From Prediction to Decision in Financial Markets: Deep Learning in Market Heterogeneity and Multi-Agent Portfolio Optimization

XIAOBIN LU

THE UNIVERSITY OF
SYDNEY

Supervisor: Josiah Poon
Associate Supervisor: Matloob Khushi

A thesis submitted in fulfilment of
the requirements for the degree of
Master of Philosophy

School of Computer Science
Faculty of Engineering
The University of Sydney
Australia

30 March 2026

Statement of Originality

This is to certify that the content of this thesis is my own work. This thesis has not been submitted for any other degree or purpose.

I certify that the intellectual content of this thesis is the product of my own work, and that all assistance received in preparing this thesis and all sources have been acknowledged.

Xiaobin Lu

Signature:

Use of Generative AI Statement

During the preparation of this thesis, the author used generative artificial intelligence tools, specifically GPT–4, Gemini, and Grok, for the sole purpose of minor text enhancement. The use of these tools was limited to improving clarity, correcting grammar, and adjusting sentence structure in selected sections of the manuscript. No generative AI tools were used to generate ideas, conduct analyses, contribute conceptual content, or produce any substantive parts of the research design, results, or discussion.

The author confirms that all text modified by generative AI was carefully reviewed to ensure accuracy, eliminate potential errors, and assess any risk of bias. All intellectual content, arguments, interpretations, analytical decisions, and academic contributions presented in this thesis are entirely the author's own work.

The author takes full responsibility for the originality, accuracy, and integrity of the submitted thesis and confirms that the use of generative AI tools has complied fully with the University of Sydney's Generative AI Guidelines for Researchers, the Academic Integrity Policy, and all other relevant institutional protocols.

Xiaobin Lu

Signature:

Authorship Attribution Statement

Chapter 3 of this thesis has been published as: Lu, X., Poon, J., and Khushi, M. (2024). Bridging the gap between machine and human in stock prediction: Addressing heterogeneity in stock market. IEEE Access. I am the sole first author of this publication. I designed the study, conducted all research activities, performed the analyses, and drafted the manuscript independently.

Chapter 4 of this thesis has been submitted to Knowledge Systems. I am the sole first author of this manuscript. I designed the study, conducted all research activities, performed the analyses, and drafted the manuscript independently.

Chapter 5 of this thesis has been published as: Lu, X., Poon, J., and Khushi, M. (2025). Leveraging BiLSTM-GAT for enhanced stock market prediction: A dual-graph approach to portfolio optimization. Applied Intelligence, 55(7), 601. I am the sole first author of this publication. I designed the study, conducted all research activities, performed the analyses, and drafted the manuscript independently.

In addition to the authorship attribution statements above, in cases where I am not the corresponding author of a published item, permission to include the published material in this thesis has been granted by the corresponding author.

Xiaobin Lu
Signature:

As supervisor for the candidature upon which this thesis is based, I confirm that the authorship attribution statements above are correct.

Josiah Poon
Signature:

Abstract

This thesis presents a systematic progression from predictive modelling to autonomous decision-making in financial markets, unified by the insight that financial time series are fundamentally distinguished from traditional domains by extreme asset heterogeneity, severe data sparsity, pronounced non-stationarity, and intricate inter-asset dependencies that cannot be adequately captured by historical price data alone. The journey begins with the demonstration that aggregating stocks into universal numerical models neglects firm-specific characteristics and yields poor generalisation. A hybrid BiLSTM-Transformer architecture is proposed that incorporates BERT-encoded business descriptions capturing unique attributes such as industry sector and operational scale combined with a series of technical indicators and dimensionality reduction via Restricted Boltzmann Machines. This framework achieves superior accuracy and robust performance on previously unseen companies, establishing that effective stock prediction requires explicit modelling of corporate heterogeneity. Building upon this foundation, the distinctive constraints of financial seq2seq forecasting are addressed through a novel patch-based Mamba-CrossAttention Network. By combining residual-connected Mamba blocks with localized patch processing and adaptive cross-layer attention, the architecture efficiently extracts temporal dynamics under conditions of extreme volatility and low signal-to-noise ratios, delivering state-of-the-art results across major global stock indices. The analysis then extends to portfolio construction, revealing that isolated stock forecasts ignore systemic relational structures. A dual-graph BiLSTM-GAT framework is introduced, simultaneously learning technical similarity from price co-movements and fundamental affinity from industry relationships, with alignment performed via an attention mechanism. Backtesting demonstrates consistent better performance of the S&P 500 in risk-adjusted returns, confirming the critical role of relational modelling in translating predictions into optimised allocations. These contributions culminate in a neuro-symbolic multi-agent system that integrates deep learning forecasts, technical chart patterns, and news sentiment

within a collaborative LLM-orchestrated architecture. Specialised analyst agents generate structured reports that a trader agent synthesises iteratively under real-time market feedback and portfolio constraints. Extensive evaluation on representative equities across bull, bear, and sideways regimes reveals substantially enhanced robustness, interpretability, and profitability compared to conventional deep learning pipelines. Collectively, this work establishes a coherent paradigm for financial artificial intelligence: progression from firm-specific prediction, through purpose-built sequential and relational modelling, to neuro-symbolic multi-agent decision-making, offering a foundational framework for next-generation autonomous trading systems.

Acknowledgements

I would like to express my deepest gratitude to my supervisors, Associate Professor Josiah Poon and Dr. Matloob Khushi, for their invaluable guidance, unwavering support, and continuous encouragement throughout the course of my research. Their extensive knowledge, scholarly insight, and constructive feedback have played a pivotal role in shaping both the direction and quality of this thesis. I am sincerely appreciative of their dedication to academic excellence and their commitment to fostering rigorous, independent inquiry. This research journey has been intellectually demanding and personally transformative. The challenges encountered along the way have strengthened my analytical skills, refined my methodological understanding, and deepened my engagement with the field. I would like to extend my heartfelt thanks to my fellow students and friends—Jack Lou, Haeri Min, Michael Mai, Yuben Yang, and Jinrong Fu—as well as to the entire postgraduate community in Room 4E of the School of Computer Science, including other students and professors. Their camaraderie, support, and everyday exchanges have created a collaborative and intellectually engaging environment that greatly enriched my research experience. The sense of community and shared dedication in Room 4E has been both motivating and deeply appreciated throughout this journey. Finally, I wish to thank the School of Computer Science at the University of Sydney for providing the academic infrastructure, research resources, and vibrant intellectual environment essential to this work. The opportunity to undertake the Master of Philosophy program within such a dynamic setting has been a defining and enriching stage of my academic and professional development.

Contents

Statement of Originality ii

Use of Generative AI Statement iii

Authorship Attribution Statement iv

Abstract v

Acknowledgements vii

Contents viii

List of Figures xv

List of Publications 1

List of Publications 1

Chapter 1 Introduction 2

1.1 Overview 2

1.2 Background and Motivation 3

1.3 Research Aim and Questions 5

1.4 Thesis Structure 7

Chapter 2 Literature review 9

2.1 Deep learning in financial time series forecasting 9

2.2 Portfolio Management and Decision Making in Deep Learning 11

2.3 Multi-agent and multimodal trading system 14

2.4 Gap from current literature 16

Chapter 3 Modeling Market Heterogeneity Using Embedded Stock Descriptions 18

3.1 Introduction 18

3.2 Motivation 19
3.3 Method 20
3.3.1 Technique indicators 20
3.3.1.1 Simple Moving Averages (SMA) 20
3.3.1.2 Bollinger Bands 21
3.3.1.3 Relative Strength Index (RSI) 22
3.3.1.4 Price Percentage Change Features 23
3.3.1.5 Moving Average Percentage Change Features 23
3.3.1.6 RSI Percentage Change Features 24
3.3.1.7 Bollinger Band Percentage Change Features 24
3.3.1.8 Rolling Maximum and Minimum 25
3.3.1.9 Close Price Shifts 26
3.3.1.10 Trading time 26
3.3.2 Bidirectional Encoder Representations from Transformers (BERT) 26
3.3.3 Restricted Boltzmann machine (RBM) 28
3.3.4 Long Short-Term Memory (LSTM) 29
3.3.5 Transformer encoder 30
3.3.5.1 Multi-Headed Self-Attention 31
3.3.5.2 Position-Wise Feedforward Network 31
3.3.5.3 Layer Normalization and Residual Connections 32
3.3.6 Proposed model 32
3.3.7 Evaluation metrics 33
3.4 Experimental Setup 34
3.4.1 Data source 34
3.4.2 Data Pre-processing 34
3.4.2.1 Vectorised company description 34
3.4.2.2 Scaling 35
3.4.2.3 Train test split 36
3.4.3 Software and Hardware Setup 37
3.5 Experimental Results and Comparative Analysis 37

3.5.1 Results on 1-100 companies and comparison with the SOTA models 38
3.5.2 Results on 101-170 (Unseen) Companies 41
3.6 Discussion of results 42
3.6.1 Comparative Performance with SOTA Models 43
3.6.2 Generalization to Unseen Companies 45
3.6.3 Time complexity analysis and experimental studies 45
3.6.4 Implications for Financial Modeling 46
3.6.5 Addressing the Limitation of Comparative Analysis 47
3.7 Conclusion and future work 48
3.7.1 Conclusion 48

Chapter 4 Cross-Attention Mamba Seq2Seq Model for Short-Term Prediction 50
4.1 Introduction 50
4.2 Motivation 51
4.3 Methodology 53
4.3.1 Technical Indicators 53
4.3.1.1 Moving Averages (MA) 53
4.3.1.2 Moving Average Convergence Divergence (MACD) 54
4.3.1.3 Bollinger Bands (BB) 55
4.3.1.4 Indicator Parameters 56
4.3.2 Mamba block 56
4.3.3 Cross-attention block 57
4.3.4 Proposed model 59
4.4 Experimental Setup 59
4.4.1 Data source and processing 59
4.4.2 Hardware & Software 61
4.5 Experimental Results 61
4.6 Discussion 61
4.7 Conclusion and Future Work 64
4.7.1 Conclusion 64

Chapter 5 Portfolio Management With GAT-based Inter-stock Dependency Modeling 66

5.1 Introduction . . . 66
5.2 Motivation . . . 67
5.3 Methodology . . . 69
5.3.1 Technique indicators . . . 69
5.3.1.1 Simple Moving Averages (SMA) . . . 70
5.3.1.2 Bollinger Bands . . . 71
5.3.1.3 Relative Strength Index (RSI) . . . 71
5.3.1.4 Price Percentage Change Features . . . 72
5.3.1.5 Moving Average Percentage Change Features . . . 73
5.3.1.6 RSI Percentage Change Features . . . 73
5.3.1.7 Bollinger Band Percentage Change Features . . . 73
5.3.1.8 Rolling Maximum and Minimum . . . 74
5.3.1.9 Close Price Shifts . . . 75
5.3.1.10 Trading time . . . 75
5.3.2 Long Short-Term Memory (LSTM) Network . . . 75
5.3.3 Graph Construction . . . 77
5.3.3.1 Scaling . . . 77
5.3.3.2 Graph for technical similarity . . . 77
5.3.3.3 Graph for fundamental similarity . . . 78
5.3.4 Graph Neural Networks (GNN) and Graph Attention Networks (GAT) . . . 78
5.3.4.1 Graph Attention Networks (GAT) . . . 79
5.3.5 Attention Mechanism for Multi-Channel Graph Networks . . . 80
5.3.5.1 Node-Level Prediction . . . 81
5.3.6 Proposed LSTM-GAT Attention Model . . . 82
5.3.6.1 Model Architecture . . . 82
5.4 Evaluation Metrics . . . 83
5.4.1 Mean Squared Error (MSE) . . . 84
5.4.2 Sharpe Ratio . . . 84

5.4.3 Mean Absolute Error (MAE) 84
5.4.4 Annual Return and Maximum Drawdown 85
5.5 Experimental Setup 85
5.5.1 Data source 85
5.5.2 Data Pre-processing 86
5.5.2.1 Scaling 86
5.5.2.2 Train test split 86
5.5.3 Graph preparing 87
5.5.3.1 Graph with technical analysis 87
5.5.3.2 Graph with fundamental analysis 87
5.5.4 Software and Hardware Setup 87
5.6 Experimental Results and Comparative Analysis 88
5.6.1 Results on prediction 89
5.6.2 Results on portfolio return 90
5.7 Discussion of results and ablation studies 91
5.7.1 Ablation Study Robustness 93
5.8 Conclusion and future work 93
5.8.1 Conclusion 93

Chapter 6 Integrated Multi-Agent Multimodal Trading Framework 96
6.1 Introduction 96
6.2 Motivation 97
6.3 Methodology 100
6.3.1 Technical Indicators 100
6.3.1.1 Simple Moving Average (SMA) 100
6.3.1.2 Exponential Moving Average (EMA) 100
6.3.1.3 Relative Strength Index (RSI) 100
6.3.1.4 Moving Average Convergence Divergence (MACD) 101
6.3.1.5 Bollinger Bands 101
6.3.1.6 Summary of Features 102
6.3.2 Pre-trained Financial Time Series Prediction Model 102

6.3.2.1 Data Preparation 102
6.3.2.2 Windowed Dataset Construction 103
6.3.2.3 Model Architecture 103
6.3.2.4 Training Procedure 104
6.3.2.5 Evaluation and Metrics 104
6.3.3 Multi-agent trading system 104
6.3.3.1 Market analysis agent 104
6.3.3.2 Technical pattern analysis agent 105
6.3.3.3 Financial news analysis agent 105
6.3.3.4 Trader agent 106
6.3.4 Proposed Framework 106
6.4 Experimental Setup 107
6.4.1 Data Collection 107
6.4.1.1 Technical Indicators and Output Prediction 108
6.4.1.2 Financial News 109
6.4.1.3 Technical Patterns 109
6.4.2 Backtesting Setup 110
6.5 Experimental Results 111
6.6 Discussion & Limitation 113
6.6.1 Discussion of results 113
6.6.2 Limitations 115
6.7 Conclusion 115

Chapter 7 Discussion, Conclusion and Future work 117

7.1 Discussion 117
7.1.1 From Representation to Financial Decision Intelligence 117
7.1.2 Stage 1: Representation Learning under Market Heterogeneity (Chapter 3) 117
7.1.3 Stage 2: Robust Temporal Modeling in Noisy and Data-Scarce Environments (Chapter 4) 118
7.1.4 Stage 3: Relational Modeling for Portfolio-Level Intelligence (Chapter 5) 119
7.1.5 Stage 4: Multimodal and Multi-Agent Decision Systems (Chapter 6) 119

7.2 Conclusion 120
7.3 Research Contributions 121
7.4 Future Work 123

Bibliography **125**

1 Appendix A 141

List of Figures

1.1 Comparison of traditional ECG and ETT datasets with financial time series data. While traditional time series data often display clear periodic patterns, financial data is characterized by subtle and irregular patterns. 4

3.1 Proposed model structure 33

3.2 The split interval of train, validation, and test set 37

3.3 Results for the 1-100 (seen) companies on the first 6 trading days of the testing set, dated from 2023-08-01. 39

3.4 Results for the 1-100 (seen) companies on the first 6 trading days of the testing set, dated from 2023-08-01. 40

3.5 Results for the 1-100 (seen) companies group by industry sectors of the testing set, dated from 2023-08-01. 41

3.6 Results for the 101-170 (unseen) companies on the first 6 trading days of the testing set, dated from 2023-08-01. 43

3.7 Results for the 101-170 companies group by industry sectors of the testing set, dated from 2023-08-01. 44

4.1 Structure of the proposed Model 60

5.1 Visualisation of proposed LSTM-GAT-AM model 83

5.2 Visualisation of the process of data pre-processing 86

5.3 The split interval of train, validation, and test set 87

5.4 Visualization of the adjacency matrix using the Fruchterman-Reingold algorithm. Each company is represented as a unique dot. Companies are connected with edges where the weight is inversely related to their DTW distance, with closer companies having heavier connections. The size of each dot is scaled based on the number of connections to enhance visual clarity. 88

5.5 Visualization of the adjacency matrix using the Fruchterman-Reingold algorithm. Each company is represented as a unique dot. Companies are connected within the same industry sector. The size of each dot is scaled based on the number of connections to enhance visual clarity. 89

5.6 Comparison of the final balance of the proposed model and baseline models on testing period. 91

6.1 Overall architecture of the proposed Neuro-Symbolic Multi-Agent Trading Framework, which integrates deep learning-based forecasting, LLM-driven reasoning, and portfolio back-testing through three specialized analyst agents: *Market*, *Pattern*, and *News*. 108

List of Publications

(1) Lu, X., Poon, J., & Khushi, M. (2024). *Bridging the gap between machine and human in stock prediction: Addressing heterogeneity in stock market*. IEEE Access.

(2) Lu, X., Poon, J., & Khushi, M. (2025). *Leveraging BiLSTM-GAT for enhanced stock market prediction: A dual-graph approach to portfolio optimization*. *Applied Intelligence*, 55(7), 601.

(3) Lu, X., Khushi, M., & Poon, J. (2025). *Patch-Based Sequential Modeling for Stock Index Forecasting via Mamba-CrossAttention Networks*. Submitted to *Knowledge-Based Systems*, under review.

(4) Lu, X., Khushi, M., & Poon, J. (2026). *Neuro-Symbolic Multi-Agent Multimodal AI for Financial Forecasting and Decision-Making*. Preparing for submission.

CHAPTER 1

Introduction

1.1 Overview

The rapid evolution of artificial intelligence (AI) and deep learning has profoundly transformed financial modeling, reshaping the way markets are analyzed, predicted, and interpreted. Despite this progress, achieving reliable, robust, and adaptive decision-making in financial markets remains a formidable challenge due to their inherent heterogeneity, volatility, and non-stationary dynamics. Traditional econometric and statistical approaches—such as autoregressive integrated moving average (ARIMA) and generalized autoregressive conditional heteroskedasticity (GARCH) models—are constrained by their linear assumptions and inability to capture complex temporal dependencies and inter-asset relationships. In contrast, deep learning models have demonstrated strong capacity to learn nonlinear patterns and high-dimensional representations, but they often suffer from limited transparency and robustness, particularly when applied to data-scarce or heterogeneous market environments.

This thesis situates itself at the intersection of deep learning, graph-based modeling, and large language model (LLM) to advance financial prediction toward adaptive decision-making. It addresses four fundamental challenges that constrain the performance of current deep learning approaches in financial contexts: (1) **market heterogeneity**, where different companies and sectors exhibit distinct behavioral and structural characteristics; (2) **data scarcity**, where daily-level or low-frequency datasets provide limited observations relative to market complexity; (3) **inter-stock dependency**, reflecting the intricate and dynamic correlations among financial assets that conventional independent modeling approaches often overlook; and (4) **multimodal data integration**, involving the effective fusion of diverse information sources such as

technical indicators, price patterns, and financial news to capture the multifaceted nature of market behavior. By addressing these challenges, this research aims to enhance both the predictive capability and decision-making intelligence of AI-driven financial systems operating in complex and uncertain environments.

The aim of this research is to advance financial artificial intelligence from predictive modeling toward intelligent decision-making through four progressive studies. The first two studies focus on enhancing predictive capability: the first introduces a contextualized representation model that embeds company-level textual descriptions to capture firm-specific heterogeneity and improve model generalization, while the second develops a state-space-inspired sequential framework for short-term financial forecasting that maintains robustness under data-scarce and noisy market conditions. Building upon these foundations, the third study transitions from prediction to decision-making by formulating a dual-graph learning model that integrates temporal and relational dependencies among stocks to optimise portfolio allocation and risk management. The fourth study further extends this trajectory by establishing a multimodal, multi-agent framework that fuses numerical, visual, and textual market information through large language model reasoning to support dynamic, explainable, and cooperative trading strategies. Together, these studies trace a systematic progression from predictive accuracy to adaptive financial decision-making, contributing to the development of robust, transparent, and autonomous systems capable of learning, reasoning, and acting effectively in complex and uncertain market environments.

1.2 Background and Motivation

Financial markets are inherently complex, exhibiting high volatility, structural non-stationarity, and dynamic interdependencies among assets. These characteristics make reliable forecasting and adaptive decision-making particularly challenging. Classical econometric approaches, such as Autoregressive Integrated Moving Average (ARIMA) and Generalized Autoregressive Conditional Heteroskedasticity (GARCH) models, offer interpretability and theoretical rigor but rely on assumptions of linearity and stationarity that are rarely met in real-world market

conditions. As a result, they often fail to capture the nonlinear patterns, regime shifts, and stochastic dynamics that characterize modern financial systems. The advent of deep learning has transformed financial forecasting by enabling models to learn complex temporal and cross-sectional dependencies from raw data. Architectures such as Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), and Temporal Convolutional Network (TCN) have achieved notable improvements over traditional methods by leveraging sequence modeling to identify latent temporal structures. More recently, Transformer-based architectures have further advanced the field through their global attention mechanisms, offering superior ability to capture long-range dependencies. Nevertheless, most of these models are designed for generic benchmark datasets with regular periodicity and abundant observations, such as the Electricity Transformer Temperature (ETT) or ECG datasets, and therefore struggle when applied to financial data, which is typically sparse, noisy, and non-periodic as illustrated in Figure 1.1.

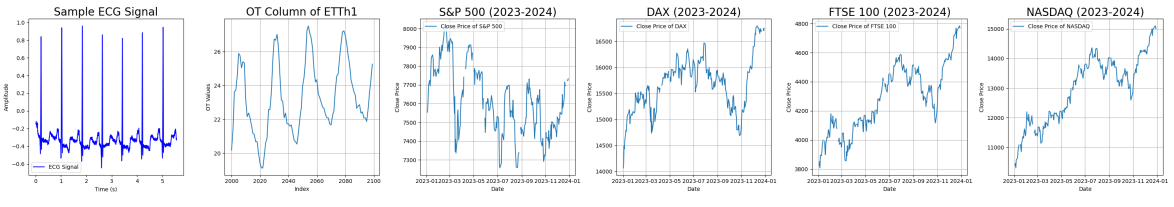

FIGURE 1.1: Comparison of traditional ECG and ETT datasets with financial time series data. While traditional time series data often display clear periodic patterns, financial data is characterized by subtle and irregular patterns.

Their limited robustness under these conditions continue to constrain their practical adoption in financial contexts. To address these limitations, this research has increasingly focused on architectures tailored to financial dynamics. State space models, exemplified by the Mamba framework, have emerged as a promising alternative for modeling temporal dependencies with linear computational complexity. By integrating structured recurrence and learnable dynamics, such models can efficiently capture long-range temporal patterns while maintaining stability under noisy and data-scarce conditions. Complementarily, graph-based neural networks, particularly Graph Attention Networks (GATs), have been employed to represent inter-stock relationships, enabling the learning of both temporal and relational dependencies crucial for portfolio-level reasoning and prediction. However, predictive accuracy alone does

not guarantee effective and profitable decision making. Modern financial environments demand systems that can integrate and reason across heterogeneous data modalities—numerical indicators, technical patterns, and unstructured textual information such as news and analyst reports. The recent emergence of large language models (LLMs) and neuro-symbolic reasoning has opened new avenues for achieving this goal. LLMs possess exceptional capabilities in abstraction and contextual reasoning, yet they struggle to handle structured numerical data directly. By combining deep learning–based forecasting with symbolic reasoning via LLMs, it becomes possible to bridge low-level quantitative modeling with high-level, interpretable decision-making processes. The motivation for this research arises from the need to unify these developments into a coherent framework for adaptive financial intelligence. Specifically, this thesis aims to (1) enhance representational learning under conditions of market heterogeneity and data scarcity, (2) capture temporal and relational dependencies among assets through hybrid state-space and graph-based architectures, and (3) extend predictive modeling toward interpretable, multimodal, and reasoning-driven decision systems. By integrating deep learning, graph reasoning, and large language model intelligence, this research seeks to advance financial AI from pattern recognition to adaptive, explainable, and autonomous decision-making in complex market environments.

1.3 Research Aim and Questions

This thesis aims to advance financial artificial intelligence from isolated predictive modelling toward adaptive, robust, and interpretable decision-making systems. To achieve this overarching goal, the research progresses through four interconnected stages, each addressing a structural limitation in existing financial modelling paradigms and giving rise to a corresponding research question.

First, the thesis aims to enhance model generalisation under market heterogeneity by incorporating contextualised representation learning. Financial assets differ fundamentally in business structure, sectoral exposure, and informational environments, yet conventional forecasting models often treat them as homogeneous entities. To overcome this limitation, this research

develops a framework that embeds firm-level textual and structural information into predictive architectures, enabling entity-aware modelling. This leads to the first research question: how can contextualised representation learning improve generalisation across heterogeneous financial entities?

Second, the thesis aims to improve robustness in data-scarce and noisy financial environments through state-space-inspired sequential modelling. Financial time series are characterised by volatility, regime shifts, and limited effective sample sizes, which challenge conventional attention-based architectures. By introducing structured state dynamics capable of capturing long-range temporal dependencies while maintaining stability, this research seeks to strengthen forecasting reliability under adverse conditions. This gives rise to the second research question: how can state-space-inspired sequential modelling enhance robustness and stability in volatile financial environments?

Third, the thesis aims to model inter-asset dependencies for portfolio-level intelligence by integrating temporal dynamics with relational graph structures. Asset prices evolve not only through individual temporal trajectories but also through interconnected market relationships such as sectoral co-movements and structural linkages. To address this, a dual-graph learning architecture is constructed to jointly capture temporal and relational dependencies, facilitating portfolio-aware prediction and risk-sensitive optimisation. This motivates the third research question: how can temporal and relational dependencies among assets be jointly modelled to improve portfolio-level optimisation?

Finally, the thesis aims to transition from prediction to interpretable and adaptive decision-making through multimodal fusion and large language model (LLM)-driven reasoning. Financial decision processes inherently integrate numerical indicators, visual price patterns, and textual sentiment signals. By developing a multimodal, multi-agent reasoning framework that unifies these information sources within an LLM-based system, this research seeks to support cooperative, transparent, and dynamically adaptive trading decisions. This leads to the fourth research question: how can multimodal fusion and LLM-driven reasoning facilitate interpretable and adaptive financial decision-making?

Collectively, these aims and questions form a coherent progression from representation learning, to robust sequential modelling, to relational portfolio intelligence, and ultimately to multimodal reasoning-driven decision systems.

1.4 Thesis Structure

This thesis is organised into six main chapters, each contributing to the overarching goal of advancing financial artificial intelligence from predictive modeling to adaptive, explainable, and decision-oriented systems. The structure of the thesis and its alignment with the corresponding studies are summarised as follows:

- **Chapter 1: Introduction**

  This chapter presents the overall background, motivation, and research context. It outlines the challenges of financial prediction—market heterogeneity, data scarcity, inter-stock dependency, and multimodal integration—and introduces the research questions, aims, objectives, and contributions of the thesis.
- **Chapter 2: Literature Review**

  This chapter provides a critical review of existing work in financial time series forecasting, covering traditional econometric models, machine learning approaches, and recent deep learning frameworks. It highlights the research gaps that motivate the design of this thesis, emphasizing the limitations of current methods in handling heterogeneity, volatility, and multimodal data integration.
- **Chapter 3: Modeling Market Heterogeneity Using Embedded Stock Descriptions**

  The first study (Lu et al. 2024) introduces a contextualized representation framework that integrates firm-level textual descriptions into financial prediction models. By embedding semantic features using BERT and performing dimensionality reduction with Restricted Boltzmann Machines (RBM), this approach captures company-specific characteristics, enhances generalization, and bridges the gap between machine and human interpretation in stock prediction.

- **Chapter 4: Cross-Attention Mamba Seq2Seq Model for Short-Term Prediction**
  The second study proposes a state-space-inspired sequential modeling framework based on the Mamba architecture. It introduces patch-based temporal segmentation and a Cross-Attention mechanism to model short-term dependencies under noisy and data-scarce financial conditions. The resulting model achieves state-of-the-art performance in multivariate-to-univariate stock index forecasting tasks and has been submitted for publication in *Knowledge-Based Systems*.
- **Chapter 5: Portfolio Management with GAT-Based Inter-Stock Dependency Modeling**
  The third study (Lu et al. 2025) develops a dual-graph learning model that combines BiLSTM-based temporal modeling with Graph Attention Networks (GAT) to capture both sequential and relational dependencies among stocks. This dual-graph framework enhances portfolio-level forecasting and optimisation, improving overall profitability and risk-adjusted performance.
- **Chapter 6: Integrated Multi-Agent Multimodal Trading Framework**
  The fourth study extends the research toward decision-making by introducing a multimodal, multi-agent architecture that integrates numerical, visual, and textual market data through large language model (LLM)-based reasoning. This framework enables cooperative and interpretable decision-making, marking a step toward autonomous financial agents capable of adaptive and explainable trading behavior. The study is currently under review for publication.
- **Chapter 7: Conclusion and Future Work**
  The final chapter summarizes the major findings and theoretical contributions of the thesis, discusses its implications for both academia and industry, and outlines promising directions for future research—particularly in hierarchical multimodal integration, explainability, and large-scale pretraining for financial decision intelligence.

CHAPTER 2

Literature review

2.1 Deep learning in financial time series forecasting

Deep learning has transformed the modeling of financial time series by providing a data-driven framework capable of capturing complex, nonlinear dependencies that traditional econometric methods fail to represent (Chen et al. 2023b; Kumar et al. 2021). Financial data are inherently noisy, non-stationary, and often exhibit regime shifts and abrupt volatility clustering, making them difficult to approximate with linear statistical models (Lim and Zohren 2021; Masini et al. 2023; Benidis et al. 2022). Deep learning architectures, however, can automatically extract latent temporal and cross-dimensional structures from raw sequential inputs, producing representations that are both expressive and adaptive to market dynamics (Chandra et al. 2021). Early applications of deep learning in finance focused on recurrent neural networks (RNNs) and their gated variants, particularly Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) networks (Rezaei et al. 2021; Wang et al. 2022a; Huang et al. 2022; Soni et al. 2022). These models introduced mechanisms for retaining long-term dependencies within sequential data, addressing vanishing-gradient issues that hindered conventional RNNs (Broby 2022). In financial forecasting, they demonstrated the ability to model extended temporal correlations between price movements, technical indicators, and trading volumes, yielding improved performance in trend identification and volatility tracking. The bidirectional extension of LSTM further allowed simultaneous learning from both past and future contexts within fixed windows, enhancing temporal coherence and smoothing short-term fluctuations (Chandra et al. 2021).

Despite their success, recurrent architectures remain limited by their sequential nature and difficulty in learning hierarchical representations of multi-scale market behavior. This has motivated the rise of convolutional and attention-based structures, which leverage parallel processing and global context modeling (Durairaj and Mohan 2022; Ahmed et al. 2022). Convolutional Neural Networks (CNNs) were adapted to time series by applying one-dimensional filters that capture local temporal patterns analogous to short-term market momentum (Tashiro et al. 2021). Combined CNN–LSTM hybrids emerged to exploit CNNs' capacity for local feature extraction together with LSTM's temporal memory, producing more stable predictions under high volatility (Dudukcu et al. 2023; Sirisha et al. 2022; Zhang et al. 2022).

More recently, attention mechanisms and transformer architectures have reshaped financial forecasting by overcoming the recurrence bottleneck. Attention allows models to weigh historical observations dynamically according to contextual relevance, enabling long-range dependency modeling with greater interpretability (Tao et al. 2024; Bala, Singh et al. 2022; Zeng et al. 2023). Transformer-based models use self-attention to process entire sequences in parallel and to identify critical time points that influence current market states (Xiang et al. 2022; Wang et al. 2022a). This global receptive field improves the identification of trend reversals, cross-asset dependencies, and structural breaks within financial data (Su et al. 2025). Extensions employing multi-head attention and positional encodings further refine the representation of temporal ordering and inter-feature correlation.

Beyond purely temporal modeling, deep learning in finance has increasingly incorporated graph-structured and multimodal data. Graph Neural Networks (GNNs) and their attention-based variant, Graph Attention Networks (GATs), represent companies as nodes linked by correlations derived from technical similarity, industry relations, or shared fundamentals (Cheng et al. 2022; Xiang et al. 2022; Wu et al. 2022). Such frameworks exploit inter-stock dependencies that traditional univariate or panel models neglect (Lazcano et al. 2023). When integrated with sequential encoders like BiLSTM, graph attention layers propagate contextual information across related assets, allowing the system to adjust forecasts based on collective market behavior (Hsu et al. 2021; Ye et al. 2022). This paradigm captures both temporal

continuity and spatial-relational structure, aligning prediction with the systemic nature of financial markets.

A parallel research direction emphasizes hybrid architectures combining sequence modeling with feature-learning modules such as autoencoders, Restricted Boltzmann Machines (RBMs), and state-space models (Bieganowski and Ślepaczuk 2025; Devrani et al. 2024; Triantafyllopoulos 2021; Jiang et al. 2025b). These designs aim to condense high-dimensional technical and fundamental indicators into compact latent representations, reducing noise and enhancing generalization in data-scarce environments. State-space formulations like Mamba networks have recently demonstrated efficiency in modeling long-term temporal dependencies through continuous-time dynamics, offering linear computational complexity while retaining expressiveness comparable to attention-based transformers (Wang et al. 2025; Mehrabian et al. 2025). When coupled with cross-attention or patch-based segmentation, such architectures achieve strong robustness against non-stationarity and sparse sampling—common challenges in daily-frequency financial data (Yang et al. 2021).

