Adaptive Stock Market Prediction Using Node Transformer Architectures with Sentiment Analysis and Reinforcement Learning Control

Mohammad Al Ridhawi

Thesis submitted to the University of Ottawa in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Electrical and Computer Engineering

School of Electrical Engineering and Computer Science
Faculty of Engineering
University of Ottawa

Supervisor: Professor Hussein Al Osman

uOttawa

© Mohammad Al Ridhawi, Ottawa, Canada, 2026

Examining Committee Members

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner: Ali Karime
Associate Professor
Royal Military College of Canada,
Department of Electrical and Computer Engineering

Internal Member: Shichao Liu
Associate Professor
Carleton University, Faculty of Engineering

Internal Member: Shervin Shirmohammadi
Professor
University of Ottawa, School of Electrical
Engineering and Computer Science

Internal Member: Abdulmotaleb El Saddik
Distinguished Professor
University of Ottawa, School of Electrical
Engineering and Computer Science

Supervisor: Hussein Al Osman
Full Professor
University of Ottawa, School of Electrical
Engineering and Computer Science

Abstract

Stock market prediction remains a central problem in quantitative finance, complicated by non-stationary price dynamics, complex inter-stock dependencies, and the influence of investor sentiment on price formation. Traditional forecasting methods, including statistical models and standard deep learning architectures, typically treat stocks as independent time series and process all market conditions through a single model, limiting their capacity to capture cross-sectional structure and adapt to regime shifts. This thesis develops a series of progressively more capable architectures for stock price forecasting, evaluated on 20 S&P 500 equities spanning January 1982 to March 2025.

The first contribution introduces a node transformer architecture that models the stock market as a graph in which individual stocks form nodes and learnable edges encode sectoral affiliations and return correlations. A fine-tuned BERT model extracts sentiment signals from approximately 4.2 million social media posts, integrated with quantitative market features through an attention-based fusion mechanism that dynamically adjusts the relative contribution of each modality based on volatility and sentiment magnitude. This system achieves 0.80% mean absolute percentage error (MAPE) for one-day-ahead predictions and 65% directional accuracy. Ablation studies confirm that graph structure contributes a 15% relative error reduction and sentiment integration a further 10%, with the largest sentiment gains occurring during earnings announcements where error decreases by 25%.

The second contribution addresses a persistent weakness of the first system: degraded performance during volatile market periods. An autoencoder trained exclusively on stable market data identifies regime shifts through reconstruction error, gating data flow into dual node transformer pathways specialized for normal and event-driven conditions respectively. The event pathway incorporates additional context features including volatility regime embeddings, sentiment spikes, earnings proximity, and cross-asset stress indicators. A Soft Actor-Critic reinforcement learning controller learns optimal regime detection thresholds and pathway blending weights directly from prediction performance feedback, discovering useful regime boundaries without requiring manual labeling or predefined volatility rules. The complete adaptive system achieves 0.59% MAPE and 72% directional accuracy, maintaining performance below 0.85% MAPE during high-volatility episodes where the baseline single-path model exceeds 1.5%.

The third contribution proposes a framework for evaluating the behavioral quality of autonomous prediction systems using large language models as judges. Conventional metrics such as MAPE and directional accuracy describe only the final prediction output, revealing nothing about whether intermediate decisions along the pipeline were sound. The proposed framework assesses decision quality across six dimensions: regime detection accuracy, routing appropriateness, adaptation responsiveness, risk calibration, strategy coherence, and error recovery. Perturbation-based validation yields cross-model consistency up to Krippendorff's $\alpha = 0.85$, and composite scores correlate with realized Sharpe ratios at $\rho = 0.72$. The framework closes the evaluation loop by translating LLM diagnostic outputs into auxiliary reward signals for the reinforcement learning controller, yielding further improvement to 0.54% MAPE and 74% directional accuracy.

Across all three contributions, paired t-tests confirm statistical significance of performance improvements ($p < 0.05$ for all comparisons), and systematic ablation studies isolate the marginal contribution of each architectural component. The progression from a single graph transformer to a regime-aware adaptive system with closed-loop behavioral evaluation demonstrates that jointly modeling inter-stock dependencies, market regime dynamics, and investor sentiment produces forecasts that are both more accurate and more robust than approaches addressing these dimensions in isolation.

Acknowledgements

First and foremost, I would like to thank my supervisor, Dr. Hussein Al Osman, for his continued support and guidance throughout my graduate studies, from sharing his deep knowledge in this field, to being a friend to talk to when needed. You have been very important in this journey, and I will forever be grateful for providing me with this opportunity.

I would also like to thank all my friends and colleagues at the University of Ottawa for their continuous support and encouragement throughout my PhD journey.

Finally, and most importantly, I would like to thank my family for their support, patience, and encouragement throughout these years. Without my family, none of this would ever be possible.

A very special and heartfelt thank you to my mother and father for their unconditional love, sacrifices, and constant belief in me. Your unwavering support and countless sacrifices have been the foundation of my success and have carried me through every challenge.

I owe everything to the two of you, whose endless love, wisdom, and encouragement have guided me to this point. This achievement is as much yours as it is mine.

Mohammad
Ottawa, 2026

Author Contributions

This thesis follows a thesis-by-publication format. The contributions of each author to the publications that form the basis of the three contribution chapters are detailed below.

Chapter 4: Al Ridhawi, M., & Al Osman, H. (2026). Stock market prediction using node transformer architecture integrated with BERT sentiment analysis. *IEEE Access*, vol. 14, pp. 72613–72631. doi: 10.1109/ACCESS.2026.3691980.

Mohammad Al Ridhawi (first author): Conceptualization of the node transformer architecture and sentiment fusion mechanism; design and implementation of all software components including the graph construction pipeline, node transformer network, BERT fine-tuning, and attention-based fusion module; execution of all experiments including baseline comparisons, ablation studies, and multi-horizon evaluation; statistical analysis of results; writing of the complete original manuscript draft.

Hussein Al Osman (supervisor): Supervision and guidance throughout the research process; review and editing of the manuscript; provision of computational resources.

Chapter 5: Al Ridhawi, M., & Al Osman, H. (2026). Adaptive regime-aware stock price prediction using autoencoder-gated dual node transformers with reinforcement learning control. *Submitted to Applied Intelligence*. Under revision. arXiv:2603.19136.

Mohammad Al Ridhawi (first author): Conceptualization of the adaptive regime-aware architecture including the autoencoder-based regime detection, dual-pathway specialization, and Soft Actor-Critic reinforcement learning controller; design and implementation of all software components and the four-stage training pipeline; execution of all experiments including baseline comparisons, per-regime analysis, ablation studies, and multi-horizon evaluation; statistical analysis of results; writing of the complete original manuscript draft.

Hussein Al Osman (supervisor): Supervision and guidance throughout the research process; review and editing of the manuscript; provision of computational resources.

Chapter 6: Al Ridhawi, M., & Al Osman, H. (2026). Multi-dimensional behavioral evaluation of agentic stock prediction systems using LLM judges with closed-loop reinforcement learning feedback. *Submitted to Applied Artificial Intelligence*. Under review. arXiv:2605.05739.

Mohammad Al Ridhawi (first author): Conceptualization of the LLM-as-a-Judge behavioral evaluation framework, including the six evaluation dimensions, rubric design, prompt engineering, credit assignment mechanism, and closed-loop reward integration; design and implementation of all software components including the behavioral trace logging, LLM evaluation pipeline, perturbation generation, and SAC reward modification; execution of all experiments including perturbation-based validation, cross-model agreement analysis, predictive validity assessment, closed-loop improvement cycles, and framework ablation studies; statistical analysis of results; writing of the complete original manuscript draft.

Hussein Al Osman (supervisor): Supervision and guidance throughout the research process; review and editing of the manuscript; provision of computational resources.

