W_o \in \mathbb{R}^{Hd^* \times d^*}$ to yield the output embeddings $\mathbf{Z} \in \mathbb{R}^{n \times d^*}$. Following the common practice, we apply layer normalization [127] before attention and incorporate skip connections after concatenating the heads, aiming at enhancing optimization efficiency [128, 111]. The Asset Influence Attention can be treated as one layer and extended to multiple layers.

Asset-wise Importance Encoding Unlike the position encoding in the original transformer, which signifies the order of the sequence, the nodes within the graph lack such inherent sequence. However, it is crucial to identify the importance of each asset and its potential impact within its sector or the broader financial market. Such insights can be learned from the topological structure of the graph. Therefore, we introduce an Importance Encoding to capture asset-wise structural information. Specifically, for each asset u, a higher node degree implies that the asset has a strong correlation with a larger number of other assets, indicating its potential market impact. Let $\mathbf{D}_u$ denote the degree encoding of asset u, which is a learnable embedding vector determined by its degree, and $\mathbf{D} \in \mathbb{R}^{n \times d}$ denote the degree encoding of all assets in the graph. This encoding serves similarly to position encodings and is added to node features before feeding them into the attention mechanism. Therefore, the $\mathbf{X}^*$ can be replaced by $\tilde{\mathbf{X}}^* = \mathbf{X}^* + \mathbf{D}$.

Pair-wise Spatial Encoding In addition to the attention mechanism that comprehensively considers every pair of assets globally, we complement the local information by extracting pairwise structure insights from the graph, specifically targeting pairs of assets that are connected by a path. When two assets are not directly connected by an

edge but a path exists between them, this indirect connection implies that although the assets do not directly affect each other, they are linked through intermediary assets, shared market factors, or underlying economic dynamics. Analyzing these paths between assets allows us to uncover hidden connections and better understand the underlying structure of the financial market network. In this study, we utilize the shortest path distance between two assets as a key metric. This distance represents the most direct route between assets, considering how far the influence can be transmitted by the intermediary assets between them. Let $\mathbf{S} \in \mathbb{R}^{n \times n}$ denote the Pair-wise Spatial Encoding of the graph, where $S_{u,v}$ is a learnable scalar based on the shortest path distance between asset u and asset v. If no path exists between them, the shortest path distance will be represented by a large value, representing that they are not reachable from each other. This encoding is then incorporated as a piece of complementary information to the dot product in the attention; therefore, the attention matrix can be modified:

$$\mathbf{A}_h^* = {\mathbf{M_0}}^* \odot softmax(\frac{(\tilde{\mathbf{X}}^* \mathbf{W}_q^*)(\tilde{\mathbf{X}}^* \mathbf{W}_k^*)^\top}{\sqrt{d_k^*}} + \mathbf{S} + {\mathbf{M}_\infty}^*) \tag{6.5}$$

Edge-wise Correlation Encoding Recall that the edge attributes encapsulate multi-scale excess return correlation between assets. For two assets exhibiting strong long-term correlations, leveraging their edge attributes becomes imperative to reveal how their relationship evolves over time. It provides insights into whether the correlation remains consistent or fluctuates across various time frames, spanning from short-term to long-term durations.

$$E_{u,v} = \frac{1}{p}\sum_{i}^{p}(\mathbf{c}_{u,v}\mathbf{w}_e^{\top})_i, \quad \mathbf{c}_{u,v} = \begin{cases} \mathcal{E}_{u,v} & if \mathcal{A}_{u,v} \neq 0 \\ \mathbf{0} & if \mathcal{A}_{u,v} = 0 \end{cases} \tag{6.6}$$

$$\mathbf{A}_h^* = \mathbf{M_0}^* \odot softmax(\frac{(\tilde{\mathbf{X}}^*\mathbf{W}_q^*)(\tilde{\mathbf{X}}^*\mathbf{W}_k^*)^{\top}}{\sqrt{d_k^*}} + \mathbf{S} + \mathbf{E} + \mathbf{M}_\infty{}^*) \tag{6.7}$$

For a given pair of assets u and v, we first compute $\mathbf{c}_{u,v} \in \mathbf{R}^p$. If an edge exists between them, then $\mathbf{c}_{u,v}$ corresponds to their edge attributes $\mathcal{E}_{u,v}$; otherwise, it defaults to a zero vector. Subsequently, $\mathbf{c}_{u,v}$ is transformed by a learnable weight matrix $w_e \in \mathbb{R}^{p\times p}$ and then averaged across the scale dimension. Similarly to the Pair-wise Spatial Encoding, the Edge-wise Correlation Encoding $\mathbf{E}$ can be seamlessly integrated into the attention matrix, complementing the model's understanding of asset interactions within the topological framework.

6.3.3 Temporal module

Given the sequence of graph inputs $\mathbb{G} = (\mathcal{G}^{t-k}, ..., \mathcal{G}^t)$, upon passing through the Topological Module, we obtain a sequence of outputs $(\mathbf{Z}^{t-k}, ..., \mathbf{Z}^t)$. Each output embedding $\mathbf{Z}^t$ comprises a set of node embeddings $\mathbf{z}_u^t$ corresponding to each node $u \in \mathcal{V}$, Thus, for every node u, we obtain a sequence of historical node embeddings $\mathbf{H}_u = (\mathbf{z}_u^{t-k}, ..., \mathbf{z}_u^t) \in \mathbb{R}^{k\times d^*}$, encapsulating both global and local structure information. The Temporal Module aims to investigate patterns from the historical embeddings along the temporal dimension for each node individually. The node index u is omitted in the remaining discussion within this module for simplicity.

Similar to the Asset Influence Attention, we adopt the multi-head attention mechanism to capture temporal dynamics. The input $\mathbf{H}$ is firstly mapped by a weight matrices $\mathbf{W}^\star \in \mathbb{R}^{d^*\times d^\star}$, where superscript $\cdot^\star$ indicates temporal module, and

$d^\star$ denotes the hidden dimension in this module. Leveraging the time step due to the sequential nature of inputs, the position encoding is defined as $\mathbf{P} \in \mathbb{R}^{k \times d^\star}$ where each $\mathbf{P}_t \in \mathbb{R}^{d^\star}$ is a learnable embedding vector determined by its time index. It is added to $\mathbf{H}$, yielding $\tilde{\mathbf{H}} = \mathbf{H}\mathbf{W}^\star + \mathbf{P}$ which is then utilized to generate temporal attention matrix $\mathbf{A}^\star_{h'}$, where $h' \in \{1, ..., H'\}$ and H' represents the number of heads. Here, each element $A^\star_{i,j,h'}$ captures the interrelationship of node embeddings between time step i and j:

$$\mathbf{A}^\star_{h'} = softmax(\frac{(\tilde{\mathbf{H}}\mathbf{W}^\star_q)(\tilde{\mathbf{H}}\mathbf{W}^\star_k)^\top}{\sqrt{d^\star_k}} + \mathbf{M}_\infty{}^\star) \tag{6.8}$$

$$\mathbf{S} = \mathbf{H} + [\mathbf{A}^\star_1(\tilde{\mathbf{H}}\mathbf{W}^\star_v), ..., \mathbf{A}^\star_H(\mathbf{H}\mathbf{W}^\star_v)]\mathbf{W}^\star_o \tag{6.9}$$

Recognizing that embeddings from future time steps cannot be foreseen in the past, we address this by applying a mask matrix $\mathbf{M}_\infty{}^\star$ to mitigate such issues. For each element $M^\star_{ij}$, if time step i is ahead of j, then it is set to negative infinity to guarantee the softmax result is 0; otherwise it is set to 0. Additionally, we employ multiple heads in temporal attention, concatenating their outputs to obtain $\mathbf{S} \in \mathbb{R}^{k \times d^*}$, the final output embeddings from Temporal Module.

6.3.4 Node-level prediction task

Our task is to forecast the excess return for each node in the subsequent future time step. For each asset u, we derive the embeddings $\mathbf{S}_u = (\mathbf{S}^{t-k}_u, ..., \mathbf{S}^t_u)$ after undergoing temporal attention. To obtain the node-level prediction, we extract the embedding vector $\mathbf{S}^t_u \in \mathbb{R}^{d^*}$ at the final time step t, which wraps all structural information from related assets and historical patterns. A multi-layer perceptron (MLP) layer and tanh

activation function are then applied to generate $\tilde{y}_u$ in the range of (-1,1), representing the predicted excess return for asset u on the time step $t+1$.

To exclude non-existing nodes from consideration, we incorporate a mask $M^{(P)}$ into both target and predictions for all shared assets. This mask is assigned a value of 1 if asset u exists for both time steps $t+1$ and t. This criterion ensures that for predicting future returns, the asset must exist in the future time step and at least one historical data point immediately preceding it must be available; otherwise, the mask is set to 0. For optimization, we adopt Mean Squared Error Loss as our objective function.

$$\hat{y}_u = M^{(P)} \cdot \tilde{y}_u, \quad \tilde{y}_u = tanh(MLP(\mathbf{S_u})) \tag{6.10}$$

$$\mathcal{L} = \sum_{u \in \mathcal{V}} (\hat{y}_u - M^{(P)} \cdot y_u)^2 \tag{6.11}$$

6.4 Experiment

To comprehensively evaluate our proposed model and the effectiveness of node representations learned from dynamic graphs, we raise four research questions and conduct experiments from the following perspectives.

- **RQ1**: Does *DySTAGE* effectively learn dynamic graphs compared with other available *dynamic* and *static* approaches regarding asset return prediction?
- **RQ2**: Can the model *generate positive portfolio return* in real-world scenarios?
- **RQ3**: Does *graph representation* learned from *DySTAGE* effectively reflect asset patterns?
- **RQ4**: What is the contribution of each component in *DySTAGE*?

Table 6.1 Summary of Statistics for Dynamic Asset Graph Datasets

Statistic	**Russell 3000**	**MLFI**	**S&P 500**
# Snapshots	193	172	2,396
# Nodes	2,151	990	460
# Edges	1,290K	387K	123K
# Features	166	93	24
Lag	12	12	20
Horizon	1	1	1
Frequency	Monthly	Monthly	Daily
Range	2000/01 – 2021/12	2000/01 – 2019/03	2011/01/03 – 2020/12/31

6.4.1 Dataset

In this study, we utilize three dynamic asset graph datasets, each sampled at different frequencies — monthly and daily — to assess our model's performance with dynamic graphs.

Russell 3000 Monthly Dataset [129] includes monthly data for all Russell 3000 index constituents. The data provides security monthly returns for all index constituents, with 166 firm-specific characteristics that have been identified as proven predictors for future returns. This includes variables related to firm size, profitability, accruals, momentum, earnings surprises, intangibles, and trading frictions.

MLFI Monthly Dataset is publicly available from Machine Learning for Factor Investing (MLFI) [130], including 1,207 U.S.-listed stocks, with 990 stocks retained after data cleaning. It has 93 features on crucial company fundamentals such as financial ratios, cash flow metrics, and performance indicators. It also incorporates liquidity data and market capitalization figures.

Table 6.2 Descriptive Statistics for Selected Asset Pricing Features

Feature	**Mean**	**STD**	**Min**	**Max**
ExcessReturn	0.009	0.163	-0.995	24.00
VolSD	-4.548	35.52	-6121	-0.000
VolumeTrend	-0.006	0.020	-0.065	0.056
Accruals	-0.009	0.108	-6.265	5.750
Beta	0.974	0.716	-7.747	52.64
AbnormalAccruals	-0.002	0.135	-2.218	6.909
zerotrade	1.229	3.008	0.000	19.85
Activism1	14.88	49.00	6.000	23.00
VolMkt	-0.155	1.792	-1330	0.000
VarCF	-0.887	55.77	-20078	0.000

S&P 500 Daily Dataset covers daily market performance of 460 stocks from 2011 to 2020, providing detailed data on price movements and trading volumes. It enables the examination of short-term market dynamics and investor behaviors by tracking daily trends, patterns, and potential trading signals through fluctuations in stock prices, volumes, and volatility indicators.

The dynamic evolution of assets varies across all datasets: over 30% of asset change in Russell 3000, and over 20% in MLFI. While assets in S&P remain constant, the graphs still retain dynamism due to changing edge correlations. For all datasets, we use monthly (daily) excess returns after the risk-free rate as the target. The stock returns data are sourced from CRSP, while the fundamental variables come from Compustat. Each dataset offers unique variables suitable for the intended frequency and type of analysis. Detailed data descriptions are provided in Table 6.1. and selected feature statistics in Table 6.2. Datasets are split along the temporal

dimension into training, validation, and test sets, in a 70%:15%:15% ratio, with testing periods of 2020/01 - 2021/12, 2017/01 - 2019/03, and 2019/08 - 2020/12, respectively. Additionally, features are normalized to the range of $(-1, 1)$.

6.4.2 Baselines

We compare *DySTAGE* against a variety of traditional and state-of-the-art time series and GNN models. Our return prediction benchmarks are categorized into three main types.

Time series models: ARIMA [131] is a traditional financial forecasting model, and N-Beats [132] is a neural basis expansion model for interpretable time series forecasting.

Static GNN approaches: Among static GNN models, we employ GAT [66], GraphSAGE [65], ARMAConv [133], and UniMP [67]. These models differ in their approaches to aggregating network information. For instance, GAT and GraphSAGE aggregate neighboring embeddings, ARMAConv leverages recurrent neural networks (RNNs), while UniMP utilizes a self-attention mechanism. These models handle only static graphs and cannot deal with temporal patterns. Hence, we utilize the latest historical graph as input with all historical features as node features.

Dynamic GNN models: Our selection includes RNN-based models such as T-GCN [72], EvolveGCN [75], and GCLSTM [73], as well as attention-based models DY-GAP [70] for asset pricing and DySAT [74] utilizing self-attention to learn node representations. We also include two state-of-the-art models, DyTed [76] in a contrastive learning framework, and DGIB [77] for robust representations.

6.4.3 Implementation details

DySTAGE is implemented in PyTorch. Hyperparameters are tuned on the validation set. The window size for Pearson correlation w is 36 and 61, representing 3 years and 3 months, and the historical time step T is set to 12 and 20 for monthly and daily

Table 6.3 Comparison Results for Asset Pricing

Model	Russell 3000			MLFI			S&P 500		
	RMSE	**MAE**	**MAPE**	**RMSE**	**MAE**	**MAPE**	**RMSE**	**MAE**	**MAPE**
ARIMA	0.1798	0.1422	117.8364	0.1254	0.0970	87.0359	0.0223	0.0178	17.8370
N-Beats	0.1327	0.1050	83.8827	0.1005	0.0793	69.7604	0.0195	0.0158	15.8619
GAT	0.1073	0.0874	66.3047	0.0802	0.0651	55.6068	0.0154	0.0131	13.1286
GraphSAGE	0.1060	0.0864	65.4026	0.0811	0.0656	56.0802	0.0156	0.0131	13.1464
ARMAConv	0.1081	0.0877	66.2636	0.0808	0.0653	55.8012	0.0158	0.0133	13.2557
UniMP	0.1078	0.0853	64.1338	0.1078	0.0853	64.1331	0.0156	0.0134	13.3536
DySAT	0.1039	0.0840	63.2542	0.0806	0.0652	55.7200	0.0155	0.0132	13.2118
DY-GAP	0.1357	0.1089	87.7335	0.0806	0.0652	55.7378	0.0155	0.0131	13.0723
T-GCN	0.1078	0.0882	67.0031	0.0813	0.0657	56.0604	0.0168	0.0145	14.5104
EvolveGCN	0.1064	0.0845	63.5845	0.0806	0.0651	55.5625	0.0155	0.0132	13.1989
GCLSTM	0.1033	0.0839	62.9582	0.0807	0.0650	55.5467	0.0156	0.0134	13.4025
DyTed	0.1040	0.0844	63.3368	0.0800	0.0647	55.3379	0.0155	0.0132	13.1666
DGIB	0.1031	0.0837	62.8114	0.0802	0.0649	55.4483	0.0154	0.0131	13.0751
DySTAGE	**0.1026**	**0.0833**	**62.5027**	**0.0797**	**0.0644**	**54.9632**	**0.0154**	**0.0131**	**13.0602**

Note: Performance of benchmarks and our model. MAPE results are in the form of percentages (%). The top panel includes time series models, the middle one consists of static graph models, and the bottom panel is for dynamic graph models.

datasets, respectively. The threshold of correlation γ for graph construction and the adjacency matrix is 0.3. We employ the AdamW optimizer for training, running for up to 300 epochs with early stopping after 30 epochs. The learning rate is 0.0001. Both modules use 16 heads and a concatenated hidden dimension of 128. For other baselines, we fine-tune the hyperparameters based on optimal values reported in their studies and choose the best-performing model for implementation.