2.2 Portfolio Management and Decision Making in Deep Learning

The evolution of deep learning has redefined portfolio management by transforming it from a static optimization problem into a dynamic, data-driven process that learns and adapts to the ever-changing conditions of financial markets (Wang et al. 2021; Betancourt and Chen 2021). Traditional portfolio theory, grounded in mean–variance optimization, assumes linearity, normality, and stationarity of returns. However, financial markets exhibit nonlinear dependencies, structural breaks, and heavy-tailed distributions that challenge these assumptions (Wu et al. 2021; Aithal et al. 2023; Pantoja Robayo et al. 2025). Deep learning introduces a flexible framework capable of capturing such nonlinear and time-varying relationships, enabling portfolios to be managed through continuous learning and real-time decision refinement (Jeribi et al. 2024).

In the deep learning paradigm, portfolio management extends beyond predicting future prices to directly optimizing trading decisions (Jang and Seong 2023; Gunjan and Bhattacharyya 2023). Neural architectures such as Long Short-Term Memory (LSTM) networks, Temporal Convolutional Networks (TCNs), and Transformer-based models extract latent temporal dependencies from multivariate financial time series, generating features that represent expected trends, risk levels, and volatility regimes (Petrozziello et al. 2022; Soleymani and Paquet 2021; Du 2022). These learned representations serve as dynamic inputs to portfolio allocation layers or optimization modules that determine asset weights in accordance with risk–return trade-offs (Jahnavi et al. 2024; Chen et al. 2024). In contrast to static covariance-based optimization, deep learning models can incorporate nonlinear interactions among assets, adapt to changing correlations, and dynamically rebalance portfolios in response to evolving market signals (Ko and Lee 2025).

Reinforcement learning (RL) plays a pivotal role in bridging prediction and decision-making within deep portfolio management systems (Faheem et al. 2022; Ma and Nan 2024). In this framework, portfolio allocation is framed as a sequential decision process where an agent interacts with the market environment, observes state variables, and selects actions—such as reallocation or trading—based on learned policies (Wang et al. 2021; Sattar et al. 2025; Guan and Liu 2021). Reward functions are designed to maximize cumulative returns while penalizing risk, transaction costs, or drawdowns. Deep reinforcement learning (DRL) algorithms such as Deep Q-Networks (DQN), Policy Gradient, and Actor–Critic methods extend this concept by leveraging deep neural networks to approximate optimal policies over continuous and high-dimensional action spaces (Wu et al. 2021; Shi et al. 2021). Through repeated interactions, the agent learns to balance exploration and exploitation, developing adaptive strategies that evolve with market conditions (Betancourt and Chen 2021).

Beyond single-agent paradigms, multi-agent reinforcement learning introduces coordination and specialization among different agents performing complementary tasks within a unified portfolio framework (Chen et al. 2023a; Ma et al. 2023; Ram et al. 2024). For instance, one agent may focus on short-term trading based on high-frequency features, another on long-term trend stability, and a third on volatility hedging (Li and Hai 2024; Shavandi and Khedmati

2022). Their interactions—cooperative or competitive—generate emergent decision structures that more closely resemble the collective intelligence of real financial markets (Qiu et al. 2024; Zhang et al. 2024; Sarin et al. 2024). This decomposition enhances both the interpretability and robustness of the decision process, as agents specialize in distinct aspects of market understanding and contribute unique perspectives to portfolio-level optimization (Lussange et al. 2021; Cui et al. 2024).

Recent developments further integrate deep learning models with differentiable optimization layers, allowing portfolio weights to be computed as continuous, trainable parameters within end-to-end learning pipelines (Lim et al. 2022; Yu et al. 2024). These differentiable solvers embed financial objectives such as the Sharpe ratio, Sortino ratio, or downside risk minimization directly into the loss function (Ko and Lee 2024; Abe et al. 2024). Consequently, the model learns not only to predict but also to act optimally, aligning the forecasting stage with investment performance (Huang et al. 2024b). This represents a major conceptual shift—from using predictions as inputs to optimization, to optimizing decisions directly under a unified differentiable objective function (Yu et al. 2025b; Ding et al. 2024).

Another emerging trend is the inclusion of multimodal information sources in portfolio decision-making (Kong et al. 2024; Fatemi and Hu 2024). Deep learning enables the integration of heterogeneous modalities such as numerical market indicators, textual news sentiment, and visual technical patterns within a unified decision model. Cross-attention and fusion mechanisms align these modalities into a coherent latent representation, allowing portfolio strategies to account for both quantitative signals and qualitative narratives (Li et al. 2025b; Jiang et al. 2024; Yu et al. 2025b). This multimodal awareness enhances interpretability and robustness, particularly during macroeconomic or sentiment-driven events where traditional numerical models underperform (Huang et al. 2024b; Galarnyk et al. 2025).

Risk management also benefits significantly from deep learning integration (Joshi 2025b; Teixeira et al. 2023). Models employing probabilistic forecasting, heteroscedastic regression, or Bayesian neural networks estimate not only expected returns but also predictive uncertainty (Feng et al. 2021; Feng and Qu 2022; Han et al. 2025a; Shi et al. 2022). These uncertainty-aware signals allow portfolio managers to calibrate exposure dynamically based on model

confidence, leading to improved risk-adjusted returns. Hierarchical attention and uncertainty weighting mechanisms further enhance this capability by prioritizing decisions with stronger predictive confidence and limiting those in uncertain regions (Nicholls et al. 2021).

2.3 Multi-agent and multimodal trading system

The rapid expansion of deep learning in financial research has led to increasing interest in multi-agent and multimodal architectures for trading and decision-making (Fatemi and Hu 2024; Sun et al. 2025; Li et al. 2025b). These systems aim to emulate the distributed intelligence, specialization, and communication dynamics observed in real financial markets, where multiple agents—human or algorithmic—interact under varying objectives, information asymmetries, and risk preferences (Pandey et al. 2000; Luo et al. 2002). By integrating multiple agents and heterogeneous data modalities into a coordinated framework, modern trading systems can achieve a level of adaptability, robustness, and interpretability unattainable by monolithic predictive models (Shavandi and Khedmati 2022; Korczak et al. 2013).

A multi-agent trading system models the market as an ecosystem of autonomous entities that perceive environmental states, perform reasoning, and execute trading actions according to defined reward functions (Fang et al. 2023; Cheng et al. 2024). Each agent typically specializes in a specific analytical domain—such as short-term trend prediction, sentiment interpretation, or volatility estimation—and collaborates with other agents through structured communication channels or shared representations (Zhang et al. 2024; Huang et al. 2024a). This distributed design mirrors the division of labor in real trading desks, where technical analysts, fundamental researchers, and strategists contribute complementary perspectives. Coordination mechanisms, often implemented via reinforcement learning or hierarchical policy optimization, allow these agents to integrate their individual insights into a coherent portfolio-level decision (Lussange et al. 2023).

In a typical configuration, analyst agents focus on distinct market dimensions. A price-forecasting agent analyzes sequential numerical data to predict short-term returns or directional movement. A pattern-recognition agent interprets visualized technical indicators, such

as candlestick formations and moving-average crossovers, using convolutional or vision-transformer encoders. A sentiment agent processes unstructured text from financial news or social media, translating linguistic cues into quantitative sentiment scores. The integration of their outputs requires a higher-level trader agent, which synthesizes symbolic and numeric signals to determine buy, sell, or hold actions under portfolio constraints (Ahmed et al. 2021; Li et al. 2025a; Liu et al. 2025b). This hierarchy creates a feedback-rich architecture in which low-level models handle quantitative precision, while high-level reasoning modules ensure interpretability and decision consistency.

The multi-agent paradigm also enhances learning efficiency and system resilience. By distributing computational and functional responsibilities, the framework mitigates overfitting and allows parallel exploration of diverse strategies (Vahdati et al. 2025; Sun et al. 2025). Agents can be trained collaboratively through shared reward mechanisms or adversarially to reflect market competition. Cooperative learning encourages agents to share intermediate representations—such as predicted returns, volatility indicators, or confidence scores—facilitating ensemble-like synergy without explicit parameter sharing (Joshi 2025a; Jiang et al. 2025a). Competitive setups, in contrast, simulate real market interactions, where opposing agents (e.g., buyers and sellers) co-evolve policies to achieve equilibrium. This dynamic interplay produces more realistic behavioral patterns and improves robustness to regime shifts or unseen market conditions (Xu et al. 2024).

Complementing the multi-agent dimension, multimodal integration expands the information space available to trading systems. Financial markets are inherently multimodal environments: numerical time series describe price evolution; textual reports convey investor sentiment and policy implications; and visual patterns encapsulate structural cues used by human traders (Satapathy et al. 2025; Yu et al. 2025a; Liu et al. 2025a). Multimodal trading architectures unify these disparate sources by embedding each modality into a shared latent space. Transformer-based encoders, cross-attention mechanisms, or contrastive alignment networks allow models to align and fuse numerical, textual, and visual representations into coherent reasoning tokens (Thenuwara et al. 2022). This fusion process enriches the contextual

understanding of market states, enabling decisions that are both data-driven and semantically grounded.

The combination of multi-agent coordination and multimodal learning establishes a powerful paradigm for interpretable and adaptive trading intelligence (Peigné et al. 2025; He et al. 2024). Each agent contributes modality-specific expertise, while the fusion layer or large-language-model (LLM) component acts as a symbolic reasoning interface that interprets, summarizes, and reconciles agent reports (Föhring and Zelewski 2015; Li et al. 2025c). This neuro-symbolic structure bridges deep quantitative modeling with natural-language reasoning, allowing the system to justify its actions through interpretable explanations. By translating low-level numerical outputs into symbolic narratives—such as "momentum weakening under declining sentiment but within upward trend bounds"—the system moves beyond opaque prediction toward human-aligned, explainable decision-making (Yadla 2025).

Furthermore, multi-agent multimodal systems support closed-loop adaptation through feedback between forecasting and execution. After executing a trading decision, the system evaluates performance metrics such as realized profit, drawdown, and risk exposure. These outcomes are then propagated back to individual agents for policy refinement (Yang et al. 2025; Han et al. 2025b). This continual learning cycle enables agents to adjust both their local models and cooperative strategies in response to evolving market conditions, fostering long-term robustness and reduced sensitivity to overfitting.

2.4 Gap from current literature

Although the existing literature has extensively explored deep learning architectures for financial forecasting and portfolio optimization, several critical research gaps remain. First, a substantial proportion of prior studies rely predominantly on historical price series or derived technical indicators as the sole input modality, implicitly assuming that temporal dynamics alone are sufficient to characterize market behavior. This univariate or narrowly multivariate formulation overlooks the inherently multi-source and structurally interdependent nature of financial markets. Second, while transformer-based and state-space models have improved

long-range dependency modeling, many approaches treat forecasting and decision-making as loosely coupled stages rather than jointly optimized components within a unified architecture. Third, multi-agent and multimodal systems are often conceptual or evaluated under simplified settings, lacking rigorous integration of temporal representation learning, inter-asset relational modeling, and decision-level coordination in a coherent end-to-end framework. Moreover, most empirical studies emphasize predictive accuracy metrics such as MSE or directional accuracy, with limited discussion of practical deployment constraints and structural robustness under regime shifts. In response to these limitations, the subsequent chapters propose a structured framework that begins with historical price-based time series modeling as a controlled baseline, then progressively incorporates enhanced representation learning, structured interaction mechanisms, and decision-oriented optimization. This progression aims to bridge the gap between isolated forecasting models and integrated, adaptive trading intelligence systems capable of aligning prediction with actionable portfolio strategies.

CHAPTER 3

Modeling Market Heterogeneity Using Embedded Stock Descriptions

---

3.1 Introduction

Accurately predicting stock prices remains a formidable challenge in financial markets. Traditional predictive models often aggregate data from multiple companies, failing to account for the unique characteristics of each firm, which can hinder the model's ability to identify company-specific patterns. Moreover, existing research on stock price prediction frequently trains and tests models within the same group of companies, neglecting to assess their generalizability on 'Out-of-Sample' companies.

This study addresses these limitations by employing BERT to encode business descriptions into vectors, capturing the distinctive attributes of each company. We further enhance the predictive modeling framework by developing features that describe the percentage change of existing indicators, adding significant novelty to the existing research. Additionally, we apply a Restricted Boltzmann Machine (RBM) for dimensionality reduction after the BERT encoding process. In our approach, both the technical indicators and the vectorized descriptions are treated as distinct elements within the transformer encoder. By integrating these representations, our model is better equipped to differentiate between firms and recognize their individual patterns. The proposed model demonstrates superior performance over baseline models, particularly when tested on 'Out-of-Sample' companies, highlighting its ability to learn, understand, and analyze company-specific descriptions for more accurate predictions. This research offers novel insights into addressing the heterogeneity in stock price prediction.

3.2 Motivation

The accurate prediction of stock prices has long been a critical focus in financial markets due to its significant implications for investment strategies and economic forecasting (Hu et al. 2021). This area of research has seen substantial development, with numerous models and approaches being proposed over the years to enhance prediction accuracy (Jiang 2021). Traditional statistical methods, such as time series analysis and econometric models, have been widely used in this domain (Nabipour et al. 2020). However, the inherent complexity and volatility of financial markets often limit the predictive power of these conventional approaches.

In recent years, the advent of machine learning and deep learning techniques (Hu et al. 2024)has sparked renewed interest in stock price prediction (Shah et al. 2022). These advanced methodologies offer the ability to learn from large volumes of data, uncovering intricate patterns and relationships that are not easily detectable by traditional models (Nikou et al. 2019). Among these approaches, the application of Natural Language Processing (NLP) and transformer models has gained considerable attention (Kang et al. 2020). These models excel at understanding and processing textual data, enabling more nuanced analysis of market sentiment, news articles, and other relevant text-based inputs that influence stock prices (Li et al. 2021a).

Despite the progress made, much of the existing work in stock price prediction continues to aggregate data from multiple companies without differentiating between the unique characteristics of each firm (Soni et al. 2022). This approach, while increasing the dataset size and potentially improving general model performance, tends to overlook the distinct patterns that are specific to individual companies. Although some research has attempted to address this issue by grouping similar companies through clustering techniques (Marti et al. 2021), these methods still fall short of fully capturing the unique business models, market positions, and defining attributes of individual companies.

To address this gap, we propose a novel approach that enhances stock price prediction by explicitly distinguishing between companies using vectorized business descriptions. Our

method introduces a new way of integrating textual descriptions as static, descriptive vectors that encapsulate the specific characteristics of each firm, such as industry sector, geographic location, and market scale. These vectors are embedded into predictive modeling frameworks, enabling the model to differentiate between companies more effectively and learn from the unique patterns associated with each firm.

The core contributions of this study are twofold. First, we introduce a novel approach by incorporating textual business descriptions into the model, providing a richer and more detailed representation of each company. This allows the model to capture the inherent differences and similarities between firms, leading to more accurate predictions. Second, we rigorously test and compare the performance of the model on unseen companies with new descriptions, highlighting the model's ability to generalize and apply learned patterns to new, out-of-sample data. This underscores the potential of transfer learning in stock price prediction, demonstrating how the model can leverage learned knowledge to perform well on new companies with different characteristics.

3.3 Method

3.3.1 Technique indicators

This study leverages a set of basic and derived technical indicators to analyze stock price movements and inform trading decisions. The fundamental indicators used are the Open, High, Low, and Close (OHLC) prices, as well as tick volume and spread. These indicators form the basis for generating an additional 44 features, categorized as follows in Table 3.1.

3.3.1.1 Simple Moving Averages (SMA)

The Simple Moving Average (SMA) is a widely used technical indicator that smooths out price data by creating a constantly updated average price. It helps in identifying the direction of the trend over a specified period Johnston et al. 1999.

Indicators	Labels	Count
Simple Moving Averages	'mv100','mv50','mv9'	3
Bollinger Bands	'bb_bbm','bb_bbh','bb_bbl'	3
Relative Strength Index	'rsi15','rsi9','rsi50'	3
Percentage Change Features	'f1' to 'f10'	10
Moving Average Comparisons	'f11' to 'f16'	6
RSI Comparisons	'f17' to 'f18'	2
Bollinger Band Comparisons	'f19' to 'f22'	4
Rolling Maximum and Minimum	'f23' to 'f28'	6
Close Price Shifts	'f29' to 'f33'	5
Trading time	'h1','w1'	2
Total		44

TABLE 3.1: List of indicators, labels, and number of indicators used

The SMA is calculated by taking the arithmetic mean of a given set of prices over a specific number of periods.

$$\text{SMA}_n(t) = \frac{1}{n} \sum_{i=0}^{n-1} P(t-i) \tag{3.1}$$

where:

- $\text{SMA}_n(t)$ is the Simple Moving Average at time t over n periods.
- $P(t-i)$ is the close price at time $t-i$.
- n is the number of periods over which the average is calculated.

In this research the 'mv100', 'mv50', 'mv9' are moving averages over 100, 50, and 9 periods, respectively. SMAs help in smoothing out price data to identify trends over different time frames.

3.3.1.2 Bollinger Bands

Bollinger Bands consist of a set of lines plotted two standard deviations (positively and negatively) away from a simple moving average (SMA) of the price which provide a relative definition of high and low prices of a financial instrument Bollinger 2002.

Middle Band (MB): The middle band is the simple moving average (SMA) of the close price, typically over 20 periods.

$$\text{MB}(t) = \text{SMA}_{20}(t) = \frac{1}{20} \sum_{i=0}^{19} P(t-i) \tag{3.2}$$

Upper Band (UB): The upper band is calculated by adding two standard deviations to the middle band.

$$\text{UB}(t) = \text{MB}(t) + 2 \times \sigma_{20}(t) \tag{3.3}$$

where $\sigma_{20}(t)$ is the standard deviation of the close price over 20 periods.

Lower Band (LB): The lower band is calculated by subtracting two standard deviations from the middle band.

$$\text{LB}(t) = \text{MB}(t) - 2 \times \sigma_{20}(t) \tag{3.4}$$

The 'bb_bbm', 'bb_bbh', 'bb_bbl' represent the middle band (moving average), upper band, and lower band respectively.

3.3.1.3 Relative Strength Index (RSI)

The Relative Strength Index (RSI) Gumparthi et al. 2017is a momentum oscillator that measures the speed and change of close price movements. It is used to identify overbought or oversold conditions in a market. The RSI oscillates between 0 and 100 and is typically used with a 14-period setting.

(1) Calculate the average gains and losses over the specified period (e.g., 14 or 50 periods).
(2) Calculate the Relative Strength (RS):

$$\text{RS} = \frac{\text{Average Gain}}{\text{Average Loss}} \tag{3.5}$$

(3) Calculate the RSI:

$$\text{RSI} = 100 - \left(\frac{100}{1 + \text{RS}} \right) \tag{3.6}$$

'rsi14', 'rsi50' are RSI over 14 and 50 periods, respectively, measures the speed and change of close price movements to identify overbought or oversold conditions. 'rsimv9' is a 9-period moving average of the 14-period RSI.

3.3.1.4 Price Percentage Change Features

'f1' to 'f10' calculate the percentage change between different prices (open, close, high, low) and their shifts over different periods.

$$\text{f1} = \left(\frac{\text{Close} - \text{Open}}{\text{Open}}\right) \times 100 \tag{3.7}$$

$$\text{f2} = \left(\frac{\text{High} - \text{Low}}{\text{Low}}\right) \times 100 \tag{3.8}$$

$$\text{f3} = \left(\frac{\text{High}_{t-1} - \text{Low}_{t-1}}{\text{Low}_{t-1}}\right) \times 100 \tag{3.9}$$

$$\text{f4} = \left(\frac{\text{High}_{t-2} - \text{Low}_{t-2}}{\text{Low}_{t-2}}\right) \times 100 \tag{3.10}$$

$$\text{f5} = \left(\frac{\text{High}_{t-3} - \text{Low}_{t-3}}{\text{Low}_{t-3}}\right) \times 100 \tag{3.11}$$

$$\text{f6} = \left(\frac{\text{High}_{t-4} - \text{Low}_{t-4}}{\text{Low}_{t-4}}\right) \times 100 \tag{3.12}$$

$$\text{f7} = \left(\frac{\text{High} - \text{Open}}{\text{Open}}\right) \times 100 \tag{3.13}$$

$$\text{f8} = \left(\frac{\text{High} - \text{Close}}{\text{Close}}\right) \times 100 \tag{3.14}$$

$$\text{f9} = \left(\frac{\text{Open} - \text{Low}}{\text{Low}}\right) \times 100 \tag{3.15}$$

$$\text{f10} = \left(\frac{\text{Close} - \text{Low}}{\text{Low}}\right) \times 100 \tag{3.16}$$

3.3.1.5 Moving Average Percentage Change Features

'f11' to 'f13' calculate the percentage change between the closing price and the moving averages (50-period, 9-period, and 100-period, respectively). Features 'f14' to 'f16' compute

the percentage changes between different moving averages themselves.

$$\text{f11} = \left(\frac{\text{Close} - \text{MV}_{50}}{\text{MV}_{50}} \right) \times 100 \tag{3.17}$$

$$\text{f12} = \left(\frac{\text{Close} - \text{MV}_{9}}{\text{MV}_{9}} \right) \times 100 \tag{3.18}$$

$$\text{f13} = \left(\frac{\text{Close} - \text{MV}_{100}}{\text{MV}_{100}} \right) \times 100 \tag{3.19}$$

$$\text{f14} = \left(\frac{\text{MV}_{9} - \text{MV}_{50}}{\text{MV}_{50}} \right) \times 100 \tag{3.20}$$

$$\text{f15} = \left(\frac{\text{MV}_{9} - \text{MV}_{100}}{\text{MV}_{100}} \right) \times 100 \tag{3.21}$$

$$\text{f16} = \left(\frac{\text{MV}_{50} - \text{MV}_{100}}{\text{MV}_{100}} \right) \times 100 \tag{3.22}$$

3.3.1.6 RSI Percentage Change Features

f17, f18' calculate the percentage difference between different RSI values (rsi14, rsi50, rsimv9). f17 computes the percentage change between the 14-period RSI and the 50-period RSI, while f18 calculates the percentage change between the 50-period RSI and a 9-period simple moving average of the 14-period RSI.

$$\text{f17} = \left(\frac{\text{RSI}_{14} - \text{RSI}_{50}}{\text{RSI}_{50}} \right) \times 100 \tag{3.23}$$

$$\text{f18} = \left(\frac{\text{RSI}_{50} - \text{RSI}_{\text{mv9}}}{\text{RSI}_{\text{mv9}}} \right) \times 100 \tag{3.24}$$

3.3.1.7 Bollinger Band Percentage Change Features

f19' to 'f23' calculate the percentage difference between the close price and Bollinger Bands (bb_bbm, bb_bbh, bb_bbl), and between the bands themselves. Specifically, f19 computes the percentage change between the closing price and the middle Bollinger Band (20-period simple moving average), f20 calculates the percentage change between the closing price and the upper Bollinger Band, and f21 calculates the percentage change between the closing price and the lower Bollinger Band. Additionally, f22 computes the percentage change between the

lower and upper Bollinger Bands.

$$\text{f19} = \left(\frac{\text{Close} - \text{BB}_{\text{Middle}}}{\text{BB}_{\text{Middle}}} \right) \times 100 \tag{3.25}$$

$$\text{f20} = \left(\frac{\text{Close} - \text{BB}_{\text{Upper}}}{\text{BB}_{\text{Upper}}} \right) \times 100 \tag{3.26}$$

$$\text{f21} = \left(\frac{\text{Close} - \text{BB}_{\text{Lower}}}{\text{BB}_{\text{Lower}}} \right) \times 100 \tag{3.27}$$

$$\text{f22} = \left(\frac{\text{BB}_{\text{Lower}} - \text{BB}_{\text{Upper}}}{\text{BB}_{\text{Upper}}} \right) \times 100 \tag{3.28}$$

3.3.1.8 Rolling Maximum and Minimum

'f23' to 'f28' calculate the percentage difference between the close price and its rolling maximum or minimum over different periods (20, 50, 100). Specifically, 'f23' to 'f25' compute the percentage change between the rolling maximum closing prices over 20, 50, and 100 periods, respectively, and the current closing price. Conversely, 'f26' to 'f28' calculate the percentage change between the rolling minimum closing prices over the same periods and the current closing price.

$$\text{f23} = \left(\frac{\max(\text{Close}_{t-20:t}) - \text{Close}}{\text{Close}} \right) \times 100 \tag{3.29}$$

$$\text{f24} = \left(\frac{\max(\text{Close}_{t-50:t}) - \text{Close}}{\text{Close}} \right) \times 100 \tag{3.30}$$

$$\text{f25} = \left(\frac{\max(\text{Close}_{t-100:t}) - \text{Close}}{\text{Close}} \right) \times 100 \tag{3.31}$$

$$\text{f26} = \left(\frac{\min(\text{Close}_{t-20:t}) - \text{Close}}{\text{Close}} \right) \times 100 \tag{3.32}$$

$$\text{f27} = \left(\frac{\min(\text{Close}_{t-50:t}) - \text{Close}}{\text{Close}} \right) \times 100 \tag{3.33}$$

$$\text{f28} = \left(\frac{\min(\text{Close}_{t-100:t}) - \text{Close}}{\text{Close}} \right) \times 100 \tag{3.34}$$

3.3.1.9 Close Price Shifts

'f29' to 'f33' calculate the percentage change of the close price compared to its previous values over different periods (1 to 5).'f29' computes the percentage change from the closing price of the previous day to the current closing price. Features 'f30' to 'f33' extend this calculation to the closing prices from 2 to 5 days prior, respectively.

$$\text{f29} = \left(\frac{\text{Close}_{t-1} - \text{Close}}{\text{Close}} \right) \times 100 \tag{3.35}$$

$$\text{f30} = \left(\frac{\text{Close}_{t-2} - \text{Close}}{\text{Close}} \right) \times 100 \tag{3.36}$$

$$\text{f31} = \left(\frac{\text{Close}_{t-3} - \text{Close}}{\text{Close}} \right) \times 100 \tag{3.37}$$

$$\text{f32} = \left(\frac{\text{Close}_{t-4} - \text{Close}}{\text{Close}} \right) \times 100 \tag{3.38}$$

$$\text{f33} = \left(\frac{\text{Close}_{t-5} - \text{Close}}{\text{Close}} \right) \times 100 \tag{3.39}$$

3.3.1.10 Trading time

'h1' captures the hour of the day from the datetime values. The second line creates a new column wd that captures the day of the week (with Monday as 0 and Sunday as 6) from the datetime values, which could be useful for identifying patterns related to different weekdays.

$$\text{h1} = \text{Hour}(\text{datetime}) \tag{3.40}$$

$$\text{wd} = \text{Weekday}(\text{datetime}) \tag{3.41}$$

3.3.2 Bidirectional Encoder Representations from Transformers (BERT)

BERT is a pre-trained model that leverages the encoder component of the Transformer architecture, distinguishing itself from convolutional and recurrent neural networks. The core strength of BERT lies in the powerful Transformer encoder, which allows the model to be extended to considerable depths, thereby fully exploiting the properties of deep neural networks and enhancing model accuracy. The BERT model employs a multi-headed attention

mechanism, where the input vector $X_a \in \mathbb{R}^k$ undergoes multiple linear transformations to generate different linear values, which are then input into the attention block to compute attention weights. This is mathematically represented as:

$$\text{Att}(Q, K, V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{3.42}$$

where h denotes the number of heads in the attention mechanism. The final output of the multi-headed self-attention mechanism is derived by concatenating the outputs from each head and performing another linear transformation, as expressed by:

$$V_l = \text{Linear}\left(W_l \cdot \text{concat}(\text{Att}_1, \text{Att}_2, \ldots, \text{Att}_h) + b_l\right) \tag{3.43}$$

Here, Q, K, and V are the word-embedding representations of the input text, where $X_a = Q = K = V$. The similarity between words is calculated using dot product, followed by scaling by a factor $\frac{1}{\sqrt{d_k}}$ to prevent excessively large values that could adversely affect gradient backpropagation. This is then followed by the application of the softmax function to compute the attention weights, which are subsequently multiplied by V to yield the attentional output V_l.

After obtaining V_l through the multi-headed attention mechanism, a new vector $V_a = V_l + X_a$ is formed via a residual connection. This vector V_a is then normalized and passed through a feedforward network, and the final output of the Transformer is computed by applying another residual connection, as shown below:

$$V_t = \text{Feed}(W_f V_a + b_f) + V_a \tag{3.44}$$

In this equation, "Feed" denotes a linear function, and the final output of the Transformer is represented by V_t. The entire computation within the Transformer for any input vector X_a is encapsulated by the expression "Trans." The BERT model itself is constructed by stacking multiple Transformer layers, allowing it to model complex dependencies and capture deep contextual information within the text. This deep architecture enables BERT to excel in a wide range of natural language processing tasks.

3.3.3 Restricted Boltzmann machine (RBM)

Restricted Boltzmann Machines (RBMs) are stochastic neural networks that are particularly effective for unsupervised learning, allowing them to learn a probability distribution over a set of inputs. An RBM consists of two layers: a visible layer (representing the input data) and a hidden layer (capturing the underlying features). The connection between the visible and hidden layers is undirected, and there are no connections within a layer, making the architecture bipartite.

The energy function, which the RBM minimizes, is defined as:

$$E(v, h) = -\sum_i v_i a_i - \sum_j h_j b_j - \sum_{i,j} v_i W_{ij} h_j \tag{3.45}$$

where: v_i and h_j represent the binary states of the visible and hidden units, respectively, a_i and b_j are the biases associated with the visible and hidden units, W_{ij} represents the weight between visible unit i and hidden unit j.

The probability of a particular visible vector v is defined as:

$$P(v) = \frac{1}{Z} \sum_h e^{-E(v,h)} \tag{3.46}$$

where Z is the partition function, calculated as:

$$Z = \sum_{v,h} e^{-E(v,h)} \tag{3.47}$$

RBMs are typically trained using contrastive divergence, an efficient approximation to maximum likelihood learning. During training, the model updates the weights and biases to minimize the difference between the data distribution and the model distribution. The updates are computed as follows:

$$\Delta W_{ij} = \epsilon \left(\langle v_i h_j \rangle_{\text{data}} - \langle v_i h_j \rangle_{\text{model}} \right) \tag{3.48}$$

$$\Delta a_i = \epsilon \left(\langle v_i \rangle_{\text{data}} - \langle v_i \rangle_{\text{model}} \right) \tag{3.49}$$

$$\Delta b_j = \epsilon \left(\langle h_j \rangle_{\text{data}} - \langle h_j \rangle_{\text{model}} \right) \tag{3.50}$$

where ϵ is the learning rate, and $\langle \cdot \rangle_{\text{data}}$ and $\langle \cdot \rangle_{\text{model}}$ denote expectations under the data and model distributions, respectively.

Once trained, the RBM can be used to extract features by computing the hidden layer activations given the visible layer inputs. The hidden unit activations are calculated as:

$$P(h_j = 1 \mid v) = \text{sigmoid}\left(b_j + \sum_i v_i W_{ij}\right) \tag{3.51}$$

These learned features can then be used as inputs to downstream tasks, such as classification or regression models, significantly enhancing the model's ability to capture complex patterns in the data. RBMs have been successfully applied in various domains, including financial data analysis, where they have proven to be effective in feature extraction and improving prediction accuracy.

3.3.4 Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) networks are a type of recurrent neural network (RNN) that are particularly well-suited for modeling sequential data, such as time series or natural language. Unlike standard RNNs, which suffer from the vanishing gradient problem, LSTMs are designed to capture long-term dependencies by incorporating a memory cell that can maintain information across long sequences.

An LSTM cell consists of three gates: the input gate, the forget gate, and the output gate. These gates control the flow of information into and out of the memory cell. The equations governing the operation of an LSTM cell are as follows:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{3.52}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{3.53}$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{3.54}$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{3.55}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{3.56}$$

$$h_t = o_t * \tanh(C_t) \tag{3.57}$$

In these equations: - f_t is the forget gate, which determines what information from the previous cell state C_{t-1} should be forgotten. - i_t is the input gate, which decides what new information should be stored in the current cell state. - $\tilde{C}_t$ is the candidate cell state, which is generated based on the current input x_t and the previous hidden state h_{t-1}. - C_t is the updated cell state, which is a combination of the previous cell state and the new candidate cell state, modulated by the forget and input gates. - o_t is the output gate, which controls the output of the LSTM cell. - h_t is the hidden state, which is the output of the LSTM cell and also serves as the input to the next time step.

Here, σ represents the sigmoid function, and $\tanh$ represents the hyperbolic tangent function. The weight matrices W_f, W_i, W_C, W_o and the bias vectors b_f, b_i, b_C, b_o are learned during the training process.

LSTMs have been extensively used in various applications, including stock price prediction, where their ability to capture both short-term and long-term dependencies in financial time series data leads to more accurate predictions. For example, by maintaining a memory of past stock prices and other relevant financial indicators, LSTM networks can better forecast future trends compared to traditional models.