Contents

Examining Committee Members **ii**

Abstract **iii**

Acknowledgements **v**

Author Contributions **vi**

List of Figures **xiii**

List of Tables **xiv**

List of Abbreviations **xvi**

1 Introduction **1**
- 1.1 Background . . . 1
- 1.2 Challenges in Stock Price Forecasting . . . 3
  - 1.2.1 Market Noise and Non-Stationarity . . . 3
  - 1.2.2 The Efficient Market Hypothesis . . . 4
  - 1.2.3 Behavioral Dynamics and Investor Sentiment . . . 5
  - 1.2.4 Inter-Stock Dependencies and Market Structure . . . 6
  - 1.2.5 Regime Shifts and Structural Breaks . . . 8
  - 1.2.6 Regulatory and Macroeconomic Factors . . . 9
- 1.3 Motivation . . . 10
- 1.4 Thesis Contributions . . . 12
- 1.5 Research Publications . . . 15
- 1.6 Thesis Organization . . . 16

2 Literature Review **18**
- 2.1 Statistical Time Series Models . . . 19
  - 2.1.1 Autoregressive Integrated Moving Average Models . . . 20
  - 2.1.2 Vector Autoregression . . . 21
  - 2.1.3 GARCH Family Models . . . 21
  - 2.1.4 Limitations of Statistical Models for Financial Prediction . . . 22
- 2.2 Classical Machine Learning for Stock Prediction . . . 23
  - 2.2.1 Support Vector Machines . . . 24
  - 2.2.2 Ensemble Methods . . . 24
  - 2.2.3 Bayesian Methods . . . 26
- 2.3 Deep Learning Architectures . . . 26
  - 2.3.1 Convolutional Neural Networks . . . 27
  - 2.3.2 Recurrent Neural Networks . . . 28

2.3.3 Attention Mechanisms in Deep Neural Networks . . . 30
2.3.4 Hybrid Deep Learning Approaches . . . 31
2.4 Transformer Architectures for Financial Forecasting . . . 33
2.4.1 Self-Attention Mechanism . . . 33
2.4.2 Positional Encoding . . . 34
2.4.3 Transformer Architecture for Sequence Modeling . . . 35
2.4.4 Financial Transformer Variants . . . 35
2.4.5 Efficient Transformer Architectures . . . 36
2.5 Graph Neural Networks for Stock Market Prediction . . . 37
2.5.1 Message Passing Framework . . . 38
2.5.2 Graph Convolutional Networks . . . 38
2.5.3 Graph Attention Networks . . . 38
2.5.4 GNNs for Financial Applications . . . 39
2.5.5 The Node Transformer . . . 41
2.6 Sentiment Analysis for Financial Forecasting . . . 43
2.6.1 Lexicon-Based Approaches . . . 43
2.6.2 Machine Learning Approaches to Sentiment Classification . . . 44
2.6.3 BERT and Transformer-Based Sentiment Analysis . . . 45
2.6.4 Multimodal Fusion of Sentiment and Price Data . . . 46
2.7 Regime Detection and Market State Identification . . . 49
2.7.1 Hidden Markov Models . . . 49
2.7.2 Threshold-Based Approaches . . . 50
2.7.3 Autoencoders for Unsupervised Regime Detection . . . 51
2.7.4 Change Point Detection . . . 52
2.8 Reinforcement Learning in Financial Applications . . . 52
2.8.1 Foundations of Deep Reinforcement Learning . . . 53
2.8.2 Soft Actor-Critic . . . 53
2.8.3 Reinforcement Learning for Trading and Portfolio Management . . . 54
2.8.4 RL as Meta-Controller for Prediction Systems . . . 55
2.9 LLM-Based Evaluation and Autonomous Agent Assessment . . . 56
2.9.1 LLM as a Judge . . . 56
2.9.2 Evaluation Design: Rubrics, Multi-Judge Agreement, and Calibration 57
2.9.3 Agent Evaluation Frameworks . . . 58
2.9.4 Evaluation in Financial AI . . . 59
2.10 Summary and Research Gaps . . . 60

3 Dataset and Experimental Framework **63**
3.1 Stock Selection and Rationale . . . 64
3.2 Financial Market Data . . . 67
3.2.1 Prediction Target . . . 69
3.3 Feature Engineering . . . 70
3.3.1 Moving Averages . . . 71
3.3.2 Relative Strength Index . . . 72
3.3.3 Moving Average Convergence Divergence . . . 72
3.3.4 Returns and Volatility . . . 73

3.3.5 Feature Summary . . . 74
3.3.6 Router-Specific Features . . . 75
3.4 Sentiment Data . . . 76
3.4.1 Financial Microblog Sentiment Dataset . . . 76
3.4.2 Comprehensive Stock Sentiment Dataset . . . 77
3.4.3 Data Quality Considerations . . . 78
3.4.4 Text Preprocessing . . . 79
3.4.5 Sentiment Scoring and Aggregation . . . 80
3.4.6 Temporal Alignment . . . 80
3.5 Data Preprocessing . . . 81
3.5.1 Missing Value Imputation . . . 82
3.5.2 Expanding Window Normalization . . . 82
3.5.3 Stationarity Considerations . . . 83
3.5.4 Outlier Treatment . . . 84
3.6 Dataset Partitioning and Validation Protocol . . . 84
3.6.1 Partition Boundaries . . . 85
3.6.2 Walk-Forward Evaluation Protocol . . . 86
3.6.3 Feature Boundary Considerations . . . 87
3.7 Evaluation Metrics . . . 87
3.7.1 Mean Absolute Percentage Error . . . 88
3.7.2 Root Mean Squared Error . . . 89
3.7.3 Directional Accuracy . . . 89
3.7.4 Theil's U Statistic . . . 90
3.7.5 Information Coefficient . . . 91
3.7.6 Confidence Tracking Rate . . . 92
3.7.7 Sharpe Ratio . . . 92
3.7.8 Maximum Drawdown . . . 93
3.8 Summary . . . 94

4 Stock Market Prediction Using Node Transformer Architecture Integrated with BERT Sentiment Analysis **96**
4.1 Related Work and Research Gap . . . 97
4.2 Methodology . . . 99
4.2.1 System Architecture Overview . . . 99
4.2.2 Graph Representation of the Stock Market . . . 100
4.2.3 Node Transformer Architecture . . . 102
4.2.4 BERT-Based Sentiment Analysis . . . 105
4.2.5 Integration of Node Transformer and BERT . . . 107
4.2.6 Training Objective . . . 109
4.2.7 Training Process . . . 111
4.3 Experiments and Results . . . 113
4.3.1 Experimental Setup . . . 113
4.3.2 Baseline Models . . . 114
4.3.3 Sentiment Model Evaluation . . . 115
4.3.4 Results . . . 117

4.4 Discussion . . . 124
4.4.1 Interpretation of Results . . . 124
4.4.2 Implications . . . 126
4.4.3 Limitations . . . 127
4.5 Summary . . . 129

5 Adaptive Regime-Aware Stock Price Prediction Using Autoencoder-Gated Dual Node Transformers with Reinforcement Learning Control **131**
5.1 Related Work and Research Gap . . . 132
5.2 Methodology . . . 133
5.2.1 System Overview . . . 135
5.2.2 Feature Engineering . . . 138
5.2.3 Autoencoder for Regime Detection . . . 139
5.2.4 Dual Node Transformer Architecture . . . 142
5.2.5 Soft Actor-Critic Controller . . . 146
5.2.6 Training Pipeline . . . 150
5.2.7 Loss Functions . . . 151
5.3 Experiments and Results . . . 152
5.3.1 Experimental Setup . . . 152
5.3.2 Evaluation Metrics . . . 153
5.3.3 Baseline Models . . . 154
5.3.4 Main Results . . . 156
5.3.5 Prediction Visualization . . . 159
5.3.6 Per-Stock Results . . . 161
5.3.7 Ablation Study . . . 166
5.3.8 Volatility Regime Analysis . . . 171
5.3.9 Threshold Sensitivity Analysis . . . 172
5.3.10 SAC Controller Behavior . . . 174
5.3.11 Economic Significance and Trading Backtest . . . 178
5.3.12 Statistical Significance . . . 180
5.3.13 Baseline Hyperparameter Configuration . . . 181
5.3.14 Computing Platform and Execution Times . . . 184
5.4 Discussion . . . 187
5.4.1 Interpretation of Results . . . 187
5.4.2 Economic Interpretation . . . 189
5.4.3 Limitations . . . 190
5.5 Summary . . . 192

6 Multi-Dimensional Behavioral Evaluation of Agentic Stock Prediction Systems Using LLM Judges with Closed-Loop Reinforcement Learning Feedback **193**
6.1 Related Work and Research Gap . . . 194
6.2 Methodology . . . 197
6.2.1 System Overview and Behavioral Context . . . 197

6.2.2 Behavioral Trace Formalization . . . 200
6.2.3 Evaluation Episodes . . . 203
6.2.4 Evaluation Dimensions . . . 204
6.2.5 Rubric Design . . . 209
6.2.6 Prompt Design and Evaluation Pipeline . . . 212
6.2.7 Closed-Loop Reward Integration . . . 217
6.3 Experiments and Results . . . 222
6.3.1 Experimental Setup . . . 222
6.3.2 Perturbation Design . . . 224
6.3.3 Baseline System Performance . . . 227
6.3.4 Perturbation-Based Validation . . . 228
6.3.5 Cross-Model Agreement . . . 230
6.3.6 Per-Regime Score Analysis . . . 232
6.3.7 Predictive Validity Analysis . . . 234
6.3.8 Qualitative Analysis of LLM Judgments . . . 236
6.3.9 Closed-Loop Improvement . . . 238
6.3.10 Statistical Significance . . . 242
6.3.11 Framework Ablation . . . 244
6.4 Discussion . . . 247
6.4.1 Interpretation of Results . . . 247
6.4.2 Implications . . . 249
6.4.3 Limitations . . . 250
6.4.4 Connection to Prior Contributions . . . 252
6.5 Summary . . . 253