6.4.4 Comparison of baselines (RQ1)

We implement *DySTAGE* and benchmarks on all monthly and daily datasets. As per common practice, model performance is evaluated using three metrics: Root Mean Absolute Error (RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE). Table 6.3 presents the numerical results, and we draw the

Table 6.4 Portfolio Management Results

Model	Russell 3000			MLFI			S&P 500		
	CR	**AR**	**SR**	**CR**	**AR**	**SR**	**CR**	**AR**	**SR**
ARIMA	42.1388	20.1368	1.0047	0.9074	0.5435	0.1198	17.6767	12.1484	0.7275
N-Beats	49.5191	23.3519	1.1667	9.2134	5.0084	0.4069	25.9006	18.0936	1.0282
GAT	42.7684	20.4142	1.1837	8.0401	4.7492	0.3549	10.5096	7.4825	0.5219
GraphSAGE	49.8937	23.5131	**1.2077**	-0.7568	-0.4547	0.0483	24.5304	17.1641	0.9445
ARMAConv	25.7004	12.6751	0.7184	-0.5407	-0.3247	0.0675	14.1315	10.0146	0.6152
UniMP	40.0290	19.2031	1.0231	5.8182	3.4514	0.2748	15.6653	1.0802	0.7083
DySAT	37.9606	18.2812	1.0156	-0.3293	-0.1977	0.0808	23.3353	16.3512	0.9530
DY-GAP	40.8272	19.5572	1.1609	9.8075	5.7740	0.4285	19.0134	13.3927	0.8330
T-GCN	35.7086	17.2698	0.9543	7.6595	4.5486	0.3626	6.0337	4.3211	0.3410
EvolveGCN	29.7708	14.5642	0.8721	2.2540	1.3464	0.1611	5.1677	3.7051	0.2837
GCLSTM	41.9357	20.4726	1.0616	-3.2744	-1.9777	-0.0330	8.0919	5.7793	0.4216
DyTed	40.1514	19.2575	0.9667	-0.7692	-0.4622	0.0643	9.4853	6.5678	0.5208
DGIB	29.2723	14.3344	0.9369	6.1102	3.6226	0.3047	8.5284	6.0876	0.4149
DySTAGE	**50.3428**	**23.7152**	1.1975	**10.2829**	**6.0486**	**0.4614**	**31.5506**	**21.8969**	**1.2945**

Note: CR and AR are in the format of percentage (%).

following observations. (1) *DySTAGE* stands out as the most reliable and accurate model across diverse datasets, consistently surpassing time series models, static GNNs, and dynamic benchmarks across all evaluation metrics, especially regarding MAPE. It achieves an impressive MAPE of 62.50% and 54.96% on Russell and MLFI, and 13.06% on S&P daily assets. (2) Time series models fall short in performance as they rely solely on a historical series of excess returns, disregarding other fundamental features and failing to capture the intricate relationship between assets in the financial networks. (3) Among static GNNs, UniMP and GAT demonstrate superior performance in terms of MAPE. Nevertheless, static GNNs generally fall behind dynamic counterparts across multiple datasets, primarily due to their lack of temporal dependencies and exclusive focus on fixed topological information. (4) In dynamic GNNs, other than *DySTAGE*, DyTed and DGIB emerge as the most powerful models for two monthly datasets, while DY-GAP achieves the lowest errors on S&P daily

data. This suggests the superiority of dynamic models in capturing inherent variations in stock market data. *DySTAGE* outperforms other dynamic GNNS due to its enhanced structural attention and graph encodings. (5) Notably, all models exhibit better performance on daily S&P data compared with the two monthly datasets. This can be attributed to the lower volatility and more predictable patterns observed in daily prices, facilitating more accurate predictions due to smoother and more consistent trends. Therefore, the difference in performance among models on S&P data is smaller compared to monthly data, yet *DySTAGE* maintains its superior performance in terms of MAPE.

6.4.5 Portfolio management (RQ2)

To demonstrate how *DySTAGE* provides insights for investors and generates profits, we utilize our prediction results to construct portfolios from the dynamic asset pool and observe the returns. We conduct portfolio management on GNNs only due to their comparable performance. At the beginning of each period, we invest long positions in the top 10% assets with the highest predicted excess returns and assign them equal weight, then liquidate them at the end of the period. The portfolio performance is evaluated by the following metrics:

1. **Cumulative Return (CR)** is the total return achieved over testing period.
2. **Annual Return (AR)** is the percentage change of equity over 1-year period.
3. **Sharpe Ratio (SR)** is a risk-adjusted metric to calculate the excess return relative to its risk. A higher value indicates a more favorable strategy.

The numerical results are shown in the table 6.4. We have the following observations: (1) *DySTAGE* consistently generates the highest profits, achieving cumulative returns of 50% on Russell, 10% on MLFI, and 31% on S&P. This demonstrates *DySTAGE*'s ability to offer lucrative investment recommendations in real-world scenarios. (2) *DySTAGE* achieves a Sharpe Ratio exceeding 1 on S&P.

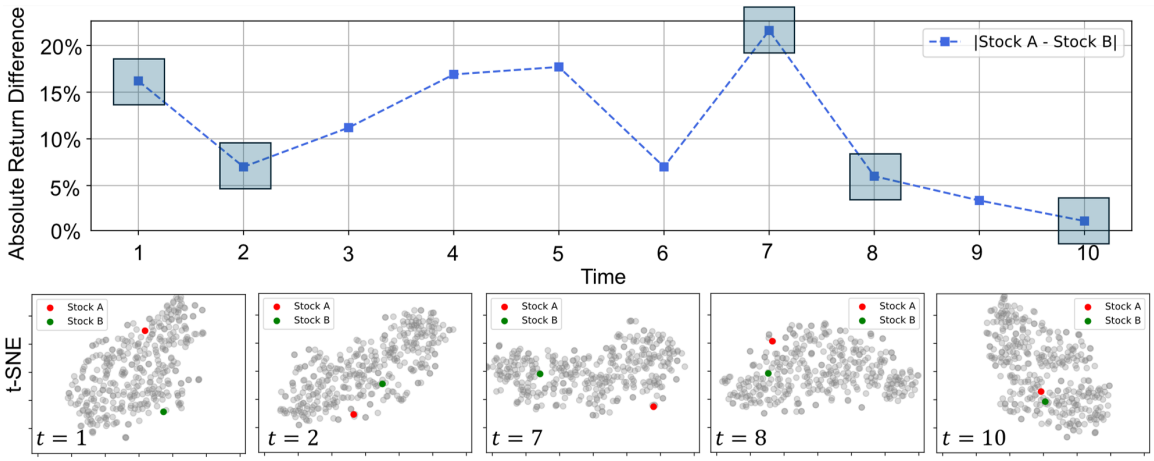


Figure 6.3 Similarity comparison between node embeddings from *DySTAGE* and ground truth.

Although GraphSAGE exhibits the highest Sharpe ratios on Russell, *DySTAGE* maintains a competitive Sharpe ratio close to 1.2, indicating a strong balance between profitability and risk management. (3) N-Beats is competitive in portfolio management but weak in prediction performance. This may be because it learns parameters for each asset series individually, which leads to less accurate predictions, but top-return assets can be identified and fall into the top 10% group. (4) Portfolio performance on MLFI is generally poorer due to higher volatility compared to the other datasets. DY-GAP and GAT show robustness during volatility periods but carry higher associated risk compared with *DySTAGE*. Conversely, GraphSAGE and DySAT perform well during the recovery period marked by an upward trend but struggle more with volatility. It is worth noting that *DySTAGE* showcases stability across all test periods.

6.4.6 Graph learning (RQ3)

Figure 6.3 illustrates *DySTAGE*'s capability to accurately capture and represent temporal dynamics within financial markets. The top panel, which depicts the absolute return differences between two selected stocks over ten months, establishes a baseline for financial performance fluctuations between these assets. The bottom

Table 6.5 Ablation Study Results

Model	Russell	MLFI	S&P
w/o Importance	62.7537	55.3018	13.0674
w/o Temporal	62.6115	55.0411	13.1801
w/o Spatial	62.5868	54.8906	13.0734
w/o Edge	62.5943	54.9654	13.0690
DySTAGE	**62.5027**	**54.9632**	**13.0602**

Note: All results are based on MAPE.

panel, a t-SNE visualization of the *DySTAGE*'s node embeddings, shows the relative positioning of these two stocks within the market's broader context at critical timestamps. Notably, the distance between the two stocks in this embedding space is consistent with observed absolute return differences, suggesting that the spatial distribution in the model's embeddings effectively mirrors the actual financial performance disparities. This consistency supports the conclusion that *DySTAGE* exhibits robust graph representational learning capabilities. By effectively mapping the temporal changes and maintaining fidelity to the ground truth, the model demonstrates its potential utility for predictive analytics in finance. The accuracy of these embeddings in reflecting true market behavior highlights the effectiveness of *DySTAGE* as a powerful tool for financial analysis and decision-making.

6.4.7 Ablation study (RQ4)

To quantify the improvement from each component, we implement the following variants of *DySTAGE* to examine the impact of removing each component:

- ***DySTAGE* w/o Importance** removes Asset-wise Importance Encoding
- ***DySTAGE* w/o Spatial** removes Pair-wise Spatial Encodings
- ***DySTAGE* w/o Edge** removes Edge-wise Correlation Encodings

- ***DySTAGE* w/o Temporal** replaces the Temporal module with a simple MLP layer

From Table 6.5, it is obvious that the temporal module significantly boosts the performance, especially a decrease of 0.11% on Russell and 0.12% on S&P in MAPE. Moreover, it demonstrates that *DySTAGE* equipped solely with the Topological Module is remarkably powerful, outperforming almost all benchmarks. In addition, all graph encodings consistently contribute to the model performance. Among them, *Asset-wise Importance Encoding* proves to be the most influential, especially on monthly data. Removing it leads to a 0.25% increase in Russell and a 0.34% in MLFI. Additionally, *Pair-wise Spatial Encoding* effectively improves the MAPE of *DySTAGE* by revealing hidden connections between asset pairs.

6.5 Conclusion

This research addresses a critical gap in the existing asset pricing literature. While prior research focuses on learning the interrelationships among assets within financial networks, it often overlooks how asset composition and relationships change over time. To bridge this gap, we introduce *DySTAGE*, a novel framework for dynamic graph representation learning in asset pricing from two aspects: the first universal formulation from asset time series to dynamic graphs accomodating changes in asset composition and relationships, and a graph learning model enriched with spatio-temporal and graph encodings with financial insights to capture the evolving nature of financial networks over time. Extensive experiments on monthly and daily assets demonstrate that *DySTAGE* outperforms popular benchmarks in price prediction and offers valuable insights for profitable investment strategies in real-world scenarios.

Future research may enhance *DySTAGE*'s dynamic feature-capturing capabilities by integrating multimodal data, such as news sentiment and economic indicators, for richer market context. Additionally, incorporating advanced

techniques, including graph generative adversarial networks and Large Language Models, could improve the model's adaptability, predictive precision, and and interpretability in response to evolving market structures.

CHAPTER 7

MARGIN TRADER: A REINFORCEMENT LEARNING FRAMEWORK FOR PORTFOLIO MANAGEMENT WITH MARGIN AND CONSTRAINTS

7.1 Introduction

Portfolio management plays a critical role in the dynamic landscape of the stock market. Investors curate a diverse selection of financial assets, including stocks, futures, bonds, foreign currency, and cryptocurrencies. These assets are periodically reallocated, guided by the overarching objective of maximizing capital growth over an extended time horizon while simultaneously minimizing risks [134, 135]. With the widespread utilization of deep learning within the financial market [12, 13, 87], Deep Reinforcement Learning (DRL) [136] has emerged as a prominent and widely adopted approach for portfolio management, primarily owing to its inherent capacity for sequential decision-making [137, 138, 139, 140]. In contrast to conventional investment strategies, DRL-based approaches have demonstrated considerable potential in adapting to dynamic market conditions and attaining superior returns [141, 142, 143, 144].

The majority of existing works using DRL have been confined to trading with their own cash, limiting the exploration of *margin accounts* that allow investors to amplify their exposure to the market and potentially enhance their returns through leveraging. By enabling both long and short positions, margin trading opens up a wide range of strategies, including hedging, short-selling, and arbitrage, which can help traders manage risk and capitalize on diverse market trends [145]. However, the potential for excessive leverage can amplify losses and introduce greater volatility into the portfolio. Proper risk management becomes critical when using margin trading. Margin accounts have constraints, such as initial and maintenance margins, to prevent such situations [146]. Some studies assumed that borrowing stocks from brokers has

no limitations, even when using a zero-investment strategy (i.e., the market value for long positions equals that of short positions) [31], ignoring that the short sale requires the margin account in reality. Therefore, the incorporation of margin accounts into RL-based portfolio management is still under-explored in the current research, and it is essential for creating more realistic and practical strategies. Designing mechanisms to ensure compliance with constraints and regulate investors' risk is also a non-trivial task.

To fill these gaps, we developed *Margin Trader*, a foundational and flexible reinforcement learning framework tailored for margin trading in the stock market. Our framework goes beyond the cash-only trading approach by incorporating margin accounts and constraints via financial data engineering solutions, thereby allowing traders to leverage their positions in both long and short directions. This integration provides a more realistic trading environment. The primary objective of *Margin Trader* is to strike a delicate balance between profit maximization and risk management. To achieve it, we implement two key modules: the *Margin Adjustment Module* timely updates the buying power to ensure traders' maximum potential returns, and the *Maintenance Detection Module* protects the stability of the portfolio by promptly alerting traders when the margin level approaches critical points.

One of *Margin Trader*'s key strengths lies in its exceptional flexibility. It allows traders to customize crucial settings for the trading environment. Traders can adapt their data-centric strategies to ever-changing market conditions by fine-tuning equity allocation to long and short positions, manage risk tolerance levels by adjusting maintenance requirements, and cater to either conservative or aggressive trading styles based on their individual preferences by controlling margin ratios. This level of customization enables traders to identify more profitable opportunities while effectively mitigating unnecessary risk. Moreover, although this study focuses on using stocks as trading securities for presentation, *Margin Trader* is versatile and can

be extended to encompass various financial markets and instruments, such as futures and cryptocurrency markets. By appropriately defining the trading environment, setting relevant parameters, and adapting the RL algorithms, the framework can be tailored to suit the characteristics of different asset classes, making it applicable in a broader range of financial scenarios.

To comprehensively quantify the performance of *Margin Trader*, we implement *Margin Trader* in the portfolio pool of Dow Jones Industrial Average (DJIA) from 2020/05 to 2023/05, encompassing the COVID pandemic recovery in 2020 and market volatility in 2022. *Margin Trader* enables trading on both long and short positions and outperforms cash-only trading baselines by achieving the lowest volatility (0.163) while the highest Sharpe ratio (1.16) in both bullish and bearish markets. Even in the long position alone, it exhibits a remarkable threefold return (1.234) compared with the DJIA index (0.437) and the highest Calmar ratio (1.199). To evaluate the *Margin Trader* with various market conditions to demonstrate its generalization and robustness, *Margin Trader* is also assessed under rolling-based testing periods, which often lack in existing works for portfolio management evaluation. The superior performance highlights *Margin Trader*'s effective learning of profitable and safer trading strategies, making it a promising option for traders seeking to optimize their trading approaches. Our contributions are summarized as follows:

1. We propose *Margin Trader*, the first data-driven framework that integrates margin account and constraints into reinforcement learning for portfolio management.
2. *Margin Trader* provides versatility, enabling traders to customize various settings to align with market conditions, risk tolerance, and individual preferences.
3. The experiment in rolling-based testing periods showcases that *Margin Trader* effectively learns trading strategies that employ margin to amplify profits and hedge risks simultaneously, both in bullish and bearish market conditions.

7.2 Preliminary

7.2.1 Trading using a margin account with long and short positions

This research introduces a more realistic trading setting, allowing trading on a margin account with both long and short positions. The portfolio pool contains equity positions, enabling traders to either long a stock when expecting price increases or short a stock to profit from anticipated price declines. Utilizing a margin account, trading is not limited to cash; instead, traders can borrow funds or stocks from brokers, increasing their trading capacity [146].

The initial margin requirement, typically set at 50%, necessitates a deposit into the margin account. In a long position, with a deposit D^l, traders can borrow up to D^l in cash as a loan and purchase stocks valued at a maximum of $2D^l$. In a short position, with a deposit D^s, they can borrow stocks valued at $2D^s$ from brokers, resulting in a total credit balance of $3D^s$. A 50% margin requirement implies a leverage of 2, amplifying any price fluctuation's impact on the portfolio value. If the equity falls below the maintenance margin requirement (usually 40% of the holding market value), a margin call is triggered, requiring additional cash deposits or position closure. As the margin account's value increases, additional buying power is obtained, allowing further borrowing without additional cash deposits.

The timeline is divided into holding periods, during which traders adjust their portfolios based on price fluctuations. At the beginning of the holding period t, traders hold a portfolio of N stocks with $n_{i,t}$ shares for stock i at a price of $p_{i,t}$. Throughout the holding period, the fluctuation in prices leads to the change in equity value from e_t to e_{t+1}. At the end of the holding period $t+1$, traders make decisions regarding buying or selling stocks to adjust the portfolio accordingly. If a margin call occurs at the first time, new trading is forbidden; for the second time, the position will be closed. The details about how to handle margin calls are presented in Subsections 7.3.3 and 7.3.3.

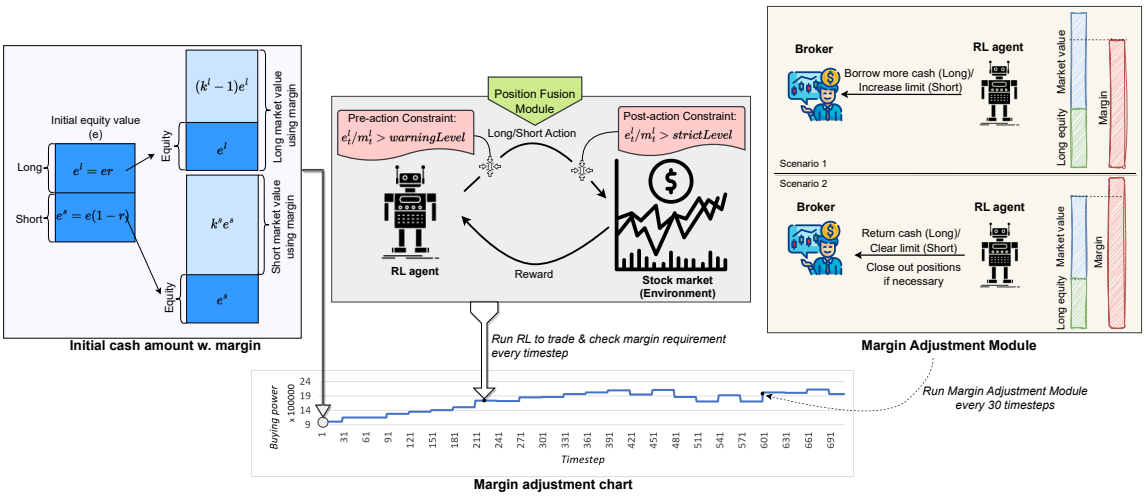


Figure 7.1 The interaction process between agent and environment in the *Margin Trader*.