3.3.5 Transformer encoder

The Transformer encoder is a key component of the Transformer architecture, which has revolutionized natural language processing by allowing for efficient handling of sequential data. Unlike traditional models such as recurrent neural networks (RNNs), the Transformer encoder processes the entire sequence of data in parallel, enabling faster training and better capture of long-range dependencies.

The Transformer encoder consists of multiple layers, each composed of two main components: a multi-headed self-attention mechanism and a position-wise fully connected feedforward network. The self-attention mechanism allows the model to focus on different parts of

the input sequence when encoding each element, while the feedforward network further transforms these representations.

3.3.5.1 Multi-Headed Self-Attention

The self-attention mechanism computes a weighted sum of input vectors, where the weights are determined by the similarity between different elements of the sequence. The equations for self-attention are as follows:

$$\text{Attention}(Q, K, V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{3.58}$$

Here: Q (queries), K (keys), and V (values) are the input matrices obtained by linearly projecting the input sequence. d_k is the dimension of the key vectors. The dot product QK^T computes the similarity between the queries and keys. The Softmax function normalizes the result to obtain the attention weights.

In a multi-headed attention mechanism, this process is repeated multiple times (with different linear projections), allowing the model to focus on different parts of the sequence simultaneously:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O \tag{3.59}$$

where each head head_i is computed as:

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \tag{3.60}$$

W_i^Q, W_i^K, W_i^V, and W^O are learned weight matrices.

3.3.5.2 Position-Wise Feedforward Network

After the multi-headed self-attention mechanism, the output is passed through a fully connected feedforward network, which is applied identically to each position in the sequence:

$$\text{FFN}(x) = \text{ReLU}(xW_1 + b_1)W_2 + b_2 \tag{3.61}$$

Here, W_1 and W_2 are learned weight matrices, and b_1 and b_2 are bias vectors. The ReLU function introduces non-linearity, allowing the network to capture complex patterns.

3.3.5.3 Layer Normalization and Residual Connections

Each sub-layer (self-attention and feedforward network) is followed by layer normalization and a residual connection, which helps stabilize training and allows the model to learn more effectively:

$$\text{Output} = \text{LayerNorm}(x + \text{SubLayer}(x)) \tag{3.62}$$

The output of the Transformer encoder is a set of encoded vectors, one for each input element, which can be used for various downstream tasks such as classification, translation, or sequence modeling. The ability of the Transformer encoder to process sequences in parallel and capture long-range dependencies has made it highly effective in tasks such as stock price prediction, where it can model complex temporal relationships in financial data.

3.3.6 Proposed model

In this research, we propose a novel predictive model designed to forecast the next day's closing price of stocks, with a focus on differentiating between specific companies. The model architecture consists of two primary channels. The first channel processes the Open, High, Low, and Close (OHLC) data along with various generated technical indicators. These inputs are fed into a Bidirectional Long Short-Term Memory (BiLSTM) network, which is capable of capturing temporal dependencies and patterns within the time-series data. The second channel handles the vectorized long business descriptions of the companies, utilizing a Restricted Boltzmann Machine (RBM) for feature extraction.

The outputs from both channels are then analyzed using a Transformer model, which integrates the results to identify patterns specific to individual companies. This approach 3.3.6 allows the model to differentiate between companies and improve the accuracy of the stock price predictions.

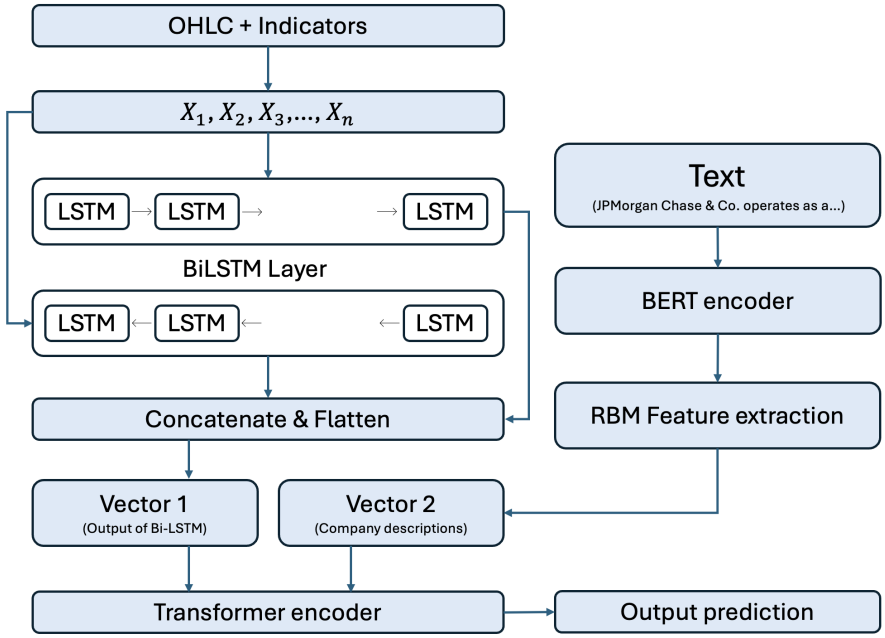

FIGURE 3.1: Proposed model structure

3.3.7 Evaluation metrics

In this study, we aim to predict the next day's closing price of selected companies, framing the problem as a regression task. We will employ various models to perform the prediction, and the effectiveness of these models will be evaluated using the two key metrics: Mean Squared Error (MSE) and Mean Absolute Error (MAE). The formulas for these metrics are as follows:

- **Mean Squared Error (MSE):**

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{3.63}$$

- **Mean Absolute Error (MAE):**

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{3.64}$$

Here, y_i represents the actual value, $\hat{y}_i$ represents the predicted value, and n is the total number of data points. These metrics will allow us to comprehensively assess the performance of the models and their suitability for predicting financial time series data.

3.4 Experimental Setup

3.4.1 Data source

The data utilized in this study were sourced from Yahoo Finance. For the first dataset, the companies selected were the 100 largest by market capitalization, spanning from Apple Inc. (AAPL) to European Metal Recycling Limited (EMR), as shown in Table 3.2. An additional dataset was compiled for companies ranked 101st to 170th by market capitalization, covering firms from O'Reilly Automotive (ORLY) to WEC Energy Group (WEC), as shown in Table 3.3. Both datasets consist of daily prices from January 1, 2020, to December 31, 2023.

In this study, we did not select companies that had undergone stock splits within the sample period to ensure the consistency and integrity of the dataset. Stock splits can introduce sudden, non-fundamental changes in stock prices, which could distort the true underlying patterns that the model aims to learn. By excluding companies with stock splits, we eliminate this potential source of noise, allowing the model to focus on capturing the genuine relationships between the company-specific characteristics and stock price movements. This approach helps to enhance the accuracy and reliability of the predictive model.

For the unseen companies, we selected companies ranked 101th to 170th in market share. After removing companies that underwent stock splits during the period, the remaining companies are as follows:

3.4.2 Data Pre-processing

3.4.2.1 Vectorised company description

The long business descriptions of the companies were retrieved from Yahoo Finance, specifically from the 'Long Business Description' section under the category of 'asset_profile'. These descriptions were then vectorized using the BERT (Bidirectional Encoder Representations from Transformers) model. Subsequently, an independent Restricted Boltzmann Machine

Industry	Companies	Count
Healthcare	UNH, LLY, JNJ, MRK, PFE, ABBV, AMGN, ABT, TMO, ISRG, SYK, GILD, BMY, ZTS, HCA, ELV, VRTX	17
Technology	AAPL, MSFT, AMZN, NVDA, GOOGL, GOOG, META, ORCL, ADBE, AMD, TXN, QCOM, CSCO, MU, CDNS, PLD	16
Consumer Staples	PG, KO, PEP, MDLZ, PM, CL, STZ, GIS, KMB, ADM	10
Finance	V, MA, JPM, MS, SPGI, AXP, BLK, SCHW, C	9
Consumer Discretionary	TSLA, HD, MCD, NKE, COST, TGT, GM, F, DG	9
Industrials	UPS, HON, RTX, CAT, LMT, UNP, DE, EMR, ITW	9
Energy	XOM, CVX, SLB, NEE, D, SO	6
Meterials	LIN, APD, ADM, WM	4
Utilities	NEE, D, SO	3
Total		83

TABLE 3.2: Dataset 1: Companies with No Stock Split Among the Top 100 Largest Companies in US

(RBM) was trained for feature extraction, reducing the dimensionality of each vector to a length of 100.

3.4.2.2 Scaling

To ensure that all features contribute equally to the prediction model and to enhance the performance of the regression algorithms, data scaling will be applied as part of the preprocessing step. Specifically, we will use the *StandardScaler()* to standardize the features and the stock prices by removing the mean and scaling to unit variance. The transformation can be expressed as follows:

$$\mathrm{z}_i = \frac{x_i - \mu}{\sigma} \tag{3.65}$$

where x_i is the original feature value, μ is the mean of the feature values, and σ is the standard deviation. This standardization process transforms the data to have a mean of 0 and a standard deviation of 1, ensuring that each feature is on the same scale. This step is crucial for models

Industry	Companies	Count
Finance	USB, TFC, AON, MET, STT, ICE, CME, PNC, EWBC, AFL, AIG	11
Healthcare	REGN, BDX, BSX, MRNA, DXCM, IDXX, MCK, CNC, BAX	9
Industrials	NSC, FDX, CTAS, ROP, TRV, TT, ROK	7
Consumer Discretionary	ORLY, APTV, SBUX, FDX, CHTR, LUV, DHIT	7
Energy	EOG, OXY, PSX, MPC, VLO, HAL, WMB	7
Materials	SHW, CTVA, DOW, PPG, LYB, GLW	6
Technology	ADI, PANW, IT, KLAC, HPQ, FTNT	6
Consumer staples	MO, KHC, WBA, KR, HSY	5
Real Estate	EQIX, PSA, SPG, WELL	4
Utilities	AEP, SRE, WEC	3
Insurance	PGR, ALL	2
Communication Services	WMG	1
Total		68

TABLE 3.3: Dataset 2: Companies with No Stock Split Among the 101-170 Largest Companies in US

that rely on the assumption that the input data is normally distributed or models sensitive to the scale of the input features.

3.4.2.3 Train test split

To prevent the model from learning from future data, which could lead to overfitting, the dataset is split into training, validation, and test sets based on chronological order. The training data spans from 2020-01-01 to 2023-04-01, the validation set covers the period from 2023-04-01 to 2023-08-01, and the test set includes data from 2023-08-01 to 2023-12-31 as in Figure 3.2. It is important to note that the scaling of features is performed using the statistics (mean and standard deviation) derived only from the training set to ensure that no future information is leaked into the model during the scaling process. This approach helps maintain the integrity of the model evaluation.

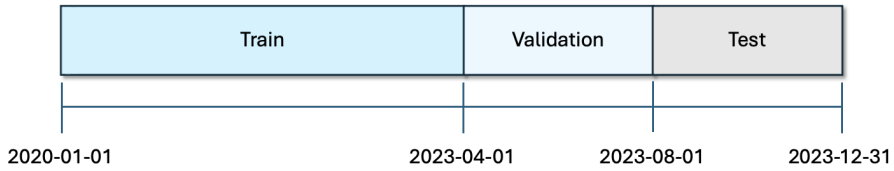

FIGURE 3.2: The split interval of train, validation, and test set

3.4.3 Software and Hardware Setup

The experiments in this study were conducted using the following software environment: PyTorch 2.4.0, TensorFlow 2.13.0, Keras 2.13.1, Pandas 2.0.3, and Numpy 1.24.3. The hardware configuration consisted of an Apple Silicon processor (ARM architecture) with 12 CPU cores (12 physical, 12 logical) and 32.0 GB of RAM, running on macOS (Darwin 23.6.0). GPU acceleration was leveraged using the MPS Backend (Metal Performance Shaders), which was enabled and available for PyTorch, with the MPS device specified as mps.

3.5 Experimental Results and Comparative Analysis

First, we demonstrate that our proposed model, which incorporates additional company-specific features such as detailed business descriptions, outperforms the baseline model that does not include these features. We will then compare the performance of our model against several recent models from the existing literature.

Next, we assess the generalization capability of our model by testing a pre-trained version (trained on data from Companies 1-100) on a new set of companies (Companies 101-170) using the same data preprocessing steps. We will also train the baseline model on this new dataset (Companies 101-170) for comparison. Our proposed model continues to outperform the baseline, even when evaluated on unseen business descriptions from new companies. This indicates that our model not only differentiates between companies based on their descriptions but also generalizes the knowledge gained from previous descriptions, effectively transferring this understanding to new, unseen companies.

To ensure the robustness of our results, we conducted the experiments 20 times independently and calculated the average of the evaluation metrics for each model.

3.5.1 Results on 1-100 companies and comparison with the SOTA models

In this section, we explore the advantages of the proposed model by comparing its performance against six other methods, including a baseline model trained exclusively on the generated features. The baseline model is trained and validated on the same dataset, and the comparison is conducted using evaluation metrics such as Mean Squared Error (MSE) and Mean Absolute Error (MAE).

The other four models represent state-of-the-art techniques in the domain of stock price prediction, including transformer and LSTM architectures. These models were chosen as baselines for several reasons. First, they represent the current state-of-the-art (SOTA) in predictive modeling, making them suitable benchmarks to evaluate the performance of the proposed model. Second, their underlying architectures share similarities with the proposed model, such as the use of deep learning components like LSTMs, GRUs, transformers, and hybrid structures. This allows for a fair comparison in terms of model structure and methodology. Additionally, these baseline models are all specifically designed for stock price prediction, making them highly relevant for assessing the effectiveness of the proposed approach in this specific application domain.

For instance, Li and Qian 2022 integrates the Empirical Mode Decomposition (EMD) algorithm with LSTM, GRU, and transformer units to enhance feature extraction and capture complex patterns in stock data. Similarly, Zhang et al. 2019 employs a CNN-LSTM hybrid model, leveraging the strengths of convolutional layers for spatial feature extraction and LSTMs for sequential data processing. Arévalo et al. 2016 utilizes a deep attention network, which aligns closely with transformer-based models in emphasizing important temporal patterns in stock sequences. Lastly, Wang 2023 implements a hybrid LSTM model that combines multiple techniques to improve prediction accuracy. By selecting these models, the comparison not only demonstrates the novelty and effectiveness of the proposed model but

also highlights its advantages when benchmarked against well-established and competitive approaches in the field detailed is shown in Table 3.4.

Models	MSE	MAE
Proposed Model	**0.0019**	**0.0272**
BiLSTM with indicators only	0.0036	0.0430
DeepLOB Zhang et al. 2019	0.0302	0.0945
DeepAtt Arévalo et al. 2016	0.0317	0.0932
FDG-Trans Li and Qian 2022	0.0291	0.0917
BiLSTM-MTRAN-TCN Wang 2023	0.0143	0.0874

TABLE 3.4: Comparison of results including baseline model and existing literature on 1-100 (seen) companies

Figure 3.3 illustrates the prediction results from our proposed model for the seen companies (1-100) on the first 6 trading days of the testing set, dated from 2023-08-01.

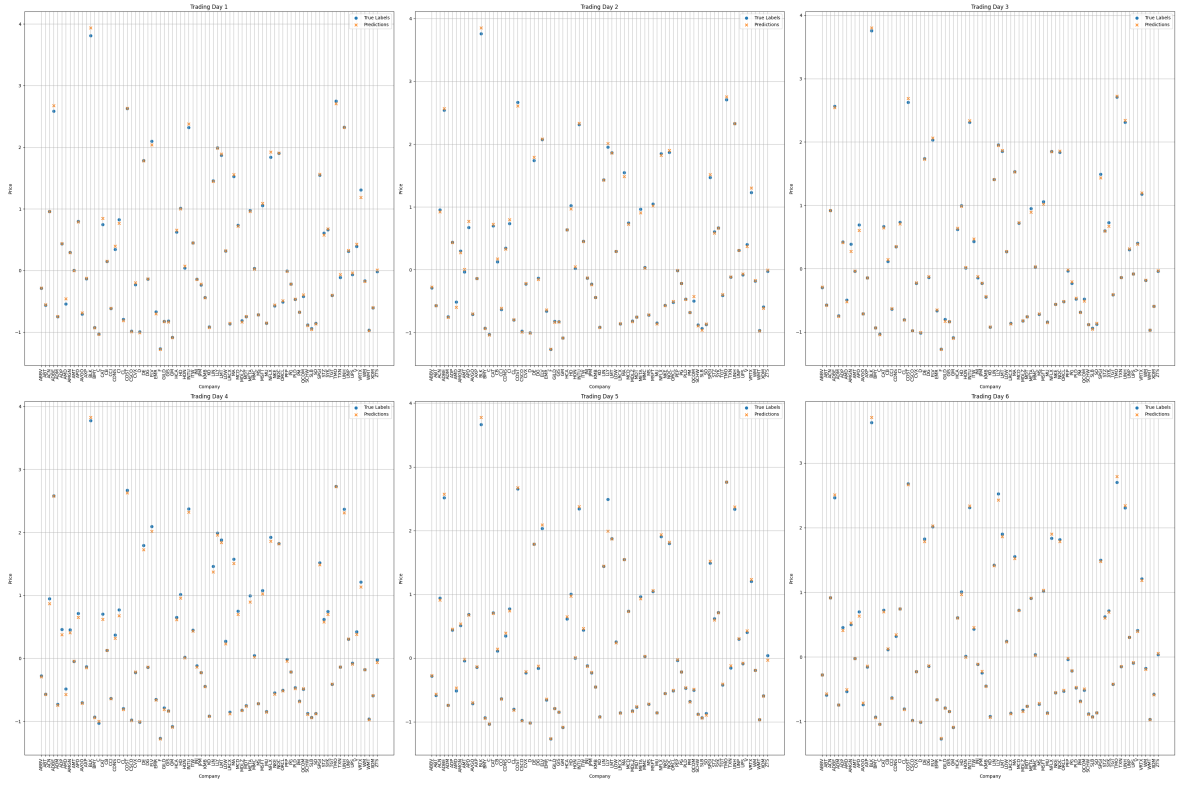

FIGURE 3.3: Results for the 1-100 (seen) companies on the first 6 trading days of the testing set, dated from 2023-08-01.

Figure 3.4 provided demonstrates the training process, showcasing the Mean Squared Error (MSE) and Mean Absolute Error (MAE) metrics for the training, and validation sets. The

close alignment of these error metrics across all three sets suggests that the model is not overfitting. The absence of significant divergence between the training and validation errors indicates that the model is generalizing well to unseen data, as the errors remain consistent across different datasets. Thus, this plot provides strong evidence that the model maintains a balanced performance and does not exhibit signs of overfitting.

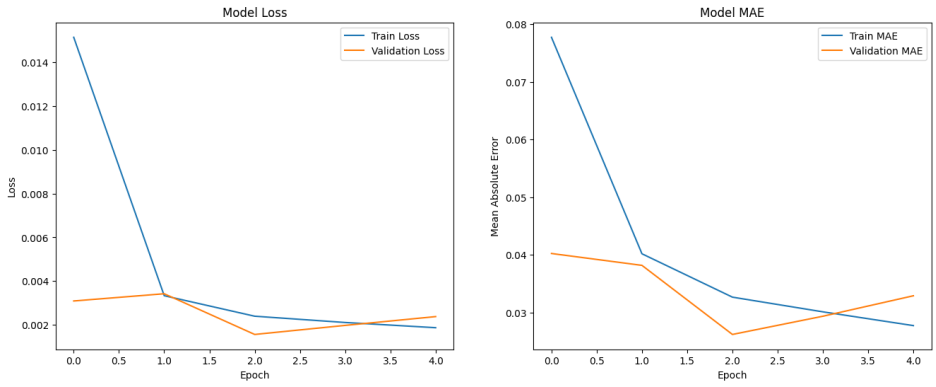

FIGURE 3.4: Results for the 1-100 (seen) companies on the first 6 trading days of the testing set, dated from 2023-08-01.

The MSE and MAE metrics for the training, validation, and test sets are presented in Table 3.5. With the implementation of early stopping during the training process, we believe that significant overfitting has been effectively mitigated.

Daraset	MSE	MAE
Training	0.0014	0.0233
Validation	0.0032	0.0229
Test	0.0019	0.0272

TABLE 3.5: Comparison of training, validation, and test set of proposed model on 1-100 (seen) companies

Based on the sector-specific analysis, as is shown in Figure 3.5, we observe varying performance across different sectors as measured by the MSE (Mean Square Error). The analysis reveals that certain sectors, such as Consumer Staples and Utilities, tend to have lower MSE values, indicating that the model performs better in these industries where market dynamics might be more stable or predictable. On the other hand, sectors such as Consumer Discretionary and Healthcare exhibit relatively higher RMSE, which could be attributed to the more

volatile nature of these sectors. This variability in performance suggests that while the model provides accurate predictions in some industries, its applicability may be less reliable in sectors characterized by rapid changes in external factors, such as geopolitical influences or fluctuating commodity prices. Hence, understanding these sectoral differences is crucial for refining the model and ensuring it is tailored to specific market environments.

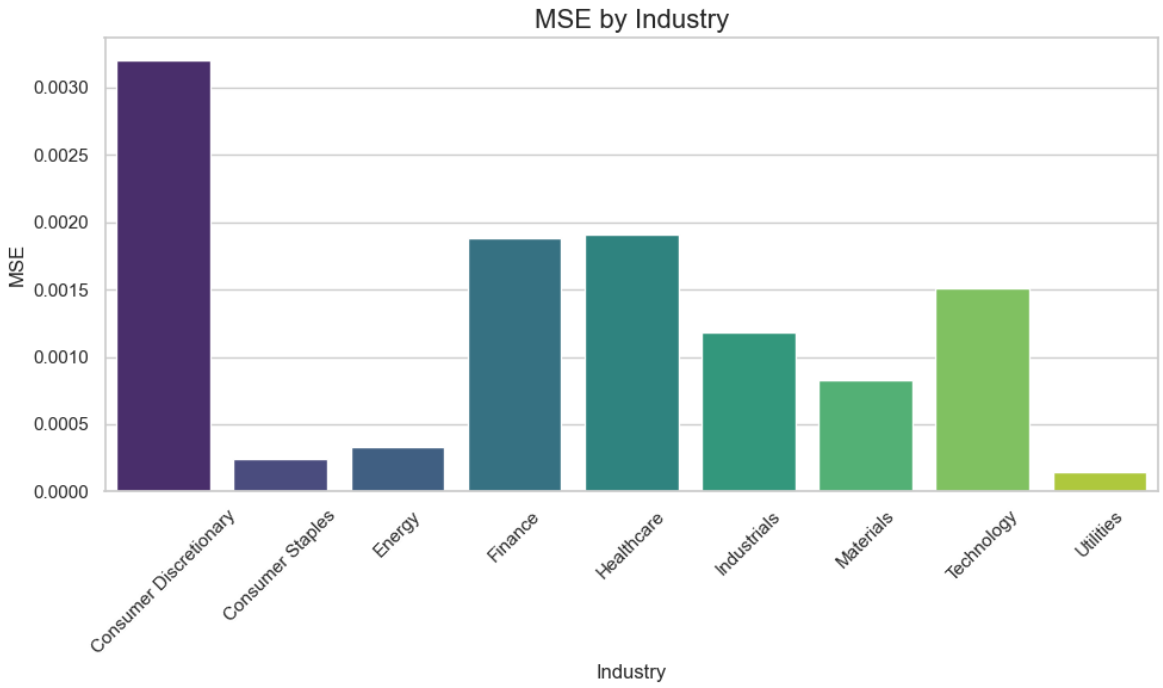

FIGURE 3.5: Results for the 1-100 (seen) companies group by industry sectors of the testing set, dated from 2023-08-01.

3.5.2 Results on 101-170 (Unseen) Companies

In this section, the proposed model was tested on a set of 70 additional companies (101-170) that were not seen during its training phase and compared with a baseline BiLSTM model, which was both trained and tested on the same set of companies (101-170). The results in Table 3.6 demonstrate that by incorporating vectorized features, our model can not only differentiate between different companies but also effectively learn from the provided descriptions and generalize this knowledge to unseen companies with new descriptions.

Models	MSE	MAE	Time
Proposed Model	**0.0014**	**0.0233**	**36s**
BiLSTM with no additional features	0.0028	0.0351	45s

TABLE 3.6: Comparison of results with baseline model on 101-170 (unseen) companies

The proposed model utilizes an additional channel for vectorized descriptions and introduces a transformer encoder to integrate the outputs of both channels into a fully connected layer. While this architecture increases the model's size, it enhances the learning efficiency, allowing the model to converge faster and requiring fewer epochs for complete training. In contrast, the baseline model, although taking less time per epoch, requires significantly more epochs to achieve full training. As a result, the overall training time of the proposed model is shorter compared to the baseline model.

Figure 3.6 illustrates the prediction results from proposed model for the unseen companies on the first trading 6 days of the testing set, dated from 2023-08-01.

On the sector-specific analysis as is shown in Figure 3.7, we observe varying performance across different sectors as measured by the MSE (Mean Squared Error). The analysis reveals that certain sectors, such as Communication Services and Utilities, tend to have lower MSE values, indicating that the model performs better in these industries where market dynamics are relatively stable and predictable. Conversely, sectors like Consumer Discretionary and Technology exhibit higher MSE, which could be attributed to the more volatile and fast-changing nature of these markets. This variability in performance suggests that while the model demonstrates strong predictive accuracy in some industries, it may be less reliable in sectors influenced by external factors such as rapid technological advancements or fluctuating consumer preferences. Therefore, understanding these sector-specific differences is essential for further refinement of the model and its application across varying market environments.

3.6 Discussion of results

In this section, we provide a detailed analysis of the results obtained from our proposed model, specifically focusing on its comparative performance with state-of-the-art (SOTA)

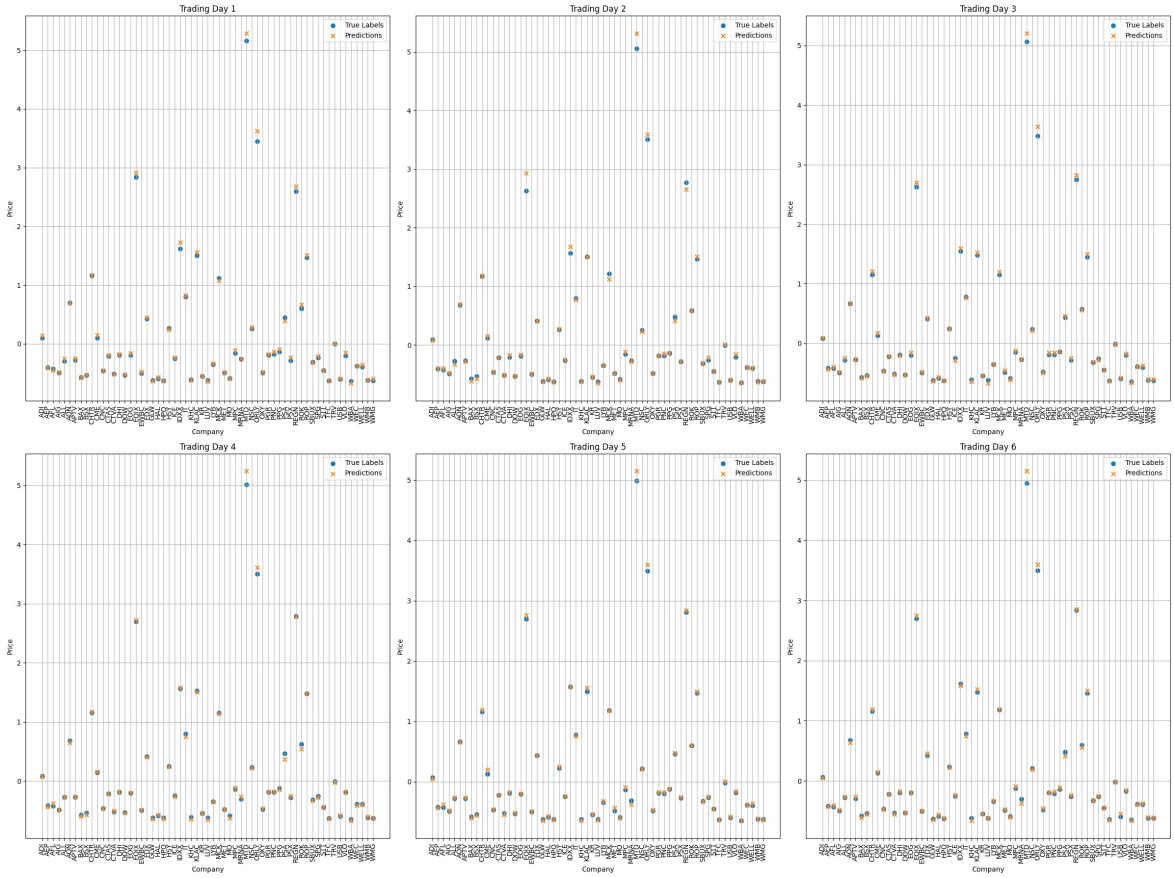

FIGURE 3.6: Results for the 101-170 (unseen) companies on the first 6 trading days of the testing set, dated from 2023-08-01.

stock prediction models and its efficacy in predicting stock prices for companies outside the training set.

3.6.1 Comparative Performance with SOTA Models

The results of our experiments demonstrate that our proposed model, which incorporates vectorized company descriptions as static features, significantly outperforms existing SOTA models in stock price prediction tasks. Traditional models typically aggregate indicators from multiple companies into a unified training set without distinguishing the unique characteristics inherent to each company. This approach, while it increases the dataset's size, inherently overlooks the heterogeneity among companies, which can lead to sub-optimal prediction accuracy. Each company operates under unique market conditions and possesses

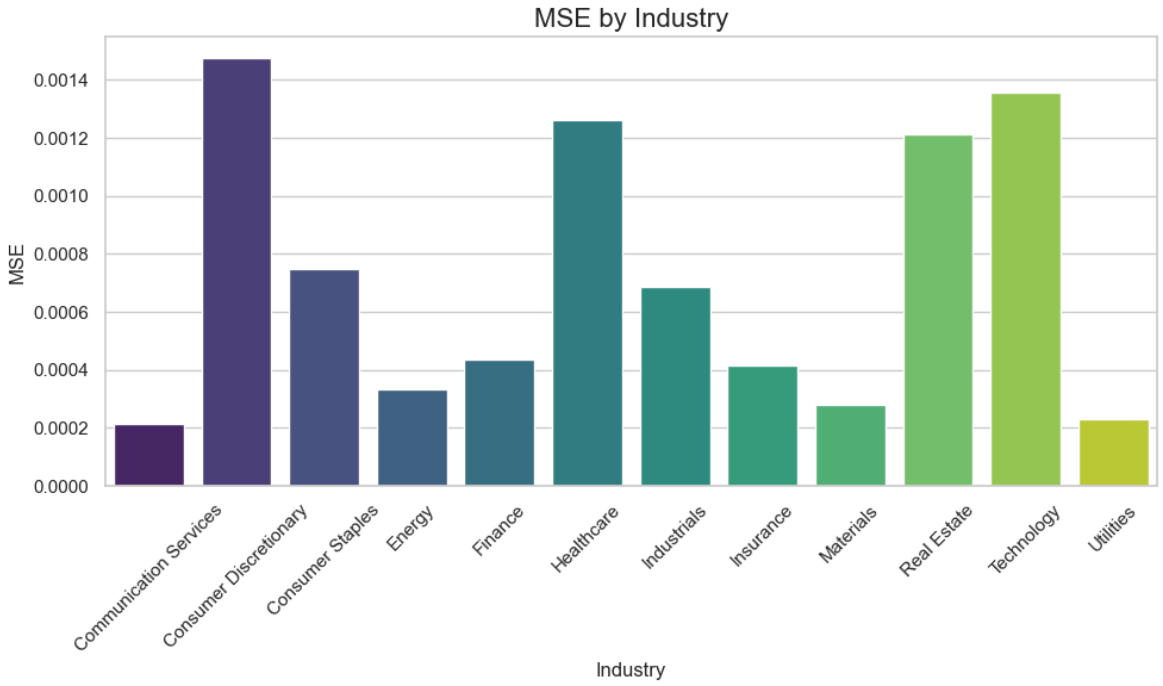

FIGURE 3.7: Results for the 101-170 companies group by industry sectors of the testing set, dated from 2023-08-01.

distinct financial and operational characteristics, which influence its stock price movements. Aggregating all companies into a single dataset assumes a homogeneity that does not exist, resulting in a model that may fail to capture the nuanced patterns that are specific to individual companies.

Our model addresses this critical limitation by integrating company-specific descriptions into the predictive modeling framework. These descriptions, encoded into vectors using advanced natural language processing techniques, allow the model to recognize and differentiate between companies, much like how an experienced human trader would consider both quantitative data (e.g., price history, trading volumes) and qualitative information (e.g., company fundamentals, market position) before making trading decisions. The incorporation of these descriptive vectors enables the model to identify distinct patterns within each company's data, leading to more accurate and reliable stock price predictions. This is particularly crucial in financial markets where the same market event can have varying effects on different companies, depending on their individual characteristics.