7 Conclusion and Future Work **255**
7.1 Summary of Contributions . . . 256
7.1.1 Graph-Based Multimodal Prediction with Sentiment Fusion . . . 256
7.1.2 Adaptive Regime-Aware Prediction with RL Control . . . 257
7.1.3 LLM-Based Behavioral Evaluation with Closed-Loop Feedback . . . 259
7.1.4 Dataset and Evaluation Framework . . . 261
7.2 Integrated Discussion . . . 261
7.3 Limitations . . . 265
7.4 Future Research Directions . . . 268
7.4.1 Scaling to Larger Stock Universes . . . 268
7.4.2 Dynamic Graph Construction . . . 269
7.4.3 Cross-Market and Cross-Asset Generalization . . . 269
7.4.4 LLM Judge Distillation . . . 270
7.4.5 Alternative Sentiment Sources . . . 271
7.4.6 Real-Time Deployment Architecture . . . 271
7.4.7 Multi-Pathway Regime Extension . . . 272
7.5 Concluding Remarks . . . 273

References **275**

A LLM Judge Prompt Template **284**
A.1 System Message 284
A.2 Behavioral-Trace Serialization 286
A.3 Output Schema and Failure-Label Vocabulary 286

B Supplementary Material for the Literature Review **288**
B.1 Standard Model Formulations 288
B.1.1 Statistical Time Series Models 288
B.1.2 Recurrent Architectures 289
B.1.3 Transformer Architectures 290
B.1.4 Graph Neural Networks 291
B.1.5 Soft Actor-Critic 292
B.2 Literature Survey Tables 292

List of Figures

1.1 Progression of thesis contributions. 3

2.1 Taxonomy of methodological areas reviewed in Chapter 2. 19

3.1 Data pipeline overview. 64
3.2 Temporal partitioning with key market events. 86

4.1 System architecture of the node transformer with BERT sentiment fusion. . 100
4.2 Single transformer layer architecture. 104
4.3 BERT sentiment extraction pipeline. 107
4.4 Adaptive fusion mechanism. 109

5.1 Overall method flowchart of the framework. 134
5.2 System architecture of the regime-aware prediction framework. 135
5.3 Context vector on two illustrative trading days. 137
5.4 Feature engineering pipeline. 138
5.5 Autoencoder architecture for regime detection. 139
5.6 Dual node transformer architecture. 142
5.7 Graph representation of stock relationships. 143
5.8 Soft Actor-Critic network architecture. 147
5.9 Multi-stage training pipeline. 150
5.10 Predicted versus actual closing prices for two representative stocks. 160
5.11 1-Day MAPE across all 16 evaluated models. 161
5.12 Learned threshold trajectory over the test period. 176
5.13 Cumulative equity curves of the trading strategy. 179

6.1 Agentic prediction architecture with LLM judge feedback. 200
6.2 LLM-as-a-Judge evaluation pipeline. 215
6.3 Sensitivity of closed-loop outcomes to feedback strength λ. 218
6.4 Perturbation validation: targeted versus off-target score drops. 229
6.5 Per-regime behavioral scores across evaluation dimensions. 233
6.6 Closed-loop progression across three validation cycles. 240

7.1 Complete integrated system architecture with cumulative performance. . . . 263

List of Tables

2.1 Summary of identified research gaps and thesis contributions. 62

3.1 Stock selection used in all experiments. 66
3.2 Cashtag search terms for sentiment data collection. 77
3.3 Summary of financial and sentiment datasets. 81
3.4 Temporal partitioning of the dataset. 85
3.5 Summary of evaluation metrics. 94

4.1 Model hyperparameters for the node transformer. 112
4.2 BERT sentiment classification performance. 116
4.3 MAPE and Theil's U across prediction horizons. 117
4.4 Directional accuracy for 1-day predictions. 118
4.5 Ablation study results. 119
4.6 MAPE by market volatility regime. 120
4.7 Paired t-test results. 122
4.8 Diebold-Mariano test results. 122
4.9 Portfolio backtest statistics. 124
4.10 Top-5 and bottom-5 learned edge weights. 125

5.1 Model hyperparameters for the regime-aware framework. 152
5.2 1-Day ahead closing price prediction results. 157
5.3 5-Day ahead closing price prediction results. 158
5.4 20-Day ahead closing price prediction results. 159
5.5 Per-stock 1-day ahead closing price results. 162
5.6 Ablation study of architectural components. 167
5.7 Cost profile of each ablation variant. 171
5.8 1-Day MAPE by volatility regime. 171
5.9 Sensitivity of 1-day MAPE to the AE cutoff and routing-threshold initialization. 172
5.10 Effect of the SAC stability penalty on threshold-trajectory statistics and MAPE. 177
5.11 Trading backtest results (2017 to 2025). 178
5.12 Statistical significance tests. 180
5.13 Per-seed 1-day MAPE and directional accuracy across five random seeds. . . 181
5.14 Hyperparameter search ranges for baseline models. 182
5.15 Per-model training time and inference latency. 185

6.1 Six evaluation dimensions for the LLM-as-a-Judge assessment. 208
6.2 Rubric anchor descriptions for each evaluation dimension. 210
6.3 Sensitivity of closed-loop outcomes to feedback strength λ. 218
6.4 Credit assignment mapping from dimensions to SAC subspaces. 220
6.5 Evaluation framework configuration parameters. 224
6.6 Baseline one-day-ahead prediction performance. 227
6.7 Perturbation validation results. 228

6.8 Mean dimension scores by volatility regime. 232
6.9 Spearman correlations between dimension scores and trading metrics. 234
6.10 Per-cycle progression of deficient scores and prediction metrics. 239
6.11 Closed-loop outcomes: single-judge variants versus three-judge ensemble. . . 244
6.12 Targeted versus uniform credit assignment comparison. 245

A.1 Failure-label vocabulary and default SAC action-subspace mapping. 287

B.1 Comparison of classical machine learning methods for stock prediction. . . . 292
B.2 Comparison of deep learning architectures for stock prediction. 293
B.3 Comparison of graph-based methods for stock prediction. 293
B.4 Sentiment analysis approaches for financial forecasting. 294

List of Abbreviations

A2C	Advantage Actor-Critic
AD	Adaptation Responsiveness
AE	Autoencoder
AI	Artificial Intelligence
AIC	Akaike Information Criterion
API	Application Programming Interface
AR	Autoregressive
ARCH	Autoregressive Conditional Heteroscedasticity
ARIMA	Autoregressive Integrated Moving Average
ATR	Average True Range
AUC	Area Under the Curve
BERT	Bidirectional Encoder Representations from Transformers
BIC	Bayesian Information Criterion
BiLSTM	Bidirectional Long Short-Term Memory
BNN	Bayesian Neural Network
CART	Classification and Regression Tree
CBOE	Chicago Board Options Exchange
CLS	Classification Token
CNN	Convolutional Neural Network
CoT	Chain-of-Thought
CRSP	Center for Research in Security Prices
CSS	Comprehensive Stock Sentiment
CTR	Confidence Tracking Rate
DA	Directional Accuracy
DCC	Dynamic Conditional Correlation
DCGAN	Deep Convolutional Generative Adversarial Network
DDPG	Deep Deterministic Policy Gradient
DGDNN	Decoupled Graph Diffusion Neural Network
DJIA	Dow Jones Industrial Average
DM	Diebold-Mariano
DNN	Deep Neural Network
DQN	Deep Q-Network
ECA	Efficient Channel Attention
EGARCH	Exponential Generalized Autoregressive Conditional Heteroscedasticity
EMA	Exponential Moving Average
EMH	Efficient Market Hypothesis
ER	Error Recovery
ESN	Echo State Network
F1	F1-Score
FC	Fully Connected
FDG	Frequency-Domain Graph

FFN	Feed-Forward Network
FMD	Financial Market Data
FMS	Financial Microblog Sentiment
GAF	Gramian Angular Field
GAN	Generative Adversarial Network
GARCH	Generalized Autoregressive Conditional Heteroscedasticity
GAT	Graph Attention Network
GCN	Graph Convolutional Network
GICS	Global Industry Classification Standard
GJR	Glosten-Jagannathan-Runkle
GNN	Graph Neural Network
GPOMS	Google Profile of Mood States
GPS	General, Powerful, Scalable
GPT	Generative Pre-trained Transformer
GPU	Graphics Processing Unit
GRU	Gated Recurrent Unit
HMM	Hidden Markov Model
HTML	Hypertext Markup Language
IC	Information Coefficient
ICC	Intra-Class Correlation Coefficient
IPO	Initial Public Offering
JSON	JavaScript Object Notation
KL	Kullback-Leibler
KNN	K-Nearest Neighbors
KPCA	Kernel Principal Component Analysis
LLM	Large Language Model
LM	Loughran-McDonald
LSTM	Long Short-Term Memory
MACD	Moving Average Convergence Divergence
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
MDD	Maximum Drawdown
MDP	Markov Decision Process
MiFID	Markets in Financial Instruments Directive
ML	Machine Learning
MLP	Multi-Layer Perceptron
MPNN	Message Passing Neural Network
MSI	Markov-Switching Intercept
MSIH	Markov-Switching Intercept and Heteroscedasticity
MSM	Markov-Switching Mean
MS-VAR	Markov-Switching Vector Autoregression
NF	NodeFormer
NLP	Natural Language Processing
OBV	On-Balance Volume
OGB	Open Graph Benchmark