7.2.2 Assumptions

In line with existing research [26, 33, 34], we adopt the following assumptions.

Zero slippage: We assume that the market exhibits high liquidity, enabling trades to be executed instantly at the specified order price. Consequently, the actual trade price aligns precisely with the price indicated in the order.

Zero market impact: Our investment capital remains relatively small, exerting minimal influence on the market. As a result, our trading activities do not disrupt the supply and demand dynamics or significantly affect market prices.

Transaction cost: Each trade incurs an additional fee known as transaction cost, denoted as c for either buying or selling shares of stocks.

7.3 Margin Trader

Margin Trader is tailored for real-world trading scenarios involving margin accounts with both long and short operations. It harnesses the power of Reinforcement Learning to equip its agent with the ability to effectively observe and interact with the dynamic trading environment, enabling the learning and development of robust

trading strategies. We delve into the core functionalities and modules of *Margin Trader* in this section. Figure 7.1 provides a big picture of *Margin Trader*.

7.3.1 Customizable settings and notations

Margin Trader provides traders with customizable settings, allowing flexibility to adapt to prevailing market conditions and specific requirements:

Ratio of long position to equity (r): Traders can customize the allocation of the initial capital value to long and short positions at t_0, according to their preferences. This ratio, denoted by r, determines the distribution of equity between the two positions and remains fixed during trading. For a given ratio r for the long position, the initial cash in the long position is $e^l = er$, while the initial cash in the short position is $e^s = e(1 - r)$.

Initial margin requirement (k): The initial margin requirement, denoted by k, governs the extent to which traders can leverage their equity. When allocating an equity amount of e^l to the long position, traders can borrow up to $(k^l - 1)e^l$ in cash and trade stocks with a market value of up to $k^l e^l$. Similarly, in the short position, with a deposit of e^s, traders can borrow stocks from brokers with a market value of up to $k^s e^s$. For our experiments, we set $k^l = k^s = 2$, resulting in a leverage ratio of 2.

In addition to transaction cost c, we denote the initial equity value as e. That means, at the start of trading, the trading account is initialized with a cash amount of e.

7.3.2 Reinforcement learning framework

In this study, we model the margin trading scenario as a Markov Decision Process (MDP) to apply reinforcement learning (RL). The RL agent acquires decision-making capabilities by interacting with an environment represented as an MDP. The MDP consists of several components, including state space S, action space A, transition probabilities P, a reward function R, and a discount factor γ. At each time step, the

agent observes a state S_t, selects an action A_t based on a policy π, and transitions to a new state S_{t+1} according to the transition probability $T(S_{t+1}|S_t, A_t)$. The agent receives a reward R_t for its action. The action-value function Q^π evaluates the quality of taking a specific action in a given state. It is expressed as the expected cumulative reward:

$$Q^\pi(s, a) = \mathbb{E}[\sum_{i=t}^{\infty} \gamma^i r_{t+i} | s_t = s, a_t = a] \tag{7.1}$$

The goal of the RL agent is to learn the optimal policy π^* that maximizes the action-value function Q^π.

State The state representation in *Margin Trader* is designed to capture both the agent's equity condition and the macro market environment at the current time step t, enabling effective portfolio tracking and management. It is represented by the state vector $S_t = [b_t^l, l_t^l, e_t^l, l_t^s, b_t^s, e_t^s, \mathbf{p}_t, \mathbf{n}_t, \mathbf{t}_t]$, where each component carries specific information:

1. Available cash balance in long position b_t^l: the remaining cash in the long position available for purchasing stocks.
2. Loan l_t^l: the amount of cash borrowed by the agent.
3. Long equity e_t^l: the equity held by the agent in the long position. It is the value of the holding asset (available cash and holding long market value m_t^l) after returning the borrowed loan:

$$e_t^l = b_t^l + m_t^l - l_t^l \tag{7.2}$$

4. Available limit in short position l_t^s: the remaining market value that can be shorted.
5. Credit balance b_t^s: the sum of the initial limit of borrowed funds and initial deposit.
6. Short equity e_t^s: the equity held by the agent in the short position. It is calculated by subtracting the credit balance from the available limit and holding

the short market value m_t^s:

$$e_t^s = b_t^s - m_t^s - l_t^s \tag{7.3}$$

7. Close prices $\mathbf{p}_t = \{p_{i,t}\} \in \mathbb{R}^N$: close price for each stock i.
8. Holding shares $\mathbf{n}_t = \{n_{i,t}\} \in \mathbb{R}^N$: the number of shares held by the agent for each stock i. The value of $n_{i,t}$ indicates the agent's position in the stock i. A positive value shows that it is in a long position. A negative value signifies that it is in short positions. A value of 0 implies that it is not traded in either long or short positions.
9. Technical indicators $\mathbf{n}_t = \{\mathbf{n}_{i,t}\} \in \mathbb{R}^{4N}$: the technical indicators associated with each stock. Following the specific settings in [147], $\mathbf{n}_{i,t}$ includes Moving Average Convergence Divergence (MACD), Relative Strength Index (RSI), Commodity Channel Index (CCI), and Average Directional Index (ADX) for the stock i.

In the reset step, the agent allocates the initial equity to long and short positions. In the long position, the agent deposits the initial equity e^l and borrows a loan of $l_0 = (k^l - 1)e^l$. Hence, the available cash for stock purchases is $b_0^l = k^l e^l$. In the short position, the agent provides an initial deposit of e^s and the initial limit for short-selling is $l_0^s = k^s e^s$, hence, the credit balance reaches $b_0^s = (k^s + 1)e^s$. As no stocks are initially held, all elements in $\mathbf{n}_0$ are initialized to 0.

Action $A_t = [\mathbf{a}_t^l, \mathbf{a}_t^s, \mathbf{a}_t^r]$ consists of actions for long position $\mathbf{a}_t^l = \{a_{i,t}^l\} \in \mathbb{R}^N$, short position $\mathbf{a}_t^s = \{a_{i,t}^s\} \in \mathbb{R}^N$ and a single action for long/short ratio changes $\mathbf{a}_t^r = \{a_{i,t}^r\} \in \mathbb{R}^N$. For each stock i, the agent chooses an integer from a discrete space $A^l = \{-h, ..., 0, ..., h\}$ and $A^s = \{-h, ..., 0, ..., h\}$as the action defined as the number of shares to trade. A positive value for $a_{i,t}^l$ or $a_{i,t}^s$ represents the number of shares to buy in long or buy in short, and a negative value implies selling the stock in long or short. h is a parameter to control the maximal quantity of shares that can be traded for a single stock at each step. The action to change the long and short ratio is a continuous sequence of natural numbers in the range between $-h^r$ and h^r: $A^r = \{-h^r, ..., 0, ..., h^r\}$. A positive number indicates an increase in the proportion of equity in the long position, and conversely, a negative number indicates an increase

in the proportion of the short position. The action representation enables the agent to make discrete decisions, providing flexible control over the trading strategy and maintaining interpretability.

Reward We define the reward function by profit and risk with reward shaping:

$$R_t(S_t, A_t, S_{t+1}) = \lambda_1 profit_t + \lambda_2 risk_t \tag{7.4}$$

The profit is determined by the difference in equity between two consecutive time steps: $e_{t+1} - e_t$. It can be decomposed into two parts: transaction costs incurred due to trading actions and price changes in the macro market. First, the agent adjusts the portfolio based on the action A_t. The quantity of holding shares is updated to $\mathbf{n}_t + \mathbf{a}_t^l + \mathbf{a}_t^s$. The traded market values are given by $\sum_i |a_{i,t}^l| p_{i,t}$ for long position and $\sum_i |a_{i,t}^s| p_{i,t}$ for short position, which impact the available cash balance b_t^l and available limit in short sale l_t^s, respectively. The transaction cost associated with the trading activities affects the long equity, short equity, available cash balance, and credit balance accordingly. As the time step progresses to $t+1$, the price fluctuates to $\mathbf{p}_{t+1}$, resulting in the updates to the holding market value in long and short positions:

$$m_{t+1}^l = \sum_{i, n_{i,t+1} > 0} n_{i,t+1} p_{i,t+1} \tag{7.5}$$

$$m_{t+1}^s = \sum_{i, n_{i,t+1} < 0} |n_{i,t+1}| p_{i,t+1} \tag{7.6}$$

The available cash balance, loan, available limit in the short sale, and credit balance remain unchanged. Consequently, the next state S_{t+1} can be observed and the profit can be calculated:

$$b^l_{t+1} = b^l_t - \mathbf{a}^{l\top}_t \mathbf{p}_t - c \sum_i |a^l_{i,t}| p_{i,t} \tag{7.7}$$

$$e^l_{t+1} = b^l_{t+1} + m^l_{t+1} - l^l_{t+1} \tag{7.8}$$

$$l^l_{t+1} = l^l_t \tag{7.9}$$

$$b^s_{t+1} = b^s_t - c \sum_i |a^s_{i,t}| p_{i,t} \tag{7.10}$$

$$e^s_{t+1} = b^s_{t+1} - l^s_{t+1} - m^s_{t+1} \tag{7.11}$$

$$l^s_{t+1} = l^s_t + \mathbf{a}^{s\top}_t \mathbf{p}_t \tag{7.12}$$

$$\mathbf{n}_{t+1} = \mathbf{n}_t + \mathbf{a}^l_t + \mathbf{a}^s_t \tag{7.13}$$

$$profit_t = e_{t+1} - e_t = (e^l_{t+1} + e^s_{t+1}) - (e^l_t + e^s_t) \tag{7.14}$$

The risk component of the reward function is represented by the Sharpe ratio [148], a widely used metric for assessing the risk-adjusted return of a portfolio. We calculate the Sharpe ratio based on the equity values of the most recent five steps:

$$SR = \frac{AR_p - AR_f}{\sigma_p} \tag{7.15}$$

where AR_p is the average return of the portfolio, AR_f is the risk-free rate, and σ_p is the standard deviation of the return of the portfolio.

The reward function provides a comprehensive evaluation of the agent's trading performance. It incorporates actual profits, accounting for transaction costs and the market value changes, while also factoring in the risk profile of the portfolio.

DRL Algorithms *Margin Trader* is designed to accommodate a variety of Deep Reinforcement Learning (DRL) algorithms to facilitate effective learning and interaction with the trading environment. There are three main categories of algorithms widely used for stock trading: policy-based [149], value-based [150, 151], and actor-critic-based algorithms [152, 153]. In *Margin Trader*, we choose actor-critic models due to their combination of value-based and policy-based approaches. Specifically, an actor-critic algorithm employs both a critic and an actor. The critic estimates the action-value function, while the actor learns the policy distribution in the direction guided by the critic. Both actor and critic are parameterized with neural networks. We apply four popular actor-critic models, Advantage Actor-Critic (A2C) [154], Proximal Policy Optimization (PPO) [155], Deep Deterministic Policy Gradient (DDPG) [156], and Soft Actor-Critic (SAC) [157]. In *Margin Trader*, the actions generated from the actor-network are normalized in the range of $[-1, 1]$ and then scaled by h to determine the number of shares to trade.

7.3.3 Optimization and margin handling

Margin Trader incorporates modules to optimize trading and ensure compliance with margin requirements. These modules focus on enhancing financial stability, minimizing risks, and maximizing the efficient use of leverage for higher profit.

Position Fusion Module The action output from the RL agent can be long and short on the same stock. To manage this conflict, we enforce a rule that stocks can only be traded either in a long position or a short position, but not both simultaneously. Once the action A_t is generated, it is passed through a *Position*

Fusion Module. For each stock i, two actions $a^l_{i,t}$ and $a^s_{i,t}$ are produced, corresponding to long and short positions, respectively. The signs of two actions reflect the market sentiment and are consistent between the two positions. For instance, a negative value in the long position indicates selling the stock to avoid potential losses, while in the short position, it represents selling the stock to profit from the anticipated price declines. This alignment ensures that the actions correspond to the intended market direction for each position. Therefore, we sum them up to obtain the net quantity $a^{'}_{i,t} = a^l_{i,t} + a^s_{i,t}$ and assign it to the current holding position. If $n_{i,t} > 0$, indicating the stock is already held in a long position, then $a^l_{i,t} = a^{'}_{i,t}$ and $a^s_{i,t} = 0$. Conversely, if $n_{i,t} < 0$, indicating a short position, then $a^s_{i,t} = a^{'}_{i,t}$ and $a^l_{i,t} = 0$. In cases where the stock has not been traded before, and the agent intends to initiate a trade in the current time step, the positive net quantity is assigned to the long position, while the negative quantity is assigned to the short position. After passing through the *Position Fusion Module*, the updated action is executed by the agent to make adjustments to the portfolio.

Margin and Buying Power Adjustment This module is designed to regularly update the buying power, allowing traders to leverage their positions to maximize potential returns. The maximal amount of loan that can be borrowed and the maximal market value that can be shorted in the *Margin Trader* are initially determined based on the initial equity e. However, as the market fluctuates and the equity value changes, it becomes necessary to update these values to ensure they align with the agent's current holding equity. To facilitate it, we design a *Margin Adjustment Module*. It is activated for periodic evaluation and updates of positions.

For the long position, the module begins by comparing the borrowed loan l^l_t with the maximal amount of loan that can be borrowed based on the current equity $(k^l - 1)e^l_t$. This comparison leads to two different scenarios. (1) If the borrowed loan

is lower than the maximal limit, it indicates that the agent has earned a profit during this period. The increased equity provides additional buying power, prompting the agent to enlarge the loan. By borrowing more cash, the agent leverages its position further and aligns itself with the desired margin ratio:

$$l_t^l = (k^l - 1)e_t^l \tag{7.16}$$

$$b_t^l = k^l e_t^l - m_t^l \tag{7.17}$$

(2) Conversely, if the borrowed loan exceeds the maximal limit, it implies that the agent has experienced a loss in their holdings. In this situation, the agent must take appropriate actions to reduce the loan and align it with the initial margin requirement $(k^l - 1)e_t^l$. This process involves two steps: Firstly, if the agent holds available cash, then return it to reduce the borrowed funds and bring the loan in line with the initial margin requirement. Otherwise, if returning available cash is insufficient to meet the margin requirement, the agent is forced to progressively close out positions to generate necessary funds. The closure process starts with the stock with the lowest holding market value, prioritizing the liquidation of the least significant portfolio positions, and ensuring the loan is reduced to the required level ultimately. When dealing with the last asset in this process, two options arise: (a) Partial closure involves liquidating a minimum portion of the position or shares to satisfy the margin requirement, or (b) complete closure entails closing out all positions for this asset to ensure compliance with the margin requirement.

Similarly, for the short position, the module compares the current credit balance b_t^s with the maximal credit balance based on current equity $(k^s+1)e_t^s$. (1) If the current credit balance is lower than the maximal limit, the agent is permitted to sell more stocks in the short sale. Consequently, the credit balance is increased to $(k^s + 1)e_t^s$,

and the available limit for short sale l_t^s is adjusted to fill the difference:

$$b_t^s = (k^s + 1)e_t^s \tag{7.18}$$

$$l_t^s = k^s e_t^s - m_t^s \tag{7.19}$$

(2) If the current credit balance exceeds the maximum limit, the module takes steps to reduce it. Firstly, the agent clears the available limit l_t^s to decrease the credit balance. Otherwise, if it is not sufficient, the agent is forced to progressively close out the positions by buying back the stocks and returning them to the brokers. The process starts from the stocks with the lowest shorted market value and continues until the credit balance is reduced to meet the necessary requirements. Liquidation of the last asset also has two options: partial closure or complete closure.

The update cycle is flexible, allowing updates every n_{count} ($n_{count} \in \mathbb{Z}+$) days, weeks, months, or years. Since trades operate in real-world time, the module's update schedule synchronizes with the actual calendar to maintain alignment with real-world dates. For example, if the model is set to update every two months, there would naturally be fewer trading dates during February's cycle due to its fewer days compared to other months.

By incorporating the *Margin Adjustment Module*, *Margin Trader* dynamically adjusts the loan and credit balance in long and short positions to maintain alignment with the agent's current equity. This adaptive approach optimizes the leveraging of profitable periods while mitigating risk during downturns, mimicking real-world trading environments, and contributing to the overall effectiveness of the trading strategy.

Margin Maintenance Detection Module To ensure compliance with the maintenance margin requirement in the margin account, which mandates that the

ratio of equity to market value in a position (e_t^l/m_t^l or e_t^s/m_t^s) must exceed a certain threshold, we implement *Maintenance Detection Module* equipped with a pre-action constraint and post-action constraint to continuously monitor the margin levels and trigger necessary actions if it is not met, safeguarding the portfolio's safety. Specifically, it establishes two levels associated with the maintenance margin requirement: the warning level and the strict level.

For the pre-action constraints, the warning level is employed to promptly notify the agent before purchasing long stocks or selling short positions. If the maintenance margin is detected to fall below the requirement level, additional trading actions on this stock are forced to halt, and a warning message is displayed. It warns the agent that the margin level is approaching a critical point, and it should take action.

For the post-action constraints, the strict level is applied after all actions A_t for both long and short positions have been executed at each step. If a margin call is triggered for any position, the *Margin Adjustment Module* is activated. The portfolio is required to be reallocated, specifically reducing the loan limit or credit balance limit, to meet the necessary maintenance level. It mitigates the potential risks associated with inadequate equity coverage.

Maximum Quantity of Shares to Trade When the agent executes buying or selling actions for stocks, the actual quantity of shares that can be traded is subject to certain constraints and adherence to available resources.