3.6.2 Generalization to Unseen Companies

To further validate the robustness and generalizability of our model, we conducted experiments on a dataset comprising companies ranked 101st to 170th by market capitalization, which were not included in the training set. This experiment aimed to assess the model's ability to apply learned pattern from seen companies to predict the stock prices of unseen companies based on their descriptive vectors and technical indicators.

The results were promising and indicated that the model successfully generalized the patterns and descriptions learned from the training companies to unseen companies. This was evidenced by the model's ability to analysis unseen companies based on their business descriptions together with technical indicators and predict their price movements with a higher degree of accuracy. The model achieved this by leveraging the underlying similarities between the vectorized descriptions of unseen companies and those of companies in the training set. For instance, companies within the same sector often share common market dynamics and risk factors, which influence their stock price movements in similar ways. Our model was able to capture these specific patterns and apply them effectively to unseen and similar companies, thereby supporting the hypothesis that companies with similar characteristics tend to exhibit similar price behaviors.

Moreover, the success of our model in predicting the stock prices of unseen companies highlights its potential for practical applications in real-world trading scenarios. The ability to generalize from a known set of companies to new companies without requiring retraining on the entire dataset presents a significant advantage in dynamic financial markets where new companies frequently emerge and existing companies undergo changes.

3.6.3 Time complexity analysis and experimental studies

In this section, we analyze the time complexity and discuss the experimental results comparing the proposed model with the baseline model in terms of training efficiency. The proposed model introduces an additional channel for vectorized descriptions and employs a transformer encoder to fuse the outputs from the two channels into a fully connected layer. While this

architectural enhancement increases the size and complexity of the model, it leads to a faster learning process.

From a time complexity perspective, the use of a transformer encoder, despite adding computational overhead, significantly improves the model's ability to capture complex patterns in the data. This results in faster convergence, allowing the proposed model to be fully trained in fewer epochs. The transformer encoder's self-attention mechanism efficiently captures relationships across input sequences, leading to more robust feature representations.

In contrast, the baseline model, while simpler and requiring less time per epoch, lacks the enhanced learning capabilities provided by the transformer. Consequently, it requires a greater number of epochs to achieve comparable performance. Although the baseline model benefits from lower per-epoch computational costs, the overall training time is extended due to the larger number of epochs required for convergence.

Our experimental studies corroborate this analysis. The proposed model, despite the higher per-epoch complexity, exhibited faster convergence, resulting in a shorter total training time compared to the baseline model. This demonstrates the trade-off between per-epoch time complexity and the total number of epochs required for training. Ultimately, the proposed model offers a more efficient training process, achieving superior performance with fewer epochs, thus reducing the overall computational cost in practice.

3.6.4 Implications for Financial Modeling

The implications of our findings are multifaceted. Firstly, the integration of company-specific descriptive vectors represents a significant advancement in stock price prediction models, particularly in addressing the issue of heterogeneity among companies. By moving away from a one-size-fits-all approach and towards a more nuanced model that accounts for individual company characteristics, our approach offers a more precise tool for financial forecasting. Secondly, the model's ability to generalize from known to unknown entities opens up new possibilities for the application of transfer learning in financial markets. This approach not

only enhances prediction accuracy but also reduces the need for frequent retraining, thus offering a more efficient and scalable solution for stock price prediction.

3.6.5 Addressing the Limitation of Comparative Analysis

A key limitation of this study is the comparative analysis between our proposed model and other state-of-the-art (SOTA) stock prediction models. Although our research highlights the benefits of integrating vectorized company descriptions into the predictive framework, the comparison with existing models was not as comprehensive as it could have been. This limitation might raise questions regarding the relative superiority of our model.

There are several factors contributing to this challenge. Firstly, each SOTA model in the field of stock prediction typically employs its own set of technical indicators and methodologies. These unique characteristics make it difficult to establish a direct and fair comparison across models. The diversity in the indicators used means that models are often optimized for specific tasks or datasets, which complicates attempts to compare their performance on a more general level.

Secondly, SOTA models are frequently trained and tested on different datasets, drawn from various markets, time periods, and companies. The lack of a standardized dataset for stock price prediction research means that models are often evaluated under different conditions. This variation is inherent to financial prediction, where market dynamics can differ significantly depending on the time period, geographic region, and sector of the companies involved. Consequently, it is challenging to compare models directly, as they may have been optimized for distinct market environments or temporal contexts.

In practice, researchers typically present their results using metrics such as Mean Squared Error (MSE) and Mean Absolute Error (MAE) to evaluate and compare the performance of their models. However, these metrics alone do not fully account for the variations in market conditions and datasets, making it difficult to draw definitive conclusions about one model's superiority over another.

Despite these challenges, our research has demonstrated the potential of our proposed model to outperform a selected set of baseline models, particularly in its ability to generalize to unseen companies. However, we acknowledge that a more extensive and standardized comparison with a broader range of SOTA models could have provided a stronger validation of our model's performance. Future research should aim to address this limitation by utilizing more standardized datasets or developing methodologies that allow for fairer comparisons across different models and market conditions.

While our study makes a valuable contribution to the field of stock price prediction, the inherent challenges of comparing models with different technical indicators and datasets must be recognized. Addressing these challenges in future work will be essential to provide a more robust and convincing case for the adoption of our model in financial prediction tasks.

3.7 Conclusion and future work

3.7.1 Conclusion

This research presents a significant advancement in the field of stock price prediction by introducing a model that integrates vectorized company descriptions as static features. Unlike traditional state-of-the-art (SOTA) models, which often rely on aggregated data from multiple companies, our approach acknowledges the unique characteristics and heterogeneity inherent in different firms. This recognition of company-specific attributes represents a substantial departure from existing methods, which tend to treat all companies as if they operate under similar conditions. By incorporating descriptive vectors into the predictive framework, our model is able to discern and leverage the distinct patterns that are specific to each company, leading to more accurate and reliable stock price predictions.

The novelty of our approach lies in its ability to simulate the decision-making process of human traders, who consider both quantitative data, such as historical prices, and qualitative information, such as a company's market position and business model. This holistic view allows the model to capture a broader range of factors that influence stock prices, offering a

more comprehensive and nuanced prediction tool. The success of our model in outperforming existing SOTA models highlights the importance of addressing the heterogeneity in stock markets, a factor that has been largely overlooked in previous research.

In summary, our research contributes a novel and effective method to the existing body of work on stock price prediction, addressing critical gaps in current SOTA models by incorporating company-specific features. This advancement not only improves prediction accuracy but also opens new avenues for integrating qualitative data into predictive models, paving the way for future research and practical applications in finance.

CHAPTER 4

Cross-Attention Mamba Seq2Seq Model for Short-Term Prediction

---

4.1 Introduction

Financial time series forecasting is central to decision-making in economics, investment, and risk management, yet it remains challenging due to pronounced non-stationarity, extreme volatility, and low signal-to-noise ratios. These difficulties are further amplified by the typical formulation of financial forecasting as a multivariate-to-univariate sequence-to-sequence (seq2seq) problem over short horizons, often with limited daily-level data.

In this paper, we introduce the Mamba-CrossAttention Network, a deep learning framework tailored to these characteristics. To clearly distinguish our contributions, we separate architectural novelty from system-level novelty. At the architectural level, our model is built from established components—such as Mamba blocks, residual connections, patch-based representations, and attention mechanisms—but adapts them to the specific demands of financial seq2seq forecasting. Multivariate input sequences are first processed through a stack of Mamba blocks with residual connections, enabling the capture of long-range temporal dependencies while maintaining stable gradient propagation. The resulting representations are then segmented using a patch-based module, which emphasizes localized temporal patterns and helps mitigate noise. On top of this, we introduce a Cross-Attention mechanism that facilitates information exchange across patch-level representations at different depths of the network.

At the system level, novelty arises from the integration and coordination of these components rather than from any single module in isolation. Instead of relying on a single attention output,

we aggregate information from all Cross-Attention layers via a learned weighted summation. This design allows the model to adaptively prioritize the most salient temporal patterns across layers when forming its final prediction. Such a system-level formulation is specifically motivated by the high variance and data scarcity typical of financial time series and goes beyond a straightforward recombination of existing architectures.

We evaluate the proposed framework on four publicly available stock index datasets spanning diverse markets and conditions. Across multiple prediction horizons, the Mamba-CrossAttention Network consistently achieves state-of-the-art performance, outperforming strong baselines, including Transformer, LSTM, and ablation variants, on both Mean Squared Error (MSE) and Mean Absolute Error (MAE). These results demonstrate that while the building blocks themselves are largely established, their targeted architectural adaptation and system-level integration together yield a robust and generalizable solution for challenging financial forecasting scenarios.

4.2 Motivation

Forecasting financial time series is a foundational task in economics and quantitative finance, underpinning critical decisions in algorithmic trading, asset allocation, and risk management(Lu et al. 2025; Lu et al. 2024). Accurate short-term predictions of market dynamics—such as future stock prices or returns—can yield significant strategic advantages(Chen et al. 2023b; Yang and Wang 2022). Yet, this task remains notoriously challenging due to the highly stochastic, non-stationary, and noise-dominated nature of financial data(Durairaj and Mohan 2022). These properties undermine many of the assumptions held by classical forecasting methods, rendering conventional time series models suboptimal in financial contexts(Widiputra et al. 2021).

A central difficulty lies in the structural divergence of financial forecasting from standard time series tasks. While benchmark datasets such as ETT or Weather (Zhou et al. 2021)often assume multivariate-to-multivariate (multi2multi) modeling(Hahn et al. 2023), financial applications are typically framed as multivariate-to-univariate (multi2one) problems, wherein

a single target variable—commonly the next-day closing price—is predicted from multivariate historical inputs (e.g., open, high, low, close, volume)(Mehtab and Sen 2022). The challenge is further exacerbated by data sparsity: financial datasets sampled at daily intervals offer significantly fewer data points than high-frequency counterparts, heightening the risk of overfitting and limiting model capacity(Lu and Xu 2024).

As illustrated in Figure 1.1, there is a distinct contrast between traditional time series and financial time series. Unlike traditional time series, which often exhibit clear periodic patterns, financial time series are characterized by their lack of periodicity, highlighting the complex and unpredictable nature of financial markets

Although deep learning has seen increasing adoption in financial forecasting, most recent studies rely on basic sequence-to-one architectures such as LSTMs, TCNs, or standard Transformers(Sirisha et al. 2022; Kontopoulou et al. 2023). These models tend to focus solely on next-day predictions, overlooking the opportunity for modeling more complex, multi-step trajectories(Fan et al. 2023; Wen and Li 2023). Furthermore, they often lack architectural components specifically adapted to the irregularities and constraints of financial time series data(Rezaei et al. 2021). As a result, the field faces two critical limitations: (1) the underutilization of architectures tailored to financial signal properties, and (2) the prevalent restriction to seq2one forecasting horizons.

To address these limitations, we present the Mamba-CrossAttention Network, a deep learning architecture expressly designed for short-term financial time series forecasting in data-constrained regimes. Our model introduces several key architectural innovations. First, multivariate input sequences are passed through a stack of Mamba blocks with residual connections, enabling robust extraction of long-range temporal dependencies while ensuring efficient gradient flow. The output features are then segmented into patches, allowing the model to isolate localized temporal patterns and reduce noise. These patches are processed by a newly designed Cross-Attention mechanism that aggregates temporal representations across multiple layers. By computing a weighted sum over all Cross-Attention layer outputs, the model dynamically highlights salient temporal cues to enhance its forecasting capacity.

Notably, the entire pipeline is designed to support sequence-to-sequence (seq2seq) forecasting, enabling richer, multi-step prediction trajectories beyond traditional next-day estimates.

We validate our approach on four publicly available stock index datasets, encompassing a broad range of market conditions and asset types. Our experiments demonstrate that the Mamba-CrossAttention Network consistently outperforms strong deep learning baselines—including Transformer and LSTM—across multiple forecast horizons, achieving state-of-the-art results in both Mean Squared Error (MSE) and Mean Absolute Error (MAE). These findings affirm the effectiveness and generalizability of our architecture for robust, short-term financial forecasting in low-data environments.

4.3 Methodology

4.3.1 Technical Indicators

In this study, the raw dataset comprises fundamental market variables, including "Open, High, Low, Close, and Volume (OHLCV)". To enhance the predictive power and interpretability of the data, we incorporate a set of technical indicators that serve as additional features. These indicators provide deeper insights into market dynamics, capturing trends, momentum, volatility, and mean reversion patterns. The integration of these derived features facilitates more robust modeling and improved decision-making in financial analytics.

4.3.1.1 Moving Averages (MA)

Moving averages are one of the simplest and most widely used technical indicators for trend analysis. A moving average smooths the fluctuations in the price data over a specified time period n, thereby helping to identify the underlying trend in a time series.

The mathematical formulation for the simple moving average (SMA) is:

$$\text{MA}_n(t) = \frac{1}{n} \sum_{i=0}^{n-1} P(t-i) \tag{4.1}$$

where:

- $\text{MA}_n(t)$ represents the moving average at time t,
- $P(t-i)$ is the closing price at time $t-i$,
- n is the window size (number of periods).

Moving averages can be categorized into different types based on their weighting mechanism. In this work, we use simple moving averages for $n = 10, 20,$ and 50 to analyze short-term, medium-term, and long-term trends, respectively.

4.3.1.2 Moving Average Convergence Divergence (MACD)

The Moving Average Convergence Divergence (MACD) is a momentum-based indicator that quantifies the relationship between two exponential moving averages (EMAs) of closing prices. It is widely used to identify trends, their strength, and potential reversals.

The MACD comprises three components:

(1) **MACD Line:** The difference between a short-term EMA (typically 12 periods) and a long-term EMA (typically 26 periods):

$$\text{MACD}(t) = \text{EMA}_{12}(t) - \text{EMA}_{26}(t) \tag{4.2}$$

where the Exponential Moving Average (EMA) is calculated as:

$$\text{EMA}_n(t) = \alpha \cdot P(t) + (1-\alpha) \cdot \text{EMA}_n(t-1) \tag{4.3}$$

Here, $\alpha = \frac{2}{n+1}$ is the smoothing factor.

(2) **Signal Line:** A 9-period EMA of the MACD Line:

$$\text{Signal}(t) = \text{EMA}_9(\text{MACD}(t)) \tag{4.4}$$

(3) **MACD Histogram:** The difference between the MACD Line and the Signal Line:

$$\text{Hist}(t) = \text{MACD}(t) - \text{Signal}(t) \tag{4.5}$$

These components allow for the identification of crossovers (between the MACD Line and Signal Line), divergences, and overbought/oversold conditions.

4.3.1.3 Bollinger Bands (BB)

Bollinger Bands are volatility-based bands that consist of three lines: the middle band, which is a simple moving average (SMA) of the closing price over a defined period, and two outer bands positioned at a distance of two standard deviations (σ) above and below the middle band.

The components are defined as follows:

- **Middle Band:** A simple moving average over n periods:

$$\text{MiddleBB}(t) = \frac{1}{n} \sum_{i=0}^{n-1} P(t-i) \tag{4.6}$$

- **Upper Band:** Positioned two standard deviations above the middle band:

$$\text{UpperBB}(t) = \text{MiddleBB}(t) + 2 \cdot \sigma(t) \tag{4.7}$$

- **Lower Band:** Positioned two standard deviations below the middle band:

$$\text{LowerBB}(t) = \text{MiddleBB}(t) - 2 \cdot \sigma(t) \tag{4.8}$$

where $\sigma(t)$ represents the standard deviation of the closing prices over the past n periods:

$$\sigma(t) = \sqrt{\frac{1}{n} \sum_{i=0}^{n-1} \left(P(t-i) - \text{MiddleBB}(t)\right)^2} \tag{4.9}$$

Bollinger Bands dynamically adjust to market conditions, expanding during periods of high volatility and contracting during periods of low volatility, making them effective tools for identifying potential breakouts and reversals.

4.3.1.4 Indicator Parameters

The specific parameters used for the technical indicators in this study are summarised in Table 4.1.

TABLE 4.1: Parameters of Technical Indicators

Indicator	**Parameters**
Moving Average	Window sizes: 10, 20, 50
MACD	EMAs: 12-period (short-term), 26-period (long-term), 9-period (Signal)
Bollinger Bands	SMA window size: 20, Multiplier: 2

These indicators collectively provide insights into market trends, momentum, and volatility, forming the basis for subsequent analysis.

4.3.2 Mamba block

To capture long-range temporal dependencies efficiently, we incorporate the *Mamba state space model* as the backbone for sequential feature extraction. Mamba belongs to the family of structured state space models (SSMs), which model latent dynamics via continuous-time recurrence and offer linear-time computation, making them highly suitable for time series tasks.

Let $\mathbf{X} \in \mathbb{R}^{T \times D}$ denote the input sequence of length T and feature dimension D. The continuous-time formulation of the Mamba state space model is given by:

$$\frac{d\mathbf{h}(t)}{dt} = \mathbf{A}\mathbf{h}(t) + \mathbf{B}\mathbf{u}(t), \quad \mathbf{y}(t) = \mathbf{C}\mathbf{h}(t) \tag{4.10}$$

where $\mathbf{u}(t)$ is the input signal, $\mathbf{h}(t)$ is the hidden state, and $\mathbf{y}(t)$ is the output. The discretized version used in practice is:

$$\mathbf{h}_{t+1} = \mathbf{A}\mathbf{h}_t + \mathbf{B}\mathbf{u}_t, \quad \mathbf{y}_t = \mathbf{C}\mathbf{h}_t \tag{4.11}$$

These dynamics are implemented via convolutional operations and parameterized kernels, allowing for efficient unrolling across time steps.

In our architecture, each Mamba block acts as a feature extractor, transforming the input sequence as:

$$\mathbf{X}_{\text{mamba}} = \text{Mamba}(\mathbf{X}) \tag{4.12}$$

To stabilize training and enable deeper representations, we adopt a residual connection around each Mamba block:

$$\mathbf{X}^{(l+1)} = \text{Mamba}(\mathbf{X}^{(l)}) + \mathbf{X}^{(l)} \tag{4.13}$$

where l denotes the block index. This residual formulation promotes gradient flow and allows effective stacking of multiple Mamba layers without degradation. Empirically, the Mamba blocks serve as a powerful temporal encoder, enabling the model to capture both short- and long-range dependencies in financial time series with high fidelity.

4.3.3 Cross-attention block

To aggregate temporal and feature-wise patterns across the extracted representations, we employ a multi-layer Cross-Attention mechanism designed to operate over both temporal and channel dimensions. Inspired by the structure of axial attention, each Cross-Attention block consists of two sequential operations: **row-wise attention** and **column-wise attention**, each followed by residual normalization.

Let $\mathbf{X} \in \mathbb{R}^{B \times R \times D \times E}$ denote the intermediate representation output by the patching module, where B is the batch size, R is the number of patches, D is the channel (feature) dimension, and E is the embedding dimension. For each Cross-Attention block, we first apply row-wise self-attention across the patch dimension:

$$\mathbf{X}_{\text{row}} = \text{RowAttn}(\mathbf{X}) = \text{MultiHead}(\mathbf{X}_{\text{row}}, \mathbf{X}_{\text{row}}, \mathbf{X}_{\text{row}}) \tag{4.14}$$

Here, $\mathbf{X}_{\text{row}}$ is reshaped to flatten the patch dimension across all channels: $\mathbf{X}_{\text{row}} \in \mathbb{R}^{(B\cdot D)\times R\times E}$. The result is then restructured and passed to the column-wise attention:

$$\mathbf{X}_{\text{col}} = \text{ColAttn}(\mathbf{X}_{\text{row}}) = \text{MultiHead}(\mathbf{X}_{\text{col}}, \mathbf{X}_{\text{col}}, \mathbf{X}_{\text{col}}) \tag{4.15}$$

with $\mathbf{X}_{\text{col}} \in \mathbb{R}^{(B\cdot R)\times D\times E}$. Residual connections and layer normalization are applied after each attention operation to promote stability and enable deep stacking:

$$\mathbf{X}_{\text{row}} \leftarrow \text{LayerNorm}(\mathbf{X}_{\text{row}} + \text{RowAttn}(\mathbf{X}_{\text{row}})) \tag{4.16}$$

$$\mathbf{X}_{\text{col}} \leftarrow \text{LayerNorm}(\mathbf{X}_{\text{col}} + \text{ColAttn}(\mathbf{X}_{\text{col}})) \tag{4.17}$$

The final output of each block is added to the input embedding $\mathbf{X}$ to produce the updated embedding representation. We stack multiple such Cross-Attention blocks and aggregate their outputs via a learnable weighted sum. Let L denote the number of cross-attention layers, and let $\mathbf{X}^{(l)}$ be the output of layer l. Then, the final encoder representation is computed as:

$$\mathbf{X}_{\text{enc}} = \sum_{l=1}^{L} \alpha_l \cdot \mathbf{X}^{(l)}, \quad \text{where} \quad \alpha_l = \frac{\exp(w_l)}{\sum_{j=1}^{L} \exp(w_j)} \tag{4.18}$$

Here, α_l denotes the normalized weight for each layer, learned through a trainable vector $\mathbf{w} \in \mathbb{R}^L$. This weighted summation enables the model to dynamically attend to informative layers during training, facilitating the extraction of salient temporal and cross-channel features.

4.3.4 Proposed model

We introduce the Mamba-CrossAttention, a unified architecture tailored for multivariate-to-univariate financial time series forecasting under short horizons and noisy, data-scarce conditions. The model begins by applying a stack of Mamba blocks with residual connections to the input sequence, enabling efficient and expressive extraction of long-range temporal features through structured state space modeling. The resulting representations are segmented into non-overlapping temporal patches, which are linearly projected and enriched with positional encodings to retain sequence order. These patch embeddings are processed through a hierarchy of Cross-Attention blocks, each composed of sequential row-wise and column-wise multi-head attention layers, designed to model intra-patch and inter-variable interactions respectively. The outputs from all Cross-Attention layers are aggregated using a learnable, softmax-normalized weighted sum, allowing the network to emphasize temporally salient features across layers. A global token is prepended to the flattened token sequence, which is then passed through a multilayer perceptron (MLP) for final sequence-to-sequence prediction of future target values. This end-to-end design bridges efficient temporal encoding, localized denoising, and hierarchical attention-based aggregation, offering a robust solution for short-term financial prediction tasks.

4.4 Experimental Setup

4.4.1 Data source and processing

We collect daily stock index data from `Yahoo Finance`, covering the period from January 1, 2013 to December 31, 2023. For each index, we use five standard financial indicators: `Open`, `High`, `Low`, `Close`, and `Volume`, forming a five-channel multivariate input. All data are sampled at a daily frequency, and no forward-looking leakage is introduced. To normalize the input scale and improve convergence, we apply *Min-Max normalization* to each feature independently using statistics computed on the training set, scaling all values to the $[0, 1]$ range. The validation and test sets are normalized using the same transformation.

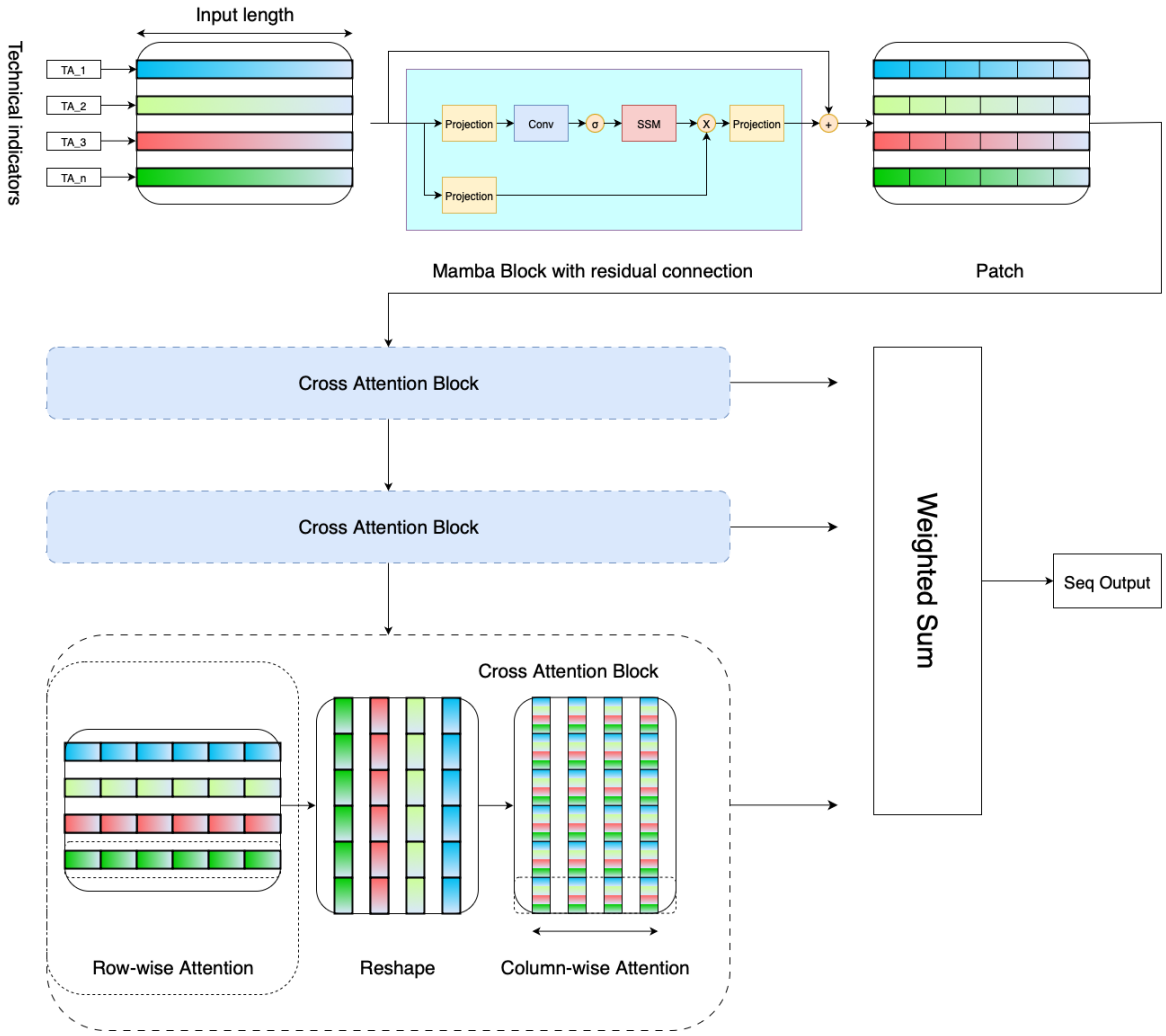

FIGURE 4.1: Structure of the proposed Model

Because financial data is inherently sequential, we maintain strict temporal order when splitting the dataset: data from `2013-01-01` to `2022-01-01` is used for training, from `2022-01-01` to `2023-01-01` for validation, and from `2023-01-01` to `2023-12-31` for testing. We adopt a sliding window strategy to generate training samples for sequence-to-sequence forecasting, using multiple configurations with input lengths $T = \{15, 30, 50\}$ and corresponding prediction lengths $L = \{3, 5, 10\}$ to evaluate the model's performance under varying temporal contexts.

4.4.2 Hardware & Software

All experiments were conducted using a standard Google Colab environment equipped with an NVIDIA Tesla T4 GPU (16GB VRAM) and 12GB system RAM. The software environment was based on Python 3.10 and PyTorch 2.4.0, with CUDA 12.1 support. The primary libraries and dependencies used include `torch==2.4.0`, `torchvision==0.19.0`, `torchaudio==2.4.0`, `causal-conv1d==1.4.0`, and `mamba-ssm==2.2`, all installed via pip using the official PyTorch and Python package indices. Code execution and model training were performed using Jupyter notebooks within the Colab environment, and all results were obtained under consistent hardware and software settings to ensure reproducibility.

4.5 Experimental Results

In this research, we evaluate the proposed Mamba-CrossAttention Network on four publicly available stock index datasets to assess its effectiveness in short-term financial time series forecasting. We conduct a comprehensive empirical study comparing our model against a suite of strong baseline models, including Transformer and LSTM architectures. The evaluation focuses on both **predictive accuracy** and **model robustness**, using two standard error metrics: Mean Squared Error (MSE) and Mean Absolute Error (MAE). Additionally, we perform detailed ablation studies to examine the contributions of each architectural component—namely, the Mamba feature extractor, patch segmentation, cross-attention mechanism, and layer-weighted aggregation. These analyses are conducted across multiple input-output configurations to validate generalizability under varying forecast horizons. All experiments are conducted under consistent settings to ensure fair comparison and reproducibility.

4.6 Discussion

The results of empirical evaluation underscore the effectiveness and robustness of the Mamba-CrossAttention architecture for short-term financial time series forecasting. Across all four

TABLE 4.2: Model comparison and ablation study on four stock indices using MSE and MAE. Input/prediction lengths shown as In/Out.

Dataset	**In/Out**	**Model Comparison**						**Ablation Study**	
		Proposed Model		LSTM		Transformer		Cross-Atten Only	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
DAX	15/3	**0.0043**	**0.0507**	0.0098	0.0726	0.0056	0.0551	0.0052	0.0535
	30/5	**0.0052**	**0.0539**	0.0094	0.0702	0.0065	0.0587	0.0066	0.0610
	50/10	**0.0101**	**0.0746**	0.0170	0.0945	0.0112	0.0770	0.0122	0.0813
S&P500	15/3	**0.0010**	**0.0244**	0.0021	0.0320	0.0015	0.0259	0.0014	0.0271
	30/5	**0.0015**	**0.0290**	0.0033	0.0388	0.0026	0.0364	0.0018	0.0321
	50/10	**0.0023**	**0.0366**	0.0033	0.0391	0.0025	0.0343	0.0029	0.0409
Nasdaq	15/3	**0.0008**	**0.0212**	0.0016	0.0272	0.0011	0.0245	0.0013	0.0255
	30/5	**0.0013**	**0.0272**	0.0023	0.0324	0.0016	0.0293	0.0018	0.0325
	50/10	**0.0028**	**0.0376**	0.0043	0.0448	0.0036	0.0428	0.0031	0.0389
FTSE	15/3	**0.0123**	**0.0844**	0.0218	0.1092	0.0161	0.0948	0.0155	0.0944
	30/5	**0.0157**	**0.0931**	0.0302	0.1276	0.0162	0.0923	0.0202	0.1064
	50/10	**0.0541**	**0.1622**	0.0655	0.1797	0.0603	0.1705	0.0587	0.1680

stock indices—DAX, S&P500, Nasdaq, and FTSE—the model consistently outperforms established baselines such as LSTM and Transformer across various prediction horizons and on both MSE and MAE metrics. This consistent superiority points to the strength of the proposed model's structural innovations, which are specifically tailored to the idiosyncrasies of financial data, such as high volatility, low signal-to-noise ratios, and non-stationarity.

A key factor contributing to the model's performance is its modular design, which combines three core components: (1) Mamba blocks for efficient long-range temporal encoding, (2) patch-based segmentation for localized noise suppression, and (3) a dual-axis Cross-Attention mechanism for refined inter-variable and inter-patch representation learning. This pipeline represents a departure from monolithic seq2one models and instead constructs a structured and interpretable flow for learning from complex financial sequences.

The Mamba blocks serve as a dynamic feature extractor, leveraging structured state space modeling to capture temporal dependencies with linear computational complexity. In contrast to attention-based models that suffer from scaling issues on long sequences, Mamba's use of linear recurrence with learned dynamics enables expressive temporal modeling while remaining resource-efficient. Furthermore, the residual connections in the Mamba stack improve gradient flow, stabilize training, and support deeper hierarchical learning—a critical need in the low-data regimes typical of financial time series.

Following temporal encoding, the patch segmentation module partitions the sequence into local regions, which offers multiple advantages: it suppresses high-frequency noise, allows the model to focus on coherent temporal patterns, and creates a more structured input for the attention module. The transformation from raw sequences into patch embeddings effectively localizes learning, mirroring the benefits of tokenization in NLP and patch embedding in vision transformers, but tailored for sequential financial data.

The Cross-Attention mechanism, designed with both row-wise (temporal) and column-wise (feature) attention, further enhances the model's ability to capture complex interdependencies. This axial design allows the model to attend separately to patch-wise patterns and feature-level correlations, which are particularly valuable in financial domains where correlations between variables (e.g., OHLCV and technical indicators) change over time and may exhibit non-linear interactions. Importantly, this design enables a decomposition of attention across dimensions, improving both interpretability and learning capacity.

Additionally, the layer-wise aggregation module, which computes a learnable weighted sum of outputs from all Cross-Attention layers, introduces a dynamic fusion mechanism. Rather than relying on the final layer's output alone, the model adaptively emphasizes more informative intermediate representations, thereby capturing both low- and high-level abstractions. This not only adds modeling flexibility but also acts as a form of implicit ensemble, boosting robustness.

The ablation studies further validate the effectiveness of each component. When the Mamba blocks are removed or replaced, we observe a substantial drop in both MSE and MAE, especially at longer input-output configurations (e.g., 50/10), indicating the importance of deep temporal memory and structured recurrence. Likewise, replacing the full Cross-Attention mechanism with simpler alternatives results in degraded performance, suggesting that both row-wise and column-wise attention are necessary to capture the temporal and cross-feature intricacies of financial data. The removal of layer aggregation also shows a measurable decline in predictive accuracy, highlighting the role of multi-layer fusion in extracting comprehensive representations.