OHLCV	Open, High, Low, Close, Adjusted Close, and Volume
PCA	Principal Component Analysis
PIP	Perceptually Important Points
PPO	Proximal Policy Optimization
RBF	Radial Basis Function
RC	Risk Calibration
RD	Regime Detection
ReLU	Rectified Linear Unit
RL	Reinforcement Learning
RLHF	Reinforcement Learning from Human Feedback
RMSE	Root Mean Squared Error
RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristic
RSI	Relative Strength Index
RT	Routing Appropriateness
S&P	Standard and Poor's
SAC	Soft Actor-Critic
SAN	Spectral Attention Networks
SC	Strategy Coherence
SEC	Securities and Exchange Commission
SETAR	Self-Exciting Threshold Autoregressive
SMA	Simple Moving Average
SOFNN	Self-Organizing Fuzzy Neural Network
SVM	Support Vector Machine
SVR	Support Vector Regression
TCN	Temporal Convolutional Network
TF-IDF	Term Frequency-Inverse Document Frequency
TLBO	Teaching-Learning-Based Optimization
URL	Uniform Resource Locator
VAE	Variational Autoencoder
VAR	Vector Autoregression
VIX	CBOE Volatility Index

1

Introduction

1.1 Background

On March 16, 2020, the S&P 500 fell 12% in a single session as COVID-19 lockdowns spread across North America. Within four weeks the index had lost over 30% of its value. By August it had recovered entirely and was setting new highs. An LSTM trained on the preceding decade of daily returns would have entered that period with parameters calibrated for a low-volatility, steadily rising market; the model's learned representations of "normal" one-day moves would have offered no useful guidance once daily swings exceeded five standard deviations. This is not a hypothetical failure. It is the defining weakness of the dominant paradigm in financial prediction: a single model, trained on aggregate historical data, applied identically regardless of whether the market is calm, panicking, or recovering.

The 2020 crash and recovery illustrate three interrelated limitations that this thesis sets out to address. First, stock prices are shaped not only by the quantitative features of each individual company but also by a web of cross-sectional dependencies: sector peers, supply chain partners, and competitors that transmit shocks across the market. When Apple reports weaker iPhone demand, the effect propagates to semiconductor suppliers, component manufacturers, and logistics firms within hours. Models that treat each stock as an isolated time series cannot represent this structure. Second, investor sentiment, amplified through social media, can drive price movements that have no basis in quantitative fundamentals. The GameStop short squeeze of January 2021 demonstrated how coordinated retail sentiment expressed on Reddit and Twitter could overwhelm institutional positioning and invalidate predictions grounded entirely in price and volume data. Third, the statistical properties of financial returns are not stationary: what constitutes a useful feature, an appropriate model

capacity, or a reliable pattern depends on whether the market is in a low-volatility trend, a crisis-driven drawdown, or a post-shock recovery phase [1].

These three limitations have each been partially addressed in isolation. Graph neural networks can encode inter-stock relationships [2], and transformer architectures capture long-range temporal dependencies through self-attention [3]. On the qualitative side, BERT-based models have shown the ability to extract structured sentiment from unstructured text at scale [4]. For non-stationary environments, autoencoders provide unsupervised anomaly detection [5] and reinforcement learning agents offer adaptive control policies that adjust to shifting reward distributions [6]. Yet no existing system integrates these capabilities into a coherent, progressively refined architecture where each component addresses a specific shortcoming of the previous stage.

This thesis develops such a system through three contributions that build on one another. The first introduces a node transformer architecture that represents the stock market as a graph and fuses quantitative features with BERT-derived sentiment signals. The second adds an autoencoder-based regime detector, dual specialized prediction pathways, and a Soft Actor-Critic reinforcement learning controller that learns to route data through the appropriate pathway based on prediction performance. The third proposes a framework for evaluating the behavioral quality of the resulting autonomous system using large language models as judges, closing the loop between evaluation and improvement. Figure 1.1 illustrates how these three contributions relate to one another and how each addresses a specific limitation of the preceding stage.

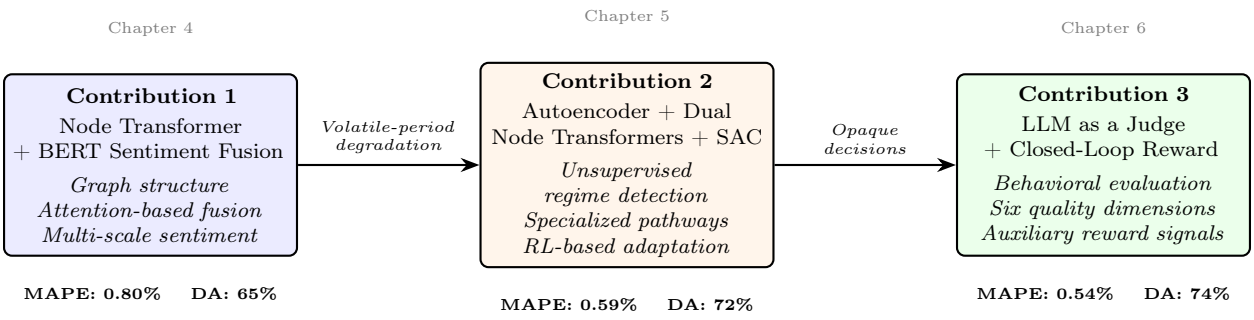

Figure 1.1: Progression of thesis contributions. Each stage addresses a specific limitation identified in the preceding architecture. Arrows indicate the motivation for the subsequent contribution, and performance metrics show the cumulative improvement across all three stages.

1.2 Challenges in Stock Price Forecasting

1.2.1 Market Noise and Non-Stationarity

Financial time series are contaminated by noise at every scale of observation. At tick level, microstructure effects such as bid-ask bounces, discrete price increments, and transient order book imbalances produce fluctuations that carry no information about future prices [7]. At the daily level, a stock's closing price reflects the last trade of the session, a single transaction that may not represent the consensus valuation implied by the full day's volume. Earnings surprises, analyst revisions, dividend announcements, and index rebalancing events inject transient dislocations that decay over hours or days.

The challenge for any predictive model is that the signal-to-noise ratio in financial data is extremely low. The predictable component of one-day-ahead equity returns is on the order of basis points, buried beneath daily moves that routinely span 1% to 3% of the stock price. A model that minimizes mean squared error on training data without adequate regularization will fit noise rather than signal, producing in-sample metrics that do not generalize. This overfitting failure mode is particularly severe for high-capacity models such as deep neural networks, where millions of parameters can memorize patterns in training data that are artifacts of the specific historical sample rather than stable statistical relationships.

Non-stationarity introduces a distinct but related problem. The joint distribution of returns, volumes, and volatility shifts over time as market microstructure evolves, new instruments are introduced, regulatory frameworks change, and the composition of market participants varies. The covariance structure among S&P 500 stocks during the 2003-2007 period of steady growth bore little resemblance to the correlation spike observed during the 2008 crisis, when pairwise correlations across sectors converged toward one and diversification benefits collapsed. A model trained on the pre-crisis period would have learned relationships that were not merely unhelpful during the crisis but actively misleading. Without explicit mechanisms to detect and respond to distributional shifts, even architecturally sophisticated models degrade at precisely the moments when accurate forecasts have the greatest economic value.

1.2.2 The Efficient Market Hypothesis

The efficient market hypothesis (EMH) provides the theoretical framework within which any prediction effort must position itself [8]. In its strong form, the EMH asserts that prices instantaneously incorporate all information, both public and private, rendering systematic prediction impossible. The semi-strong form restricts this claim to public information, while the weak form asserts only that historical prices are fully reflected in current prices.

Decades of empirical investigation have established that markets are neither perfectly efficient nor grossly inefficient [9, 10]. Large-capitalization, liquid equities in developed markets are priced efficiently enough that simple technical rules and linear models cannot reliably extract profit after transaction costs. However, several well-documented deviations from efficiency persist. Post-earnings announcement drift, in which stocks continue to move in the direction of an earnings surprise for weeks after the announcement, suggests that complex information takes time to be fully incorporated [8]. Momentum effects, where past winners continue to outperform over horizons of 3 to 12 months, have been documented

across markets and time periods. Sentiment-driven overreaction and subsequent correction create short-term reversals that can be detected with sufficiently granular data.