1. For the stocks to be bought in the long position, the market value of the trade cannot exceed the available cash. It ensures that the cash is not depleted beyond the limit.

$$a_{i,t}^l = \min(a_{i,t}^l, \frac{b_t^l}{p_{i,t}(1+c)}), \quad if\ a_{i,t}^l > 0 \tag{7.20}$$

2. For the stocks to be sold in the short position, the market value of the trade cannot surpass the available limit for the short sale.

$$a_{i,t}^s = -\min(|a_{i,t}^s|, \frac{l_t^s}{p_{i,t}(1+c)}), \quad if\ a_{i,t}^s < 0 \tag{7.21}$$

3. For the stocks to be sold in the long position or bought back in the short position, the quantity of shares traded cannot exceed the current holding shares.

$$a_{i,t}^l = -\min(|a_{i,t}^l|, n_{i,t}), \quad if\ a_{i,t}^l < 0 \tag{7.22}$$

$$a_{i,t}^s = \min(a_{i,t}^s, |n_{i,t}|), \quad if\ a_{i,t}^l > 0 \tag{7.23}$$

7.4 Experiments

7.4.1 Portfolio pool

We choose the Dow Jones Industrial Average (DJIA) as our portfolio pool and execute trading actions among the stocks of DJIA [1]. We follow the data cleaning and feature engineering procedure in FinRL-Meta [34]. The stock data and technical indicators can be collected from Yahoo Finance. The stock data is divided along with the temporal dimension into three datasets for training, tuning parameters, and testing. The model is trained on the period from 2009/01 to 2018/12; the hyperparameters are tuned based on the validation set from 2019/01 to 2020/04; the performance is evaluated over the testing period from 2020/05 to 2023/05, encompassing over 700 trading days. The testing period includes the recovery period from the COVID pandemic in 2020 and a notable phase of market volatility experienced in 2022. Our code is publicly available [2].

7.4.2 Baselines

We compare the performance of *Margin Trader* with two agents, DJIA and FinRL. (1) The DJIA agent tracks the DJIA index to trade. (2) FinRL [147] is chosen as one of the baselines since it is a state-of-the-art benchmark to apply DRL algorithms in

[1] Our security pool can be easily extended to other stock indices or other financial products, e.g., cryptocurrency.

[2] `https://github.com/JingyiGu/Margin-Trader`

portfolio management. Its trading environment is limited to the long position only and does not include margin trading capabilities. (3) *Margin Trader* enables both long and short positions, it is necessary to compare their impact and performance under different market conditions. Considering that the long-term market trend generally exhibits an upward trajectory with short-term fluctuations, we implement two distinct modes for the position settings: margin on long position, and margin on both long and short positions. For a fair comparison, four popular DRL algorithms, A2C, PPO, DDPG, and SAC, are applied in FinRL and two modes of *Margin Trader*.

7.4.3 Evaluation metrics

We follow [34] to select six metrics to comprehensively evaluate the performance of the framework.

1. **Accumulated Return** R: total return achieved over the testing period.
2. **Annual Return** AR: the percentage change in equity over a one-year period, widely used to assess long-term performance.
3. **Annual Volatility** VOL: the standard deviation of annualized return. A higher value indicates higher fluctuations in the return.
4. **Maximum Drawdown** MDD: the maximum loss from the peak to trough during the trading period. A lower value indicates a volatile downside movement.
5. **Sharpe Ratio** SR: a risk-adjusted metric to evaluate the excess return relative to its risk. A higher SR implies a more favorable risk-to-reward strategy.
6. **Calmar Ratio** CR: a risk-adjusted performance measuring the return relative to its maximum loss. A higher CR shows a better performance.

To summarize, R and AR evaluate the profit, VOL and MDD measure volatility and loss, SR and CR indicate both risk and return.

7.4.4 Implementation

For the parameters in the trading environment: the transaction cost c is set to 0.1% of traded market value; the λ_1 in the reward function is set to 0.00001, and λ_2 is tuned

Table 7.1 Hyperparameters for Margin Trader and FinRL

Model	Hyperparameter	Range
A2C	Entropy coefficient	{0, 0.005, 0.01}
	Number of steps per update	{5, 10, 15}
	Learning rate	[0.00001, 0.005]
PPO	Entropy coefficient	{0, 0.005, 0.01}
	Batch size	{64, 128, 256}
	Learning rate	[0.00001, 0.005]
DDPG	Buffer size	{50000, 100000}
	Batch size	{64, 128, 256}
	Learning rate	[0.00001, 0.005]
SAC	Buffer size	{50000, 100000}
	Entropy coefficient	{0, 0.01, 0.1}
	Batch size	{32, 64, 128}
	Learning rate	[0.00001, 0.005]
	Warm-up steps	{100, 500}

in the range of {0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05}; in *Maintenance Detection Module*, the warning level is set to 40% and strict level is set to 30%; the maximal quantity of shares h is set to 100 for FinRL (long) and *Margin Trader* in long, it is set to 50 for *Margin Trader* in long & short, considering that it reaches 100 after *Position Fusion Module*.

The actor-critic algorithms are implemented by stable baselines3 [158], and an MLP policy is used as the network in actors and critics. We follow the FinRL framework to tune the hyperparameters of DRL algorithms. Table 7.1 shows the range of hyperparameters for tuning during the training process.

7.4.5 Results comparison

We initialize each agent with an initial equity of $100,000. FinRL trades up to a market value of $100,000 exclusively in the long position. *Margin Trader* on the long position allocates its entire initial equity to the long market, thereby allowing trades with a maximum market value of $200,000. In the case of margin on both long and

Table 7.2 Numerical Results on Portfolio Management

Agent	Model	Accumulated Return	Annual Return	Annual Volatility	Maximum Drawdown	Sharpe Ratio	Calmar Ratio
DJI		43.7%	12.9%	17.3%	-21.9%	0.790	0.587
FinRL (Long)	A2C	38.3%	11.5%	18.4%	-24.2%	0.684	0.474
	PPO	35.2%	10.6%	17.0%	**-13.9%**	0.680	0.764
	DDPG	60.6%	17.2%	18.4%	-23.4%	0.956	0.733
	SAC	54.4%	15.6%	18.2%	-19.5%	0.890	0.801
Margin Trader (Long & Short)	A2C	66.3%	18.5%	25.6%	-26.1%	0.793	0.710
	PPO	69.0%	19.2%	**16.3%**	-18.2%	**1.160**	1.057
	DDPG	70.8%	20.9%	27.2%	-25.7%	0.835	0.812
	SAC	65.1%	18.3%	21.2%	-19.2%	0.901	0.950
Margin Trader (Long)	A2C	118.9%	30.0%	29.3%	-27.1%	1.044	1.106
	PPO	**123.4%**	**30.9%**	30.1%	-25.8%	1.046	**1.199**
	DDPG	110.0%	30.1%	30.7%	-32.9%	1.012	0.915
	SAC	113.4%	28.9%	29.4%	-31.5%	1.011	0.916

Note: Performance of three agents (DJI, FinRL, and Margin Trader) and different position settings implemented by actor-critic algorithms. The best performance is marked in bold. The first four metrics are expressed as percentages.

short positions, half of the equity is allocated to each position, enabling trades with a maximum market value of $100,000 for each position.

Table 7.2 presents the numerical results. Figure 7.2 displays the equity change of *Margin Trader* and FinRL over the testing period. We have the following key observations.

Margin Trader on the long position achieves the highest final equity and exhibits superior performance regarding risk-adjusted metrics such as SR and CR. Its annualized return is twice that of FinRL on long positions only, demonstrating that trading strategies involving margin accounts can effectively earn more profit than trading with cash only, particularly in the bullish market from 2020/07 to 2022/01. In contrast, FinRL, which trades with its own cash in long positions, essentially follows the trend of the DJIA agent.

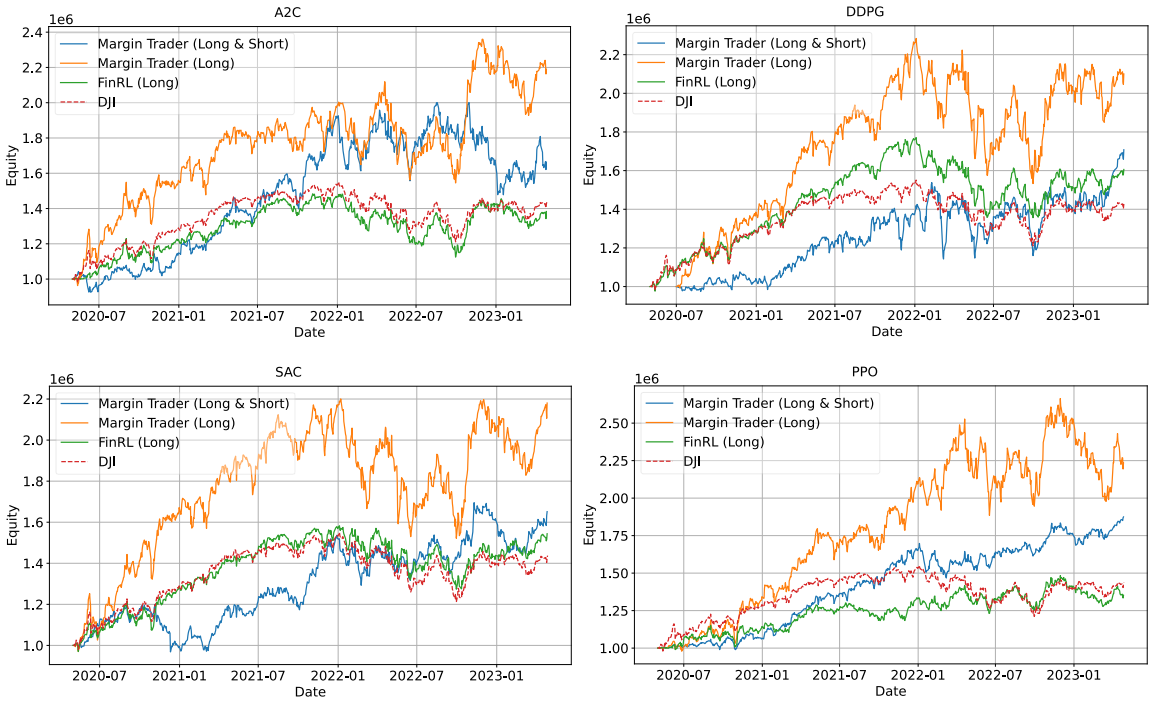


Figure 7.2 The equity changes of DJI, FinRL and *Margin Trader*.

During the volatile period, especially from 2022/01 to 2022/10, DJIA, FinRL, and Margin Trade in long positions show decreasing trends. Margin Trading in long & short positions exhibits an upward trend, particularly with SAC and PPO. A2C shows fluctuating equities in both modes of *Margin Trader*, but it does not lose many positions due to the help of maintenance margin constraints. This finding validates that incorporating short positions during volatile periods can hedge the risk and guarantee profit. However, this strategy is not robust in a bullish market since half of the equity is allocated to short positions, which makes it difficult to profit from price increases.

Among different types of DRL algorithms, PPO consistently provides the best trading strategies for all positions regarding all metrics, regardless of bullish and bearish market conditions.

The two different modes of *Margin Trader* demonstrate the importance of selecting positions based on dynamic market conditions. The long position outperforms during the first half of the period in a bullish market, while the short

Table 7.3 Time Span for Rolling Window Analysis

Window No.	Training		Validation		Testing	
	Start date	End date	Start date	End date	Start date	End date
1	2009-01	2018-01	2018-01	2019-01	2019-01	2020-01
2	2009-01	2019-01	2019-01	2020-01	2020-01	2021-01
3	2009-01	2020-01	2020-01	2021-01	2021-01	2022-01
4	2009-01	2021-01	2021-01	2022-01	2022-01	2023-01

position makes a profit during the second half of the period in a bearish market. Properly choosing allocation ratios for long and short positions, as well as margin ratios, can magnify the profit and hedge the risk simultaneously. This flexibility in *Margin Trader* enables users to adapt their strategies according to market conditions and individual preferences.

7.4.6 Rolling window evaluation

To fully assess the generalization ability and robustness of *Margin Trader*, we apply rolling-based testing periods, a crucial methodology often lacking in existing works for portfolio management evaluation. Specifically, the training, validation, and testing periods are rolled forward to demonstrate the performance when *Margin Trader* and baselines consistently learn from recent past data and are deployed for near-future portfolio management. The time span is shown in the table 7.3, where test periods describe various market conditions:

1. Rolling window 1 reflects a bullish market with fluctuations.
2. Rolling window 2 encompasses the dramatic market downturn during the COVID-19 pandemic.
3. Rolling window 3 represents the abnormal recovery with the economic stimulus package.
4. Rolling window 4 depicts the market impacted by interest-rate rises.

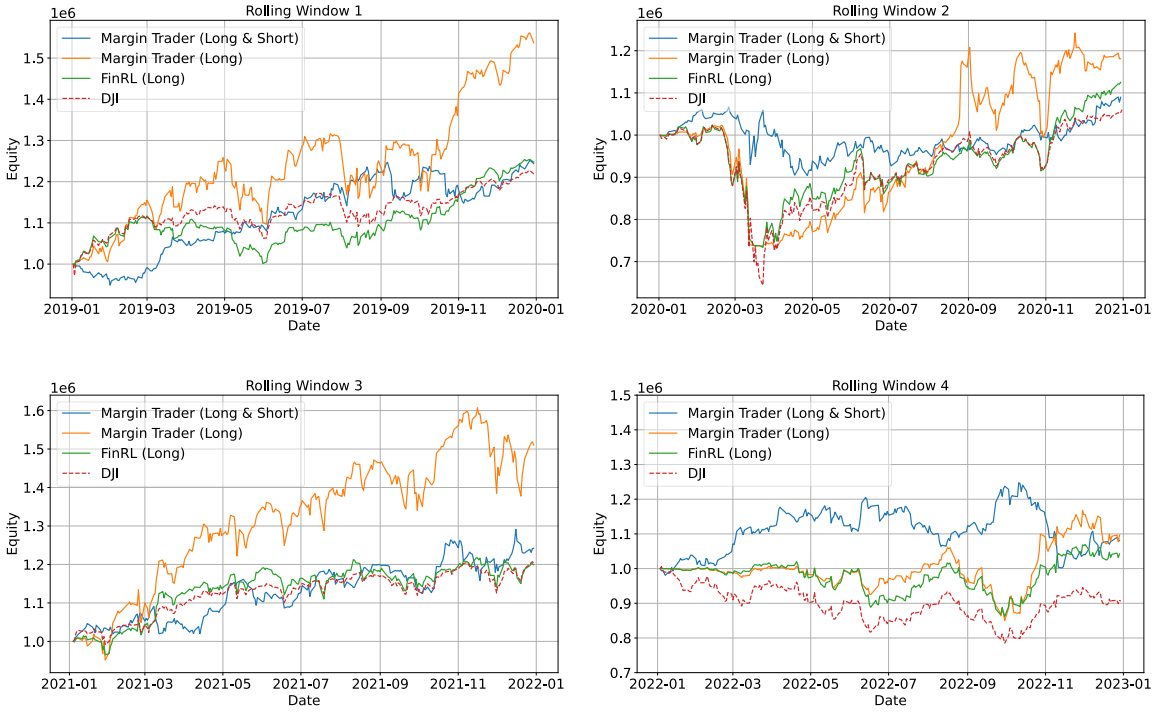


Figure 7.3 The equity changes of DJI, FinRL, and *Margin Trader* under four rolling windows.

Among four RL algorithms, we choose PPO due to its superior performance and exclusively apply it to both FinRL and *Margin Trader*. Table 7.4 displays the numerical result, and Figure 7.3 visualizes the portfolio changes.

During market upswings, as observed in rolling windows one and three, *Margin Trader* employing only long positions consistently yields the highest annual return of over 50%, double that of other agents. Its equity curve notably surpasses others. Despite it comes with the highest volatility and severe MDD, the robust risk-adjusted metrics (Sharpe ratio and Calmar ratio) indicate that the returns are proportionally higher, making the risk potentially worthwhile. *Margin Trader* engaging in both long and short positions is competitive with FinRL's long-only approach. Meanwhile, DJI earns the least profit, yet maintains the highest stability. It is typically representative of the market as a whole and indicates a lower-risk investment.

In times of market distress, such as the COVID pandemic impact around March 2020 in rolling window two, and the raised interest rates in the first half

Table 7.4 Numerical Results of Three Agents in Rolling Window Analysis

Window No.	Agent	Accumulated Return	Annual Return	Annual Volatility	Maximum Drawdown	Sharpe Ratio	Calmar Ratio
1	DJI	22.24%	22.24%	**12.46%**	**-6.91%**	1.681	3.220
	FinRL	24.74%	24.85%	13.04%	-10.34%	**1.774**	2.403
	MT (L&S)	24.39%	24.50%	14.28%	-7.97%	1.613	3.074
	MT	**53.70%**	**53.97%**	26.69%	-12.88%	1.758	**4.191**
2	DJI	6.02%	5.99%	36.81%	-37.09%	0.344	0.162
	FinRL	12.50%	12.50%	29.09%	-28.30%	0.553	0.442
	MT (L&S)	8.98%	8.98%	**21.51%**	**-15.26%**	0.509	0.588
	MT	**18.09%**	**18.09%**	36.21%	-28.28%	**0.644**	**0.640**
3	DJI	20.23%	20.23%	**12.38%**	**-6.62%**	1.556	3.058
	FinRL	19.92%	20.01%	15.69%	-6.85%	1.246	2.922
	MT (L&S)	24.23%	24.34%	16.91%	-8.38%	1.379	2.905
	MT	**51.03%**	**51.28%**	28.14%	-14.29%	**1.619**	**3.588**
4	DJI	-9.40%	-9.43%	19.85%	-21.94%	-0.402	-0.430
	FinRL	4.25%	4.29%	**18.83%**	**-15.46%**	0.318	0.277
	MT (L&S)	8.06%	8.12%	19.84%	-19.67%	0.494	0.413
	MT	**9.64%**	**9.72%**	21.91%	-19.82%	**0.534**	**0.491**

Note: DJI, FinRL, and Margin Trader operate long positions only, and Margin Trader (L&S) involves both long and short positions. The best performance in each rolling window is marked in bold. The first four metrics are expressed as percentages.

of 2022 in window four, the utilization of margin in short positions in *Margin Trader* effectively hedges against downturns. Conversely, other models experience severe asset depreciation. As the market rebounds in the latter half of 2022, *Margin Trader* with long positions grows its profit with a strong resilience, and hence achieves the highest returns and Sharpe ratio.