Taken together, these findings demonstrate that the observed performance improvements do not arise from model size or parameter count alone but from a synergistic interplay between modular components, each purpose-built to address a structural limitation in existing financial forecasting models. The proposed design effectively bridges global temporal modeling, local pattern detection, and multi-scale attention, offering a principled and generalizable solution for financial time series forecasting.

4.7 Conclusion and Future Work

4.7.1 Conclusion

This paper presents Mamba-CrossAttention, a novel deep learning architecture tailored for short-term financial time series forecasting in noisy, data-limited, and non-stationary environments. Central to our design is the use of Mamba blocks, which serve as highly effective temporal feature extractors. By leveraging structured state space modeling with linear recurrence and learnable dynamics, Mamba enables the model to efficiently capture long-range dependencies across financial sequences—an essential capability for modeling complex market behaviors over time.

Complementing the temporal encoder, the Cross-Attention module operates as a powerful and interpretable sequence encoder, utilizing both row-wise and column-wise attention to capture intra-patch temporal relationships and inter-variable dependencies. This axial attention structure allows the model to disentangle and selectively focus on relevant temporal and feature-based patterns, significantly enhancing its representational power and flexibility.

A key contribution of our work is the introduction of a layer-wise weighted aggregation mechanism, inspired by residual network principles, which adaptively fuses outputs from multiple Cross-Attention layers. This enables the model to learn from both low-level and high-level representations and prioritize informative temporal patterns during training. Our experiments demonstrate that this aggregation strategy not only improves predictive accuracy

but also provides a form of implicit regularization, crucial in the context of financial data's volatility and noise.

Empirical evaluations on four diverse stock indices confirm the superiority of the proposed architecture over strong baseline models, including LSTM and Transformer variants, across various input-output configurations. Furthermore, ablation studies affirm the individual and combined contributions of each component, highlighting the model's ability to generalize effectively in low-data, high-noise financial forecasting scenarios. Collectively, these findings demonstrate that Mamba-CrossAttention offers a structurally principled and empirically validated approach to advancing deep learning in financial time series prediction.

CHAPTER 5

Portfolio Management With GAT-based Inter-stock Dependency Modeling

---

5.1 Introduction

Stock price prediction remains a critical challenge in financial research due to its importance for strategic decision-making. Existing methods primarily address two tasks: (1) regression, which forecasts future stock prices, and (2) classification, which generates trading signals such as buy, sell, or hold. However, the effectiveness of these approaches is often constrained by the limited and noisy nature of financial data. A common strategy to alleviate data scarcity is to aggregate historical data across multiple companies, but this practice overlooks stock-specific characteristics and inter-stock dependencies, ultimately limiting predictive accuracy. To address these limitations, we propose the BiLSTM–GAT–AM model and explicitly distinguish its contributions at the architectural and system levels. At the architectural level, the model is composed of established components, including bidirectional long short-term memory (BiLSTM) networks, graph attention networks (GAT), and attention mechanisms. These components are adapted to capture complementary aspects of stock behavior: temporal dynamics through BiLSTM and relational information through graph-based representations. Rather than introducing entirely new modules, the architectural contribution lies in tailoring these well-known building blocks to jointly model sequential and relational features in stock data. System-level novelty arises from how these components are organized and integrated into a unified framework. In particular, we introduce a dual-graph system in which one graph models technical similarities among stocks, while a second graph encodes fundamental industry relationships. An attention mechanism is then used to align and fuse information

from these two graphs, enabling the system to dynamically balance technical and fundamental signals. This coordinated interaction between temporal modeling, relational learning, and attention-based fusion goes beyond a straightforward recombination of existing methods and is specifically designed to address the heterogeneity and interdependence inherent in financial markets. We conduct extensive experiments, including ablation studies and comparisons with strong baseline models, to validate the effectiveness of the proposed framework. Results show consistent improvements in predictive performance. Furthermore, using the model's forecasts, we construct an optimized investment portfolio and perform backtesting on the test dataset. The resulting portfolio outperforms both baseline strategies and the S&P 500 index, demonstrating that while individual components are well established, their system-level integration yields tangible benefits for stock prediction and portfolio optimization.

5.2 Motivation

Stock market prediction has long been a critical area of research due to its potential to offer significant financial benefits(Gandhmal and Kumar 2019). Accurate forecasting of stock prices is highly sought after in the financial industry(Pang et al. 2020), as it can enable investors and traders to make informed decisions and optimize their portfolios for maximum returns(Kumar et al. 2022). However, the complex and volatile nature of financial markets makes stock market prediction an inherently challenging task(Thakkar and Chaudhari 2021). While deep learning models have made strides in improving predictive accuracy(Bustos and Pomares-Quimbaya 2020), there remain significant limitations that hinder their effectiveness in real-world trading scenarios.

Traditional deep learning approaches to stock market prediction are often constrained by two primary factors. First, these models typically aggregate data from multiple companies into a single dataset without considering the heterogeneity inherent in the stock market(Jiang 2021; Ji et al. 2021). By treating companies as homogenous entities, these models overlook the distinct characteristics of individual companies and the intricate relationships between them(Rouf et al. 2021),(Roe 2020). Factors such as industry sector, market capitalization, and

financial health can all influence stock performance, and ignoring these nuances can result in oversimplified models that do not reflect the true dynamics of the market(Htun et al. 2023).

Second, the narrow focus on predicting stock prices alone provides little operational value for actual trading(Liu et al. 2023). While price predictions may achieve high accuracy, they often fail to offer actionable insights that can inform profitable trading decisions(Ma et al. 2023). In practice, financial markets require more than accurate price forecasts; they demand a comprehensive understanding of market structure, inter-company correlations, and risk-adjusted returns(Cagliero et al. 2023),(Cao et al. 2024). Thus, moving from pure price prediction to a framework that considers both predictive performance and practical decision-making is crucial(Ma et al. 2024).

To address these limitations, recent advances in Graph Neural Networks (GNNs) have shown significant potential in capturing the complex relationships between stocks(Wu et al. 2020),(Cheng et al. 2022). Unlike traditional models, GNNs can treat each stock as a node and define the connections between them using various criteria(Cheng et al. 2022). This is particularly useful in financial markets where inter-company relationships can significantly impact stock performance(Xu et al. 2021),(Hsu et al. 2021). For example, companies within the same industry often exhibit similar price behaviors due to shared economic factors, while companies with different business models may demonstrate weaker correlations(Han et al. 2024). Graph Attention Networks (GATs), an extension of GNNs, are particularly powerful as they assign attention weights to different edges, enabling the model to focus more on relevant relationships between companies and discount less important ones. However, the success of graph-based models is highly sensitive to the design of the adjacency matrix, which determines how connections between companies are defined(Cheng and Li 2021),(Liu et al. 2021), and (Wang et al. 2022b).

In financial markets, these connections can be understood through two major analytical frameworks: technical analysis(Nti et al. 2020) and fundamental analysis(Rahman et al. 2022). Technical analysis focuses on price movements and trends, while fundamental analysis examines a company's underlying attributes, such as industry classification, financial performance, and market position(Singh et al. 2022). Most existing models that use GNNs

for stock prediction primarily focus on one of these aspects, typically using price data to model relationships(Song et al. 2023),(Xu and Zhang 2023). However, to fully capture the complexity of stock markets, it is necessary to consider both technical and fundamental factors.

We propose a novel stock market prediction and portfolio optimization model that integrates Bi-directional Long Short-Term Memory (BiLSTM) networks, Graph Attention Networks (GATs), and Attention Mechanisms (AMs). Our approach addresses key limitations of traditional models by treating each company as a distinct node and constructing two separate adjacency matrices: one capturing technical similarities based on price movements and the other reflecting fundamental similarities based on industry sectors. This dual-graph representation enables a more comprehensive understanding of inter-company relationships.

Beyond price prediction, our model ranks stocks by predicted return rates, constructing a portfolio optimized for risk-adjusted returns. Performance evaluation against benchmark indices (e.g., S&P 500) and baseline models demonstrates superior results across key financial metrics, including internal rate of return (IRR), Sharpe ratio, and final balance. These findings highlight the model's effectiveness in generating higher returns while managing risk.

This research advances stock market prediction by integrating technical and fundamental analysis into a unified framework. Furthermore, it shifts focus from pure forecasting to actionable trading strategies, bridging the gap between financial prediction and decision-making.

5.3 Methodology

5.3.1 Technique indicators

This study leverages a set of basic and derived technical indicators to analyze stock price movements and inform trading decisions. The fundamental indicators used are the Open, High, Low, and Close (OHLC) prices, as well as tick volume and spread. These indicators

form the basis for generating an additional 44 features, categorized Table 5.1.

Indicators	Labels	Count
Simple Moving Averages	'mv100','mv50','mv9'	3
Bollinger Bands	'bb_bbm','bb_bbh','bb_bbl'	3
Relative Strength Index	'rsi15','rsi9','rsi50'	3
Percentage Change Features	'f1' to 'f10'	10
Moving Average Comparisons	'f11' to 'f16'	6
RSI Comparisons	'f17' to 'f18'	2
Bollinger Band Comparisons	'f19' to 'f22'	4
Rolling Maximum and Minimum	'f23' to 'f28'	6
Close Price Shifts	'f29' to 'f33'	5
Trading time	'h1','w1'	2
Total		44

TABLE 5.1: List of indicators, labels, and number of indicators used

5.3.1.1 Simple Moving Averages (SMA)

The Simple Moving Average (SMA) is a widely used technical indicator that smooths out price data by creating a constantly updated average price. It helps in identifying the direction of the trend over a specified period Johnston et al. 1999.

The SMA is calculated by taking the arithmetic mean of a given set of prices over a specific number of periods.

$$\text{SMA}_n(t) = \frac{1}{n} \sum_{i=0}^{n-1} P(t-i) \tag{5.1}$$

where:

- $\text{SMA}_n(t)$ is the Simple Moving Average at time t over n periods.
- $P(t-i)$ is the close price at time $t-i$.
- n is the number of periods over which the average is calculated.

In this research the 'mv100', 'mv50', 'mv9' are moving averages over 100, 50, and 9 periods, respectively. SMAs help in smoothing out price data to identify trends over different time frames.

5.3.1.2 Bollinger Bands

Bollinger Bands consist of a set of lines plotted two standard deviations (positively and negatively) away from a simple moving average (SMA) of the price which provide a relative definition of high and low prices of a financial instrument Bollinger 2002.

Middle Band (MB): The middle band is the simple moving average (SMA) of the close price, typically over 20 periods.

$$\text{MB}(t) = \text{SMA}_{20}(t) = \frac{1}{20} \sum_{i=0}^{19} P(t-i) \tag{5.2}$$

Upper Band (UB): The upper band is calculated by adding two standard deviations to the middle band.

$$\text{UB}(t) = \text{MB}(t) + 2 \times \sigma_{20}(t) \tag{5.3}$$

where $\sigma_{20}(t)$ is the standard deviation of the close price over 20 periods.

Lower Band (LB): The lower band is calculated by subtracting two standard deviations from the middle band.

$$\text{LB}(t) = \text{MB}(t) - 2 \times \sigma_{20}(t) \tag{5.4}$$

The 'bb_bbm', 'bb_bbh', 'bb_bbl' represent the middle band (moving average), upper band, and lower band respectively.

5.3.1.3 Relative Strength Index (RSI)

The Relative Strength Index (RSI) Gumparthi et al. 2017is a momentum oscillator that measures the speed and change of close price movements. It is used to identify overbought or oversold conditions in a market. The RSI oscillates between 0 and 100 and is typically used with a 14-period setting.

(1) Calculate the average gains and losses over the specified period (e.g., 14 or 50 periods).

(2) Calculate the Relative Strength (RS):

$$\text{RS} = \frac{\text{Average Gain}}{\text{Average Loss}} \tag{5.5}$$

(3) Calculate the RSI:

$$\text{RSI} = 100 - \left(\frac{100}{1 + \text{RS}}\right) \tag{5.6}$$

'rsi14', 'rsi50' are RSI over 14 and 50 periods, respectively, measures the speed and change of close price movements to identify overbought or oversold conditions. 'rsimv9' is a 9-period moving average of the 14-period RSI.

5.3.1.4 Price Percentage Change Features

'f1' to 'f10' calculate the percentage change between different prices (open, close, high, low) and their shifts over different periods.

$$\text{f1} = \left(\frac{\text{Close} - \text{Open}}{\text{Open}}\right) \times 100 \tag{5.7}$$

$$\text{f2} = \left(\frac{\text{High} - \text{Low}}{\text{Low}}\right) \times 100 \tag{5.8}$$

$$\text{f3} = \left(\frac{\text{High}_{t-1} - \text{Low}_{t-1}}{\text{Low}_{t-1}}\right) \times 100 \tag{5.9}$$

$$\text{f4} = \left(\frac{\text{High}_{t-2} - \text{Low}_{t-2}}{\text{Low}_{t-2}}\right) \times 100 \tag{5.10}$$

$$\text{f5} = \left(\frac{\text{High}_{t-3} - \text{Low}_{t-3}}{\text{Low}_{t-3}}\right) \times 100 \tag{5.11}$$

$$\text{f6} = \left(\frac{\text{High}_{t-4} - \text{Low}_{t-4}}{\text{Low}_{t-4}}\right) \times 100 \tag{5.12}$$

$$\text{f7} = \left(\frac{\text{High} - \text{Open}}{\text{Open}}\right) \times 100 \tag{5.13}$$

$$\text{f8} = \left(\frac{\text{High} - \text{Close}}{\text{Close}}\right) \times 100 \tag{5.14}$$

$$\text{f9} = \left(\frac{\text{Open} - \text{Low}}{\text{Low}}\right) \times 100 \tag{5.15}$$

$$\text{f10} = \left(\frac{\text{Close} - \text{Low}}{\text{Low}}\right) \times 100 \tag{5.16}$$

5.3.1.5 Moving Average Percentage Change Features

'f11' to 'f13' calculate the percentage change between the closing price and the moving averages (50-period, 9-period, and 100-period, respectively). Features 'f14' to 'f16' compute the percentage changes between different moving averages themselves.

$$\text{f11} = \left(\frac{\text{Close} - \text{MV}_{50}}{\text{MV}_{50}} \right) \times 100 \tag{5.17}$$

$$\text{f12} = \left(\frac{\text{Close} - \text{MV}_{9}}{\text{MV}_{9}} \right) \times 100 \tag{5.18}$$

$$\text{f13} = \left(\frac{\text{Close} - \text{MV}_{100}}{\text{MV}_{100}} \right) \times 100 \tag{5.19}$$

$$\text{f14} = \left(\frac{\text{MV}_{9} - \text{MV}_{50}}{\text{MV}_{50}} \right) \times 100 \tag{5.20}$$

$$\text{f15} = \left(\frac{\text{MV}_{9} - \text{MV}_{100}}{\text{MV}_{100}} \right) \times 100 \tag{5.21}$$

$$\text{f16} = \left(\frac{\text{MV}_{50} - \text{MV}_{100}}{\text{MV}_{100}} \right) \times 100 \tag{5.22}$$

5.3.1.6 RSI Percentage Change Features

'f17', 'f18' calculate the percentage difference between different RSI values (rsi14, rsi50, rsimv9). f17 computes the percentage change between the 14-period RSI and the 50-period RSI, while f18 calculates the percentage change between the 50-period RSI and a 9-period simple moving average of the 14-period RSI.

$$\text{f17} = \left(\frac{\text{RSI}_{14} - \text{RSI}_{50}}{\text{RSI}_{50}} \right) \times 100 \tag{5.23}$$

$$\text{f18} = \left(\frac{\text{RSI}_{50} - \text{RSI}_{\text{mv9}}}{\text{RSI}_{\text{mv9}}} \right) \times 100 \tag{5.24}$$

5.3.1.7 Bollinger Band Percentage Change Features

'f19' to 'f23' calculate the percentage difference between the close price and Bollinger Bands (bb_bbm, bb_bbh, bb_bbl), and between the bands themselves. Specifically, f19 computes the percentage change between the closing price and the middle Bollinger Band (20-period

simple moving average), f20 calculates the percentage change between the closing price and the upper Bollinger Band, and f21 calculates the percentage change between the closing price and the lower Bollinger Band. Additionally, f22 computes the percentage change between the lower and upper Bollinger Bands.

$$\text{f19} = \left(\frac{\text{Close} - \text{BB}_{\text{Middle}}}{\text{BB}_{\text{Middle}}} \right) \times 100 \tag{5.25}$$

$$\text{f20} = \left(\frac{\text{Close} - \text{BB}_{\text{Upper}}}{\text{BB}_{\text{Upper}}} \right) \times 100 \tag{5.26}$$

$$\text{f21} = \left(\frac{\text{Close} - \text{BB}_{\text{Lower}}}{\text{BB}_{\text{Lower}}} \right) \times 100 \tag{5.27}$$

$$\text{f22} = \left(\frac{\text{BB}_{\text{Lower}} - \text{BB}_{\text{Upper}}}{\text{BB}_{\text{Upper}}} \right) \times 100 \tag{5.28}$$

5.3.1.8 Rolling Maximum and Minimum

'f23' to 'f28' calculate the percentage difference between the close price and its rolling maximum or minimum over different periods (20, 50, 100). Specifically, 'f23' to 'f25' compute the percentage change between the rolling maximum closing prices over 20, 50, and 100 periods, respectively, and the current closing price. Conversely, 'f26' to 'f28' calculate the percentage change between the rolling minimum closing prices over the same periods and the current closing price.

$$\text{f23} = \left(\frac{\max(\text{Close}_{t-20:t}) - \text{Close}}{\text{Close}} \right) \times 100 \tag{5.29}$$

$$\text{f24} = \left(\frac{\max(\text{Close}_{t-50:t}) - \text{Close}}{\text{Close}} \right) \times 100 \tag{5.30}$$

$$\text{f25} = \left(\frac{\max(\text{Close}_{t-100:t}) - \text{Close}}{\text{Close}} \right) \times 100 \tag{5.31}$$

$$\text{f26} = \left(\frac{\min(\text{Close}_{t-20:t}) - \text{Close}}{\text{Close}} \right) \times 100 \tag{5.32}$$

$$\text{f27} = \left(\frac{\min(\text{Close}_{t-50:t}) - \text{Close}}{\text{Close}} \right) \times 100 \tag{5.33}$$

$$\text{f28} = \left(\frac{\min(\text{Close}_{t-100:t}) - \text{Close}}{\text{Close}} \right) \times 100 \tag{5.34}$$

5.3.1.9 Close Price Shifts

'f29' to 'f33' calculate the percentage change of the close price compared to its previous values over different periods (1 to 5).'f29' computes the percentage change from the closing price of the previous day to the current closing price. Features 'f30' to 'f33' extend this calculation to the closing prices from 2 to 5 days prior, respectively.

$$\text{f29} = \left(\frac{\text{Close}_{t-1} - \text{Close}}{\text{Close}} \right) \times 100 \tag{5.35}$$

$$\text{f30} = \left(\frac{\text{Close}_{t-2} - \text{Close}}{\text{Close}} \right) \times 100 \tag{5.36}$$

$$\text{f31} = \left(\frac{\text{Close}_{t-3} - \text{Close}}{\text{Close}} \right) \times 100 \tag{5.37}$$

$$\text{f32} = \left(\frac{\text{Close}_{t-4} - \text{Close}}{\text{Close}} \right) \times 100 \tag{5.38}$$

$$\text{f33} = \left(\frac{\text{Close}_{t-5} - \text{Close}}{\text{Close}} \right) \times 100 \tag{5.39}$$

5.3.1.10 Trading time

'h1' captures the hour of the day from the datetime values. The second line creates a new column wd that captures the day of the week (with Monday as 0 and Sunday as 6) from the datetime values, which could be useful for identifying patterns related to different weekdays.

$$\text{h1} = \text{Hour}(\text{datetime}) \tag{5.40}$$

$$\text{wd} = \text{Weekday}(\text{datetime}) \tag{5.41}$$

5.3.2 Long Short-Term Memory (LSTM) Network

In this study, we employ Long Short-Term Memory (LSTM) networks to model sequential data and capture both short-term and long-term dependencies. LSTM networks, a variant of Recurrent Neural Networks (RNNs), are particularly well-suited for tasks involving temporal sequences due to their ability to mitigate the vanishing gradient problem inherent in traditional

RNNs. This is achieved through the inclusion of a memory cell that maintains information over extended time steps.

The structure of an LSTM cell is defined by three key gates: the input gate, the forget gate, and the output gate. These gates regulate the flow of information into and out of the memory cell, enabling the network to selectively retain or discard information at each time step. The equations governing the behavior of the LSTM cell are as follows:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{5.42}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{5.43}$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{5.44}$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{5.45}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{5.46}$$

$$h_t = o_t * \tanh(C_t) \tag{5.47}$$

In these equations, f_t represents the forget gate, which determines the extent to which the previous cell state C_{t-1} should be forgotten. i_t is the input gate, controlling what new information is stored in the current cell state. The candidate cell state, $\tilde{C}_t$, is computed based on the previous hidden state h_{t-1} and current input x_t. The updated cell state C_t is a combination of the previous cell state and the candidate cell state, modulated by the forget and input gates. Finally, o_t is the output gate, which controls the output of the LSTM cell, and h_t represents the hidden state, which is passed to the next time step.

LSTMs are particularly effective in time series forecasting tasks, such as stock price prediction, as they can capture both short-term fluctuations and long-term trends in financial data. By leveraging historical stock prices and other relevant indicators, our LSTM model provides improved forecasting accuracy compared to traditional models.

The weight matrices W_f, W_i, W_C, W_o and bias vectors b_f, b_i, b_C, b_o are learned during the training process, ensuring the model adapts to the dynamics of the data.

5.3.3 Graph Construction

In our approach, each company is represented as a unique node within a graph, where edges are established based on similarity metrics. Companies exhibiting higher similarity are interconnected, effectively capturing interdependencies within the financial market. This graph-based framework enables the incorporation of relational information among stocks, allowing for the simultaneous prediction of closing prices across all companies.

5.3.3.1 Scaling

Before constructing the weighted edge graph, we first normalize the closing prices of all companies, scaling them to the range [0, 1]. Let $X_i = [x_1, x_2, \ldots, x_T]$ represent the normalized time series of closing prices for company i. The normalization is done as follows:

$$X_i = \frac{X_i - \min(X_i)}{\max(X_i) - \min(X_i)} \tag{5.48}$$

5.3.3.2 Graph for technical similarity

After normalizing the closing prices, we compute the Dynamic Time Warping (DTW) distance between every pair of companies using the fastdtw algorithm, which provides an efficient approximation of the DTW distance. Given two normalized time series X_i and X_j for companies i and j, the DTW distance $DTW(X_i, X_j)$ is calculated.

Next, we introduce a threshold τ to determine which company pairs should be connected by edges. For any two companies i and j, if their DTW distance $DTW(X_i, X_j)$ is less than the threshold τ, an edge is established between them. The weight of the edge w_{ij} is computed as:

$$w_{ij} = \tau - DTW(X_i, X_j) \tag{5.49}$$

This formula ensures that the closer the DTW distance is to 0 (i.e., the more similar the two companies' stock prices are), the larger the edge weight will be. Conversely, as the DTW distance approaches the threshold, the edge weight decreases.

The result is a weighted graph, where each node represents a company, and edges are established only between companies whose DTW distance is below the threshold τ. The edge weight reflects the similarity of the companies' stock price movements, with smaller DTW distances leading to stronger connections. This graph provides a structural representation of the relationships between companies based on their normalized closing prices.

This weighted graph as is shown in Figure 5.4 is then used in subsequent Graph Neural Network (GNN) and Graph Attention Network (GAT) models to predict future trends by incorporating both individual company data and the relationships captured in the graph.

5.3.3.3 Graph for fundamental similarity

The construction of the second graph is predicated on the industry sectors of the selected companies. We introduce this graph under the hypothesis that companies within the same industry sector exhibit similar patterns in price performance. To operationalize this concept, an edge is established between companies belonging to the same industry sector, with each edge assigned a uniform weight. This approach assumes that the shared sectorial characteristics will reflect in comparable price change behaviors, providing a structured framework to analyze the interconnectedness of market performance across similar industries.

5.3.4 Graph Neural Networks (GNN) and Graph Attention Networks (GAT)

In this study, we also incorporate Graph Neural Networks (GNNs) to model complex relationships between entities represented as graphs. GNNs are particularly suited for structured data, where the relationships between nodes can provide crucial insights that are otherwise missed by traditional neural networks. Each node in a graph represents an entity, and edges define relationships or interactions between these entities. The objective of a GNN is to learn a

node representation by aggregating features from neighboring nodes through message passing mechanisms.

The key operation in a GNN is the neighborhood aggregation or message passing, where the representation of each node is updated based on its neighbors' features. Formally, the node representation $h_v^{(k)}$ at the k-th layer is computed as:

$$h_v^{(k)} = \text{aggregate}\left(h_u^{(k-1)} : u \in \mathcal{N}(v)\right) \tag{5.50}$$

where $\mathcal{N}(v)$ represents the set of neighbors of node v, and $h_u^{(k-1)}$ is the representation of node u from the previous layer. The aggregation function can vary based on the GNN variant (e.g., summation, averaging, or more complex functions).

5.3.4.1 Graph Attention Networks (GAT)

To further enhance the expressiveness of GNNs, we utilize Graph Attention Networks (GATs), which introduce attention mechanisms to assign different importance weights to neighboring nodes during the aggregation process. In a GAT, rather than treating all neighbors equally, attention scores are learned, allowing the model to focus on the most relevant nodes. This attention mechanism is particularly useful in scenarios where not all neighbors contribute equally to a node's final representation.

For each node v, the attention coefficient α_{vu} between node v and its neighbor u is computed as:

$$\alpha_{vu} = \frac{\exp\left(\text{LeakyReLU}\left(\mathbf{a}^T[Wh_v \| Wh_u]\right)\right)}{\sum_{k \in \mathcal{N}(v)} \exp\left(\text{LeakyReLU}\left(\mathbf{a}^T[Wh_v \| Wh_k]\right)\right)} \tag{5.51}$$

where W is a weight matrix applied to the node features, $\mathbf{a}$ is a learnable attention vector, and $\|$ denotes concatenation. The attention coefficients α_{vu} are then used to compute the weighted sum of the neighboring features to update the node representation:

$$h_v^{\text{new}} = \sigma\left(\sum_{u \in \mathcal{N}(v)} \alpha_{vu} W h_u\right) \tag{5.52}$$

Here, σ is a non-linear activation function, such as ReLU. This mechanism allows GATs to adaptively attend to the most informative neighbors, improving the model's performance on tasks where the relevance of neighbors varies.

By leveraging GNNs and GATs, our methodology captures both local and global dependencies in graph-structured data, offering a robust framework for learning node representations. The graph attention mechanism further enhances the model's ability to focus on critical nodes in the graph, leading to more accurate predictions, particularly in tasks involving structured data such as social networks, molecular graphs, or citation networks.

5.3.5 Attention Mechanism for Multi-Channel Graph Networks

In our approach, we utilize an attention mechanism to combine the outputs of two separate graph networks, each representing a different channel of information for every node in the graph. By leveraging this mechanism, we are able to aggregate information from both graph channels and make more informed predictions based on the features of each node.

Given two graph networks, G_1 and G_2, the output of each network for a node v is represented by the feature vectors $h_v^{G_1}$ and $h_v^{G_2}$, respectively. Our goal is to combine these two outputs using an attention mechanism that determines the contribution of each graph channel to the final representation of node v.

The attention mechanism assigns a weight to each graph channel based on the relevance of the information provided by each channel for the specific node. For each node v, the attention coefficients $\beta_v^{G_1}$ and $\beta_v^{G_2}$ are computed as follows:

$$\beta_v^{G_i} = \frac{\exp\left(\mathbf{a}^T \cdot \text{concat}(h_v^{G_1}, h_v^{G_2})\right)}{\sum_{j \in \{1,2\}} \exp\left(\mathbf{a}^T \cdot \text{concat}(h_v^{G_1}, h_v^{G_2})\right)} \tag{5.53}$$

Here, $\mathbf{a}$ is a learnable attention vector, and concat$(h_v^{G_1}, h_v^{G_2})$ represents the concatenation of the feature vectors from both graph channels for node v. The softmax function ensures that the attention coefficients $\beta_v^{G_1}$ and $\beta_v^{G_2}$ sum to 1, allowing the model to weigh the importance of each channel dynamically.

The final representation h_v^{final} for node v is then computed as the weighted sum of the two graph channels:

$$h_v^{\text{final}} = \beta_v^{G_1} h_v^{G_1} + \beta_v^{G_2} h_v^{G_2} \tag{5.54}$$

This mechanism enables the model to focus on the most relevant graph channel for each node, depending on the specific task and data characteristics.

5.3.5.1 Node-Level Prediction

Once the final node representation h_v^{final} is obtained by combining the two graph outputs, we use it to make predictions at the node level. Specifically, for each node v, we apply a fully connected layer followed by a softmax function to output the predicted class or value for that node. The prediction $\hat{y}_v$ for node v is computed as:

$$\hat{y}_v = \text{softmax}(W \cdot h_v^{\text{final}} + b) \tag{5.55}$$

where W is a weight matrix, and b is a bias term learned during the training process. The softmax function converts the raw scores into probabilities for classification tasks, while other activation functions may be used for regression tasks.

By incorporating the attention mechanism to fuse the two graph channels and making node-specific predictions, our model is able to leverage the full potential of multi-channel graph information, leading to more accurate and context-aware predictions for each node.

5.3.6 Proposed LSTM-GAT Attention Model

In this study, we propose a novel model that combines Long Short-Term Memory (LSTM) networks, Graph Attention Networks (GATs), and an attention mechanism to capture both temporal and relational dependencies in time series data. A single input at time t consists of a 2-dimensional vector representing 82 companies, with each company characterized by 51 features, including the technical indicators mentioned previously.

5.3.6.1 Model Architecture

Our proposed model consists of three main components: the LSTM layers, the GAT layers for two graph channels, and an attention mechanism to fuse the outputs from these channels. The final output is generated through a series of fully connected layers. Below is a detailed description of each component:

LSTM Layers: The LSTM layers are used to capture the temporal dependencies in the time series data. The input time series data for each node is passed through a two-layer bidirectional LSTM, which processes the sequential information in both forward and backward directions. The LSTM output at the final time step is used as the node's representation of the sequence. This results in a feature vector for each node that summarizes the historical information.

GAT Layers: The model employs two sets of GAT layers to process graph-structured data from two distinct adjacency matrices (representing two different graph channels). Each graph is processed through three consecutive GAT layers. Each GAT layer updates the node features by attending to the features of neighboring nodes, where the attention mechanism learns to assign different importance weights to each neighbor.

For both graph channels, the node features from the LSTM layers are first passed through the GAT layers to propagate information from neighboring nodes, resulting in updated node representations for each channel.

Attention Mechanism: After processing the node features through the two sets of GAT layers, the model employs an attention mechanism to fuse the outputs from the two graph

channels. The GAT outputs from the two channels are stacked and passed through a linear layer to compute attention scores for each channel. These scores are then normalized using a softmax function to obtain attention weights, which are applied to the GAT outputs. The final node representation is computed as the weighted sum of the two GAT outputs.

Fully Connected Layers: The combined node features from the attention mechanism are passed through a series of fully connected layers. The first two layers use ReLU activations, while the final layer outputs the prediction for each node. This part of the model captures complex non-linear relationships in the learned node representations as is shown in Figure 5.1.

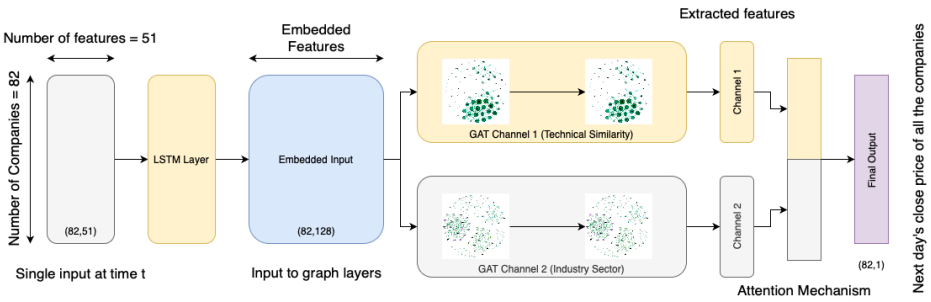

FIGURE 5.1: Visualisation of proposed LSTM-GAT-AM model

5.4 Evaluation Metrics

In this study, four primary evaluation metrics are employed to assess the performance of our proposed model: the Mean Squared Error (MSE) for prediction accuracy, the Sharpe Ratio for portfolio performance, the Mean Absolute Error (MAE) for absolute prediction deviations, and the Annual Return and Maximum Drawdown for overall financial performance. These metrics provide a comprehensive analysis of the predictive capability, risk-adjusted return, and financial resilience of the model.

5.4.1 Mean Squared Error (MSE)

The Mean Squared Error (MSE) is used to measure the accuracy of the model's predictions of the next day's stock prices. It evaluates how close the predicted values are to the actual values, with a lower MSE indicating a more accurate model. The MSE is calculated as follows:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{5.56}$$

where y_i is the actual stock price, $\hat{y}_i$ is the predicted stock price, and n represents the total number of data points.