For the purposes of this thesis, the practical implication of partial efficiency is that the predictable component of returns is small, time-varying, and concentrated around specific information events. The design of a prediction system must therefore prioritize the extraction of weak signals from high-dimensional, noisy data rather than the pursuit of large, persistent patterns that are unlikely to exist in liquid equity markets. Transaction costs, market impact, and execution delays further constrain the set of statistically significant predictions that translate into economically viable trading signals, a distinction that the experimental evaluation in this thesis explicitly acknowledges.

1.2.3 Behavioral Dynamics and Investor Sentiment

The rational agent assumption underlying the EMH breaks down in systematic and well-documented ways. Prospect theory shows that investors weight losses roughly twice as heavily as equivalent gains, leading to disposition effects where winning positions are sold prematurely and losing positions are held too long [11]. Anchoring causes investors to fixate on salient reference points such as round numbers, 52-week highs, or purchase prices, producing support and resistance levels that have no fundamental justification but influence trading behavior. Confirmation bias leads investors to seek and overweight information that supports their existing positions while discounting contradictory evidence.

These individual biases aggregate into market-level phenomena. Herding behavior, in which investors mimic the trades of others rather than forming independent valuations, amplifies price trends beyond what fundamentals justify and contributes to the formation of bubbles. Fear cascades during sell-offs accelerate drawdowns as stop-loss orders, margin calls, and panic selling feed upon one another. The result is a market whose short-term dynamics are shaped as much by psychology as by information.

Social media has transformed the observability of these dynamics. Before the mid-2000s, investor sentiment was measured indirectly through surveys, options market implied volatility, or mutual fund flows, all of which were lagging and coarsely sampled. Platforms such as X (formerly Twitter) now provide a real-time, high-frequency channel through which millions of participants express views on specific stocks, sectors, and market conditions. The volume and polarity of this discussion contain leading indicators of price moves, particularly around earnings announcements, product launches, and macroeconomic releases where sentiment diverges from consensus expectations. Natural language processing models, and BERT [4] in particular, have made it feasible to extract structured sentiment scores from these unstructured text streams at scale, creating a potential input modality that pure price-based models cannot access.

The technical challenge lies not in computing sentiment scores but in integrating them with quantitative features in a principled way. Sentiment is noisy, subject to bot activity, and its predictive value varies with market conditions: sentiment signals may be strongly informative during earnings weeks and nearly useless during low-news periods. A naive concatenation of sentiment scores with price features conflates modalities whose informativeness varies independently. An effective integration mechanism must learn when and how much to weight sentiment relative to quantitative signals, adapting the fusion dynamically based on current conditions.

1.2.4 Inter-Stock Dependencies and Market Structure

Financial markets have a rich relational structure that standard time series models discard entirely. At the most visible level, sectoral co-movement links companies exposed to common demand and cost drivers: energy stocks move with oil prices, bank stocks respond to interest rate expectations, and semiconductor companies share exposure to global chip demand cycles. Beyond sector membership, supply chain dependencies propagate firm-specific shocks

through vertical relationships, so that when a major automaker reports weak sales the effect ripples through tire manufacturers, steel suppliers, and auto-parts distributors within days. Competitive dynamics layer on top of these cooperative links, creating inverse relationships where one firm's market share gain corresponds to a peer's loss. Cutting across all of these firm-level channels, macroeconomic exposures such as rising interest rates compress price-to-earnings multiples across growth stocks regardless of individual company fundamentals, inducing correlated responses across broad market segments.

These dependencies are not static. Correlation structures shift as business relationships evolve, new competitors enter markets, and macroeconomic conditions change. During periods of market stress, the correlation structure undergoes a particularly dramatic transformation. Pairwise correlations among stocks increase sharply as idiosyncratic factors are overwhelmed by systematic risk, a pattern documented repeatedly across international equity markets and confirmed in every major crisis from the 1997 Asian financial contagion through the 2020 pandemic sell-off. The practical consequence is that a model trained on average-period correlations will underestimate the degree of co-movement during crises, precisely when cross-sectional information is most valuable for risk management and prediction.

Graph-based representations provide a natural formalism for encoding this structure. Each stock corresponds to a node, and edges encode pairwise relationships weighted by strength. The graph can combine multiple relationship types: sector membership (a binary attribute), return correlation (a continuous, time-varying weight), and supply chain proximity (derived from revenue exposure data). Graph neural networks [2] and graph transformer architectures [12] propagate information along edges during the forward pass, allowing each node's representation to be informed by the states of its neighbors. The resulting predictions for any individual stock are conditioned on the current states of related stocks, capturing cross-sectional dynamics that univariate models miss [13, 14].

The challenge in applying graph methods to stock markets lies in graph construction and adaptation. Correlation-based edges computed over a fixed historical window capture

lagged relationships that may not persist. Learnable edge weights that adapt during training can discover more robust dependency structures but risk overfitting to the training period's specific correlation regime. A successful approach must balance prior structural knowledge (sector labels, known supply chain relationships) with data-driven edge weight learning, and it must handle the fact that the optimal graph topology differs between calm and volatile market periods.

1.2.5 Regime Shifts and Structural Breaks

The assumption that a single model can handle all market conditions is arguably the most consequential simplification in published stock prediction work. Empirical evidence from decades of financial economics research establishes that equity returns are generated by a mixture of processes whose parameters vary across market regimes [1]. During low-volatility bull markets, daily returns follow a distribution with moderate mean, low variance, and weak serial correlation. During crisis episodes, the distribution shifts: the mean turns sharply negative, variance increases by a factor of three to five, serial correlation strengthens (due to momentum in selling pressure), and tail events that would be extreme outliers under the normal-regime distribution occur repeatedly.

These regime differences have direct consequences for model design. Technical momentum indicators may be informative during trending markets but misleading during mean-reverting recovery phases, while sentiment signals become relatively more important during crisis periods when fear-driven selling dominates fundamental valuation. At the same time, cross-sectional correlations tighten during stress, making graph structure more informative precisely when univariate models fail. A model with fixed parameters calibrated to average conditions will inevitably underweight features that are critical during crises and overweight features that are informative only during calm periods.

The problem of regime detection adds a layer of difficulty. Traditional approaches rely on hidden Markov models (HMMs), which model the regime as a latent state with Gaussian emission distributions and transition probabilities estimated from historical data [1]. HMMs require specifying the number of regimes in advance, and their emission distributions may be too restrictive to capture the complexity of real market behavior. Threshold-based rules on observable indicators such as VIX or realized volatility are simple to implement but arbitrary: the choice of threshold directly determines which periods are labeled as volatile, and small changes in the threshold can reclassify large portions of the data. Supervised approaches require labeled training data identifying which historical periods constitute crises, but such labels are inherently subjective and backward-looking.

An alternative is to learn regime boundaries from downstream prediction performance rather than from predefined indicators. If an autoencoder trained on stable-market data produces high reconstruction error on a new observation, that observation is statistically unusual relative to the learned representation of normalcy. The magnitude of this error can serve as a continuous anomaly score that does not require threshold specification by a human. A reinforcement learning controller can then learn to map this score to routing decisions (which prediction pathway to use) and blending weights (how to combine outputs from multiple pathways), optimizing these decisions directly against forecasting accuracy rather than against a proxy measure of regime membership.

1.2.6 Regulatory and Macroeconomic Factors

Policy interventions and regulatory changes create structural breaks in financial time series that data-driven models cannot learn from historical patterns alone, because each intervention is novel in its specifics even if the general mechanism (fiscal stimulus, monetary tightening, industry regulation) is familiar. The 2008 crisis involved a cascade of failures in securitized mortgage products that had no direct precedent. The 2020 pandemic response included

direct payments to individuals, payroll protection for businesses, and near-zero interest rates sustained for two years, a policy combination without historical analog. The Federal Reserve's aggressive rate-hiking cycle beginning in 2022 reversed a decade of accommodative policy, reshaping the relative valuation of growth versus value stocks in ways that models trained on the low-rate era could not have anticipated.

These factors matter for model design in two ways. First, they serve as a reminder that any out-of-sample evaluation period includes conditions that were not represented in training data, and a robust architecture must degrade gracefully rather than catastrophically when encountering unfamiliar dynamics. Second, while the specific policy action may be unpredictable, its effects propagate through observable channels: prices, volumes, implied volatility, and sentiment shift in detectable patterns. A regime-aware architecture that monitors these observable signatures can recognize departures from normal dynamics and adjust its processing accordingly, even if it cannot predict the policy action itself.

1.3 Motivation

Three empirical regularities of financial markets motivate the architectural commitments of this thesis. Each is well-documented in the financial economics literature, and each maps directly onto one of the design choices that distinguishes the systems developed in subsequent chapters from a conventional univariate time-series predictor.

The first regularity is that stock prices are relational. A given company's price movement is shaped not only by its own fundamentals and order flow but also by the behavior of related firms, including sector peers, supply-chain partners, and competitors that share exposure to common risk factors. Industry-level information has been shown to diffuse across firms with measurable lead-lag patterns, and economic links such as customer-supplier relationships generate predictable cross-stock return effects [15, 16]. A predictor that treats each stock as

an isolated sequence cannot represent these cross-sectional dependencies, which motivates the graph-based representation underlying the first contribution of this thesis (Chapter 4).