The performance of portfolio management varies notably across distinct time windows. 2019 (window one) appears to be the best-performing period for most strategies, characterized by robust equity growth and swift recovery from drawdowns. In contrast, 2022 (window four) presents significant challenges across all strategies,

with DJI even yielding negative returns. This suggests the critical importance of adeptly selecting agents and positions to balance profit generation and risk mitigation. DJI exhibits more moderate returns and minimal risk. FinRL operating long positions shows a middle ground in terms of returns and volatility, generally mirroring market trends. The margin accounts in *Margin Trader* can lead to substantial returns during favorable market movements, and the short positions offer protection against significant losses in volatile markets, allowing traders to capitalize on both upward and downward market shifts. It is more suitable for investors with a high risk tolerance.

7.5 Conclusions

We present *Margin Trader*, a novel RL framework specifically designed for margin trading in the stock market. Unlike traditional approaches that primarily focus on cash-only trading, *Margin Trader* incorporates margin accounts and constraints, offering a more realistic environment for portfolio management. It empowers traders to customize crucial settings and caters to specific needs. *Margin Trader* is evaluated under various DRL algorithms, rolling testing periods, and compared with popular benchmarks. The extensive experiments demonstrate its ability to generate profitable trading strategies while effectively managing risk under various market conditions.

CHAPTER 8

ENHANCED MARGIN TRADING VIA LLMS

8.1 Introduction

Recent advancements in portfolio management involve strategies that incorporate both long and short positions, often leveraging deep reinforcement learning to adjust the portfolio weights and execute trading operations. However, existing approaches either assume that short positions always maintain equal equity with long positions throughout the trading period, which does not align with real-world conditions, or simply allocate equity to both positions with a fixed long-short equity ratio at initialization, which limits their effectiveness in adapting to rapidly changing environments and achieving better portfolio performance. Therefore, a flexible strategy that incorporates the dynamic adjustment of the long-short position ratio is essential for achieving resilient and adaptive portfolio management, which has been ignored in existing research.

The development of Large Language Models (LLMs) such as GPT-4 [159] and Claude-3 [160] has significantly enhanced data analytics capabilities in complex domains such as finance [161]. LLMs stand out by processing and deriving insights from unstructured data, such as financial reports' textual narratives and social media content, eliminating the need for the extensive preprocessing required by traditional models like BERT. Furthermore, their extensive pre-training enables them to generalize knowledge to new, unseen tasks through zero-shot learning, allowing them to follow natural language instructions and adapt swiftly to the volatile financial markets. LLMs provide contextual explanations for their decisions, boosting transparency and trust among stakeholders, which is crucial in finance to justify investment decisions. They not only augment the analytical capabilities within

portfolio management but also contribute to the evolution towards more data-driven and agile financial strategies.

To fill these gaps, we develop an adaptive and explainable framework, integrating LLMs into the sequential decision-making process of RL in *Margin Trader*, to facilitate dynamic fund transfers between two positions in response to evolving market conditions, as shown in Figure 8.1. This framework features a two-stage adjustment process implemented regularly during the RL trading process: the *Market Forecasting/Reasoning Pipeline* interact with RL at regular intervals of every k step, leveraging LLMs to analyze rich external data sources to forecast the fine-grained market trends with clear reasoning and determine the optimal adjustment ratio; In *Position Reallocation stage*, the equity in the state received from RL Environment is adjusted accordingly to reallocate funds between long and short positions, and *Margin Trader* continues trading operations with updated portfolio states for the next k steps. We present an example pipeline that utilizes microeconomic firm news and macroeconomic indicator time series, each offering unique insights into general market trends. However, the framework is designed to be highly flexible, capable of incorporating more complex data sources, diverse data types, and various LLMs. Moreover, our designed pipeline not only forecasts the near-future trends and adjustment ratio but also provides a transparent reasoning path with clear explanations, which offers valuable insights into the rationale behind portfolio adjustments and enhances the decision-making process. LLMs intervene only during the deployment phase instead of the training process, thereby advancing in efficiency. The framework is conducted on DJIA from 2020/5 to 2024/4, including both bullish and bearish periods. Results demonstrate that intervention of LLM can boost performance at most 3 times of accumulated return (243.852), twice of Sharpe ratio (1.314), and almost threefold Calmar ratio (1.508) compared with the backbone without position adjustments, validating the effective adaptation

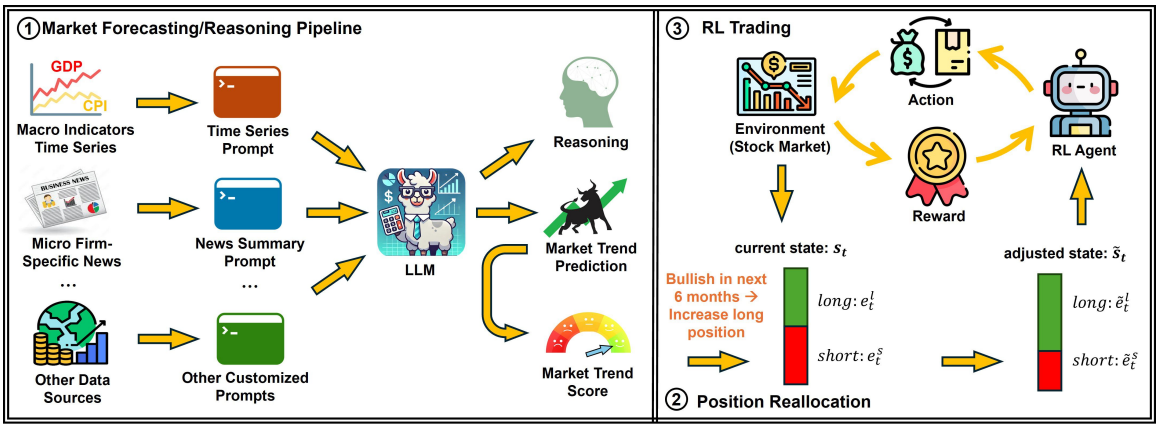


Figure 8.1 The framework presents the interaction of the two-stage adjustments with the pre-trained RL trading model. Stage 1 (Market Forecasting/Reasoning Pipeline) employs LLMs to predict and analyze future stock market trends, utilizing various data sources. Stage 2 (Position Reallocation) updates the states in the RL process by adjusting the long-short ratio based on the insights gained from LLMs.

capability of LLM to evolving markets. Our major contributions are summarized as follows[1]:

1. We propose the first framework that integrates LLMs' interaction with RL for dynamic long-short position adjustments in response to varying market conditions in margin trading.
2. Our framework offers flexibility to various external data sources from macroeconomics to individual firms, diverse data types including time series and text, as well as multiple LLMs, providing interpretation with varied perspectives.
3. Our pipeline demonstrates superior performance in enhancing profitability and balancing risks under diverse market conditions.

8.2 Methodology

8.2.1 External datasets

We utilize two distinct datasets to comprehensively analyze the U.S. economic landscape and its interplay with the stock market from different perspectives. The

[1]Jingyi Gu and Junyi Ye contributed equally to this research, which centers on an adaptive and explainable framework integrating LLMs with RL for dynamic margin trading. Specifically, Jingyi Gu focused on the RL aspects, including managing the *Margin Trader* data, developing the RL experimental pipeline, and implementing the LLM interactions with RL, while Junyi Ye concentrated on the LLM components, designing the Explainable Market Forecasting/Reasoning Pipeline, and handling LLM input data selection and processing. Both authors collaborated closely on integrating the LLM and RL components and contributed to the overall framework design and evaluation.

first dataset is a macroeconomic dataset, which includes major indicator time series that reflect broad economic conditions and trends. The second dataset is microeconomic, comprising firm-specific news within the asset pool, providing detailed insights into individual company activities. By examining these datasets, we aim to determine which perspective is more effective for predicting future market changes and enhancing trading strategies, providing a robust analysis of their respective influences on market dynamics.

Macroeconomic Time Series: The dataset comprises monthly time series data of 21 macroeconomic indicators, including CPI (CPI, CPI Less Food and Energy), Interest Rates (Federal Funds Effective Rate, Bank Prime Loan Rate), Government Debt and Deficit (Federal Debt, Federal Surplus or Deficit), Exchange Rates (Nominal Broad US Dollar Index, US Dollar to Euro, Japanese Yen to US Dollar, Chinese Yuan Renminbi to US Dollar), Money Supply (Real M1 Money Stock, Real M2 Money Stock), and Retail Sales (Advance Retail Sales: Retail Trade and Food Services, Advance Retail Sales: Retail Trade). It provides a chronological sequence of data points, enabling the analysis of trends, cycles, and patterns over time, which is crucial for understanding the broader U.S. economic landscape and its potential impact on the stock market.

Microeconomic Firm-Specific News: This dataset is specifically tailored for portfolio management. This dataset focuses on daily news articles and reports directly related to individual assets within the portfolio pool, including detailed information on company performance, earnings reports, strategic initiatives, mergers and acquisitions, and other significant corporate events. By capturing firm-specific news, it offers granular insights into the microeconomic factors that influence the performance of major corporations. While macroeconomic news covers general market conditions that may not directly reflect the trends affecting the portfolio pool,

Table 8.1 Mapping from Sentiment to Margin Changes

Market Sentiment	Expected Percentage Change
Strongly Bullish	Greater than +20%
Bullish	+10% to +20%
Slightly Bullish	+5% to +10%
Flat/Fluctuating	-5% to +5%
Slightly Bearish	-10% to -5%
Bearish	-20% to -10%
Strongly Bearish	Less than -20%

Note: The mapping between market sentiment and expected percentage change over the next 6 months. 6 out of 8 baseline LLMs agreed on this mapping. Phi-3 Medium and DeepSeek-V2 had more conservative ranges, e.g., Strongly Bullish was 10-20% and 15-20%, respectively.

firm-specific news provides more relevant and actionable information for managing and optimizing the portfolio.

8.2.2 Explainable market forecasting/reasoning

We design an explainable market reasoning pipeline via LLM to process the two distinct datasets and generate relevant insights based on corresponding prompts, taking into account the varying data types, data sizes, and the context window limitations of the language models. For each dataset, we have tailored different prompt pipelines. The pipeline leverages the capabilities of advanced language models to analyze, summarize, and interpret financial and market data, providing valuable outputs specific to each dataset type. For each data type, LLMs are tasked with predicting the near future (six months) market trend and providing explanations. The market trend prediction is categorized into one of seven fine-grained categories (Strongly Bullish to Strongly Bearish), as illustrated in Table 8.1. Compared to common categories of positive, negative, and neutral, this allows more nuanced

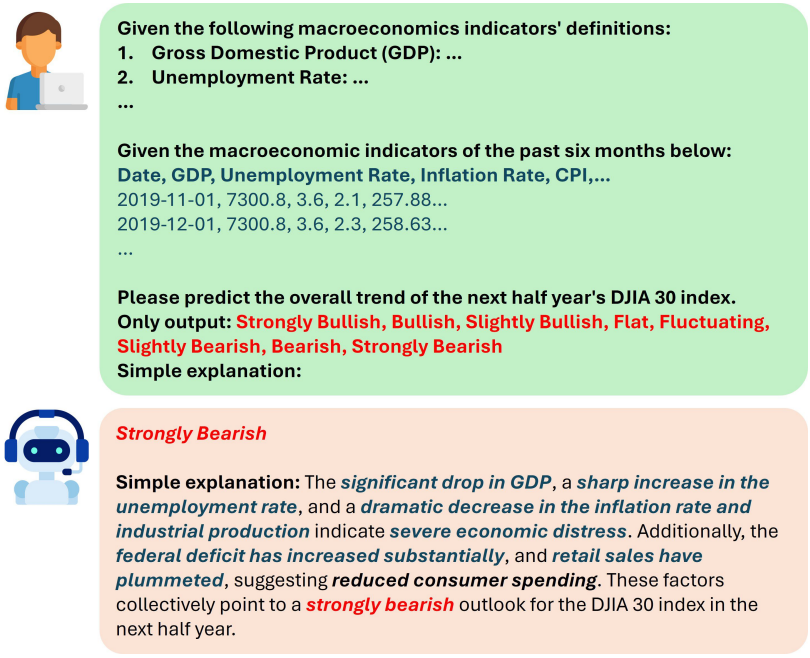


Figure 8.2 Sample prompt during the COVID-19 pandemic and GPT-4o response for predicting the overall trend of the DJIA 30 index based on macroeconomic indicator time series.

adjustments in subsequent RL algorithms. Below is a detailed description of the LLM pipeline and how it processes each dataset. We use the DJIA as a representative example of an asset pool.

Prompt Pipeline for Macroeconomic Indicator Time Series The prompt in Figure 8.2 comprises three parts: (1) macroeconomic indicators' definitions, (2) historical macroeconomic indicator time series data, and (3) instructions for LLMs' output requirements. This time series based prompt provides a structured analysis of past economic performance by incorporating all relevant indicators. It leverages reliable data from government websites, ensuring comprehensive coverage of economic conditions and trends. By integrating time series data, the pipeline offers a robust framework for understanding the continuous impact of various macroeconomic factors, delivering precise insights into future market trends. It evaluates the LLM's capability to discern relationships between macroeconomic features, detect numerical data

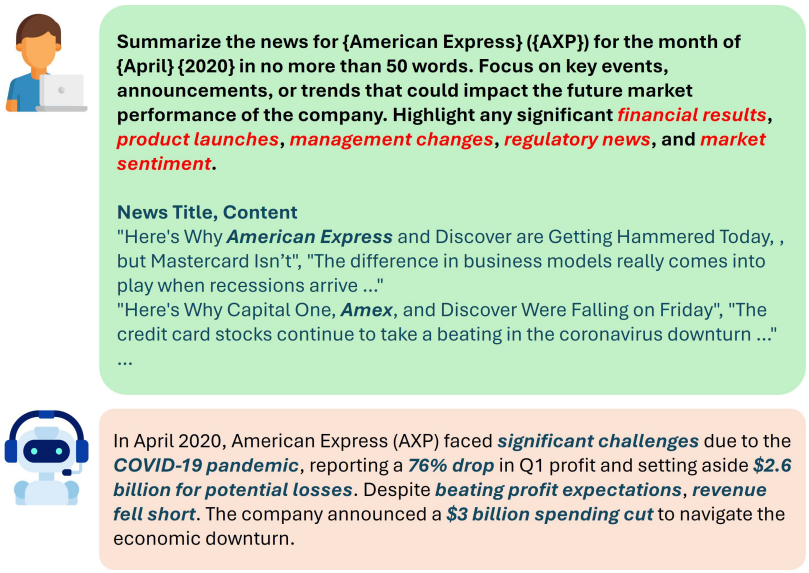


Figure 8.3 Prompt and GPT-4o response for summarizing the news for American Express (AXP) for the month of April 2020. The texts in the {} are replaced with company names, tickers, month, and year accordingly.

trends, and accurately forecast future economic conditions based on the varying sentiments of different indicators.

As shown in Figure 8.2, GPT-4o analyzes key positive and negative indicators to identify decisive features impacting economic trends. The LLM's reasoning highlights severe economic distress due to the significant drop in GDP, rise in unemployment, decreased inflation and industrial production, increased federal deficit, and plummeting retail sales. Based on this analysis, GPT-4o predicted a strongly bearish market trend during the COVID-19 pandemic, which aligns with the macroeconomic time series data.

Prompt Pipeline for Microeconomic Firm-Specific News The firm news prompt pipeline involves three key steps to effectively aggregate and summarize daily news for each firm in the pre-selected asset pool, ensuring it fits within the context limits of most open-source LLMs: (1) **Daily Aggregation of Firm News:** For each firm, the daily firm news is collected and aggregated on a monthly basis. This process involves gathering all relevant news articles for each firm, ensuring comprehensive

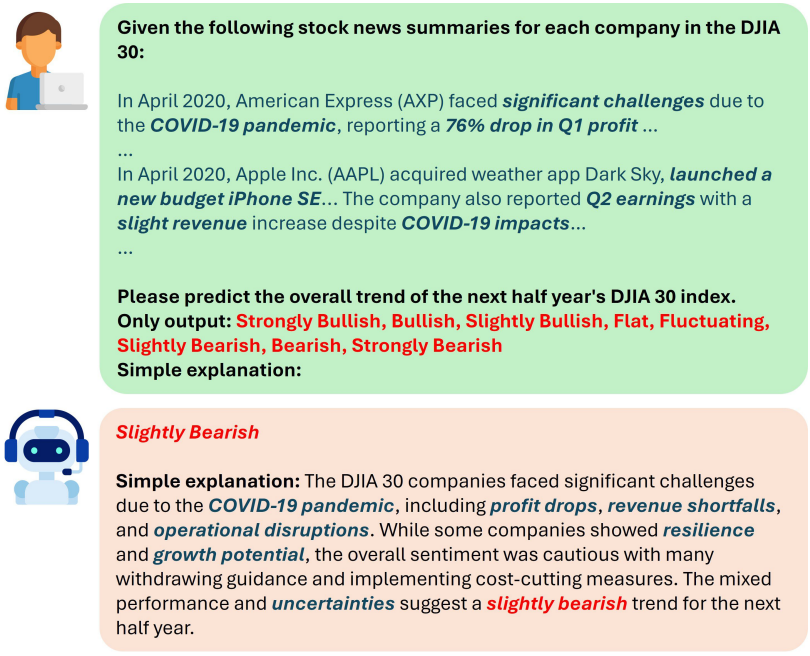


Figure 8.4 Sample prompt during COVID-19 pandemic and GPT-4o response for predicting the overall trend of the DJIA 30 index based on monthly firm news summary of the DJIA 30.

coverage of their activities and developments. (2) **Summarization to Fit LLM Context Limits:** Given the constraint of a 4K-token limit in many LLMs, the aggregated monthly news for each company is summarized to a maximum of 50 words with GPT-4o. This summary focuses on significant financial results, product launches, management changes, regulatory news, and market sentiment, illustrated in Figure 8.3. This step ensures balanced representation across companies, preventing firms with extensive news coverage (e.g., Apple with over 400 news articles per month) from overshadowing those with fewer updates (e.g., Amgen Inc. with only 30 news articles monthly). By summarizing each company's news, the prompt can accommodate summaries for all 30 companies within the 4K-token window. (3) **Fine-Grained Market Trend Prediction:** The LLM uses the aggregated monthly news summaries for all firms to predict the future fine-grained trend of the market, as illustrated in Figure 8.4. Shifting the input from macroeconomic data to microeconomic firm news data allows GPT-4o to adjust its forecast from a strongly bearish outlook to

a slightly bearish one. This adjustment reflects detailed information in firm news summaries, which indicate resilience and growth potential among companies that have implemented adjustments in response to the COVID-19 pandemic. These insights provide more accurate and actionable information not captured in macroeconomic time series.