5.4.2 Sharpe Ratio

The Sharpe Ratio is utilized to evaluate the risk-adjusted return of the portfolio generated from the model's predictions. It compares the portfolio's excess return to its volatility, with a higher Sharpe Ratio indicating a more favorable risk-adjusted return. The Sharpe Ratio is computed as follows:

$$\text{Sharpe Ratio} = \frac{\mathbb{E}[R_p - R_f]}{\sigma_p} \tag{5.57}$$

where R_p is the portfolio return, R_f is the risk-free rate, and σ_p is the standard deviation of the portfolio's excess return.

5.4.3 Mean Absolute Error (MAE)

The Mean Absolute Error (MAE) measures the average magnitude of the errors in a set of predictions, without considering their direction. It is less sensitive to outliers than the MSE and provides a clearer measure of actual prediction errors. The MAE is calculated as follows:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{5.58}$$

where y_i and $\hat{y}_i$ are defined as in the MSE section.

5.4.4 Annual Return and Maximum Drawdown

The Annual Return measures the percentage change in the portfolio value over a year, reflecting the overall profitability of the investment strategy. The Maximum Drawdown assesses the largest single drop from peak to trough in the portfolio during the investment period, providing insight into the potential risk of losses. These metrics together offer a complete picture of the financial performance and risk resilience of the portfolio.

- **Annual Return:** Calculated based on the cumulative returns at the end of the year compared to the initial portfolio value.
- **Maximum Drawdown:** Defined as the maximum observed loss from a peak to a trough of the portfolio, before a new peak is attained.

5.5 Experimental Setup

5.5.1 Data source

The data for this study were obtained from Yahoo Finance, comprising the largest 100 companies by market share in the S&P 500. The sampling period spanned from January 1, 2020, to December 31, 2023 including the Open, High, Low, Close, and Volume. Then we generate the technical indicators respectively as is shown in Figure 5.2.

To ensure data consistency, we specifically excluded companies that underwent stock splits during the sample period. Stock splits introduce abrupt, non-fundamental shifts in stock prices, which could obscure the true patterns the model is designed to detect. By omitting such companies, we minimize distortions and focus on capturing the underlying relationships

between company characteristics and stock price movements. This approach enhances the model's predictive accuracy and robustness.

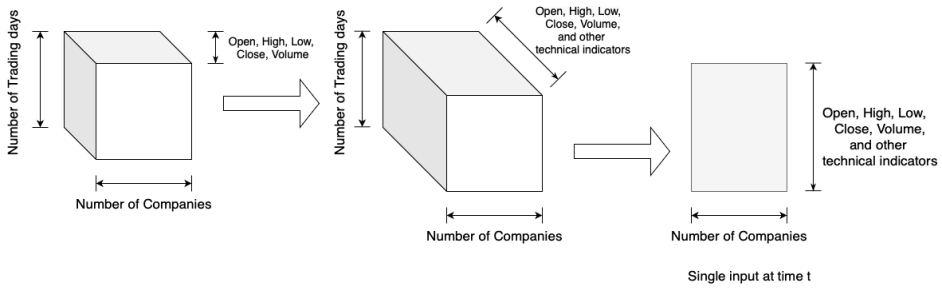

FIGURE 5.2: Visualisation of the process of data pre-processing

5.5.2 Data Pre-processing

5.5.2.1 Scaling

To ensure that all features contribute equally to the prediction model and to improve the performance of the regression algorithms, data scaling will be applied during the pre-processing step. We will employ *StandardScaler()* to standardize the features and stock prices by removing the mean and scaling to unit variance. This transformation is expressed as:

$$z_i = \frac{x_i - \mu}{\sigma} \tag{5.59}$$

where x_i represents the original feature value, μ is the mean, and σ is the standard deviation of the feature values. This standardization ensures that the data has a mean of 0 and a standard deviation of 1, aligning all features on the same scale. This step is essential for models that are sensitive to feature scales or assume normally distributed input data.

5.5.2.2 Train test split

To prevent the model from learning from future data, which could lead to overfitting, the dataset is split into training, validation, and test sets based on chronological order. The training data spans from 2020-01-01 to 2023-04-01, the validation set covers the period from

2023-04-01 to 2023-08-01, and the test set includes data from 2023-08-01 to 2023-12-31 as is shown in Figure 5.3. It is important to note that the scaling of features is performed using the statistics (mean and standard deviation) derived only from the training set to ensure that no future information is leaked into the model during the scaling process. This approach helps maintain the integrity of the model evaluation.

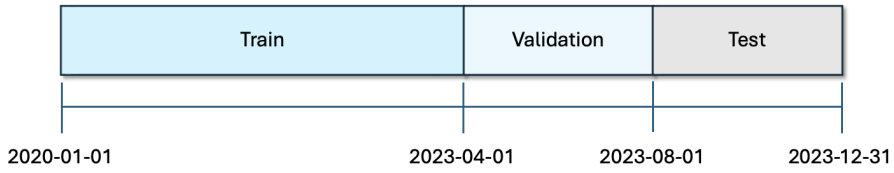

FIGURE 5.3: The split interval of train, validation, and test set

5.5.3 Graph preparing

5.5.3.1 Graph with technical analysis

As is shown in Figure 5.4 the first graph is a weighted and undirected graph based on the Dynamic Time Warping (DTW) distance between the companies. Companies with smaller distances will have a higher weight on the edges that link them.

5.5.3.2 Graph with fundamental analysis

As is shown in Figure 5.5 the second graph is an undirected and unweighted graph based on the industry sector the company belongs to. Companies within the same industry sector will be linked.

5.5.4 Software and Hardware Setup

The experiments in this study were conducted using the following software environment: PyTorch 2.4.0, TensorFlow 2.13.0, Keras 2.13.1, Pandas 2.0.3, and Numpy 1.24.3. The hardware configuration consisted of an Apple Silicon processor (ARM architecture) with 12

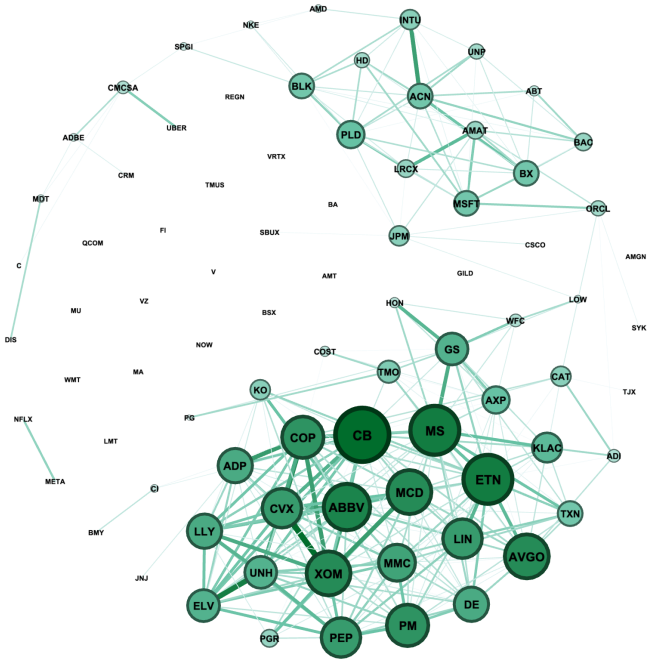

FIGURE 5.4: Visualization of the adjacency matrix using the Fruchterman-Reingold algorithm. Each company is represented as a unique dot. Companies are connected with edges where the weight is inversely related to their DTW distance, with closer companies having heavier connections. The size of each dot is scaled based on the number of connections to enhance visual clarity.

CPU cores (12 physical, 12 logical) and 32.0 GB of RAM, running on macOS (Darwin 23.6.0). GPU acceleration was leveraged using the MPS Backend (Metal Performance Shaders), which was enabled and available for PyTorch, with the MPS device specified as mps.

5.6 Experimental Results and Comparative Analysis

In this section, we present the results in a two-fold manner. Initially, we evaluate our model's ability to predict the next day's closing price, comparing its performance against other models with single graph generated by DTW distance or Industry sector using the Mean Squared Error (MSE) as the evaluation metric.

Subsequently, utilizing the predicted closing prices, we compute the daily return rate, ranking companies based on this metric. A portfolio is then formulated by selecting the top-performing companies. This portfolio is backtested with actual market data to assess its practical viability.

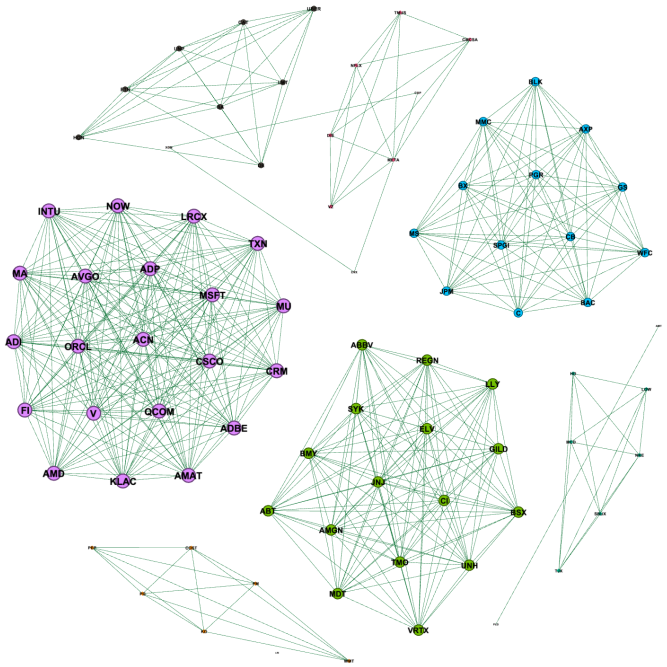

FIGURE 5.5: Visualization of the adjacency matrix using the Fruchterman-Reingold algorithm. Each company is represented as a unique dot. Companies are connected within the same industry sector. The size of each dot is scaled based on the number of connections to enhance visual clarity.

The process involves recalculating the true average return rate of the selected companies daily and adjusting the portfolio composition accordingly. This iterative procedure is replicated across each trading day to emulate the portfolio's temporal performance.

We conclude by analyzing and contrasting the final balance and Sharpe ratio achieved by our model against those of other models during the testing period.

5.6.1 Results on prediction

In this section, we conduct a detailed comparison of our proposed model against several baseline models including ablation studies, employing identical experimental setups and performance metrics to ensure a fair and rigorous evaluation. The results of this comparison are presented in table 5.2.

The baseline BiLSTM-GAT model retains its original structure, utilizing a single graph constructed from Dynamic Time Warping (DTW) distance or the industry sector graph. In contrast, the BiLSTM-GNN model replaces the Graph Attention Network (GAT) framework with a more general Graph Neural Network (GNN) architecture to highlight the differences in performance attributable to the graph modeling approach. Notably, our proposed BiLSTM-GAT-AM model introduces an enhanced architecture by incorporating two distinct graphs, enabling it to capture more complex relational structures within the data.

By comparing these models side by side, we aim to illustrate the improvements brought about by the additional graph in the BiLSTM-GAT-AM model, which allows for richer feature extraction and, ultimately, superior predictive performance.

Model	MSE (Mean Squared Error)	MAE (Mean Absolute Error)
BiLSTM-GAT-AM (DTW & Sector graph)	**0.00440**	**0.04680**
BiLSTM-GAT (DTW graph)	0.00454	0.04730
BiLSTM-GAT (Sector graph)	0.00543	0.05408
BiLSTM-GNN (DTW graph)	0.01518	0.08032
BiLSTM-GNN (Sector graph)	0.00561	0.05484

TABLE 5.2: Comparison of proposed model and baseline model in MSE/MAE for prediction of next day's close price.

5.6.2 Results on portfolio return

In this section, we compare the portfolio returns of the three models with the performance of the S&P 500 index during the testing period. The results, as illustrated in Figure 5.6, demonstrate that our proposed model achieves the highest final portfolio balance, outperforming both the baseline models and the S&P 500 index. Furthermore, the proposed model exhibits the largest Sharpe ratio, smaller drawdown, and higher annual return rate as is shown in Table 5.3, indicating a superior risk-adjusted return compared to the other models. This highlights the model's ability to generate consistent returns while effectively managing risk throughout the evaluation period.

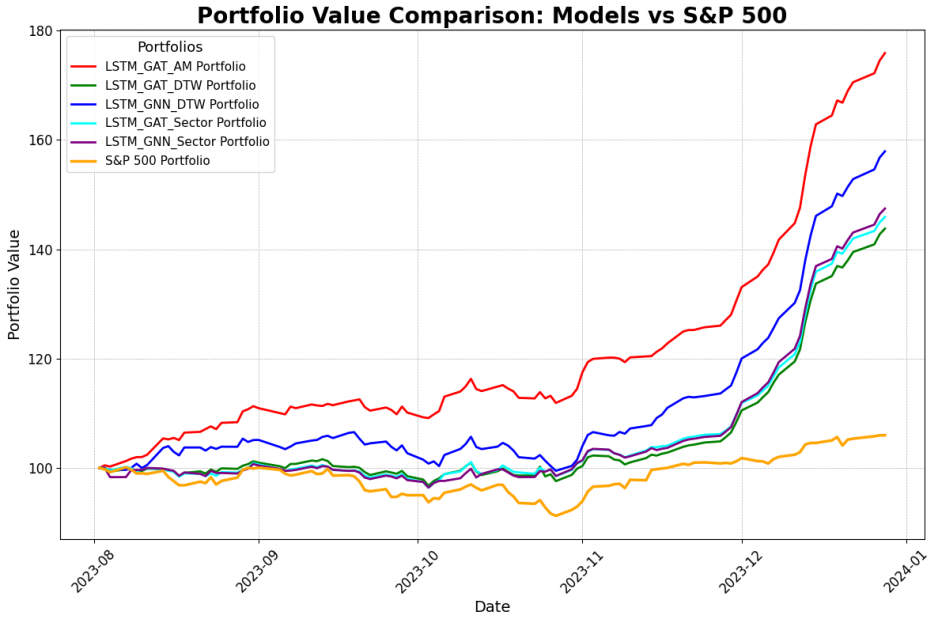

FIGURE 5.6: Comparison of the final balance of the proposed model and baseline models on testing period.

Model	Final Balance	Sharpe ratio	maximum DD	Annual RR
BiLSTM-GAT-AM (DTW & Sector graph)	**176.56**	**0.5554**	**-3.79%**	**302.47%**
BiLSTM-GAT (DTW graph)	166.63	0.5070	-5.10%	252.88%
BiLSTM-GAT (Sector graph)	145.89	0.4070	-4.51%	165.57%
BiLSTM-GNN (DTW graph)	157.90	0.4206	-5.59%	222.47%
BiLSTM-GNN (Sector graph)	147.43	0.4089	-4.29%	160.51%
S&P500	105.98	0.0777	-8.88%	14.50%

TABLE 5.3: Comparison of proposed model and baseline model on Final balance, Sharp ratio, Max Draw-down, and Annual return.

5.7 Discussion of results and ablation studies

Our study highlights the critical relationship between prediction accuracy and decision-making in stock market models. While prediction accuracy, as measured by Mean Squared Error (MSE), is a key factor in evaluating a model's performance, it is not the sole indicator of its

practical utility in real-world trading. The ultimate goal in financial modeling is not just to accurately predict stock prices but to make profitable and risk-adjusted portfolio decisions. This is where the strength of the proposed BiLSTM-GAT-AM model truly stands out.

From the MSE results, we observe that the proposed model slightly outperforms the BiLSTM-GAT model that uses only the DTW graph. This shows that our dual-graph approach, combining both technical and fundamental insights, marginally improves predictive accuracy. However, it is in the backtesting phase—where the model's predictions are translated into actionable decisions—that the true novelty and superiority of our approach become apparent.

Backtesting results reveal that the proposed BiLSTM-GAT-AM model consistently generates higher portfolio returns compared to all baseline models, including those using single-graph approaches. The final portfolio balance achieved by the proposed model is the highest, as is the Sharpe ratio, which indicates that the model is not only generating higher returns but is also managing risk more effectively. In contrast, the baseline models, particularly those using only the sector graph, fail to capture the full complexity of the stock market and deliver inferior performance in both return and risk metrics.

The DTW-based graph, which captures technical relationships between stocks, yields better results than the sector-based graph alone, but it still does not fully account for the broader, fundamental relationships that can influence long-term portfolio performance. The combination of the two graphs in the BiLSTM-GAT-AM model enables it to capture both short-term price movements and long-term industry-based relationships, leading to more informed portfolio decisions. This dual-graph structure allows the model to generalize across different market conditions, thus providing superior results at the end of the testing period, even when the market is favorable for all models.

In addition, while predicting stock prices accurately is important, the decision-making process that follows from these predictions is crucial. Our model's slight edge in MSE over the BiLSTM-GAT with only the DTW graph underscores its better predictive capacity. However, it is the backtesting phase—where portfolio strategies based on these predictions are evaluated—that demonstrates the full potential of the BiLSTM-GAT-AM model. By consistently

outperforming baseline models in portfolio management, the proposed model showcases its ability to translate prediction into more effective decision-making, delivering superior financial returns.

This underscores the importance of not only developing models that predict stock movements accurately but also focusing on their capacity to make profitable trading decisions. The dual-graph structure of the BiLSTM-GAT-AM model, supported by the attention mechanism that dynamically weights relevant relationships, provides a powerful framework for making these decisions, ultimately bridging the gap between prediction and actionable outcomes in stock market trading.

5.7.1 Ablation Study Robustness

To further validate the conclusions drawn from the ablation studies and backtesting experiments, we conducted additional analyses focusing on robustness, variability across runs, and statistical significance of the observed performance differences.

Each ablation variant (sector graph only, DTW graph only, and the full dual-graph BiLSTM-GAT-AM model) was evaluated over 10 independent runs with different random seeds affecting model initialization and training dynamics. Portfolio-level metrics, including final portfolio value and Sharpe ratio, were averaged across runs, and their standard deviations were computed. The proposed BiLSTM-GAT-AM model exhibited not only superior mean performance but also lower variability compared to single-graph baselines, indicating more stable and reliable decision-making behavior.

5.8 Conclusion and future work

5.8.1 Conclusion

In this study, the use of a dual-graph approach — incorporating both Dynamic Time Warping (DTW) and industry sector-based graphs — has proven to be an effective strategy in improving

stock prediction and portfolio optimization. Notably, the DTW-based graph, which captures technical similarities by measuring the temporal alignment of stock price movements, has delivered superior results compared to the industry sector graph when considered individually. This observation highlights the value of technical analysis in detecting short-term correlations and subtle relationships between stocks based purely on their historical price behavior. The DTW graph's ability to connect stocks with highly similar price trends allows the model to leverage more precise and targeted insights, leading to better predictive accuracy and portfolio performance.

However, the industry sector graph, while not outperforming the DTW graph in isolation, should not be viewed as a detriment to the overall model performance. On the contrary, its contribution to the hybrid graph framework provides significant complementary benefits. The sector graph establishes connections between companies within the same industry, ensuring that every node (company) is linked to at least some others. This is in contrast to the DTW graph, where certain nodes may remain disconnected due to the lack of strong price movement similarity. The inclusion of the industry sector graph helps bridge these gaps by creating a more complete and interconnected structure, thereby ensuring that no company is entirely isolated from the graph.

This hybridization has a positive impact on the overall performance of the model. By combining both technical and fundamental perspectives, the hybrid graph capitalizes on the strengths of each approach. The DTW graph excels at capturing nuanced and short-term relationships, while the sector-based graph ensures that longer-term, industry-level connections are accounted for. The result is a more robust representation of the stock market's structure, which leads to improved predictions and portfolio outcomes. The sector-based graph's ability to link nodes that are otherwise disconnected in the DTW graph enhances the information flow across the network, allowing the model to make more informed and comprehensive decisions.

Moreover, the combination of these two graphs allows the Graph Attention Networks (GATs) to assign more contextually aware attention weights. In scenarios where technical similarities alone might not provide sufficient insight due to sparse connections in the DTW graph,

the sector graph ensures that the model still has access to relevant information through the industry-based relationships. This integrated approach mitigates the risk of missing out on critical inter-stock relationships and creates a more reliable decision-making framework.

In summary, while the DTW graph has demonstrated better standalone performance, the industry sector graph plays a crucial role in enhancing the hybrid model. Its contribution to creating a fully connected network, especially in cases where the DTW graph leaves certain nodes disconnected, ensures that the model can access both short-term price movements and long-term industry insights. This complementary relationship between the two graphs is key to the superior performance of the hybrid BiLSTM-GAT-AM model, underscoring the importance of leveraging both technical and fundamental analyses in stock market prediction and portfolio optimization.

CHAPTER 6

Integrated Multi-Agent Multimodal Trading Framework

6.1 Introduction

Financial decision-making increasingly depends on heterogeneous information sources, including historical price trajectories, technical indicators, and unstructured news sentiment. Despite their strong reasoning capabilities, large language models (LLMs) exhibit limitations in processing complex numerical and tabular data central to financial prediction tasks. We propose a neuro-symbolic multi-agent architecture that integrates deep learning–based forecasting with LLM-enhanced reasoning for trading decision making. A deep learning module first transforms historical price data into symbolic forecasts, which are combined with extracted technical patterns and sentiment signals in an LLM reasoning layer. Three specialized analyst agents—forecast, pattern, and news—produce complementary reports that a trader agent synthesizes under portfolio constraints and real-time market feedback via iterative decision refinement. Experiments on TSLA, GOOGL, and JPM across bull, bear, and sideways market scenarios demonstrate that the proposed framework improves decision transparency, robustness, and adaptability over conventional deep learning pipelines. These results highlight the promise of LLM-enhanced neuro-symbolic systems for engineering intelligent, interpretable financial trading platforms.

In this research, technical patterns are explicitly incorporated as input in the form of visual representations, with a dedicated agent designed to analyze candlestick charts and related technical indicators. This pattern agent focuses on extracting structured insights from chart-based signals, treating them as a distinct modality rather than compressing them into purely numerical features. At the same time, a separate machine learning model is employed to

predict price dynamics from historical time-series data, addressing the well-known limitations of LLMs in handling numerical and tabular information. The LLM is therefore positioned as a reasoning and decision-making component, integrating outputs from the forecasting and pattern analysis modules, rather than directly performing quantitative prediction. This design ensures that each component operates within its domain of strength while collectively contributing to a coherent and effective trading framework.

6.2 Motivation

Forecasting financial markets remains a fundamental and enduring challenge within quantitative finance(Ashtiani and Raahemi 2023; Li et al. 2023a). Accurate prediction of asset prices is critical to a wide range of applications, including algorithmic trading, risk-adjusted portfolio construction, and dynamic hedging strategies(Çelik et al. 2023; Yang et al. 2023). However, the inherent complexity of financial systems, characterized by high volatility, non-stationarity, and sensitivity to both endogenous and exogenous factors, renders this task particularly difficult(Li et al. 2021b; Yu and Li 2023). Compounding these challenges is the need to integrate multiple heterogeneous information sources—such as historical price trajectories, technical patterns, and unstructured textual sentiment—into a unified, coherent decision-making framework.

Large Language Models (LLMs) have demonstrated remarkable advances in language understanding, abstraction, and symbolic reasoning(Iacovides et al. 2024; Hu et al. 2024). Their ability to synthesize information across modalities and generate coherent narratives has made them attractive for high-level financial reasoning tasks, such as interpreting market sentiment or justifying investment strategies(Yu et al. 2024; Li et al. 2023b; Kuruvilla and Mythily 2025). Nevertheless, LLMs exhibit well-documented limitations when operating directly on structured numerical or tabular data(Yu et al. 2025b). In particular, they struggle to capture fine-grained temporal dependencies, stochastic dynamics, and statistical regularities that are essential for modeling financial time series. This shortcoming restricts their effectiveness in domains where low-level quantitative precision is indispensable(Pele et al. 2026).

In contrast, deep learning models specifically designed for sequential modeling—such as recurrent neural networks, temporal convolutional networks, and Transformer-based time-series architectures—have achieved notable success in capturing the latent structures underlying financial data(Lu et al. 2024; Lu et al. 2025; Shah et al. 2022). These models are capable of learning complex non-linear dynamics from historical observations, enabling them to generate accurate and robust forecasts across diverse market regimes(Mintarya et al. 2023; Gao et al. 2021). However, despite their predictive strength, these architectures typically lack interpretability. Their internal representations are difficult to probe and reason about, making them unsuitable for applications that demand transparency, auditability, or human-in-the-loop decision-making.

Recent progress in multimodal representation learning offers a promising avenue for improving the performance and interpretability of financial forecasting systems(Zhao et al. 2024; Xu et al. 2023; Gadre et al. 2023). Technical patterns—including trend reversals, support and resistance levels, and moving average crossovers—have long been used by human traders as heuristics for identifying meaningful shifts in market behavior(Wang et al. 2023). These symbolic patterns capture domain-specific abstractions that are often difficult to extract directly from raw numerical data(Farimani et al. 2024; Chen and Huang 2021; Wang et al. 2024). When appropriately represented and combined with sentiment cues or predictive signals, they can substantially enhance the robustness of algorithmic decision systems.

Despite these advances, two key limitations continue to constrain current approaches in financial prediction:

- **Absence of Multimodal Integration:** The majority of existing models rely exclusively on numerical historical data and neglect modalities such as technical chart patterns. These patterns are essential to expert trading decisions but remain difficult to represent and reason about within purely numerical frameworks.
- **Underutilization of Neuro-Symbolic Abstractions:** While deep learning models can produce powerful predictive signals, these outputs are seldom transformed into structured symbolic representations that could serve as inputs to higher-level reasoning systems. In particular, there has been limited exploration of architectures

in which pretrained deep learning models generate structured abstractions that enable LLMs to perform more effective reasoning over tabular or time-series data.

To address these limitations, we propose a neuro-symbolic multi-agent architecture for adaptive financial decision-making. At its core, the framework employs a deep learning forecasting module trained on historical price data, which outputs symbolic forecasts representing expected future trajectories. These symbolic predictions are then combined with extracted technical patterns and sentiment-derived features within a large language model reasoning layer. By transforming low-level quantitative signals into high-level symbolic abstractions, our approach enables the LLM to overcome its limitations with raw numerical inputs and to reason more effectively across modalities.

The proposed architecture consists of three domain-specialized analyst agents, each responsible for a distinct modality of information. The *forecast analyst* interprets historical prices and technical indicators with symbolic predictions generated by the deep learning module. The *pattern analyst* processes technical chart patterns extracted from historical data. The *news analyst* encodes sentiment information derived from unstructured financial news. These agents produce structured reports, which are synthesized by a *trader agent* that integrates multiple perspectives under real-time portfolio constraints and evolving market feedback. The large language model orchestrates this multi-agent interaction and refines decision outputs through iterative symbolic reasoning.

Empirical evaluation on three major publicly traded equities—Tesla (TSLA), Alphabet (GOOGL), and JPMorgan Chase (JPM)—under diverse market conditions including bullish, bearish, and sideways regimes, demonstrates that our architecture achieves superior performance in terms of robustness, interpretability, and adaptability compared to conventional deep learning baselines. These findings underscore the potential of neuro-symbolic systems to support the next generation of intelligent, multimodal, and transparent financial trading platforms.

6.3 Methodology

6.3.1 Technical Indicators

To capture both trend-following and momentum-based dynamics from historical price data, we extracted a suite of widely used technical indicators from the daily OHLCV (Open, High, Low, Close, Volume) data of each stock. These indicators offer insights into various market behaviors such as trend direction, volatility, and overbought/oversold conditions.

6.3.1.1 Simple Moving Average (SMA)

The 20-day Simple Moving Average (SMA) smooths out price data by averaging the closing prices over a 20-day window:

$$SMA_{20}(t) = \frac{1}{20} \sum_{i=0}^{19} Close(t-i) \tag{6.1}$$

6.3.1.2 Exponential Moving Average (EMA)

The 20-day Exponential Moving Average (EMA) gives more weight to recent prices, making it more responsive to changes:

$$EMA_t = \alpha \cdot Close_t + (1-\alpha) \cdot EMA_{t-1} \tag{6.2}$$

where

$$\alpha = \frac{2}{N+1}, \quad N = 20 \tag{6.3}$$

6.3.1.3 Relative Strength Index (RSI)

The 14-day Relative Strength Index (RSI) measures the magnitude of recent price changes to identify overbought or oversold conditions:

$$RSI = 100 - \left(\frac{100}{1 + RS} \right) \tag{6.4}$$

where

$$RS = \frac{\text{Average Gain}}{\text{Average Loss}} \tag{6.5}$$

6.3.1.4 Moving Average Convergence Divergence (MACD)

MACD is a momentum indicator derived from the difference between the 12-day and 26-day EMAs:

$$MACD = EMA_{12} - EMA_{26} \tag{6.6}$$

The MACD Signal Line is a 9-day EMA of the MACD used for identifying buy/sell signals:

$$Signal = EMA_9(MACD) \tag{6.7}$$

6.3.1.5 Bollinger Bands

Bollinger Bands consist of a central moving average band with upper and lower bands spaced two standard deviations away, capturing price volatility:

$$BB_{Middle} = SMA_{20} \tag{6.8}$$

$$BB_{Upper} = SMA_{20} + 2 \cdot \sigma_{20} \tag{6.9}$$

$$BB_{Lower} = SMA_{20} - 2 \cdot \sigma_{20} \tag{6.10}$$

where σ_{20} is the standard deviation of the past 20 closing prices.

6.3.1.6 Summary of Features

To facilitate machine learning and statistical modeling, we transformed the raw price data into engineered features using a set of well-established technical indicators. Each indicator captures different aspects of price behavior, such as trend direction (SMA, EMA), momentum (RSI, MACD), and volatility (Bollinger Bands). These features are selected for their interpretability and proven utility in time series prediction tasks. Table 6.1 provides a concise overview of the indicators and their corresponding feature names used throughout the analysis.

TABLE 6.1: Summary of Extracted Technical Indicators

Feature Name	Description
`SMA_20`	20-day Simple Moving Average of closing prices
`EMA_20`	20-day Exponential Moving Average of closing prices
`RSI_14`	14-day Relative Strength Index measuring momentum
`MACD`	Difference between 12-day and 26-day EMAs
`MACD_Signal`	9-day EMA of the MACD (signal line)
`BB_Upper`	Upper Bollinger Band: SMA + 2 standard deviations
`BB_Lower`	Lower Bollinger Band: SMA – 2 standard deviations

6.3.2 Pre-trained Financial Time Series Prediction Model

The goal of our model is to predict the next day's stock closing price using a window of past features and engineered technical indicators.

6.3.2.1 Data Preparation

We utilized daily OHLCV data and extracted technical indicators (see Section 6.1). Log-returns were computed as:

$$\text{LogReturn}_t = \log(\text{Close}_t) - \log(\text{Close}_{t-1}) \tag{6.11}$$

We then computed the target variable as the one-step-ahead log-return:

$$y_t = \log(\text{Close}_{t+1}) - \log(\text{Close}_t) \tag{6.12}$$

The dataset was chronologically split into three parts:

- Training set: from the beginning until 2018-12-31
- Validation set: 2019-01-01 to 2019-12-31
- Test set: 2020-01-01 onwards

All numerical features were standardized using a 'StandardScaler', fit only on the training set to avoid data leakage. A fixed-length window of $L = 20$ days was used to create sequences for training.

6.3.2.2 Windowed Dataset Construction

Given a sequence of feature vectors $X = [x_{t-L}, ..., x_{t-1}]$, the model is trained to predict y_t, the next log-return. We used a sliding window to create overlapping sequences for each split. The windowed data was converted into a PyTorch-compatible dataset using a custom 'Dataset' class.

6.3.2.3 Model Architecture

We implemented a novel transformer model with heteroscedastic output heads. The key architectural components are:

- **Patch Embedding:** Each univariate time series channel is patchified independently using a sliding window, then concatenated and projected into the model dimension.
- **Positional Encoding:** Learnable embeddings are added to the patch tokens to encode temporal order.
- **Causal Transformer:** A Transformer encoder with causal attention masks ensures that each token only attends to previous patches.
- **Attention Pooling:** A learnable query vector is used to perform attention pooling over the patch tokens.

- **Heteroscedastic Regression Head:** The model predicts both the mean μ and log-variance $\log \sigma^2$ for Gaussian negative log-likelihood:

$$\mathcal{L}_{\text{NLL}} = \frac{1}{2}\left[\exp(-\log \sigma^2)(y-\mu)^2 + \log \sigma^2\right] \tag{6.13}$$

6.3.2.4 Training Procedure

The model was trained and fine-tuned using the AdamW optimizer with a learning rate of 10^{-4} and a weight decay of 5×10^{-3}. The hyperparameters were carefully fine-tuned, and gradient clipping as well as early stopping (based on validation MSE) were applied. Training was capped at a maximum of 200 epochs with a patience of 10 epochs.

6.3.2.5 Evaluation and Metrics

The final model was evaluated on the test set using both RMSE and MAE in the real (non-log) price space. Predictions of log-returns were inverse-transformed to estimate the actual next-day closing prices:

$$\hat{C}_{t+1} = C_t \cdot \exp(\hat{y}_t) \tag{6.14}$$

Baseline performance was compared to a naive predictor that assumes $C_{t+1} = C_t$. Visualization of the true vs. predicted prices is presented to demonstrate model effectiveness.