The second regularity is that markets respond to qualitative information that price and volume series do not directly capture. The tone of financial news, the language used in corporate disclosures, and the sentiment expressed in social media discussion have all been shown to forecast subsequent returns and trading activity [17–19]. The growing availability of high-frequency text data, combined with language models that can extract structured sentiment from unstructured text, makes the integration of qualitative signals alongside quantitative features a natural design choice for a modern prediction system. This is the rationale for coupling the graph-temporal backbone in Chapter 4 with a fine-tuned BERT sentiment pathway and an adaptive fusion gate.

The third regularity is that market behavior is non-stationary across regimes. The statistical properties of returns, volatilities, and cross-asset correlations differ substantially between calm bull markets, crisis-driven drawdowns, and post-shock recovery phases, and these regime shifts have been formally characterized through Markov-switching and related models for several decades [1, 20, 21]. A predictor whose parameters are calibrated for average conditions will tend to underperform during regime transitions and crises, the periods where accurate forecasting carries the greatest economic value. This regime non-stationarity motivates the unsupervised regime detection, dual specialized prediction pathways, and reinforcement learning controller introduced in the second contribution (Chapter 5).

Together these three regularities motivate the first two contributions of this thesis: a graph-based, sentiment-fused architecture in Chapter 4, and an adaptive regime-aware extension with reinforcement learning control in Chapter 5. As the resulting system grows in architectural complexity, with regime detectors, routing mechanisms, and an adaptive controller layered on top of the prediction backbone, its intermediate decisions become opaque to the aggregate output metrics traditionally used to evaluate prediction quality. This opacity motivates the third contribution, an LLM-based behavioral evaluation framework with closed-loop

reward integration developed in Chapter 6, which assesses the appropriateness of intermediate decisions and feeds the assessment back into the controller. The literature underlying each of these contributions, together with the specific research gaps they address, is reviewed in detail in Chapter 2.

1.4 Thesis Contributions

This thesis makes four contributions that span architecture design, adaptive control, evaluation methodology, and experimental validation. Each contribution builds on the preceding one, and together they constitute a progression from a single-model baseline to a self-evaluating, regime-aware prediction system.

The first contribution, presented in Chapter 4, is a node transformer architecture with attention-based sentiment fusion. The stock market is modeled as a fully connected graph of 20 S&P 500 equities, where learnable edge weights are initialized from sector co-membership and 60-day rolling return correlations. Each node receives a 16-dimensional input comprising OHLCV data and 11 derived technical indicators. A fine-tuned BERT model processes approximately 4.2 million social media posts to produce per-stock sentiment scores at daily resolution, which enter the node transformer through three complementary mechanisms: multi-scale temporal sentiment features, key-modulation within the attention computation, and an adaptive fusion gate whose openness is conditioned on rolling volatility and sentiment magnitude. The architecture achieves 0.80% MAPE and 65% directional accuracy on one-day-ahead forecasts. Ablation studies isolate a 15% relative error reduction attributable to graph structure and an additional 10% from sentiment integration, with the largest sentiment gains occurring during earnings announcement windows where error decreases by 25% relative to the price-only model. The novelty of this contribution lies in unifying graph-relational structure, temporal dynamics, and BERT-derived sentiment within a single end-to-end architecture, together with an adaptive fusion gate whose openness is dynamically conditioned on rolling

volatility and sentiment magnitude rather than set by the fixed weighting schemes used in prior multimodal stock prediction work.

The second contribution, presented in Chapter 5, extends the first by introducing an autoencoder-gated dual node transformer with reinforcement learning control. An autoencoder trained exclusively on stable-market data learns to reconstruct normal price patterns, and observations that produce high reconstruction error are flagged as regime deviations. Rather than applying a fixed threshold to this error signal, a Soft Actor-Critic reinforcement learning controller observes a state comprising recent prediction errors, reconstruction losses, and volatility indicators, and outputs continuous actions that set the regime detection threshold and the blending weights between dual node transformer pathways. One pathway is optimized for normal conditions while the other incorporates event-context features including volatility regime embeddings, sentiment spike indicators, earnings proximity flags, and cross-asset stress signals. The controller discovers which threshold settings and blending ratios minimize downstream prediction error, learning regime boundaries that would be difficult to specify manually. The full system achieves 0.59% MAPE and 72% directional accuracy, with high-volatility MAPE held below 0.85% compared to over 1.5% for the single-path baseline. The novelty of this contribution lies in using reconstruction error from a stable-market autoencoder as a continuous, label-free regime signal, and in delegating the regime threshold and the pathway-blending weights to a reinforcement learning controller that learns them directly from downstream prediction error, removing the manual regime specifications and fixed thresholds that constrain conventional regime-aware forecasting.

The third contribution, presented in Chapter 6, addresses the opacity of intermediate decisions in the multi-stage pipeline by introducing an LLM-based behavioral evaluation framework with closed-loop reward integration. Behavioral trace logs record the system's regime classifications, routing choices, parameter adjustments, and confidence estimates along with their temporal context. Structured prompts present these traces to large language models, which score decision quality along six dimensions: regime detection accuracy, routing

appropriateness, adaptation responsiveness, risk calibration, strategy coherence, and error recovery. Perturbation-based validation confirms that each dimension responds selectively to targeted degradation of the corresponding system component, with cross-model consistency reaching Krippendorff's $\alpha = 0.85$ [22]. Composite behavioral scores correlate with realized Sharpe ratios at Spearman $\rho = 0.72$. By translating dimensional scores into auxiliary reward signals for the SAC controller, the framework closes the evaluation loop and yields further improvement to 0.54% MAPE and 74% directional accuracy. The novelty of this contribution lies in the first systematic application of structured LLM-based behavioral evaluation to autonomous financial prediction systems, and in closing the evaluation loop by translating per-dimension judge scores into auxiliary reward signals that fine-tune the SAC controller, enabling correction of intermediate-decision weaknesses that aggregate output metrics cannot reveal.

The fourth contribution is a multi-source dataset and rigorous evaluation framework that underpins all experimental results. The dataset integrates historical OHLCV data with 11 derived technical indicators for 20 S&P 500 companies spanning January 1982 to March 2025, augmented with sentiment scores from the SemEval-2017 corpus [23] for BERT fine-tuning and approximately 4.2 million social media posts collected from 2007 to 2025. Temporal splits enforce strict chronological separation with training data from 1982 to 2010, validation from 2011 to 2016, and testing from 2017 to 2025, ensuring no information leakage across boundaries. All performance comparisons are supported by paired t-tests ($p < 0.05$), and ablation studies systematically remove each component to quantify its marginal contribution. The novelty of this contribution lies in assembling a temporally aligned, multi-source dataset that pairs more than four decades of OHLCV history with eighteen years of high-volume social media sentiment, and in establishing an evaluation protocol that applies strict chronological partitioning, paired statistical testing, and component-level ablations consistently across all three architectural contributions.

All three architectural contributions are evaluated in an offline backtesting setup rather than in live trading. Models are trained on historical data, predictions are generated on held-out chronological windows, and the reported metrics (MAPE, directional accuracy, and economic measures such as Sharpe ratio and maximum drawdown) are computed retrospectively against realized historical price series. No system component is connected to a live brokerage and no real capital is deployed. Effects that arise only in live execution, including order book interaction, market impact from the strategy's own activity, fill latency, partial fills, and broker-specific transaction frictions, are therefore outside the scope of the present evaluation, and any extension to live deployment would require a separate engineering and risk-management effort.

1.5 Research Publications

The research presented in this thesis has produced the following publications:

1. Al Ridhawi, M., & Al Osman, H. (2023). Stock price forecasting from sentiment and financial stock data using machine learning. *Proceedings of the Canadian Conference on Artificial Intelligence*. `https://doi.org/10.21428/594757db.40c1a462`

2. Al Ridhawi, M., Haj Ali, M., & Al Osman, H. (2026). Stock market prediction using node transformer architecture integrated with BERT sentiment analysis. *IEEE Access*, vol. 14, pp. 72613–72631. doi: 10.1109/ACCESS.2026.3691980.

3. Al Ridhawi, M., Haj Ali, M., & Al Osman, H. (2026). Adaptive regime-aware stock price prediction using autoencoder-gated dual node transformers with reinforcement learning control. *Submitted to Applied Intelligence*. Under revision. arXiv:2603.19136.

4. Al Ridhawi, M., Haj Ali, M., & Al Osman, H. (2026). Multi-dimensional behavioral evaluation of agentic stock prediction systems using LLM judges with closed-loop

reinforcement learning feedback. *Submitted to Applied Artificial Intelligence*. Under review. arXiv:2605.05739.