By focusing on detailed, company-specific information and events, the firm news prompt pipeline provides granular microeconomic insights, offering a more precise perspective for portfolio management. This approach complements the broader economic focus of the macroeconomic pipeline, allowing the LLM to generate nuanced predictions about overall stock market trends based on the significant events of firms in the asset pool.

LLM Response Mapped to Market Trend Score To seamlessly integrate the predicted trend from LLMs into our RL trading process, qualitative sentiment levels are converted into a quantitative Market Trend Score $m \in \{-3, -2, -1, 0, 1, 2, 3\}$. Positive integers signify varying degrees of bullish sentiment, negative integers represent bearish levels, and zero denotes a flat or fluctuating market condition. A larger absolute value of m indicates a strong market signal. The adjustment ratio $\delta = ms$ is determined by applying a scaling factor s, which controls the impact of trend prediction on position change. For example, if $s = 5\%$, then δ ranges from -15%, to 15%, enabling a precise adjustment based on the market trend prediction provided by LLMs.

8.2.3 Position reallocation

The LLM's intervention in the RL process occurs every k step. At time step t, we receive the previous reward r_{t-1} and observe the current state s_t, which includes the long equity e_t^l and short equity e_t^s. The current ratio of long equity over the total equity $r_t = e_t^l/(e_t^l + e_t^s)$ can be calculated. We adjust this long ratio proportionally to reach

the target ratio $r'_t = r_t * (1 + \delta)$, with the change in long equity being $e^l_t \delta$. Compared with directly adding δ_t to the current ratio, this proportional adjustment considers existing equity conditions, thereby preventing significant and abrupt changes in market exposure. In addition, we establish a range threshold for the target ratio $r'_t \in [\eta_1, \eta_2]$, which is set to $[10\%, 90\%]$ in this study, to prevent allocating all funds into a single position. This approach mitigates certain risks of overreacting to market fluctuations and facilitates smoother transitions between positions.

A positive δ results in an increase in long equity, necessitating a transfer of funds from short to long positions. Conversely, a negative δ prompts a movement of funds from long to short. We implement these adjustments following the margin adjustment module in *Margin Trader* to manage the buying power and states effectively. Firstly, available cash in long positions or available limits for short sales are assessed to cover the transfer amount. If sufficient funds are available, the transfer occurs directly; otherwise, the position closures are triggered, starting with the asset of lowest holding value, as it is considered the least significant portfolio, until adequate funds are available. In addition, the loan in long or the credit balance in short is also reduced to align with the margin requirement. Secondly, the transferred funds allow for increased short sales or borrowing more loans for the long trade. Consequently, the available limit and credit balance for short positions, or the loan and available cash for long positions, are increased accordingly.

After the adjustment, the state $\tilde{s}_t$ with long equity $\tilde{e}^l_t$ and short equity $\tilde{e}^s_t$ has been updated. The new state is then fed into the RL framework to continue the sequential process and refine the optimal trading policy iteratively.

8.3 Experiments

8.3.1 Data source

We use the same training and validation periods, and extend the test period to 2020/5 - 2024/2, since it is up-to-date and includes complex economic fluctuations, marked by a significant rise (COVID-19 pandemic recovery) and subsequent variations (supply chain disruptions and inflation). Two distinct external data sources are collected and tested to evaluate their impact on near-future (six-month) U.S. market trend prediction:

1. **Macroeconomic Indicator Time Series Dataset** comprises monthly time series data for 21 key US economic metrics. The inflation rate data is sourced from the US Inflation Calculator[2], while the remaining macroeconomic indicators are obtained from the Federal Reserve Bank of St. Louis[3]. To ensure consistency, data with daily or quarterly features has been appropriately downsampled or upsampled to a monthly frequency.
2. **Microeconomic Firm-Specific News Dataset** includes daily news data specific to the 30 companies listed on the DJIA. News is gathered from various sources, such as company announcements, earnings reports, and other significant events. The data is retrieved by ticker from the Stock News API[4], which indexes articles and video content from reputable sources including CNBC, Reuters, MarketWatch, Seeking Alpha, Bloomberg, and The Motley Fool.

8.3.2 Baseline

To fairly compare the portfolio performance, we employ two benchmarks: **DJIA** follows the DJIA index trend, and **Margin Trader** is the backbone RL model without LLM intervention on ratio adjustment. In addition, the following two financial NLP models and several state-of-the-art LLMs are integrated to evaluate our framework: **FinBERT**[38] is an NLP model for financial sentiment analysis, built on BERT and fine-tuned with a comprehensive financial corpus; **FinGPT v3.3**[81] is based on

[2] `https://www.usinflationcalculator.com/inflation/current-inflation-rates/`
[3] `https://www.stlouisfed.org/about-us`
[4] Stock News API: `https://stocknewsapi.com/`

LLaMA2-13B and fine-tuned on financial sentiment datasets; **GPT-4o**[162] excels in multi-language understanding, question answering, math, code evaluation, reasoning, and reading comprehension; **Claude-3.5 Sonnet**[163] significantly outperforms its predecessor, Claude 3 Opus, and competing models across a wide range of evaluations; **DeepSeek-V2**[164] is a highly efficient Mixture-of-Experts model that enables economical training and efficient inference; **Mixtral 8x22B**[165] is a cutting-edge LLM using a Sparse Mixture-of-Experts (SMoE) architecture with reduced computational costs and enhanced speed; **LLaMA-3 70B**[166] is the latest in the LLaMA family, enhancing contextual understanding and efficiency for complex natural language processing tasks; **Qwen-2 72B**[167] is the top model in the Qwen series, handling complex applications effectively; **Yi 34B**[168] excels in English and Chinese benchmarks using advanced architectures like Grouped-Query Attention for versatile applications; **Phi-3 Medium**[169] employs supervised fine-tuning and direct preference optimization for alignment with human preferences and safety.

8.3.3 Implementation details

In our implementation of large language models (LLMs), we employ greedy decoding with a maximum limit of 1024 newly generated tokens. Open-source models are deployed on 1 to 4 A100 GPUs, selected based on GPU RAM requirements, utilizing the Hugging Face library. Proprietary models, such as GPT-4o, Claude 3.5, and DeepSeek V2, are accessed via their respective APIs. To mitigate potential biases in sentiment label assignments, we randomly shuffle the order of sentiment labels in the prompt and conduct five experimental runs. The final prediction is determined through majority voting. For models with a 512-token limit, such as FinBERT and FinGPT, we segment texts into chunks and then apply majority voting. The sentiment outputs—positive, neutral, and negative—are mapped to Bullish, Flat/Fluctuating, and Bearish sentiments, respectively. This approach

Table 8.2 Portfolio Management Results on Moderate Strategy

Model	Macro Indicator			Firm News		
	AR(%)	**SR**	**CR**	**AR(%)**	**SR**	**CR**
DJI	67.795	0.917	0.646	-	-	-
Margin Trader	75.834	0.673	0.592	-	-	-
FinBERT	-	-	-	55.004	0.558	0.467
FinGPT v3.3	-	-	-	-	-	-
GPT-4o	163.643	1.140	1.077	**205.197**	**1.199**	**1.353**
Claude-3.5 Sonnet	178.530	1.174	1.237	143.189	1.019	0.996
Qwen2 72B	**198.623**	**1.198**	**1.311**	175.395	1.140	1.266
Llama-3 70B	161.010	1.066	1.139	160.060	1.037	1.134
Mixtral 8x22B	160.060	1.037	1.134	160.060	1.037	1.134
DeepSeek-V2	170.553	1.111	1.187	112.173	0.894	0.812
Phi-3 Medium	164.705	1.139	1.192	106.022	0.900	0.920
Yi 34B	160.060	1.037	1.134	82.699	0.724	0.647

Note: This table is under a scaling factor of 10%. FinBERT and FinGPT, as sentiment analysis models, are unsuitable for macro time series data, especially FinGPT produces gibberish or multiple outputs with multiple news inputs for firm news data.

ensures robust and unbiased sentiment analysis across different model architectures and token limitations.

For the implementation of RL, we choose A2C as the DRL algorithm. For parameters in the RL trading environment and hyperparameters for the A2C, we follow [170]. The initial equity e is $\$100,000$. DJIA uses cash to trade on long positions only. Rest strategies implement both long and short based on *Margin Trader*, allocating half of the equity at the initialization step. The frequency of interaction (k) between RL and LLMs is a quarter.

Table 8.3 Portfolio Management Results on Radical Strategy

Model	Macro Indicator			Firm News		
	AR(%)	**SR**	**CR**	**AR(%)**	**SR**	**CR**
DJI	67.795	0.917	0.646	-	-	-
Margin Trader	75.834	0.673	0.592	-	-	-
FinBERT	-	-	-	-15.561	-0.022	-0.112
FinGPT v3.3	-	-	-	-	-	-
GPT-4o	128.512	1.014	0.657	**243.852**	**1.314**	**1.508**
Claude-3.5 Sonnet	**198.117**	**1.252**	**1.604**	212.229	1.269	1.350
Qwen2 72B	194.996	1.170	1.270	193.232	1.173	1.377
Llama-3 70B	191.449	1.189	1.277	196.969	1.164	1.303
Mixtral 8x22B	196.969	1.164	1.303	196.969	1.164	1.303
DeepSeek-V2	163.889	1.111	1.143	123.893	0.969	0.639
Phi-3 Medium	182.110	1.224	1.494	106.022	0.900	0.920
Yi 34B	196.969	1.164	1.303	82.699	0.724	0.647

Note: This table is under 20% of scaling factor.

8.3.4 Results and analysis

Comparison with Baselines We present two different strategies based on varying scaling factors. Table 8.2 and 8.3 show the numerical results for a scaling factor of 10%, representing a common and moderate strategy, and a more radical strategy with a scaling factor of 20%. We have the following observations. (1) The frameworks integrating LLMs on a quarterly basis consistently outperform DJIA and *Margin Trader* without long-short ratio adjustment. This highlights that LLMs can effectively detect market trends and determine the position ratio, significantly enhancing portfolio performance. (2) Among LLMs, GPT-4o, Claude-3.5 Sonnet, and Qwen2 72B achieve over twice and even three times the return, a 25% higher Sharpe ratio,

and more than twice the Calmar ratio across datasets and scaling factors compared with DJI and *Margin Trader*, making them standout LLMs for this framework. This conclusion aligns with the performance of these LLMs in other tasks. In contrast, FinBERT exhibits suboptimal performance, and FinGPT often generates gibberish or multiple outputs, rendering it unsuitable for this application. This issue arises because both models are trained on individual news inputs, leading to disorganized outputs. This limitation highlights a fundamental drawback of fine-tuned NLP models on isolated tasks: their lack of generalization. Unlike LLMs, which are designed to handle diverse and complex data inputs, fine-tuned models struggle to generalize beyond their specific training data, resulting in less reliable performance in varied contexts. (3) Increasing the scaling factor from 10% to 20% results in a more significant ratio change and generally enhances the performance of most models. However, it is important to note that larger scaling factors also introduce higher risk and greater volatility in equity changes, such as Llama-3 70B in firm news and Deepseek-V2 in macro indicators. While this may boost performance in a bullish market, it can also lead to extreme position changes and excessive market exposure, increasing the risk of substantial losses in unexpected market conditions. (4) Generally, models perform better with macro indicators than firm news, as macro indicators offer extensive information on the broader economic environment. Prices of individual assets already reflect some impact of their firm-specific news, potentially reducing their additional predictive power. An exception is GPT-4o, which excels at capturing and interpreting the nuanced and immediate signals in firm news. Note that our framework offers flexibility for traders to choose data sources for ratio adjustments, ensuring informed decision-making even when certain data sources are unavailable.

Effectiveness of LLM Predictions on Market Trends To illustrate the effectiveness of predictions of LLM on market trends, Figure 8.5 compares trend

predictions by GPT-4o with DJIA movements from the test period. The DJIA close prices are shown in blue, while sentiment predictions are indicated by colored blocks: more red denotes bearish sentiment, and more green denotes bullish sentiment. The plot demonstrates the alignment of GPT-4o's predictions with actual market trends.

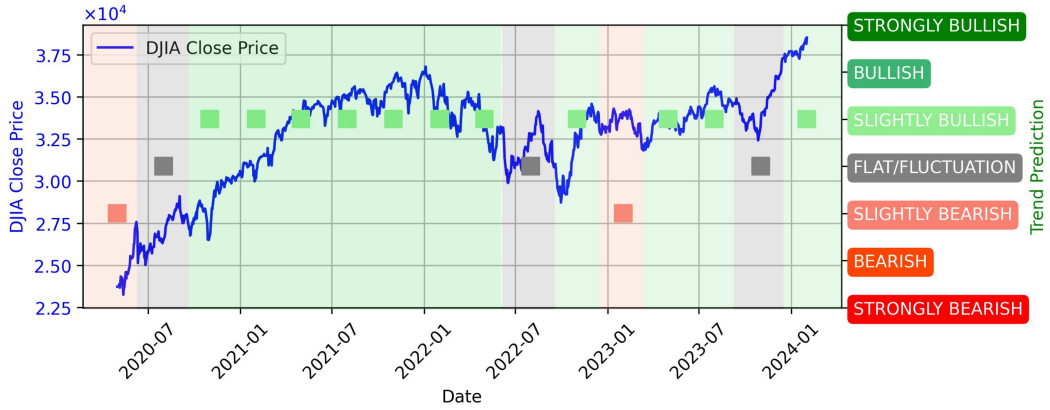


Figure 8.5 Comparison of GPT-4o trend predictions using micro firm-specific news as input with DJIA movements.

Frequency Analysis To investigate the impact of the frequency of adjusting the long-short ratio based on LLM output, we implement three different frequencies: quarterly, biannually, and annually. Figure 8.6 presents the equity changes of Claude in varying frequencies. Firstly, quarterly adjustments introduce higher volatility and lower returns compared to less frequent adjustments, such as bi-annual ones, because macro information and firm news tend to have long-term effects, and adjusting positions too frequently makes portfolios unstable and unbalanced. Besides, infrequent adjustments, such as annual adjustments, show lower performance, missing out on opportunities to capitalize on timely market movements, and thus providing suboptimal portfolio performance. Therefore, it is important to find a reasonable frequency for optimal portfolio performance. Bi-annual adjustments strike the optimal balance, avoiding the excessive volatility of quarterly adjustments and the missed opportunities of annual adjustments, achieving the highest cumulative return,

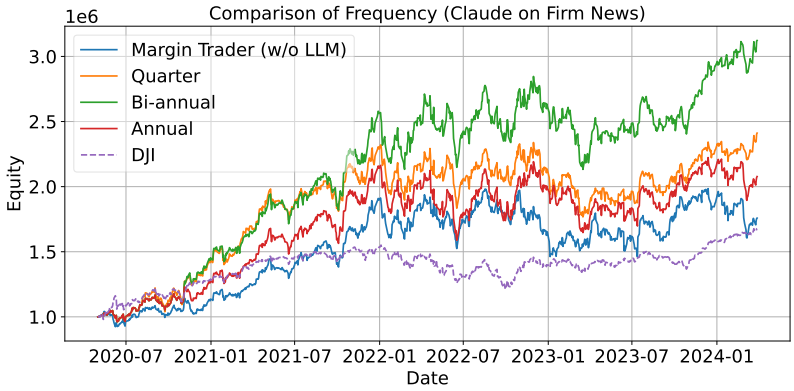


Figure 8.6 Equity changes on various adjustment frequencies.

Sharpe ratio, and Calmar ratio, particularly excelling with macro indicators and firm news.

8.4 Conclusion

Our study presents a novel framework that dynamically adjusts the long-short position ratio in portfolio management by leveraging the interaction between LLMs and RL. This framework ensures timely adaptation to evolving market conditions, offering transparency and informed decision-making through its explainable market reasoning module. Additionally, it provides traders with flexibility by enabling the integration of diverse external data sources and various LLMs. Comprehensive evaluations on real-world markets demonstrate that the intervention of LLMs significantly improves trading strategies in terms of both profitability and risk management. These findings underscore the transformative potential of LLMs and RL in revolutionizing portfolio management, paving the way for more adaptive, robust, and effective financial strategies.

CHAPTER 9

CONCLUSION AND FUTURE DIRECTIONS

9.1 Summary

This dissertation presents a unified and modular framework for robust financial intelligence that systematically addresses three critical challenges in financial modeling under uncertainty: market stochasticity, structural dynamics, and constrained decision-making. Through six studies, the framework presents tailored machine learning and AI-driven approaches that advance the state-of-the-art in forecasting, asset pricing, and portfolio management.