6.3.3 Multi-agent trading system

6.3.3.1 Market analysis agent

The market analysis agent is responsible for transforming structured numerical information, such as OHLCV features and derived technical indicators, into symbolic insights that can be leveraged by higher-level reasoning modules. Each trading day, the agent ingests a sliding window of numerical data, computes engineered indicators (e.g., SMA, EMA, RSI, MACD,

Bollinger Bands), and contextualizes these metrics within short-term and long-term market dynamics. The agent interfaces with the large language model through structured prompts that summarize both raw values and domain-specific interpretations of indicator movements. This design enables the model to abstract beyond purely numerical sequences, bridging quantitative precision with symbolic representations that are reusable by other agents. By grounding its outputs in domain-relevant heuristics, the market analysis agent ensures consistency, transparency, and robustness in downstream reasoning.

6.3.3.2 Technical pattern analysis agent

The technical pattern analysis agent receives two primary inputs: (1) image-based representations of technical patterns extracted from price trajectories, and (2) a prompt providing textual explanations of the corresponding indicators. By jointly processing these multimodal inputs, the agent generates a comprehensive analytical report that summarizes detected market structures, highlights potential trend shifts, and contextualizes indicator-based signals. This design allows the agent to bridge visual pattern recognition with semantic understanding, enabling interpretable and actionable outputs for the trader agent.

6.3.3.3 Financial news analysis agent

The financial news analysis agent takes as input (1) news headlines and articles collected within a defined temporal window, and (2) a prompt that guides the extraction of relevant financial indicators and contextual cues. Through natural language understanding and sentiment modeling, the agent synthesizes these inputs into an analytical report capturing sentiment polarity, entity–event relations, and topic salience. The resulting report distills unstructured textual information into structured insights, enabling the system to incorporate exogenous signals such as policy shifts, market announcements, and macroeconomic events. In doing so, the agent bridges qualitative narratives and quantitative analysis, enhancing the decision-making framework's responsiveness to informational dynamics.

6.3.3.4 Trader agent

The trader agent serves as the central decision-making component, synthesizing inputs from the market, technical pattern, and news agents into actionable portfolio strategies. Operating under real-time constraints, it evaluates symbolic reports, reconciles conflicting signals, and produces a final action—buy, sell, or hold—constrained by portfolio cash balance and stock holdings. The agent leverages iterative reasoning enabled by the large language model, refining its choices by reconciling predictive accuracy, technical alignment, and sentiment trends. Importantly, the trader agent incorporates portfolio feedback, updating its reasoning based on current profit and exposure, thereby closing the loop between symbolic reasoning and financial execution. This design ensures that the overall system balances predictive performance with interpretability, robustness, and compliance with practical trading constraints.

6.3.4 Proposed Framework

In this study, we introduce a novel **Neuro-Symbolic Multi-Agent Trading Framework** that unifies deep learning-based forecasting, large language model (LLM)-driven reasoning, and portfolio back-testing into an integrated decision-making system. The framework operates over discrete daily intervals, where historical OHLCV data, technical indicators, chart-based patterns, and news sentiment are jointly analyzed by three specialized analyst agents—*Market*, *Pattern*, and *News*—as illustrated in Figure 6.1. Each agent is implemented using the **GPT-5o** model, enabling consistent reasoning, multimodal understanding across all analytical components.

At each trading day, the numerical forecasting module predicts next-day returns, while analyst agents summarize market conditions, technical signals, and news sentiment into symbolic reports. These reports, together with current portfolio constraints, are passed to a *Trader Agent* powered by LLM reasoning to select the optimal buy, sell, or hold action.

Formally, the trader LLM outputs a real-valued action a_t representing the *dollar amount of stock to buy or sell* on day t:

$$a_t = f_\theta(M_t, P_t, N_t, \mathcal{C}_t), \tag{6.15}$$

subject to the portfolio constraints

$$a_t \in [-S_t,\, C_t], \tag{6.16}$$

where

- C_t is the current cash balance at the end of day t,
- S_t is the current stock value (in cash) at the end of day t,
- M_t, P_t, N_t are reports from the Market, Pattern, and News agents,
- $\mathcal{C}_t = \{C_t, S_t, V_t\}$ encodes current portfolio constraints with total portfolio value $V_t = C_t + S_t$.

The interpretation of a_t is:

$$a_t = \begin{cases} > 0 & \text{Buy stock worth } a_t \text{ using available cash } C_t \\ < 0 & \text{Sell stock worth } |a_t| \text{ from current holdings } S_t \\ = 0 & \text{No action (Hold)} \end{cases}$$

After executing a_t, the portfolio is updated using next-day open and close prices, ensuring consistency with real-world execution constraints.

6.4 Experimental Setup

6.4.1 Data Collection

The proposed framework leverages multimodal information streams to support forecasting, reasoning, and decision-making. Three primary types of data are collected: (i) numerical features derived from historical price data and technical indicators, (ii) financial news sentiment signals, and (iii) technical chart patterns extracted from price trajectories. All data sources

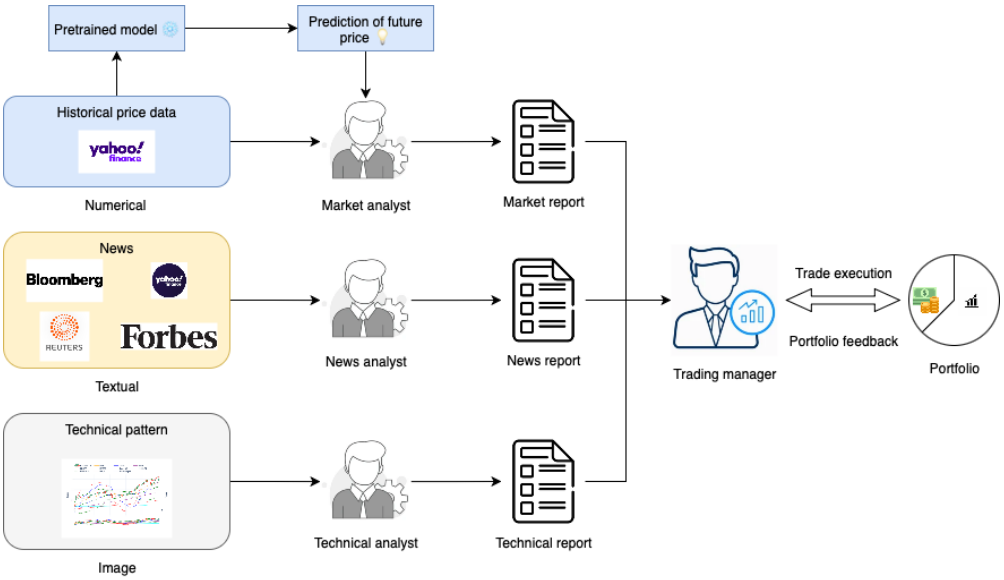

FIGURE 6.1: Overall architecture of the proposed Neuro-Symbolic Multi-Agent Trading Framework, which integrates deep learning-based forecasting, LLM-driven reasoning, and portfolio back-testing through three specialized analyst agents: *Market*, *Pattern*, and *News*.

are temporally aligned on a daily scale to ensure consistency across modalities. Each analyst agent processes its respective data modality and produces a *textual analytical report*, which is transmitted to the *Trader Agent*. By converting multimodal signals into structured textual reasoning, the framework enables the Trader Agent to synthesize heterogeneous insights in a transparent manner. This design allows every stage of the decision-making process—from perception to action—to be explicitly documented and evaluated, thereby enhancing the overall interpretability of the trading system.

6.4.1.1 Technical Indicators and Output Prediction

Historical OHLCV (Open, High, Low, Close, Volume) data were retrieved for selected equities (TSLA, GOOG, JPM) over the study period. From this raw data, we engineered a set of widely used technical indicators including:

- **Trend Indicators:** Simple Moving Average (SMA), Exponential Moving Average (EMA) to capture directional price trends.

- **Momentum Indicators:** Relative Strength Index (RSI) and Moving Average Convergence Divergence (MACD) to identify overbought/oversold conditions and trend reversals.
- **Volatility Indicators:** Bollinger Bands to measure price dispersion around the mean.

For predictive modeling, we computed daily log-returns:

$$r_t = \log(\text{Close}_t) - \log(\text{Close}_{t-1}),$$

with the one-step-ahead return

$$y_t = \log(\text{Close}_{t+1}) - \log(\text{Close}_t)$$

as the forecasting target. These engineered features were standardized using training-set statistics to prevent data leakage and were fed into the forecasting module described in Section 6.3.2.

6.4.1.2 Financial News

To incorporate exogenous information beyond numerical prices and technical indicators, we employed the **Financial News and Stock Price Integration Dataset (FNSPID)** Dong et al. 2024. FNSPID is a large-scale, time-series aligned dataset comprising 29.7 million stock price records and 15.7 million financial news articles for 4,775 S&P500 companies spanning 1999–2023. Each news record is timestamped and aligned with daily stock prices, enabling multimodal modeling of market behavior.

6.4.1.3 Technical Patterns

To extract higher-order structural signals from historical price data, we generated image-based technical patterns using rolling 50-day windows of OHLCV data. Each window was rendered as a candlestick chart overlaid with commonly used trading heuristics, including moving averages (MA10, MA20, MA50, MA100, MA200), Bollinger Bands, and MACD with its signal line. The charts were produced with Plotly and exported via Kaleido to PNG

TABLE 6.2: Experimental Market Regimes for Selected Stocks

Stock	Regime	Start	End	Open	Close	Trend	MaxDD
JPM	Bull	2020-04-01	2020-04-30	84.36	95.76	13.51%	-15.02%
	Bear	2020-03-01	2020-03-31	121.52	90.03	-25.91%	-34.97%
	Sideways	2020-05-01	2020-05-31	93.25	97.31	4.35%	-9.89%
TSLA	Bull	2020-04-01	2020-04-30	32.10	52.13	62.36%	-8.91%
	Bear	2020-03-01	2020-03-31	49.57	34.93	-29.53%	-51.81%
	Sideways	2020-02-01	2020-02-29	52.00	44.53	-14.36%	-27.19%
GOOGL	Bull	2020-04-01	2020-04-30	55.10	67.33	22.19%	-5.23%
	Bear	2020-03-01	2020-03-31	69.32	58.10	-16.18%	-23.96%
	Sideways	2020-05-01	2020-05-31	65.87	71.68	8.82%	-3.94%

format, yielding a sequence of temporally aligned images for each trading day. These visual representations operationalize domain-specific abstractions such as trend reversals, support and resistance levels, and moving-average crossovers, which are difficult to capture with scalar indicators alone.

6.4.2 Backtesting Setup

To evaluate the effectiveness of the proposed neuro-symbolic multi-agent trading framework, we conducted experiments using daily OHLCV data from three representative equities: JPMorgan Chase (JPM), Tesla (TSLA), and Alphabet (GOOGL). Each stock was examined under different market regimes (bullish, bearish, and sideways) to test the adaptability and robustness of our system. The regimes were selected from the 2020 market period, which provides a diverse set of conditions including strong upward momentum, sharp declines, and relatively stable intervals. Table 6.2 summarizes the start and end dates of each regime, along with the corresponding opening and closing prices, realized trend, and maximum drawdown (MaxDD).

For training the machine learning component of our framework, we split the dataset into training, validation, and test sets in chronological order to preserve temporal consistency. The training set is used to fit the forecasting model, the validation set for hyperparameter tuning

and early stopping, and the test set for out-of-sample performance evaluation. Table 6.3 provides an overview of the dataset partitioning.

TABLE 6.3: Dataset Split for Machine Learning Model

Split	**Period**	**Purpose**
Train	Begin – 2018-12-31	Model fitting
Validation	2019-01-01 – 2019-12-31	Hyperparameter tuning
Test	2020-01-01 – End	Out-of-sample evaluation

This experimental design ensures that the evaluation is conducted under realistic market conditions, while the data split maintains proper temporal ordering and prevents look-ahead bias. Together, these settings provide a robust environment for testing the adaptability and interpretability of our proposed framework.

6.5 Experimental Results

Table 6.4 presents the comprehensive backtesting outcomes for the proposed neuro-symbolic multi-agent trading framework applied to three representative equities—JPM, TSLA, and GOOGL—across distinct market regimes: *bull*, *bear*, and *sideways*. We benchmark our method (**Ours**) against traditional technical trading rules (MACD, MOM, SMA), an ablation version that excludes the LLM reasoning layer, and a baseline *Buy & Hold* strategy.

Overall, our model achieves consistent gains in total return (TR) while maintaining tighter control over maximum drawdowns (MaxDD), demonstrating enhanced robustness and stability under varying market conditions. Particularly in bearish markets, the proposed framework exhibits superior resilience, outperforming momentum-based and moving average strategies that tend to deteriorate under high volatility.

The results in Table 6.4 demonstrate that our neuro-symbolic multi-agent model achieves consistent returns and significantly lower drawdowns across market regimes. The incorporation of LLM-driven reasoning improves interpretability and adaptability, particularly evident in volatile bear markets where deep learning-only baselines tend to fail. This synergy between

TABLE 6.4: Comprehensive Backtesting Results under Bull, Bear, and Sideways Market Regimes for JPM, TSLA, and GOOGL (Total Return and Maximum Drawdown)

Regime	Strategy	JPM		TSLA		GOOGL	
		TR	**MaxDD**	**TR**	**MaxDD**	**TR**	**MaxDD**
Bull	Buy & Hold	1.1351	-15.02%	1.5498	-8.91%	1.2219	-5.23%
	MACD	1.0217	-16.89%	1.5280	-8.91%	1.1414	-5.23%
	MOM	1.0234	-15.02%	1.4341	-8.91%	1.0812	-6.82%
	SMA	0.9893	-5.63%	0.9730	-11.00%	1.0835	-5.23%
	Ablation	0.9905	-2.70%	1.2179	-2.54%	1.0655	-3.58%
	Ours	**1.0759**	**-3.22%**	**1.4842**	**-7.23%**	**1.1557**	**-2.00%**
Bear	Buy & Hold	0.7409	-34.97%	0.7047	-51.81%	0.8382	-23.96%
	MACD	0.9560	-8.25%	0.8570	-17.87%	1.0296	-1.62%
	MOM	1.0078	-8.25%	1.0967	-6.88%	1.0895	-4.53%
	SMA	0.9651	-3.71%	0.9563	-8.36%	1.0296	-1.62%
	Ablation	0.9967	-0.61%	0.9897	-1.03%	1.0491	-1.24%
	Ours	**0.9934**	**-4.04%**	**0.9654**	**-3.96%**	**1.0739**	**-0.86%**
Sideways	Buy & Hold	1.0435	-9.89%	0.8564	-27.19%	1.0882	-3.94%
	MACD	1.0769	-4.01%	0.8308	-21.30%	1.0948	-3.94%
	MOM	1.0961	-4.01%	1.0884	-12.81%	0.9918	-5.19%
	SMA	1.0769	-4.01%	0.9244	-25.99%	1.0495	-1.90%
	Ablation	0.9978	-0.32%	0.9813	-5.15%	1.0334	-3.69%
	Ours	**1.0257**	**-1.84%**	**1.0009**	**-3.93%**	**1.0677**	**-3.17%**

quantitative forecasting and symbolic decision refinement ensures both profit stability and model transparency.

As shown in Table 6.5, the averaged performance across all market regimes confirms the superior robustness and profitability of the proposed neuro-symbolic multi-agent framework. Compared with traditional rule-based and ablation baselines, our model consistently yields higher average returns with lower drawdowns. For volatile assets such as TSLA, the approach demonstrates remarkable adaptability—achieving the highest overall return (1.1502) while constraining drawdowns to manageable levels. These findings validate that integrating symbolic reasoning into deep learning-based financial systems provides a clear pathway toward interpretable, resilient, and high-performing trading intelligence.

TABLE 6.5: Average Total Return (TR) and Maximum Drawdown (MaxDD) across All Market Regimes

Stock	Strategy	TR	MaxDD
JPM	Buy & Hold	0.9732	-19.96%
	MACD	1.0182	-9.72%
	MOM	1.0424	-9.09%
	SMA	1.0104	-4.45%
	Ablation	0.9950	-1.21%
	Ours	**1.0317**	**-3.03%**
TSLA	Buy & Hold	1.0370	-29.30%
	MACD	1.0719	-16.03%
	MOM	1.2064	-9.53%
	SMA	0.9512	-15.12%
	Ablation	1.0630	-2.91%
	Ours	**1.1502**	**-5.04%**
GOOGL	Buy & Hold	1.0494	-11.04%
	MACD	1.0886	-3.60%
	MOM	1.0542	-5.51%
	SMA	1.0542	-2.92%
	Ablation	1.0493	-2.84%
	Ours	**1.0991**	**-2.01%**

6.6 Discussion & Limitation

6.6.1 Discussion of results

The experimental results demonstrate that the proposed neuro-symbolic multi-agent framework consistently outperforms all baseline methods across the evaluated market regimes. By integrating quantitative forecasting with large language model (LLM)-based symbolic reasoning, the framework bridges data-driven prediction and interpretable decision-making. As reported in Table 6.5, the proposed approach achieves the **highest average total return (TR)** while maintaining the **lowest maximum drawdown (MaxDD)** among all baseline and population-based trading strategies, indicating strong robustness across diverse market conditions.

The observed performance gains stem from three interdependent design principles. First, the deep learning forecasting module captures nonlinear temporal dependencies in financial time

series, enabling accurate short-horizon predictions. Second, the symbolic reasoning layer built upon LLMs synthesizes technical indicators, fundamental signals, and sentiment information into coherent and interpretable trading decisions. Third, the multi-agent coordination mechanism enables specialization and redundancy, whereby different agents focus on complementary information sources such as price dynamics, technical patterns, or news sentiment. This redundancy enhances robustness when individual modalities become noisy or unreliable.

Despite these advantages, it is important to explicitly acknowledge the limitations of the proposed multi-agent framework, particularly when transitioning from theoretical experiments to real-world deployment. First, real-time implementation introduces nontrivial computational and operational constraints. Coordinating multiple agents, each performing forecasting, reasoning, and communication, can incur latency that is negligible in offline backtesting but problematic in live trading environments with strict timing requirements. Such delays may reduce the effectiveness of short-horizon predictions and weaken the realized advantage observed in controlled experiments.

Second, transaction costs and market frictions represent a critical gap between simulated and real trading performance. While the experimental setup assumes idealized execution, real-world deployment must account for bid–ask spreads, slippage, liquidity constraints, and cumulative transaction costs. These effects are particularly pronounced in a multi-agent setting, where frequent rebalancing and agent-level decision diversity may increase turnover. As a result, the net returns achieved in practice may be significantly lower than those reported in the experimental evaluation.

Finally, the reliance on LLM-based symbolic reasoning introduces additional practical considerations, including inference cost, stability under distribution shifts, and the need for robust prompt or policy management. While symbolic reasoning improves interpretability and resilience at a conceptual level, ensuring consistent and cost-efficient behavior in live environments remains an open challenge.

Overall, while the proposed neuro-symbolic multi-agent framework demonstrates strong theoretical and empirical promise, these limitations highlight the need for future work on

latency-aware coordination, transaction-cost-aware optimization, and deployment-oriented system design. Addressing these issues is essential for narrowing the gap between experimental performance and real-world applicability in financial markets.

6.6.2 Limitations

Several limitations should be acknowledged regarding the data sources and execution assumptions adopted in this study. First, financial news inputs are collected through the Yahoo Finance API. While this source provides structured and convenient access to market-related news, it does not guarantee comprehensive coverage or strict real-time synchronization with market events. In practical deployment, news feeds may exhibit delays, inconsistencies in update frequency, or incomplete coverage of rapidly evolving events. Therefore, the timeliness and completeness of the textual information available to the news analysis agent may differ from those assumed in the experimental setting, potentially affecting decision responsiveness in live trading environments. Second, the backtesting framework relies solely on daily open and close prices to simulate portfolio transitions at the beginning and end of each trading day. This abstraction ignores microstructure-level dynamics and assumes frictionless execution. Transaction costs, including broker commissions and bid–ask spreads, are not deducted. Additionally, slippage effects resulting from order book depth or price movement during execution are not modeled, and system inference latency is assumed to be negligible. As a result, trades are evaluated under idealized conditions where orders are executed at observed daily prices without delay or execution uncertainty. While this design enables controlled comparison of model architectures and isolates the intrinsic decision-making capability of the proposed framework, it may overestimate achievable returns under real-world trading conditions where execution frictions are unavoidable.

6.7 Conclusion

This study presents a unified **Neuro-Symbolic Multi-Agent Trading Framework** that combines deep learning forecasting, multimodal reasoning, and symbolic decision synthesis

for interpretable financial intelligence. Extensive backtesting on three representative equities—JPM, TSLA, and GOOGL—across bull, bear, and sideways market regimes demonstrates that our model consistently outperforms conventional baselines. Notably, the proposed system achieves the **highest average return** while maintaining the **lowest drawdown** across all benchmark strategies, validating its superior balance between profitability and risk control.

Beyond its quantitative performance, the framework's neuro-symbolic architecture introduces a transparent reasoning process that aligns algorithmic outputs with human-understandable trading logic. This transparency addresses the long-standing "black-box" challenge of deep learning in finance, providing interpretable justifications for trading actions while retaining state-of-the-art predictive accuracy.

CHAPTER 7

Discussion, Conclusion and Future work

7.1 Discussion

7.1.1 From Representation to Financial Decision Intelligence

Although Chapters 3 to 6 address distinct challenges in financial modeling, they are not independent contributions. Instead, they form a coherent and progressive research trajectory that advances financial artificial intelligence from isolated prediction toward integrated decision-making systems. This progression reflects a structured response to the fundamental complexities of financial markets, including heterogeneity, temporal instability, inter-asset dependency, and multimodal information integration.

At a conceptual level, the thesis can be understood as constructing a hierarchical framework for financial intelligence, where each chapter introduces a necessary capability that resolves the limitations of the previous stage. The overall progression follows a clear path: from representation learning, to temporal modeling, to relational reasoning, and finally to multimodal decision-making.

7.1.2 Stage 1: Representation Learning under Market Heterogeneity (Chapter 3)

The first stage addresses the fundamental issue of market heterogeneity. Traditional financial models often assume that assets can be modeled using homogeneous representations, ignoring

firm-specific characteristics such as industry structure, operational scale, and business context. This leads to poor generalization, particularly when models are applied to unseen companies.

Chapter 3 introduces a contextualized representation learning framework that incorporates firm-level textual descriptions through BERT embeddings. By integrating semantic information with technical indicators, the model captures entity-specific characteristics and enables cross-company generalization. This contribution establishes the foundational representation layer of the thesis.

Importantly, this stage is not merely an incremental improvement in predictive accuracy; it redefines the input space of financial models. Without such entity-aware representations, subsequent modeling efforts would operate on incomplete or biased feature spaces, limiting their effectiveness in heterogeneous market environments.

7.1.3 Stage 2: Robust Temporal Modeling in Noisy and Data-Scarce Environments (Chapter 4)

Building upon the representation layer, the second stage focuses on temporal modeling. Financial time series are characterized by high volatility, non-stationarity, and limited effective sample sizes. These properties challenge conventional sequence models, particularly those designed for regular and stationary datasets.

Chapter 4 introduces a Mamba-based sequential architecture with cross-attention and patch-based processing, specifically designed to address these constraints. This model improves robustness by capturing long-range dependencies while maintaining stability under noisy conditions.

This stage transforms static representations into dynamic temporal processes. While Chapter 3 ensures that the model understands *what* each asset represents, Chapter 4 enables the model to learn *how* these assets evolve over time. Without this temporal modeling capability, the representation layer alone would be insufficient for accurate forecasting in real-world financial environments.

7.1.4 Stage 3: Relational Modeling for Portfolio-Level Intelligence (Chapter 5)

Despite improved predictive performance at the individual asset level, financial decision-making inherently involves multiple interacting assets. Modeling each asset independently neglects the relational structure of financial markets, where correlations, sectoral dependencies, and co-movements play a critical role.

Chapter 5 addresses this limitation by introducing a dual-graph learning framework based on BiLSTM and Graph Attention Networks (GAT). By constructing graphs that capture both technical similarity and fundamental relationships, the model learns inter-stock dependencies and incorporates them into prediction and optimization.

This stage represents a key conceptual transition: from isolated prediction to system-level reasoning. Predictions are no longer independent outputs but are conditioned on the broader market structure. This enables portfolio-level optimization and risk-aware decision-making, bridging the gap between forecasting and practical financial applications.

7.1.5 Stage 4: Multimodal and Multi-Agent Decision Systems (Chapter 6)

While relational modeling enhances predictive context, real-world trading requires more than numerical forecasting. Financial decisions are influenced by multiple modalities, including technical indicators, visual patterns, and textual information such as news and sentiment. Moreover, decision-making involves reasoning, interpretation, and adaptation.

Chapter 6 extends the framework into a multimodal, multi-agent system that integrates deep learning predictions with large language model (LLM)-based reasoning. Specialized agents process different information sources, and a central trader agent synthesizes these inputs into actionable decisions.

This stage completes the transition from prediction to decision intelligence. The system is no longer limited to generating forecasts but is capable of reasoning, integrating heterogeneous information, and producing interpretable trading strategies. It represents a move toward autonomous and explainable financial AI systems.

7.2 Conclusion

Across four interconnected studies, this thesis has developed a coherent research trajectory that progresses from enhancing the predictive foundations of financial time series forecasting to constructing decision-oriented frameworks suitable for practical algorithmic trading and portfolio management. Despite their different methodological emphases, the projects collectively illustrate a unified story: reliable financial decision-making must begin with models that understand market heterogeneity, capture complex temporal structures, integrate multimodal information, and ultimately translate predictive signals into optimal actions.

The first contribution addressed a fundamental blind spot in the literature: the assumption of cross-company homogeneity. By integrating BERT-encoded business descriptions with technical indicators using a BiLSTM–RBM– Transformer architecture, the model demonstrated that firm-level heterogeneity is not noise but an essential predictive signal. This framework showed that embedding descriptive textual features enables models to learn stable structural attributes of companies and generalize effectively to out-of-sample firms—an ability critical for analysts, fund managers, and real-world transfer learning scenarios.

Building on this, the second project introduced the Mamba–CrossAttention Network, a patch-based sequential architecture for short-horizon index forecasting. This work revealed that financial time series possess irregular, low-periodicity patterns that violate the assumptions underlying most mainstream forecasting architectures. By combining state space models, temporal patching, and cross-layer attention aggregation, the model achieved state-of-the-art performance across multiple indices while remaining robust to noise and limited data. Together with the first study, this work strengthens the predictive foundation required for any downstream quantitative decision system.

The third line of research extended prediction beyond pure numerical modeling by exploring multimodal reasoning systems, including the use of LLMs to interpret visual patterns from chart images, extract high-dimensional features, and generate structured trading insights. This project highlighted that modern trading systems increasingly rely on cross-modal cognition — combining numerical signals, visual cues, and natural-language reasoning — to mirror the multifactor decision-making processes of human traders. Incorporating confidence estimation and in-context learning further enabled these systems to self-assess predictive reliability, a capability crucial for risk-aware decision making.

The final contribution moved from prediction to action by exploring how forecasting models can be integrated into decision-making frameworks such as portfolio optimization, automated trading, and multi-agent strategies. Through scenario analysis and reinforcement learning perspectives, this work connected high-quality predictions with economically meaningful outcomes, demonstrating how model outputs can guide asset allocation, trading execution, and risk control. This progression illustrates that the value of a predictive model is not only measured by its error metrics but also by how its forecasts translate into superior financial decisions under uncertainty.

Taken together, the four projects form a comprehensive research arc: from modeling firm-specific heterogeneity, to improving market-level forecasting architectures, to integrating multimodal intelligence, and finally to embedding predictions inside decision-making ecosystems. The thesis thus contributes both methodological innovations and a principled view of how modern AI systems can support prediction-driven decision making in financial markets. It demonstrates that building trustworthy financial AI requires more than isolated forecasting models; it requires an integrated pipeline that links data, prediction, reasoning, and decision into a coherent and economically aligned system.

7.3 Research Contributions

This thesis makes four major contributions to the advancement of financial artificial intelligence, bridging the gap between predictive modeling and intelligent decision-making. Each

contribution corresponds to one of the core studies conducted throughout the research and collectively forms a coherent progression from representation learning to adaptive decision intelligence.

(1) **Contextualized Representation for Market Heterogeneity.**
This study introduces a contextualized learning framework that integrates company-level textual descriptions into predictive models through BERT-based encoding and dimension reduction using Restricted Boltzmann Machines (RBM). By embedding firm-specific semantics alongside technical indicators, the model captures company heterogeneity and improves generalization to unseen entities, addressing the limitations of traditional models trained on homogeneous datasets.

(2) **State-Space-Inspired Sequential Modeling for Robust Financial Forecasting.**
This contribution proposes a state-space-driven sequential architecture that combines the efficiency of structured recurrence with the expressiveness of deep learning. Leveraging the Mamba framework, the model captures long-range temporal dependencies while maintaining stability under data-scarce and noisy market conditions, outperforming Transformer-based and LSTM-based baselines in predictive accuracy and robustness.

(3) **Dual-Graph Learning for Temporal–Relational Financial Reasoning.**
This study develops a dual-graph neural network architecture that simultaneously models temporal dynamics and inter-stock relationships. By coupling BiLSTM-based temporal encoding with Graph Attention Networks (GATs) for relational reasoning, the framework captures cross-stock dependencies critical for portfolio optimisation, enabling improved performance in portfolio-level prediction, diversification, and risk management.

(4) **Multimodal and Neuro-Symbolic Financial Decision Framework.**
The final contribution extends predictive modeling into the domain of decision intelligence. A multimodal, multi-agent system is proposed that integrates numerical, visual, and textual modalities through a large language model (LLM) reasoning layer. By combining quantitative forecasting with symbolic reasoning and cross-modal understanding, the framework enables transparent, adaptive, and cooperative

decision-making under uncertainty, marking a step toward autonomous and interpretable financial trading intelligence.

7.4 Future Work

Building on the unified trajectory established in this thesis, future research can advance both the predictive and decision-making dimensions of financial AI, extending the contributions of all four projects in a coherent and practically aligned manner.

(1) Advancing Heterogeneity-Aware Prediction. The static business descriptions used in the heterogeneity-aware framework can be extended to dynamic, time-evolving embeddings derived from quarterly reports, earnings calls, and real-time textual streams. Incorporating adaptive company representations may further enhance cross-company generalization and enable real-time firm-level tracking.

(2) Enhancing Sequential Modeling in Noisy Markets. Future work may explore deeper integrations between state space models and attention mechanisms, including adaptive patching, hierarchical temporal abstractions, and uncertainty-aware variants of Mamba blocks. Such extensions may strengthen robustness under extreme volatility and improve performance across multi-horizon forecasting tasks.

(3) Expanding Multimodal Financial Reasoning. While the current multimodal systems combine numerical, visual, and language-based reasoning, future models may incorporate richer modalities (e.g., order flow, sentiment graphs, and market microstructure images). Leveraging LLM-driven chain-of-thought reasoning with calibrated uncertainty could enhance both interpretability and decision reliability.

(4) Integrating Prediction with Decision Making. A natural next step is to embed the forecasting models into decision-oriented architectures. This includes reinforcement-learning trading agents, risk-aware portfolio optimization, dynamic hedging strategies, and multi-agent simulations. Developing end-to-end systems that jointly learn to predict and act would

bridge the final gap between algorithmic forecasting and economically meaningful financial decisions.

In summary, future research should pursue deeper multimodal integration, greater adaptability to market heterogeneity, and closer coupling between prediction models and decision-making frameworks. Such developments will further advance the creation of intelligent, reliable, and economically aligned financial AI systems.

Bibliography

Abe, Yoshia et al. (2024). 'Leveraging Large Language Models for Institutional Portfolio Management: Persona-Based Ensembles'. In: *2024 IEEE International Conference on Big Data (BigData)*. IEEE, pp. 4799–4808.

Ahmed, Dozdar Mahdi, Masoud Muhammed Hassan and Ramadhan J Mstafa (2022). 'A review on deep sequential models for forecasting time series data'. In: *Applied computational intelligence and soft computing* 2022.1, p. 6596397.

Ahmed, Muhammed Kabir et al. (2021). 'Multi-agent based capital market management system: a distributed framework for trading and regulation'. In: *International Journal of Managing Information Technology (IJMIT) Vol* 13.

Aithal, Prakash K et al. (2023). 'Real-time portfolio management system utilizing machine learning techniques'. In: *IEEE access* 11, pp. 32595–32608.

Arévalo, Andrés et al. (2016). 'High-frequency trading strategy based on deep neural networks'. In: *International conference on intelligent computing*. Springer, pp. 424–436.

Ashtiani, Matin N and Bijan Raahemi (2023). 'News-based intelligent prediction of financial markets using text mining and machine learning: A systematic literature review'. In: *Expert Systems with Applications* 217, p. 119509.