1.6 Thesis Organization

The remainder of this thesis is organized into six chapters. The literature review in Chapter 2 traces the evolution of stock prediction from statistical time series models through deep learning to graph-based and transformer-based architectures, surveys the application of sentiment analysis, autoencoders, and reinforcement learning to financial forecasting, and positions the contributions of this thesis relative to the existing body of work.

Chapter 3 establishes the shared experimental foundation for all three contributions, describing the 20 stocks drawn from the S&P 500, the financial market data and feature engineering pipeline, the sentiment data collection and preprocessing methodology, the temporal partitioning strategy, and the evaluation metrics used throughout the subsequent chapters.

Chapter 4 presents the node transformer with BERT sentiment integration, covering the graph representation, architecture design, BERT fine-tuning pipeline, the three-part fusion mechanism, and experimental results that include comparisons against eight baseline models and ablation studies isolating the contribution of each component.

The regime-aware extension is developed in Chapter 5, which details the autoencoder architecture and its training on stable-market data, the dual node transformer design with event-context features, the SAC controller formulation and its training procedure, and experiments demonstrating improvements across both aggregate metrics and regime-specific performance.

Chapter 6 turns to the problem of evaluating intermediate decision quality, formalizing the six evaluation dimensions used by the LLM-based framework, describing the structured

prompt design, validating dimension specificity through perturbation experiments, and demonstrating how closed-loop reward integration converts qualitative behavioral assessments into quantitative learning signals.

The thesis concludes in Chapter 7 with a synthesis of findings, a discussion of implications for adaptive financial prediction systems and autonomous AI evaluation, an acknowledgment of limitations, and proposals for future research directions.

2

Literature Review

Stock price forecasting sits at the intersection of financial economics, signal processing, and machine learning. The problem has attracted sustained attention from all three communities because equity prices encode a dense, noisy, and time-varying mixture of fundamental value, investor psychology, and market microstructure effects. This chapter surveys the methodological landscape that frames the contributions of this thesis. The review begins with classical statistical models that established the foundational concepts of stationarity, volatility clustering, and serial dependence. It then traces the progression through classical machine learning, deep neural networks, transformers, and graph-based architectures that collectively expanded the representational capacity available for financial prediction. Subsequent sections examine three complementary dimensions that the thesis integrates with these predictive architectures: sentiment analysis from unstructured text, unsupervised regime detection, and reinforcement learning for adaptive control. The chapter concludes with a review of LLM-based evaluation frameworks for autonomous systems and a synthesis of research gaps that motivate the present work. Figure 2.1 provides an overview of the methodological areas covered in this chapter and indicates how each connects to the contributions of this thesis.

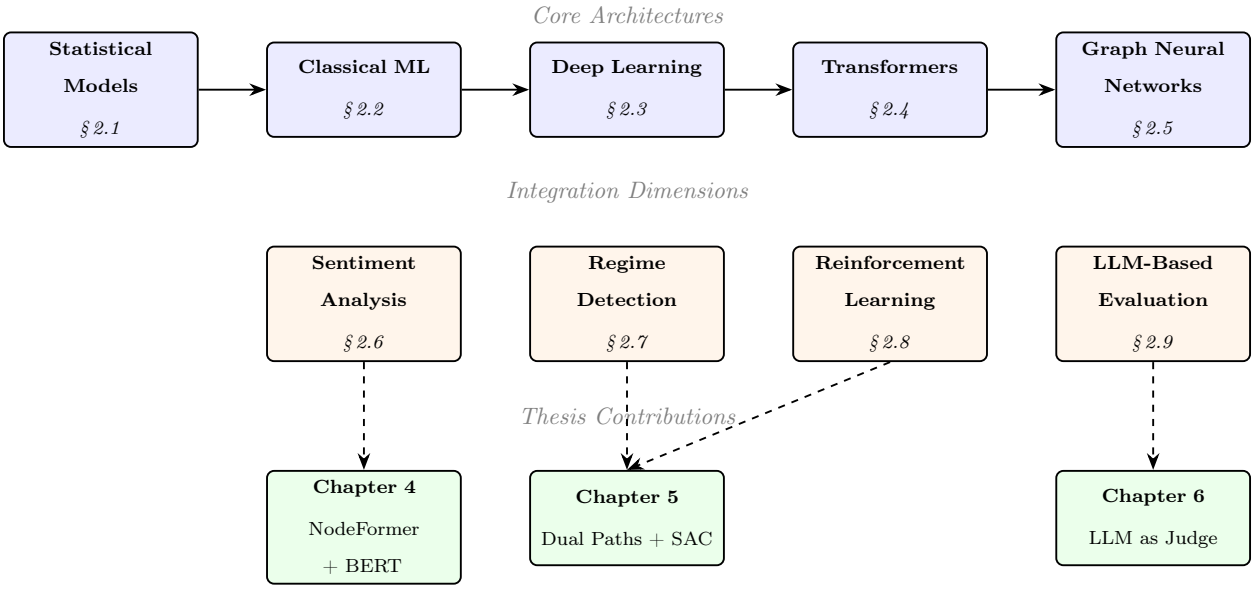

Figure 2.1: Taxonomy of the methodological areas reviewed in this chapter. The top row traces the evolution of core prediction architectures (§ 2.1–2.5). The middle row shows the integration dimensions (§ 2.6–2.9) that this thesis combines with the core models. Dashed arrows indicate the primary methodological connections to the thesis chapters; Chapter 4 additionally extends the graph neural network framework of § 2.5, and Chapter 5 builds on the deep learning and transformer architectures of § 2.3–2.4.

2.1 Statistical Time Series Models

The quantitative study of financial time series began with the recognition that asset returns exhibit statistical regularities that simple random walk models cannot explain. Mandelbrot [24] first demonstrated that commodity price changes follow heavy-tailed distributions with variance that clusters in time, an observation that challenged the Gaussian assumptions underlying classical portfolio theory. This foundational insight has been repeatedly confirmed in modern markets; Cont [25] documented that the same stylized facts (heavy tails, volatility clustering, absence of linear autocorrelation, and leverage effects) persist across asset classes, time periods, and market microstructures, establishing them as universal properties of financial returns.

2.1.1 Autoregressive Integrated Moving Average Models

The ARIMA framework, formalized by Box et al. [26], provides a systematic methodology for modeling univariate time series through three components: autoregression, differencing, and moving average smoothing. An ARIMA(p, d, q) model combines an autoregressive polynomial of order p, d orders of differencing, and a moving-average polynomial of order q; its formal specification and terms are given in Appendix B (Equation B.1). The Box-Jenkins methodology selects model order through iterative identification, estimation, and diagnostic checking, with the Akaike Information Criterion (AIC) providing a principled trade-off between goodness of fit and model complexity.

In the context of stock forecasting, Low and Sakk [27] compared ARIMA and LSTM networks across ten exchange-traded funds and found that ARIMA achieved comparable accuracy to LSTM for longer-term predictions, a result that challenges the assumption that complex architectures uniformly dominate simpler ones. Wahyudi [28] applied ARIMA to Indonesian stock prices and demonstrated effectiveness for short-term volatility prediction using AIC-based model selection. These findings establish ARIMA as a meaningful baseline: when a deep learning model cannot outperform ARIMA on a given dataset, the deep model is likely fitting noise rather than signal.

The strength of ARIMA lies in its interpretability. Each coefficient carries a clear statistical meaning, confidence intervals are well-defined, and diagnostic tests for residual autocorrelation and heteroscedasticity are standard. The weakness is equally definitive. Financial returns exhibit nonlinear dependencies, asymmetric responses to positive and negative shocks, and regime-dependent behavior that violate the linearity and stationarity assumptions upon which the ARIMA framework rests.

2.1.2 Vector Autoregression

Vector Autoregression (VAR), introduced by Sims [29] and widely used in modern financial econometrics [30], extends the univariate ARIMA framework to multivariate settings by modeling each variable as a linear function of its own lags and the lags of all other variables in the system. For a K-dimensional time series $\mathbf{y}_t$, the VAR(p) model captures these dependencies through $K \times K$ coefficient matrices, an intercept vector, and a multivariate white-noise term; its full specification appears in Appendix B (Equation B.2). VAR models the dynamic interactions among multiple time series simultaneously, and Granger [31] formalized the concept of Granger causality within this framework, providing a statistical test for whether the lagged values of one variable improve forecasts of another.

For stock prediction, VAR captures lead-lag relationships between related securities and the directional influence of macroeconomic variables on equity returns. Its limitation is computational and structural: the number of parameters grows quadratically with the number of variables and linearly with the lag order, making it impractical for high-dimensional systems such as the 20-stock universe considered in this thesis. Sparse VAR variants and Lasso-regularized estimation address the dimensionality issue partially, but the linearity constraint remains binding.