To address market stochasticity, Chapter 3 explores sentiment-based modeling, demonstrating how investor mood, especially around weekends and holidays, impacts next-day returns. By combining novel sentiment indices with GRU-based recurrent networks, the study outperforms conventional models in forecasting DJIA and S&P 500 movements. Chapter 4 introduces *IndexGAN*, a generative adversarial framework capable of producing multi-step stochastic forecasts for index returns. Unlike deterministic models, it captures volatility and external signals, improving forecast quality across different horizons. Chapter 5 presents *RAGIC*, a risk-aware generative-inference model for constructing adaptive prediction intervals. It balances sharpness and coverage better than conformal or quantile methods, providing a more reliable view of potential return ranges under uncertainty.

For structural dynamics, considering asset prices are not independent—they evolve within a dynamic, interconnected financial network shaped by sector relationships, macroeconomic shifts, and firm-level actions. Chapter 6 introduces *DySTAGE*, a dynamic spatiotemporal graph learning framework that captures both evolving inter-asset relationships and time-varying market composition. It offers

interpretable representations and achieves state-of-the-art performance in multi-asset price forecasting.

For constrained financial decision-making in real-world trading, strategies must adhere to operational constraints such as leverage limits, margin requirements, and short-selling rules. Chapter 7 develops *Margin Trader*, a constraint-aware reinforcement learning (RL) system that models realistic trading conditions. It generates robust long-short strategies while maintaining regulatory compliance and outperforming unconstrained RL agents. Chapter 8 further enhances decision-making under uncertainty by integrating LLMs to guide dynamic position reallocation based on textual and macroeconomic inputs. It enables explainable, context-sensitive adjustments and further boosts performance.

Collectively, these contributions form a cohesive, end-to-end pipeline that reflects real-world trading complexity and offers both predictive strength and operational realism. The framework is designed to be modular and scalable, making it suitable not only for academic exploration but also for practical deployment in diverse financial settings.

9.2 Potential Research Directions

Building on this foundation, several promising avenues for future research are identified, aiming at further enhancing the granularity, responsiveness, and practical utility of the proposed models in real-world financial systems.

Asset-level interval forecasting with option-based features While *RAGIC* has demonstrated strong performance in predicting interval returns for broad market indices, it can be extended to the level of individual stocks, where volatility is often driven by firm-specific events and idiosyncratic risks. Modeling such asset-level uncertainty would enable finer-grained prediction intervals tailored to specific stocks. In this context, features derived from option markets, particularly implied volatility

(IV) derived from call options via models like Black-Scholes, can serve as informative signals of anticipated volatility for the individual stock. Traders usually look at the IV of call options to gauge how much the market expects the stock to move. If IV rises without a corresponding price movement in the stock, it can signal anticipation of future volatility. Incorporating such features into interval prediction models can improve the characterization of uncertainty at the asset level. Moreover, these interval forecasts can be incorporated into downstream trading workflows, serving as input to portfolio allocation strategies, position sizing rules, or risk-based order execution frameworks. This integration would enhance decision-making by offering both predictive accuracy and quantified uncertainty.

Subgraph-based dynamic modeling The current *DySTAGE* model treats the financial market as a single evolving graph, which can be computationally intensive. Future research can explore subgraph-level modeling, where the dynamic graph is decomposed into meaningful clusters based on sector affiliation, correlation strength, or causal structure. Hierarchical attention mechanisms or localized message passing can then be applied to efficiently capture both local dependencies and global trends. Such granularity would improve scalability and computational efficiency, as well as reveal finer structural insights into market dynamics.

Multi-agent RL for long-short allocation The current *Margin Trader* separates position adjustment logic from portfolio execution. A natural extension is to formulate a multi-agent RL system, where one agent governs position rebalancing (e.g., long-short ratio) while another handles day-to-day execution under constraints. This layered architecture could improve adaptability and decision stability in volatile or segmented markets.

Event-triggered adjustment mechanisms Instead of fixed adjustment intervals for long-short ration, which may miss sudden regime shifts or volatility spikes, future models can incorporate event-triggered adjustment mechanisms that detects key signals, such as abrupt changes in volatility, liquidity stress, or macroeconomic shocks, for immediate, risk-sensitive position reallocation in response to real-time market signals, improving robustness during periods of market stress.

Real-time adaptation to evolving LLMs for market interpretation As new large language models are released, trading systems can dynamically integrate the latest models into the portfolio decision-making loop. This avoids data leakage from retrospectively applying outdated models and ensures that the system continually benefits from state-of-the-art language understanding. Real-time LLM switching can also enhance macro signal interpretation, sentiment tracking, and explainability for both traders and compliance stakeholders.

These directions will advance the framework toward becoming a responsive, interpretable, and institution-ready financial system, capable of adapting to fast-changing environments, learning from multi-modal inputs, and acting under realistic operational constraints.

REFERENCES

[1] C. Gollier, *The economics of risk and uncertainty.* Edward Elgar Publishing Limited, 2018.

[2] E. F. Fama, "Efficient capital markets: A review of theory and empirical work," *The Journal of Finance*, vol. 25, no. 2, pp. 383–417, 1970.

[3] ——, "The behavior of stock-market prices," *The Journal of Business*, vol. 38, no. 1, pp. 34–105, 1965.

[4] B. G. Malkiel, "The efficient market hypothesis and its critics," *Journal of Economic Perspectives*, vol. 17, no. 1, pp. 59–82, 2003.

[5] B. K. Nelson, "Time series analysis using autoregressive integrated moving average (arima) models," *Academic Emergency Medicine*, 1998.

[6] M. Ballings, D. Van den Poel, N. Hespeels, and R. Gryp, "Evaluating multiple classifiers for stock price direction prediction," *Expert Systems with Applications*, vol. 42, no. 20, pp. 7046–7056, 2015.

[7] J. Wang and J. Wang, "Forecasting stock market indexes using principle component analysis and stochastic time effective neural networks," *Neurocomputing*, vol. 156, pp. 68–78, 2015.

[8] M. U. Gudelek, S. A. Boluk, and A. M. Ozbayoglu, "A deep learning based stock trading model with 2-d cnn trend detection," in *2017 IEEE Symposium Series on Computational Intelligence (SSCI).* IEEE, 2017, pp. 1–8.

[9] Y. Qin, D. Song, H. Chen, W. Cheng, G. Jiang, and G. Cottrell, "A dual-stage attention-based recurrent neural network for time series prediction," *arXiv preprint arXiv:1704.02971*, 2017.

[10] F. Feng, H. Chen, X. He, J. Ding, M. Sun, and T.-S. Chua, "Enhancing stock movement prediction with adversarial training," *arXiv preprint arXiv:1810.09936*, 2018.

[11] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[12] Q. Ding, S. Wu, H. Sun, J. Guo, and J. Guo, "Hierarchical multi-scale gaussian transformer for stock movement prediction." in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI-20)*, 2020.

[13] Y. Duan, L. Wang, Q. Zhang, and J. Li, "Factorvae: A probabilistic dynamic factor model based on variational autoencoder for predicting cross-sectional stock returns," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 4, 2022.

[14] C. Hu and L. T. He, "An application of interval methods to stock market forecasting," *Reliable Computing*, vol. 13, no. 5, pp. 423–434, 2007.

[15] S. De, A. K. Dey, and D. K. Gouda, "Construction of confidence interval for a univariate stock price signal predicted through long short term memory network," *Annals of Data Science*, pp. 1–14, 2020.

[16] D. J. MacKay, "A practical bayesian framework for backpropagation networks," *Neural Computation*, vol. 4, no. 3, pp. 448–472, 1992.

[17] R. M. Neal, *Bayesian learning for neural networks*. Springer Science and Business Media, 2012, vol. 118.

[18] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *International Conference on Machine Learning*. PMLR, 2016, pp. 1050–1059.

[19] R. Koenker and K. F. Hallock, "Quantile regression," *Journal of Economic Perspectives*, vol. 15, no. 4, pp. 143–156, 2001.

[20] R. Wen, K. Torkkola, B. Narayanaswamy, and D. Madeka, "A multi-horizon quantile recurrent forecaster," *arXiv preprint arXiv:1711.11053*, 2017.

[21] A. Khosravi, S. Nahavandi, D. Creighton, and A. F. Atiya, "Lower upper bound estimation method for construction of neural network-based prediction intervals," *IEEE Transactions on Neural Networks*, vol. 22, no. 3, pp. 337–346, 2010.

[22] D. Shrestha, N. Kayastha, and D. Solomatine, "A novel approach to parameter uncertainty analysis of hydrological models using neural networks," *Hydrology and Earth System Sciences*, vol. 13, no. 7, pp. 1235–1248, 2009.

[23] S. Fort, H. Hu, and B. Lakshminarayanan, "Deep ensembles: A loss landscape perspective," *arXiv preprint arXiv:1912.02757*, 2019.

[24] G. Shafer and V. Vovk, "A tutorial on conformal prediction." *Journal of Machine Learning Research*, vol. 9, no. 3, 2008.

[25] K. Stankeviciute, A. M Alaa, and M. van der Schaar, "Conformal time-series forecasting," *Advances in Neural Information Processing Systems*, vol. 34, pp. 6216–6228, 2021.

[26] Z. Jiang, D. Xu, and J. Liang, "A deep reinforcement learning framework for the financial portfolio management problem," *arXiv preprint arXiv:1706.10059*, 2017.

[27] J. Wang, Y. Zhang, K. Tang, J. Wu, and Z. Xiong, "Alphastock: A buying-winners-and-selling-losers investment strategy using interpretable deep reinforcement attention networks," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.

[28] R. Wang, H. Wei, B. An, Z. Feng, and J. Yao, "Commission fee is not enough: A hierarchical reinforced framework for portfolio management," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021.

[29] M. Yang, X. Zheng, Q. Liang, B. Han, and M. Zhu, "A smart trader for portfolio management based on normalizing flows," in *Proceedings of the International Joint Conference on Artificial Intelligence*, 2022.

[30] Y. Liu, Q. Liu, H. Zhao, Z. Pan, and C. Liu, "Adaptive quantitative trading: An imitative deep reinforcement learning approach," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 02, 2020.

[31] Z. Wang, B. Huang, S. Tu, K. Zhang, and L. Xu, "Deeptrader: a deep reinforcement learning approach for risk-return balanced portfolio management with market conditions embedding," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 1, 2021, pp. 643–650.

[32] H. Niu, S. Li, and J. Li, "Metatrader: An reinforcement learning approach integrating diverse policies for portfolio optimization," in *Proceedings of the 31st ACM International Conference on Information and Knowledge Management*, 2022.

[33] K. Xu, Y. Zhang, D. Ye, P. Zhao, and M. Tan, "Relation-aware transformer for portfolio policy learning," in *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, 2021, pp. 4647–4653.

[34] X.-Y. Liu, Z. Xia, J. Rui, J. Gao, H. Yang, M. Zhu, C. Wang, Z. Wang, and J. Guo, "Finrl-meta: Market environments and benchmarks for data-driven financial reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 35, 2022.

[35] X. Li, Y. Li, H. Yang, L. Yang, and X.-Y. Liu, "Dp-lstm: Differential privacy-inspired lstm for stock prediction using financial news," *arXiv preprint arXiv:1912.10806*, 2019.

[36] X. Zhang, H. Fuehres, and P. A. Gloor, "Predicting stock market indicators through twitter "i hope it is not as bad as i fear"," *Procedia-Social and Behavioral Sciences*, vol. 26, pp. 55–62, 2011.

[37] E. J. De Fortuny, T. De Smedt, D. Martens, and W. Daelemans, "Evaluating and understanding text-based stock price prediction models," *Information Processing and Management*, vol. 50, no. 2, pp. 426–441, 2014.

[38] D. Araci, "Finbert: Financial sentiment analysis with pre-trained language models," *arXiv preprint arXiv:1908.10063*, 2019.

[39] J. Tang and X. Chen, "Stock market prediction based on historic prices and news titles," in *Proceedings of the 2018 International Conference on Machine Learning Technologies*, 2018, pp. 29–34.

[40] J. B. Nascimento and M. Cristo, "The impact of structured event embeddings on scalable stock forecasting models," in *Proceedings of the 21st Brazilian Symposium on Multimedia and the Web*, 2015, pp. 121–124.

[41] Y. Liu, Q. Zeng, H. Yang, and A. Carrio, "Stock price movement prediction from financial news with deep learning and knowledge graph embedding," in *Knowledge Management and Acquisition for Intelligent Systems: 15th Pacific Rim Knowledge Acquisition Workshop, PKAW 2018, Nanjing, China, August 28-29, 2018, Proceedings 15*. Springer, 2018, pp. 102–113.

[42] Z. S. Harris, "Distributional structure," *Word*, vol. 10, no. 2-3, pp. 146–162, 1954.

[43] C.-Y. Lee and V.-W. Soo, "Predict stock price with financial news based on recurrent convolutional neural networks," in *2017 Conference on Technologies and Applications of Artificial Intelligence (TAAI)*. IEEE, 2017, pp. 160–165.

[44] D. L. Minh, A. Sadeghi-Niaraki, H. D. Huy, K. Min, and H. Moon, "Deep learning approach for short-term stock trends prediction based on two-stream gated recurrent unit network," *IEEE Access*, vol. 6, pp. 55 392–55 404, 2018.

[45] W. Li, R. Bao, K. Harimoto, D. Chen, J. Xu, and Q. Su, "Modeling the stock relation with graph network for overnight stock movement prediction," in *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, 2021, pp. 4541–4547.

[46] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543.

[47] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[48] A. Dash, J. Gu, and G. Wang, "Hi-gan: Hierarchical inpainting gan with auxiliary inputs for combined rgb and depth inpainting," *arXiv preprint arXiv:2402.10334*, 2024.

[49] A. Dash, J. Gu, G. Wang, and C. Chen, "Eye-see-you: Reverse pass-through vr and head avatars," *arXiv preprint arXiv:2505.18869*, 2025.

[50] M. Saito, E. Matsumoto, and S. Saito, "Temporal generative adversarial nets with singular value clipping," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2830–2839.

[51] C. Esteban, S. L. Hyland, and G. Rätsch, "Real-valued (medical) time series generation with recurrent conditional gans," *arXiv preprint arXiv:1706.02633*, 2017.

[52] Y. Zhang, Z. Gan, and L. Carin, "Generating text via adversarial training," in *NIPS Workshop on Adversarial Training*, vol. 21. academia. edu, 2016, pp. 21–32.

[53] A. Gupta and J. Zou, "Feedback gan for dna optimizes protein functions," *Nature Machine Intelligence*, vol. 1, no. 2, pp. 105–111, 2019.

[54] S. Takahashi, Y. Chen, and K. Tanaka-Ishii, "Modeling financial time-series with generative adversarial networks," *Physica A: Statistical Mechanics and its Applications*, vol. 527, p. 121261, 2019.

[55] H.-W. Dong, W.-Y. Hsiao, L.-C. Yang, and Y.-H. Yang, "Musegan: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[56] L. Yu, W. Zhang, J. Wang, and Y. Yu, "Seqgan: Sequence generative adversarial nets with policy gradient," in *Proceedings of the Association for the Advancement of Artificial Intelligence (AAAI) Conference on Artificial Intelligence*, vol. 31, no. 1, 2017.

[57] J. Yoon, D. Jarrett, and M. Van der Schaar, "Time-series generative adversarial networks," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[58] M. Wiese, R. Knobloch, R. Korn, and P. Kretschmer, "Quant gans: Deep generation of financial time series," *Quantitative Finance*, vol. 20, no. 9, pp. 1419–1440, 2020.

[59] Y. Xu, Z. Zhang, L. You, J. Liu, Z. Fan, and X. Zhou, "scigans: single-cell rna-seq imputation using generative adversarial networks," *Nucleic Acids Research*, vol. 48, no. 15, pp. e85–e85, 2020.

[60] L. Wu, Y. Xia, F. Tian, L. Zhao, T. Qin, J. Lai, and T.-Y. Liu, "Adversarial neural machine translation," in *Asian Conference on Machine Learning*. PMLR, 2018, pp. 534–549.

[61] K. Zhang, G. Zhong, J. Dong, S. Wang, and Y. Wang, "Stock market prediction based on generative adversarial network," *Procedia Computer Science*, vol. 147, pp. 400–406, 2019.

[62] M. Faraz and H. Khaloozadeh, "Multi-step-ahead stock market prediction based on least squares generative adversarial network," in *2020 28th Iranian Conference on Electrical Engineering (ICEE)*. IEEE, 2020, pp. 1–6.

[63] P. Sonkiya, V. Bajpai, and A. Bansal, "Stock price prediction using bert and gan," *arXiv preprint arXiv:2107.09055*, 2021.

[64] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *International Conference on Learning Representations*, 2017.

[65] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[66] P. Veli**v**cković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=rJXMpikCZ

[67] Y. Shi, Z. Huang, S. Feng, H. Zhong, W. Wang, and Y. Sun, "Masked label prediction: Unified message passing model for semi-supervised classification," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence (IJCAI-20)*, 2021.

[68] R. Kim, C. H. So, M. Jeong, S. Lee, J. Kim, and J. Kang, "Hats: A hierarchical graph attention network for stock movement prediction," *arXiv preprint arXiv:1908.07999*, 2019.

[69] R. Cheng and Q. Li, "Modeling the momentum spillover effect for stock prediction via attribute-driven graph attention networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021.

[70] A. Uddin, X. Tao, and D. Yu, "Attention based dynamic graph learning framework for asset pricing," in *Proceedings of the 30th ACM International Conference on Information and Knowledge Management*, 2021.

[71] J. Yoo, Y. Soun, Y.-c. Park, and U. Kang, "Accurate multivariate stock movement prediction via data-axis transformer with multi-level contexts," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2021.

[72] L. Zhao, Y. Song, C. Zhang, Y. Liu, P. Wang, T. Lin, M. Deng, and H. Li, "T-gcn: A temporal graph convolutional network for traffic prediction," *IEEE Transactions on Intelligent Transportation Systems*, 2019.