Bala, Rajni, Ram Pal Singh et al. (2022). 'A dual-stage advanced deep learning algorithm for long-term and long-sequence prediction for multivariate financial time series'. In: *Applied Soft Computing* 126, p. 109317.

Benidis, Konstantinos et al. (2022). 'Deep learning for time series forecasting: Tutorial and literature survey'. In: *ACM Computing Surveys* 55.6, pp. 1–36.

Betancourt, Carlos and Wen-Hui Chen (2021). 'Deep reinforcement learning for portfolio management of markets with a dynamic number of assets'. In: *Expert Systems with Applications* 164, p. 114002.

Bieganowski, Bartosz and Robert Ślepaczuk (2025). 'Supervised autoencoder MLP for financial time series forecasting'. In: *Journal of Big Data* 12.1, p. 207.

Bollinger, John (2002). *Bollinger on Bollinger bands*. McGraw-Hill New York.

Broby, Daniel (2022). 'The use of predictive analytics in finance'. In: *The Journal of Finance and Data Science* 8, pp. 145–161.

Bustos, Oscar and Alexandra Pomares-Quimbaya (2020). 'Stock market movement forecast: A systematic review'. In: *Expert Systems with Applications* 156, p. 113464.

Cagliero, Luca, Jacopo Fior and Paolo Garza (2023). 'Shortlisting machine learning-based stock trading recommendations using candlestick pattern recognition'. In: *Expert Systems with Applications* 216, p. 119493.

Cao, Sean et al. (2024). 'From man vs. machine to man+ machine: The art and AI of stock analyses'. In: *Journal of Financial Economics* 160, p. 103910.

Çelik, Taha Buğra, Özgür İcan and Elif Bulut (2023). 'Extending machine learning prediction capabilities by explainable AI in financial time series prediction'. In: *Applied Soft Computing* 132, p. 109876.

Chandra, Rohitash, Shaurya Goyal and Rishabh Gupta (2021). 'Evaluation of deep learning models for multi-step ahead time series prediction'. In: *Ieee Access* 9, pp. 83105–83123.

Chen, Yu-Fu and Szu-Hao Huang (2021). 'Sentiment-influenced trading system based on multimodal deep reinforcement learning'. In: *Applied Soft Computing* 112, p. 107788.

Chen, Luyang, Markus Pelger and Jason Zhu (2024). 'Deep learning in asset pricing'. In: *Management Science* 70.2, pp. 714–750.

Chen, Min-You, Chiao-Ting Chen and Szu-Hao Huang (2023a). 'Knowledge distillation for portfolio management using multi-agent reinforcement learning'. In: *Advanced Engineering Informatics* 57, p. 102096.

Chen, Weisi et al. (2023b). 'Deep learning for financial time series prediction: A state-of-the-art review of standalone and hybrid models'. In: *CMES-Computer Modeling in Engineering and Sciences*.

Cheng, Chi et al. (2024). 'Quantum Finance and Fuzzy Reinforcement Learning-Based Multi-agent Trading System'. In: *International Journal of Fuzzy Systems* 26.7, pp. 2224–2245.

Cheng, Dawei et al. (2022). 'Financial time series forecasting with multi-modality graph neural network'. In: *Pattern Recognition* 121, p. 108218.

Cheng, Rui and Qing Li (2021). 'Modeling the momentum spillover effect for stock prediction via attribute-driven graph attention networks'. In: *Proceedings of the AAAI Conference on artificial intelligence*. Vol. 35. 1, pp. 55–62.

Cui, Xiangyu et al. (2024). 'Dual Core Portfolio Strategy: A Deep RL & Multi-Agent Portfolio Strategy'. In: *Proceedings of the 2024 International Conference on Mathematics and Machine Learning*, pp. 147–153.

Devrani, Rekha, Prerna Mahajan and Dinesh Kumar (2024). 'The Smart Predictive Analysis of Stock Market Performance using Restricted Boltzmann Machines'. In: *2024 International Conference on Optimization Computing and Wireless Communication (ICOCWC)*. IEEE, pp. 1–6.

Ding, Wei et al. (2024). 'Trend-heuristic reinforcement learning framework for news-oriented stock portfolio management'. In: *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 5120–5124.

Dong, Zihan, Xinyu Fan and Zhiyuan Peng (2024). 'Fnspid: A comprehensive financial news dataset in time series'. In: *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 4918–4927.

Du, Juan (2022). 'Mean–variance portfolio optimization with deep learning based-forecasts for cointegrated stocks'. In: *Expert Systems with Applications* 201, p. 117005.

Dudukcu, Hatice Vildan et al. (2023). 'Temporal Convolutional Networks with RNN approach for chaotic time series prediction'. In: *Applied soft computing* 133, p. 109945.

Durairaj, Dr M and BH Krishna Mohan (2022). 'A convolutional neural network based approach to financial time series prediction'. In: *Neural Computing and Applications* 34.16, pp. 13319–13337.

Faheem, Muhammad, Muhammad Aslam and Sridevi Kakolu (2022). 'Artificial Intelligence in Investment Portfolio Optimization: A Comparative Study of Machine Learning Algorithms'. In: *International Journal of Science and Research Archive* 6.1, pp. 335–342.

Fan, Jin et al. (2023). 'Parallel spatio-temporal attention-based TCN for multivariate time series prediction'. In: *Neural Computing and Applications* 35.18, pp. 13109–13118.

Fang, Yuchen et al. (2023). ‘Learning multi-agent intention-aware communication for optimal multi-order execution in finance’. In: *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 4003–4012.

Farimani, Saeede Anbaee, Majid Vafaei Jahan and Amin Milani Fard (2024). ‘An adaptive multimodal learning model for financial market price prediction’. In: *IEEE Access*.

Fatemi, Sorouralsadat and Yuheng Hu (2024). ‘FinVision: A multi-agent framework for stock market prediction’. In: *Proceedings of the 5th ACM International Conference on AI in Finance*, pp. 582–590.

Feng, Qi, Han Chen and Ruohan Jiang (2021). ‘Analysis of early warning of corporate financial risk via deep learning artificial neural network’. In: *Microprocessors and Microsystems* 87, p. 104387.

Feng, Ran and Xiaoe Qu (2022). ‘Analyzing the Internet financial market risk management using data mining and deep learning methods’. In: *Journal of Enterprise Information Management* 35.4/5, pp. 1129–1147.

Föhring, René and Stephan Zelewski (2015). ‘Towards decentralized electronic market places and agent-based freight exchanges for multimodal transports’. In: *2015 IEEE International Conference on Automation Science and Engineering (CASE)*. IEEE, pp. 249–254.

Gadre, Samir Yitzhak et al. (2023). ‘Datacomp: In search of the next generation of multimodal datasets’. In: *Advances in Neural Information Processing Systems* 36, pp. 27092–27112.

Galarnyk, Michael et al. (2025). ‘Videoconviction: A multimodal benchmark for human conviction and stock market recommendations’. In: *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 5447–5458.

Gandhmal, Dattatray P and Kannan Kumar (2019). ‘Systematic analysis and review of stock market prediction techniques’. In: *Computer Science Review* 34, p. 100190.

Gao, Ya, Rong Wang and Enmin Zhou (2021). ‘Stock prediction based on optimized LSTM and GRU models’. In: *Scientific Programming* 2021.1, p. 4055281.

Guan, Mao and Xiao-Yang Liu (2021). ‘Explainable deep reinforcement learning for portfolio management: an empirical approach’. In: *Proceedings of the second ACM international conference on AI in finance*, pp. 1–9.

Gumparthi, Srinivas et al. (2017). 'Relative strength index for developing effective trading strategies in constructing optimal portfolio'. In: *International Journal of Applied Engineering Research* 12.19, pp. 8926–8936.

Gunjan, Abhishek and Siddhartha Bhattacharyya (2023). 'A brief review of portfolio optimization techniques'. In: *Artificial Intelligence Review* 56.5, pp. 3847–3886.

Hahn, Yannik et al. (2023). 'Time series dataset survey for forecasting with deep learning'. In: *Forecasting* 5.1, pp. 315–335.

Han, Songqiao et al. (2024). 'Core patent forecasting based on graph neural networks with an application in stock markets'. In: *Technology Analysis & Strategic Management* 36.8, pp. 1680–1694.

Han, Xu et al. (2025a). 'Symmetry-Aware Credit Risk Modeling: A Deep Learning Framework Exploiting Financial Data Balance and Invariance.' In: *Symmetry (20738994)* 17.3.

Han, Yongyi et al. (2025b). 'ChatCAS: A Multimodal Ceramic Multi-Agent Studio for Consultation, Image Analysis and Generation'. In: *Electronics* 14.18, p. 3735.

He, Brian Yueshuai, Qinhua Jiang, Jiaqi Ma et al. (2024). 'Multi-agent multimodal transportation simulation for mega-cities: Application of los angeles'. In: *Procedia Computer Science* 238, pp. 736–741.

Hsu, Yi-Ling, Yu-Che Tsai and Cheng-Te Li (2021). 'FinGAT: Financial graph attention networks for recommending top-k k profitable stocks'. In: *IEEE transactions on knowledge and data engineering* 35.1, pp. 469–481.

Htun, Htet Htet, Michael Biehl and Nicolai Petkov (2023). 'Survey of feature selection and extraction techniques for stock market prediction'. In: *Financial Innovation* 9.1, p. 26.

Hu, Yin et al. (2024). 'PPMamba: A Pyramid Pooling Local Auxiliary SSM-Based Model for Remote Sensing Image Semantic Segmentation'. In: *arXiv preprint arXiv:2409.06309*.

Hu, Zexin, Yiqi Zhao and Matloob Khushi (2021). 'A survey of forex and stock price prediction using deep learning'. In: *Applied System Innovation* 4.1, p. 9.

Huang, Xiaoqiao et al. (2022). 'Time series forecasting for hourly photovoltaic power using conditional generative adversarial network and Bi-LSTM'. In: *Energy* 246, p. 123403.

Huang, Yuling et al. (2024a). 'A multi-agent reinforcement learning framework for optimizing financial trading strategies based on TimesNet'. In: *Expert Systems with Applications* 237, p. 121502.

Huang, Zhendai et al. (2024b). 'Leveraging enhanced egret swarm optimization algorithm and artificial intelligence-driven prompt strategies for portfolio selection'. In: *Scientific Reports* 14.1, p. 26681.

Iacovides, Giorgos et al. (2024). 'Finllama: Llm-based financial sentiment analysis for algorithmic trading'. In: *Proceedings of the 5th ACM International Conference on AI in Finance*, pp. 134–141.

Jahnavi, M et al. (2024). 'Optimizing Mutual Fund Portfolio Management through the Application of Advanced Soft Computing Techniques'. In: *2024 Third International Conference on Electrical, Electronics, Information and Communication Technologies (ICEEICT)*. IEEE, pp. 1–6.

Jang, Junkyu and NohYoon Seong (2023). 'Deep reinforcement learning for stock portfolio optimization by connecting with modern portfolio theory'. In: *Expert Systems with Applications* 218, p. 119556.

Jeribi, Fathe et al. (2024). 'A deep learning based expert framework for portfolio prediction and forecasting'. In: *IEEE Access* 12, pp. 103810–103829.

Ji, Xuan, Jiachen Wang and Zhijun Yan (2021). 'A stock price prediction method based on deep learning technology'. In: *International Journal of Crowd Science* 5.1, pp. 55–72.

Jiang, Bowen et al. (2025a). 'Towards rationality in language and multimodal agents: a survey'. In: *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 3656–3675.

Jiang, Weiwei (2021). 'Applications of deep learning in stock market prediction: recent progress'. In: *Expert Systems with Applications* 184, p. 115537.

Jiang, Yiping, Zhijing Li and CL Philip Chen (2024). 'Research on Financial Big Data Collection and Intelligent Decision-Making System Based on Multimodal Large Language Model'. In: *2024 International Conference on Fuzzy Theory and Its Applications (iFUZZY)*. IEEE, pp. 1–6.

Jiang, Yushan et al. (2025b). 'Multi-modal time series analysis: A tutorial and survey'. In: *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 6043–6053.

Johnston, FR et al. (1999). 'Some properties of a simple moving average when applied to forecasting a time series'. In: *Journal of the Operational Research Society* 50.12, pp. 1267–1271.

Joshi, Satyadhar (2025a). 'Advancing innovation in financial stability: A comprehensive review of ai agent frameworks, challenges and applications'. In: *World Journal of Advanced Engineering Technology and Sciences* 14.2, pp. 117–126.

— (2025b). 'Review of gen ai models for financial risk management'. In: *International Journal of Scientific Research in Computer Science, Engineering and Information Technology* 11.1, pp. 709–723.

Kang, Yue et al. (2020). 'Natural language processing (NLP) in management research: A literature review'. In: *Journal of Management Analytics* 7.2, pp. 139–172.

Ko, Hyungjin and Jaewook Lee (2024). 'Can ChatGPT improve investment decisions? From a portfolio management perspective'. In: *Finance Research Letters* 64, p. 105433.

— (2025). 'Portfolio management transformed: An enhanced black–litterman approach integrating asset pricing theory and machine learning'. In: *Computational Economics*, pp. 1–47.

Kong, Yaxuan et al. (2024). 'Large Language Models for Financial and Investment Management: Applications and Benchmarks.' In: *Journal of Portfolio Management* 51.2.

Kontopoulou, Vaia I et al. (2023). 'A review of ARIMA vs. machine learning approaches for time series forecasting in data driven networks'. In: *Future Internet* 15.8, p. 255.

Korczak, Jerzy, Marcin Hernes and Maciej Bac (2013). 'Risk avoiding strategy in multi-agent trading system'. In: *2013 Federated Conference on Computer Science and Information Systems*. IEEE, pp. 1131–1138.

Kumar, Deepak, Pradeepta Kumar Sarangi and Rajit Verma (2022). 'A systematic review of stock market prediction using machine learning and statistical techniques'. In: *Materials Today: Proceedings* 49, pp. 3187–3191.

Kumar, Raghavendra, Pardeep Kumar and Yugal Kumar (2021). 'Analysis of financial time series forecasting using deep learning model'. In: *2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*. IEEE, pp. 877–881.

Kuruvilla, Jeevan Sunil and M Mythily (2025). 'Financial LLM For Stock Price Analysis And Investment Recommendation'. In: *Journal of Telematics and Informatics* 13.1.

Lazcano, Ana, Pedro Javier Herrera and Manuel Monge (2023). 'A combined model based on recurrent neural networks and graph convolutional networks for financial time series forecasting'. In: *Mathematics* 11.1, p. 224.

Li, Chengyu and Guoqi Qian (2022). 'Stock price prediction using a frequency decomposition based GRU transformer neural network'. In: *Applied Sciences* 13.1, p. 222.

Li, Haifeng and Mo Hai (2024). 'Deep reinforcement learning model for stock portfolio management based on data fusion'. In: *Neural Processing Letters* 56.2, p. 108.

Li, Menggang et al. (2021a). 'Applying BERT to analyze investor sentiment in stock market'. In: *Neural Computing and Applications* 33, pp. 4663–4676.

Li, Shixuan et al. (2021b). 'A deep learning-based approach to constructing a domain sentiment lexicon: a case study in financial distress prediction'. In: *Information Processing & Management* 58.5, p. 102673.

Li, Xiangyu et al. (2025a). 'Hedgeagents: A balanced-aware multi-agent financial trading system'. In: *Companion Proceedings of the ACM on Web Conference 2025*, pp. 296–305.

Li, Xuetao, Jia Wang and Chengying Yang (2023a). 'Risk prediction in financial management of listed companies based on optimized BP neural network under digital economy'. In: *Neural Computing and Applications* 35.3, pp. 2045–2058.

Li, Yinheng et al. (2023b). 'Large language models in finance: A survey'. In: *Proceedings of the fourth ACM international conference on AI in finance*, pp. 374–382.

Li, Zhenglong, Vincent Tam and Kwan L Yeung (2025b). 'A Multimodal and Sentiment-Based Trading System for Financial Portfolio Optimisation'. In: *2025 IEEE International Conference on Consumer Electronics (ICCE)*. IEEE, pp. 1–6.

Li, Zhuo et al. (2025c). 'Embodied Multi-Agent Systems: A Review'. In: *IEEE/CAA Journal of Automatica Sinica* 12.6, pp. 1095–1116.

Lim, Bryan and Stefan Zohren (2021). 'Time-series forecasting with deep learning: a survey'. In: *Philosophical Transactions of the Royal Society A* 379.2194, p. 20200209.

Lim, Qing Yang Eddy, Qi Cao and Chai Quek (2022). 'Dynamic portfolio rebalancing through reinforcement learning'. In: *Neural Computing and Applications* 34.9, pp. 7125–7139.

Liu, Can et al. (2021). 'Intention-aware heterogeneous graph attention networks for fraud transactions detection'. In: *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pp. 3280–3288.

Liu, Chenghao et al. (2025a). 'David vs. Goliath: Cost-Efficient Financial QA via Cascaded Multi-Agent Reasoning'. In: *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 4212–4229.

Liu, Huajin et al. (2023). 'A stock rank prediction method combining industry attributes and price data of stocks'. In: *Information Processing & Management* 60.4, p. 103358.

Liu, Pei et al. (2025b). 'Hm-rag: Hierarchical multi-agent multimodal retrieval augmented generation'. In: *Proceedings of the 33rd ACM International Conference on Multimedia*, pp. 2781–2790.

Lu, Minrong and Xuerong Xu (2024). 'TRNN: An efficient time-series recurrent neural network for stock price prediction'. In: *Information Sciences* 657, p. 119951.

Lu, Xiaobin, Josiah Poon and Matloob Khushi (2024). 'Bridging the gap between machine and human in stock prediction: Addressing heterogeneity in stock market'. In: *IEEE Access*.

— (2025). 'Leveraging BiLSTM-GAT for enhanced stock market prediction: a dual-graph approach to portfolio optimization'. In: *Applied Intelligence* 55.7, p. 601.

Luo, Yuan, Kecheng Liu and Darryl N Davis (2002). 'A multi-agent decision support system for stock trading'. In: *IEEE network* 16.1, pp. 20–27.

Lussange, Johann et al. (2021). 'Modelling stock markets by multi-agent reinforcement learning'. In: *Computational Economics* 57.1, pp. 113–147.

Lussange, Johann et al. (2023). 'Stock price formation: Precepts from a multi-agent reinforcement learning model'. In: *Computational Economics* 61.4, pp. 1523–1544.

Ma, Cong and Shijing Nan (2024). 'Dynamic graph reinforcement learning algorithm for portfolio management: A novel time–frequency correlated model'. In: *Finance Research Letters* 63, p. 105373.

Ma, Cong et al. (2023). 'Multi-agent deep reinforcement learning algorithm with trend consistency regularization for portfolio management'. In: *Neural Computing and Applications* 35.9, pp. 6589–6601.

Ma, Yu et al. (2024). 'Quantitative stock portfolio optimization by multi-task learning risk and return'. In: *Information Fusion* 104, p. 102165.

Marti, Gautier et al. (2021). 'A review of two decades of correlations, hierarchies, networks and clustering in financial markets'. In: *Progress in information geometry: Theory and applications*, pp. 245–274.

Masini, Ricardo P, Marcelo C Medeiros and Eduardo F Mendes (2023). 'Machine learning advances for time series forecasting'. In: *Journal of economic surveys* 37.1, pp. 76–111.

Mehrabian, Ali et al. (2025). 'Mamba meets financial markets: A graph-mamba approach for stock price prediction'. In: *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 1–5.

Mehtab, Sidra and Jaydip Sen (2022). 'Analysis and forecasting of financial time series using CNN and LSTM-based deep learning models'. In: *Advances in Distributed Computing and Machine Learning: Proceedings of ICADCML 2021*. Springer, pp. 405–423.

Mintarya, Latrisha N et al. (2023). 'Machine learning approaches in stock market prediction: A systematic literature review'. In: *Procedia Computer Science* 216, pp. 96–102.

Nabipour, Mojtaba et al. (2020). 'Deep learning for stock market prediction'. In: *Entropy* 22.8, p. 840.

Nicholls, Jack, Aditya Kuppa and Nhien-An Le-Khac (2021). 'Financial cybercrime: A comprehensive survey of deep learning approaches to tackle the evolving financial crime landscape'. In: *Ieee Access* 9, pp. 163965–163986.

Nikou, Mahla, Gholamreza Mansourfar and Jamshid Bagherzadeh (2019). 'Stock price prediction using DEEP learning algorithm and its comparison with machine learning algorithms'. In: *Intelligent Systems in Accounting, Finance and Management* 26.4, pp. 164–174.

Nti, Isaac Kofi, Adebayo Felix Adekoya and Benjamin Asubam Weyori (2020). 'A systematic review of fundamental and technical analysis of stock market predictions'. In: *Artificial Intelligence Review* 53.4, pp. 3007–3057.

Pandey, Vikrant, Wee-Keong Ng and Ee-Peng Lim (2000). 'Financial advisor agent in a multi-agent financial trading system'. In: *Proceedings 11th International Workshop on Database and Expert Systems Applications*. IEEE, pp. 482–486.

Pang, Xiongwen et al. (2020). 'An innovative neural network approach for stock market prediction'. In: *The Journal of Supercomputing* 76, pp. 2098–2118.

Pantoja Robayo, Javier Orlando, Julián Alberto Alemán Muñoz and Diego F Tellez-Falla (2025). 'Iterative deep learning approach to active portfolio management with sentiment factors'. In: *Computational Economics* 66.1, pp. 301–322.

Peigné, Pierre et al. (2025). 'Multi-agent security tax: Trading off security and collaboration capabilities in multi-agent systems'. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 39. 26, pp. 27573–27581.

Pele, Daniel Traian et al. (2026). 'In the beginning was the word: LLM-VaR and LLM-ES'. In: *Expert Systems with Applications* 295, p. 128676.

Petrozziello, Alessio et al. (2022). 'Deep learning for volatility forecasting in asset management'. In: *Soft Computing* 26.17, pp. 8553–8574.

Qiu, Yitao, RongKai Liu and Raymond ST Lee (2024). 'The design and implementation of a deep reinforcement learning and quantum finance theory-inspired portfolio investment management system'. In: *Expert Systems with Applications* 238, p. 122243.

Rahman, SM Masudur, Kaniz Fatema and Nayan Miah (2022). 'Fundamental Analysis of Dhaka Stock Exchange (DSE) Listed Top Five NBFIs: A Study on Bangladesh'. In: *Journal of International Business and Management* 5.6, pp. 01–20.

Ram, Kottooru Saaketh Raja et al. (2024). 'Enhanced Investment Decision Making with a Reinforcement Learning-Based Multi-Agent Portfolio Management System'. In: *2024 International Conference on Data Science and Network Security (ICDSNS)*. IEEE, pp. 1–6.

Rezaei, Hadi, Hamidreza Faaljou and Gholamreza Mansourfar (2021). 'Stock price prediction using deep learning and frequency decomposition'. In: *Expert Systems with Applications* 169, p. 114332.

Roe, Mark J (2020). *Missing the target: why stock-market short-termism is not the problem*. Oxford University Press.

Rouf, Nusrat et al. (2021). 'Stock market prediction using machine learning techniques: a decade survey on methodologies, recent developments, and future directions'. In: *Electronics* 10.21, p. 2717.

Sarin, Saket et al. (2024). 'Unleashing the Power of Multi-Agent Reinforcement Learning for Algorithmic Trading in the Digital Financial Frontier and Enterprise Information Systems.' In: *Computers, Materials & Continua* 80.2.

Satapathy, Ranjan et al. (2025). 'From Earnings Calls to Investment Reports: Evaluating Role-based Multi-Agent LLM Systems'. In: *Proceedings of The 10th Workshop on Financial Technology and Natural Language Processing*, pp. 258–267.

Sattar, Asma et al. (2025). 'A novel rms-driven deep reinforcement learning for optimized portfolio management in stock trading'. In: *IEEE Access*.

Shah, Jaimin, Darsh Vaidya and Manan Shah (2022). 'A comprehensive review on multiple hybrid deep learning approaches for stock prediction'. In: *Intelligent Systems with Applications* 16, p. 200111.

Shavandi, Ali and Majid Khedmati (2022). 'A multi-agent deep reinforcement learning framework for algorithmic trading in financial markets'. In: *Expert Systems with Applications* 208, p. 118124.

Shi, Si et al. (2021). 'Xpm: An explainable deep reinforcement learning framework for portfolio management'. In: *Proceedings of the 30th ACM international conference on information & knowledge management*, pp. 1661–1670.

Shi, Si et al. (2022). 'Machine learning-driven credit risk: a systemic review'. In: *Neural Computing and Applications* 34.17, pp. 14327–14339.

Singh, Arjun et al. (2022). 'Application of neural network to technical analysis of stock market prediction'. In: *2022 3rd International Conference on Intelligent Engineering and Management (ICIEM)*. IEEE, pp. 302–306.

Sirisha, Uppala Meena, Manjula C Belavagi and Girija Attigeri (2022). 'Profit prediction using ARIMA, SARIMA and LSTM models in time series forecasting: A comparison'. In: *Ieee Access* 10, pp. 124715–124727.

Soleymani, Farzan and Eric Paquet (2021). 'Deep graph convolutional reinforcement learning for financial portfolio management–DeepPocket'. In: *Expert Systems with Applications* 182, p. 115127.

Song, Guowei et al. (2023). 'Stock ranking prediction using a graph aggregation network based on stock price and stock relationship information'. In: *Information Sciences* 643, p. 119236.

Soni, Payal, Yogya Tewari and Deepa Krishnan (2022). 'Machine learning approaches in stock price prediction: a systematic review'. In: *Journal of Physics: Conference Series*. Vol. 2161. 1. IOP Publishing, p. 012065.

Su, Liyilei et al. (2025). 'A systematic review for transformer-based long-term series forecasting'. In: *Artificial Intelligence Review* 58.3, p. 80.

Sun, Yu-Zhu et al. (2025). 'Multimodal Agent AI: A Survey of Recent Advances and Future Directions'. In: *Journal of Computer Science and Technology* 40.4, pp. 1046–1063.

Tao, Zicheng, Wei Wu and Jianxin Wang (2024). 'Series decomposition Transformer with period-correlation for stock market index prediction'. In: *Expert Systems with Applications* 237, p. 121424.

Tashiro, Yusuke et al. (2021). 'Csdi: Conditional score-based diffusion models for probabilistic time series imputation'. In: *Advances in neural information processing systems* 34, pp. 24804–24816.

Teixeira, Ana Clara et al. (2023). '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*, pp. 340–348.

Thakkar, Ankit and Kinjal Chaudhari (2021). 'A comprehensive survey on deep neural networks for stock market: The need, challenges, and future directions'. In: *Expert Systems with Applications* 177, p. 114800.

Thenuwara, Susara S, Chinthaka Premachandra and Hiroharu Kawanaka (2022). 'A multi-agent based enhancement for multimodal biometric system at border control'. In: *Array* 14, p. 100171.

Triantafyllopoulos, Kostas (2021). 'The state space model in finance'. In: *Bayesian Inference of State Space Models: Kalman Filtering and Beyond*. Springer, pp. 341–402.

Vahdati, Monica et al. (2025). 'Digital Twin AI Fitness Coach: An Intelligent Multi-Agent System for Personalized Exercise Guidance'. In: *Proceedings of the 1st International Workshop on Multi-Sensorial Media and Applications*, pp. 18–26.

Wang, Chaojie et al. (2022a). 'Stock market index prediction using deep Transformer model'. In: *Expert Systems with Applications* 208, p. 118128.

Wang, Gang, Jingling Ma and Gang Chen (2023). 'Attentive statement fraud detection: Distinguishing multimodal financial data with fine-grained attention'. In: *Decision Support Systems* 167, p. 113913.

Wang, Shuzhen (2023). 'A stock price prediction method based on BiLSTM and improved transformer'. In: *IEEE Access*.

Wang, Wenjing et al. (2025). 'An efficient framework for general long-horizon time series forecasting with Mamba and Diffusion Probabilistic Models'. In: *Engineering Applications of Artificial Intelligence* 162, p. 112525.

Wang, Yi et al. (2022b). 'Attention based spatiotemporal graph attention networks for traffic flow forecasting'. In: *Information Sciences* 607, pp. 869–883.

Wang, Zhicheng et al. (2021). '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. 1, pp. 643–650.

Wang, Zirui et al. (2024). 'Charxiv: Charting gaps in realistic chart understanding in multimodal llms'. In: *Advances in Neural Information Processing Systems* 37, pp. 113569–113697.

Wen, Xianyun and Weibang Li (2023). 'Time series prediction based on LSTM-attention-LSTM model'. In: *IEEE access* 11, pp. 48322–48331.

Widiputra, Harya, Adele Mailangkay and Elliana Gautama (2021). 'Multivariate CNN-LSTM model for multiple parallel financial time-series prediction'. In: *Complexity* 2021.1, p. 9903518.

Wu, Mu-En et al. (2021). 'Portfolio management system in equity market neutral using reinforcement learning'. In: *Applied Intelligence* 51.11, pp. 8119–8131.

Wu, Junran et al. (2022). 'Price graphs: Utilizing the structural information of financial time series for stock prediction'. In: *Information Sciences* 588, pp. 405–424.

Wu, Zonghan et al. (2020). 'A comprehensive survey on graph neural networks'. In: *IEEE transactions on neural networks and learning systems* 32.1, pp. 4–24.

Xiang, Sheng et al. (2022). 'Temporal and heterogeneous graph neural network for financial time series prediction'. In: *Proceedings of the 31st ACM international conference on information & knowledge management*, pp. 3584–3593.

Xu, Bingbing et al. (2021). 'Towards consumer loan fraud detection: Graph neural networks with role-constrained conditional random field'. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. 5, pp. 4537–4545.

Xu, Kaixian et al. (2024). 'Generalizable Multi-Agent Framework for Quantitative Trading of US Education Funds'. In: *Innovations in Applied Engineering and Technology*, pp. 1–12.

Xu, Peng, Xiatian Zhu and David A Clifton (2023). 'Multimodal learning with transformers: A survey'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45.10, pp. 12113–12132.

Xu, Yaoqun and Yuhang Zhang (2023). 'Enhancement economic system based-graph neural network in stock classification'. In: *IEEE Access* 11, pp. 17956–17967.

Yadla, Prasanth (2025). 'Collaborative Multimodal Agent Networks: Dynamic Specialization and Emergent Communication for Complex Scene Understanding'. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6468–6475.

Yang, Mo and Jing Wang (2022). 'Adaptability of financial time series prediction based on BiLSTM'. In: *Procedia Computer Science* 199, pp. 18–25.

Yang, Ping Feng Tingting et al. (2025). 'OC-HMAS: Dynamic Self-Organization and Self-Correction in Heterogeneous Multi-Agent Systems Using Multi-Modal Large Models'. In: *IEEE Internet of Things Journal*.

Yang, Yi et al. (2023). 'Unlocking the power of voice for financial risk prediction: A theory-driven deep learning design approach'. In: *Mis Quarterly* 47.1, pp. 63–96.

Yang, Zhen et al. (2021). 'AComNN: Attention enhanced Compound Neural Network for financial time-series forecasting with cross-regional features'. In: *Applied Soft Computing* 111, p. 107649.

Ye, Junchen et al. (2022). 'Learning the evolutionary and multi-scale graph structure for multivariate time series forecasting'. In: *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pp. 2296–2306.

Yu, Chenhao et al. (2025a). 'Computational Argumentation Based Multi-Agent Consensual Approach for Semantically Interpretable Credit Rating'. In: *Machine Learning with Applications*, p. 100690.

Yu, Lean and Mengxin Li (2023). 'A case-based reasoning driven ensemble learning paradigm for financial distress prediction with missing data'. In: *Applied Soft Computing* 137, p. 110163.

Yu, Yangyang et al. (2024). 'Fincon: A synthesized llm multi-agent system with conceptual verbal reinforcement for enhanced financial decision making'. In: *Advances in Neural Information Processing Systems* 37, pp. 137010–137045.

Yu, Yangyang et al. (2025b). 'Finmem: A performance-enhanced llm trading agent with layered memory and character design'. In: *IEEE Transactions on Big Data*.

Zeng, Zhen et al. (2023). 'From pixels to predictions: Spectrogram and vision transformer for better time series forecasting'. In: *Proceedings of the Fourth ACM International Conference on AI in Finance*, pp. 82–90.

Zhang, Hengxi et al. (2024). 'Optimizing trading strategies in quantitative markets using multi-agent reinforcement learning'. In: *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 136–140.

Zhang, Luyao et al. (2022). 'A data science pipeline for algorithmic trading: A comparative study of applications for finance and cryptoeconomics'. In: *2022 IEEE International Conference on Blockchain (Blockchain)*. IEEE, pp. 298–303.

Zhang, Zihao, Stefan Zohren and Stephen Roberts (2019). 'Deeplob: Deep convolutional neural networks for limit order books'. In: *IEEE Transactions on Signal Processing* 67.11, pp. 3001–3012.

Zhao, Fei, Chengcui Zhang and Baocheng Geng (2024). 'Deep multimodal data fusion'. In: *ACM computing surveys* 56.9, pp. 1–36.

Zhou, Haoyi et al. (2021). 'Informer: Beyond efficient transformer for long sequence time-series forecasting'. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 35, pp. 11106–11115.

1 Appendix A

Something