[73] J. Chen, X. Wang, and X. Xu, "Gc-lstm: Graph convolution embedded lstm for dynamic network link prediction," *Applied Intelligence*, 2022.

[74] A. Sankar, Y. Wu, L. Gou, W. Zhang, and H. Yang, "Dysat: Deep neural representation learning on dynamic graphs via self-attention networks," in *Proceedings of the 13th International Conference on Web Search and Data Mining*, 2020.

[75] A. Pareja, G. Domeniconi, J. Chen, T. Ma, T. Suzumura, H. Kanezashi, T. Kaler, T. Schardl, and C. Leiserson, "Evolvegcn: Evolving graph convolutional networks for dynamic graphs," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[76] K. Zhang, Q. Cao, G. Fang, B. Xu, H. Zou, H. Shen, and X. Cheng, "Dyted: Disentangled representation learning for discrete-time dynamic graph," in *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2023.

[77] H. Yuan, Q. Sun, X. Fu, C. Ji, and J. Li, "Dynamic graph information bottleneck," in *The Web Conference 2024*, 2024.

[78] A. C. Teixeira, V. Marar, H. Yazdanpanah, A. Pezente, and M. Ghassemi, "Enhancing credit risk reports generation using llms: An integration of bayesian networks and labeled guide prompting," in *Proceedings of the Fourth ACM International Conference on AI in Finance*, 2023, pp. 340–348.

[79] Z. Zeng, W. Watson, N. Cho, S. Rahimi, S. Reynolds, T. Balch, and M. Veloso, "Flowmind: automatic workflow generation with llms," in *Proceedings of the Fourth ACM International Conference on AI in Finance*, 2023, pp. 73–81.

[80] X. Yu, Z. Chen, and Y. Lu, "Harnessing llms for temporal data: A study on explainable financial time series forecasting," in *Proceedings of the Empirical Methods in Natural Language Processing (EMNLP) 2023 Industry Track*. Association for Computational Linguistics, 2023.

[81] H. Yang, X.-Y. Liu, and C. D. Wang, "Fingpt: Open-source financial large language models," *FinLLM Symposium at International Joint Conference on Artificial Intelligence (IJCAI) 2023*, 2023.

[82] S. Wu, O. Irsoy, S. Lu, V. Dabravolski, M. Dredze, S. Gehrmann, P. Kambadur, D. Rosenberg, and G. Mann, "Bloomberggpt: A large language model for finance," *arXiv preprint arXiv:2303.17564*, 2023.

[83] B. Zhang, H. Yang, T. Zhou, M. Ali Babar, and X.-Y. Liu, "Enhancing financial sentiment analysis via retrieval augmented large language models," in *Proceedings of the Fourth ACM International Conference on AI in Finance*, 2023, pp. 349–356.

[84] J. Gu, S. Shukla, J. Ye, A. Uddin, and G. Wang, "Deep learning model with sentiment score and weekend effect in stock price prediction," *SN Business and Economics*, vol. 3, no. 7, p. 119, 2023.

[85] A. M. Rahman, A. Uddin, and G. G. Wang, "Hodl: The hold of reddit over the stock market," in *2023 IEEE 39th International Conference on Data Engineering Workshops (ICDEW)*. IEEE, 2023.

[86] X. Deng, V. Bashlovkina, F. Han, S. Baumgartner, and M. Bendersky, "What do llms know about financial markets? a case study on reddit market sentiment analysis," in *Companion Proceedings of the ACM Web Conference 2023*, 2023, pp. 107–110.

[87] J. Gu, F. P. Deek, and G. Wang, "Stock broad-index trend patterns learning via domain knowledge informed generative network," *arXiv preprint arXiv:2302.14164*, 2023.

[88] J. Gu, W. Du, and G. Wang, "Ragic: Risk-aware generative framework for stock interval construction," *IEEE Transactions on Knowledge and Data Engineering*, 2025.

[89] Y. Yu, H. Li, Z. Chen, Y. Jiang, Y. Li, D. Zhang, R. Liu, J. W. Suchow, and K. Khashanah, "Finmem: A performance-enhanced llm trading agent with layered memory and character design," in *Proceedings of the Association for the Advancement of Artificial Intelligence (AAAI) Symposium Series*, vol. 3, no. 1, 2024, pp. 595–597.

[90] I. O. Gallegos, R. A. Rossi, J. Barrow, M. M. Tanjim, S. Kim, F. Dernoncourt, T. Yu, R. Zhang, and N. K. Ahmed, "Bias and fairness in large language models: A survey," *Computational Linguistics*, 2024.

[91] K. Kim and D. Ryu, "Sentiment changes and the monday effect," *Finance Research Letters*, vol. 47, p. 102709, 2022.

[92] K. R. French, "Stock returns and the weekend effect," *Journal of Financial Economics*, vol. 8, no. 1, pp. 55–69, 1980.

[93] R. Ren, D. D. Wu, and T. Liu, "Forecasting stock market movement direction using sentiment analysis and support vector machine," *IEEE Systems Journal*, vol. 13, no. 1, pp. 760–770, 2018.

[94] T. B. Shahi, A. Shrestha, A. Neupane, and W. Guo, "Stock price forecasting with deep learning: A comparative study," *Mathematics*, vol. 8, no. 9, p. 1441, 2020.

[95] C. Hutto and E. Gilbert, "Vader: A parsimonious rule-based model for sentiment analysis of social media text," in *Proceedings of the International Association for the Advancement of Artificial Intelligence (AAAI) Conference on Web and Social Media*, vol. 8, no. 1, 2014.

[96] W. Antweiler and M. Z. Frank, "Is all that talk just noise? the information content of internet stock message boards," *The Journal of Finance*, vol. 59, no. 3, pp. 1259–1294, 2004.

[97] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. Ieee, 2013, pp. 6645–6649.

[98] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statistics and Computing*, vol. 14, pp. 199–222, 2004.

[99] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, 2001.

[100] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 785–794.

[101] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *International Conference on Machine Learning*. PMLR, 2017, pp. 214–223.

[102] Y. Rubner, C. Tomasi, and L. J. Guibas, "The earth mover's distance as a metric for image retrieval," *International Journal of Computer Vision*, vol. 40, no. 2, pp. 99–121, 2000.

[103] F. Klinker, "Exponential moving average versus moving exponential average," *Mathematische Semesterberichte*, vol. 58, no. 1, pp. 97–107, 2011.

[104] S. Makridakis and M. Hibon, "Arma models and the box–jenkins methodology," *Journal of Forecasting*, vol. 16, no. 3, pp. 147–163, 1997.

[105] Y. Xu and S. B. Cohen, "Stock movement prediction from tweets and historical prices," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018, pp. 1970–1979.

[106] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[107] I. Yaniv and D. P. Foster, "Graininess of judgment under uncertainty: An accuracy-informativeness trade-off." *Journal of Experimental Psychology: General*, vol. 124, no. 4, p. 424, 1995.

[108] M. Lawrence, P. Goodwin, M. O'Connor, and D. Önkal, "Judgmental forecasting: A review of progress over the last 25 years," *International Journal of Forecasting*, vol. 22, no. 3, 2006.

[109] K. Rasul, C. Seward, I. Schuster, and R. Vollgraf, "Autoregressive denoising diffusion models for multivariate probabilistic time series forecasting," in *International Conference on Machine Learning*. PMLR, 2021, pp. 8857–8868.

[110] Y. Li, X. Lu, Y. Wang, and D. Dou, "Generative time series forecasting with diffusion, denoise, and disentanglement," in *Advances in Neural Information Processing Systems*, A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, Eds., 2022.

[111] R. Xiong, Y. Yang, D. He, K. Zheng, S. Zheng, C. Xing, H. Zhang, Y. Lan, L. Wang, and T. Liu, "On layer normalization in the transformer architecture," in *Proceedings of the 37 th International Conference on Machine Learning*. PMLR, 2020.

[112] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[113] C.-H. Chiang, "Stock returns on option expiration dates: Price impact of liquidity trading," *Journal of Empirical Finance*, vol. 28, pp. 273–290, 2014.

[114] G. M. Mian and S. Sankaraguruswamy, "Investor sentiment and stock market response to earnings news," *The Accounting Review*, vol. 87, no. 4, pp. 1357–1384, 2012.

[115] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint arXiv:1803.01271*, 2018.

[116] J. Bollinger, *Bollinger on Bollinger bands*. McGraw-Hill New York, 2002.

[117] Y. Du, J. Wang, W. Feng, S. Pan, T. Qin, R. Xu, and C. Wang, "Adarnn: Adaptive learning and forecasting of time series," in *Proceedings of the 30th ACM International Conference on Information and Knowledge Management*, 2021, pp. 402–411.

[118] J. Ye, B. Goswami, J. Gu, A. Uddin, and G. Wang, "From factor models to deep learning: Machine learning in reshaping empirical asset pricing," *arXiv preprint arXiv:2403.06779*, 2024.

[119] C. A. Ramírez, "Firm networks and asset returns," *The Review of Financial Studies*, vol. 37, no. 10, pp. 3050–3091, 2024.

[120] A. Uddin, X. Tao, and D. Yu, "The network factor of equity pricing: A signed graph laplacian approach," *Journal of Financial Econometrics*, 2024.

[121] L. F. Di Cerbo and S. Taylor, "Graph theoretical representations of equity indices and their centrality measures," *Quantitative Finance*, 2021.

[122] W. Yao, J. Gu, W. Du, F. P. Deek, and G. Wang, "Adpp: A novel anomaly detection and privacy-preserving framework using blockchain and neural networks in tokenomics," *International Journal of Artificial Intelligence and Applications*, 2022.

[123] O. A. Rodriguez, Z. Du, J. Patchett, F. Li, and D. A. Bader, "Arachne: An arkouda package for large-scale graph analytics," in *2022 IEEE High Performance Extreme Computing Conference (HPEC)*. IEEE, 2022, pp. 1–7.

[124] Y. Chen, Z. Wei, and X. Huang, "Incorporating corporation relationship via graph convolutional neural networks for stock price prediction," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 2018.

[125] J. Bai, J. Zhu, Y. Song, L. Zhao, Z. Hou, R. Du, and H. Li, "A3t-gcn: Attention temporal graph convolutional network for traffic forecasting," *ISPRS International Journal of Geo-Information*, 2021.

[126] B. T. Kelly, S. Pruitt, and Y. Su, "Characteristics are covariances: A unified model of risk and return," *Journal of Financial Economics*, 2019.

[127] J. Xu, X. Sun, Z. Zhang, G. Zhao, and J. Lin, "Understanding and improving layer normalization," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[128] S. Narang, H. W. Chung, Y. Tay, W. Fedus, T. Fevry, M. Matena, K. Malkan, N. Fiedel, N. Shazeer, Z. Lan *et al.*, "Do transformer modifications transfer across implementations and applications?" *arXiv preprint arXiv:2102.11972*, 2021.

[129] A. Y. Chen and T. Zimmermann, "Open source cross-sectional asset pricing," *Critical Finance Review*, 2022.

[130] G. Coqueret and T. Guida, *Machine Learning for Factor Investing: Python Version*. CRC Press, 2023.

[131] T. C. Mills, *Time series techniques for economists*. Cambridge University Press, 1990.

[132] B. N. Oreshkin, D. Carpov, N. Chapados, and Y. Bengio, "N-beats: Neural basis expansion analysis for interpretable time series forecasting," *arXiv preprint arXiv:1905.10437*, 2019.

[133] F. M. Bianchi, D. Grattarola, L. Livi, and C. Alippi, "Graph neural networks with convolutional arma filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 7, pp. 3496–3507, 2021.

[134] J. Lussange, I. Lazarevich, S. Bourgeois-Gironde, S. Palminteri, and B. Gutkin, "Modelling stock markets by multi-agent reinforcement learning," *Computational Economics*, vol. 57, pp. 113–147, 2021.

[135] B. Hambly, R. Xu, and H. Yang, "Recent advances in reinforcement learning in finance," *Mathematical Finance*, vol. 33, no. 3, pp. 437–503, 2023.

[136] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT Press, 2018.

[137] S. Sun, R. Wang, and B. An, "Reinforcement learning for quantitative trading," *ACM Transactions on Intelligent Systems and Technology*, vol. 14, no. 3, pp. 1–29, 2023.

[138] S. Almahdi and S. Y. Yang, "An adaptive portfolio trading system: A risk-return portfolio optimization using recurrent reinforcement learning with expected maximum drawdown," *Expert Systems with Applications*, vol. 87, 2017.

[139] A. Millea, "Deep reinforcement learning for trading—a critical survey," *Data*, vol. 6, no. 11, p. 119, 2021.

[140] W. Du, J. Ye, J. Gu, J. Li, H. Wei, and G. Wang, "Safelight: A reinforcement learning method toward collision-free traffic signal control," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 37, no. 12, 2023, pp. 14 801–14 810.

[141] Y. Zhang, P. Zhao, Q. Wu, B. Li, J. Huang, and M. Tan, "Cost-sensitive portfolio selection via deep reinforcement learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 1, pp. 236–248, 2020.

[142] S. Sun, W. Xue, R. Wang, X. He, J. Zhu, J. Li, and B. An, "Deepscalper: A risk-aware reinforcement learning framework to capture fleeting intraday trading opportunities," in *Proceedings of the 31st ACM International Conference on Information and Knowledge Management*, 2022.

[143] H. Yang, X.-Y. Liu, S. Zhong, and A. Walid, "Deep reinforcement learning for automated stock trading: An ensemble strategy," in *Proceedings of the First ACM International Conference on AI in Finance*, 2020.

[144] Y. Li, P. Ni, and V. Chang, "Application of deep reinforcement learning in stock trading strategies and stock forecasting," *Computing*, 2020.

[145] E. Dudley and M. Nimalendran, "Margins and hedge fund contagion," *Journal of Financial and Quantitative Analysis*, vol. 46, no. 5, pp. 1227–1257, 2011.

[146] P. Fortune *et al.*, "Margin requirements, margin loans, and margin rates: Practice and principles," *New England Economic Review*, pp. 19–44, 2000.

[147] X.-Y. Liu, H. Yang, J. Gao, and C. D. Wang, "Finrl: Deep reinforcement learning framework to automate trading in quantitative finance," in *Proceedings of the Second ACM International Conference on AI in Finance*, 2021.

[148] W. F. Sharpe, "The sharpe ratio," *Streetwise–the Best of the Journal of Portfolio Management*, vol. 3, pp. 169–85, 1998.

[149] E. Vittori, M. Trapletti, and M. Restelli, "Option hedging with risk averse reinforcement learning," in *Proceedings of the First ACM International Conference on AI in Finance*, 2020.

[150] R. A. de Oliveira, H. S. Ramos, D. H. Dalip, and A. C. M. Pereira, "A tabular sarsa-based stock market agent," in *Proceedings of the First ACM International Conference on AI in Finance*, 2020.

[151] G. Jeong and H. Y. Kim, "Improving financial trading decisions using deep q-learning: Predicting the number of shares, action strategies, and transfer learning," *Expert Systems with Applications*, vol. 117, pp. 125–138, 2019.

[152] Z. Xiong, X.-Y. Liu, S. Zhong, H. B. Yang, and A. Walid, "Practical deep reinforcement learning approach for stock trading (2023)," *arXiv preprint arXiv:1811.07522*, 2018.

[153] Y. Ye, H. Pei, B. Wang, P.-Y. Chen, Y. Zhu, J. Xiao, and B. Li, "Reinforcement-learning based portfolio management with augmented asset movement prediction states," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 01, 2020.

[154] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2016.

[155] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[156] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[157] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International Conference on Machine Learning*. PMLR, 2018.

[158] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, "Stable-baselines3: Reliable reinforcement learning implementations," *The Journal of Machine Learning Research*, vol. 22, no. 1, 2021.

[159] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat *et al.*, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[160] A. Anthropic, "The claude 3 model family: Opus, sonnet, haiku," *Claude-3 Model Card*, vol. 1, 2024.

[161] H. Zhao, Z. Liu, Z. Wu, Y. Li, T. Yang, P. Shu, S. Xu, H. Dai, L. Zhao, G. Mai *et al.*, "Revolutionizing finance with llms: An overview of applications and insights," *arXiv preprint arXiv:2401.11641*, 2024.

[162] OpenAI, "Hello gpt-4o," https://openai.com/index/hello-gpt-4o/, 2024, accessed: 2024-07-15.

[163] Anthropic, "Introducing claude 3.5 sonnet," https://www.anthropic.com/news/claude-3-5-sonnet, 2024, accessed: 2024-07-15.

[164] W. Liang, A. Liu, B. Feng *et al.*, "Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model," 2024.

[165] M. AI, "Cheaper, better, faster, stronger," https://mistral.ai/news/mixtral-8x22b/, 2024, accessed: 2024-07-15.

[166] ——, "Introducing meta llama 3: The most capable openly available llm to date," https://ai.meta.com/blog/meta-llama-3/, 2024, accessed: 2024-07-15.

[167] Q. Team, "Hello qwen2," https://qwenlm.github.io/blog/qwen2/, 2024, accessed: 2024-07-15.

[168] A. Young, B. Chen, C. Li, C. Huang, G. Zhang, G. Zhang, H. Li, J. Zhu, J. Chen, J. Chang *et al.*, "Yi: Open foundation models by 01. ai," *arXiv preprint arXiv:2403.04652*, 2024.

[169] M. Abdin, S. A. Jacobs, A. A. Awan, J. Aneja, A. Awadallah, H. Awadalla, N. Bach, A. Bahree, A. Bakhtiari, H. Behl *et al.*, "Phi-3 technical report: A highly capable language model locally on your phone," *arXiv preprint arXiv:2404.14219*, 2024.

[170] J. Gu, W. Du, A. M. Rahman, and G. Wang, "Margin trader: a reinforcement learning framework for portfolio management with margin and constraints," in *Proceedings of the Fourth ACM International Conference on AI in Finance*, 2023, pp. 610–618.