2.1.3 GARCH Family Models

The observation that financial return volatility clusters in time motivated the Autoregressive Conditional Heteroscedasticity (ARCH) model of Engle [32], which models the conditional variance of returns as a function of past squared residuals. Bollerslev [33] generalized this to the GARCH(p, q) specification, which models the conditional variance σ_t^2 as a function of past squared residuals and past conditional variances; the full expression and its parameter constraints are given in Appendix B (Equation B.3). The model captures the empirical

regularity that large returns (positive or negative) tend to be followed by large returns, and small returns by small returns.

GARCH and its extensions (EGARCH, GJR-GARCH, multivariate DCC-GARCH) remain standard tools in risk management and option pricing, with recent surveys confirming their continued relevance alongside machine learning alternatives [34]. For point prediction of returns, however, GARCH models the conditional variance rather than the conditional mean, which limits their direct applicability to the price forecasting task that is the primary objective of this thesis. The conditional mean equation in a GARCH model is typically a simple constant or AR(1) specification, providing minimal predictive power for the direction or magnitude of future returns. Nevertheless, volatility estimates from GARCH serve as valuable input features for more complex prediction architectures, and the GARCH framework provides a rigorous benchmark for evaluating whether deep learning models genuinely capture conditional heteroscedasticity or merely overfit to it.

2.1.4 Limitations of Statistical Models for Financial Prediction

Statistical time series models share a set of structural limitations that motivate the progression to machine learning and deep learning approaches. The linearity assumption at the core of ARIMA and VAR excludes interaction effects between features, threshold behaviors, and the nonlinear dynamics that characterize financial markets during stress periods. Compounding this restriction, univariate models such as ARIMA and GARCH process each stock independently, discarding the cross-sectional information that supply chain connections, sectoral co-movement, and competitive dynamics provide. VAR addresses the multivariate dimension partially, but the number of parameters grows quadratically with the number of variables, making it impractical for the high-dimensional stock universe considered in this thesis. Perhaps most fundamentally, the stationarity requirement that underpins parameter estimation is at tension with the non-stationary nature of financial time series:

structural breaks induced by regulatory changes, technological disruption, or macroeconomic regime shifts violate the assumption that the data-generating process remains constant over the training and prediction periods.

These limitations do not diminish the value of statistical models as benchmarks. ARIMA and GARCH establish interpretable baselines against which more complex architectures must demonstrate meaningful improvement. Any deep learning model that cannot consistently outperform ARIMA on out-of-sample data is likely capturing noise rather than genuine predictive signal, a test that a surprising number of published deep learning results fail when evaluated with proper temporal splitting and statistical significance testing.

2.2 Classical Machine Learning for Stock Prediction

Classical machine learning methods brought nonlinear modeling capacity to financial prediction without requiring the explicit specification of functional forms that statistical models demand. The trade-off was a shift from model interpretability to feature engineering: the quality of predictions became inseparable from the quality and relevance of the input features, which required substantial domain expertise to construct. Early work by Markowska-Kaczmar and Dziedzic combined a supervised shallow neural network with the perceptually important points (PIP) technique for data reduction, achieving 84% accuracy in classifying shortened stock time series data, though performance was limited to these reduced representations and may not generalize to full-resolution data. Sigo explored nonlinear movement patterns of leading stocks on the Bombay Stock Exchange from 2008 to 2017 using a shallow neural network, demonstrating that even simple architectures could capture meaningful patterns in emerging market equities.

2.2.1 Support Vector Machines

Support Vector Machines (SVM) learn a hyperplane that maximizes the margin between classes (for classification) or fits a tube of specified width around the data (for regression via SVR). Kernel functions project the input space into higher-dimensional feature spaces where linear separation becomes feasible, enabling SVMs to capture nonlinear decision boundaries without explicitly computing the high-dimensional representation.

Nayak et al. [35] developed a hybrid SVM-KNN model for Indian stock market indices, where the SVM provided margin-based classification and KNN contributed local neighborhood voting. The hybrid model outperformed each individual classifier in mean squared error across multiple Indian benchmark indices. Siddique and Panda [36] combined kernel PCA with SVR and teaching-learning-based optimization (TLBO), reporting that the KPCA-SVR-TLBO configuration outperformed orthogonal forward selection with SVR on the Tata Steel index, with the dimensionality reduction step reducing MSE by removing redundant correlated features before regression. Xiao et al. combined Singular Spectrum Analysis with SVM for stock price analysis and forecasting, using the decomposition to separate trend, seasonal, and noise components before feeding the cleaned signals into SVM classifiers. Ismail et al. predicted the direction of stock price movement using a hybrid method combining multiple ML techniques including SVM, achieving improved directional accuracy through multi-model consensus. These results established that SVMs could capture nonlinear patterns in engineered feature sets, though performance remained tightly coupled to the quality of input features and the specific kernel and regularization choices.

2.2.2 Ensemble Methods

Ensemble methods aggregate predictions from multiple base learners to reduce variance (bagging) or bias (boosting). Random forests construct many decision trees on bootstrapped samples with random feature subsets and average their predictions. Gradient boosting

methods, particularly XGBoost, build trees sequentially where each tree corrects the errors of the ensemble to that point.

Basak et al. [37] applied XGBoost to long-term stock price forecasting using technical stock indicators as input features, achieving F-scores ranging from 0.82 to 0.97 for 60-day and 90-day prediction periods on Apple and Yahoo stocks. The study validated that tree-based boosting could capture complex nonlinear interactions between technical indicators that linear models miss. In a comparative study, Ballings et al. evaluated Random Forest, AdaBoost, Kernel Factory, Neural Networks, Logistic Regression, SVMs, and KNN for long-term stock price prediction and found Random Forest to be the top-performing algorithm across their test universe, outperforming all other approaches in terms of area under the ROC curve. Milosevic introduced an innovative approach that classified stocks as "good" (increased by 10% or more over one year) or "bad" using 11 fundamental ratios, with Random Forest achieving the best F-score of 0.751 among all tested methods. Syukur and Istiawan investigated prediction of the LQ45 index on the Indonesia Stock Exchange and similarly found Random Forest achieving the best performance compared to C4.5, SVM, Logistic Regression, Naïve Bayes, and Linear Discriminant Analysis.

The practical advantage of ensemble methods lies in their robustness to overfitting and their built-in feature importance metrics, which provide interpretable guidance on which inputs drive predictions. Their limitation for financial forecasting is shared with all classical ML methods: they operate on fixed-length feature vectors that must be engineered manually, and the ceiling on predictive performance is set by the informativeness of those features rather than by the learning algorithm. An ensemble that aggregates hundreds of trees over poorly chosen indicators will converge to a confident but uninformative consensus. The features themselves are often calibrated to specific markets and time periods, meaning that a feature set designed for large-cap US equities during a bull market may lose its discriminative power when applied to emerging market stocks or to the same US stocks during a structural break.

2.2.3 Bayesian Methods

Bayesian approaches to stock prediction incorporate prior beliefs about model parameters and update them using observed data. Bayesian Neural Networks (BNNs) replace fixed weights with probability distributions, providing uncertainty estimates alongside point predictions. Chandra and He explored BNNs for stock forecasting and reported that the probabilistic framework helped identify periods where model confidence was low, providing a natural mechanism for risk-aware prediction. Malagrino et al. applied BNNs to predict the daily closing direction of the iBOVESPA index, achieving a mean accuracy of approximately 71% with peak accuracy near 78%.

The Bayesian framework's principal contribution to financial prediction is not raw accuracy but principled uncertainty quantification. Markets are inherently stochastic, and a model that reports high confidence in a prediction during a volatile period is arguably more dangerous than one that produces a wider prediction interval. This perspective informs the regime-aware design of the system developed in Chapter 5, where different model pathways are activated based on the estimated uncertainty of the current market state. Table B.1 in Appendix B summarizes these representative classical machine learning methods, highlighting the diversity of approaches, markets, and evaluation metrics across studies.

2.3 Deep Learning Architectures

Deep learning methods marked a fundamental transition from manual feature engineering to end-to-end representation learning. By stacking multiple nonlinear transformations, deep neural networks learn hierarchical features directly from raw or lightly processed inputs, reducing dependence on domain expertise for feature construction [38]. This capacity proved especially effective for financial time series, where the relevant features may be complex

temporal patterns, cross-frequency interactions, or nonlinear combinations of indicators that a human designer would not specify.

2.3.1 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) extract hierarchical spatial patterns through local receptive fields, weight sharing, and pooling operations. Originally designed for image recognition, CNNs were adapted for financial time series by treating temporal windows of features as two-dimensional inputs.

Chandar [39] proposed TI-CNN, a method that converts ten technical indicators into images using Gramian Angular Fields before processing them through a CNN. The approach was evaluated on multiple stock datasets and demonstrated that the image-based representation of indicator dynamics yielded improved classification accuracy for buy, sell, and hold signals compared to feeding raw indicator values into the network. Gunduz et al. applied a CNN model to stocks in the Borsa Istanbul 100 index using features derived from indicators, prices, and temporal information, reporting that the CNN's ability to learn hierarchical patterns from the specially ordered feature matrices produced competitive trading signals. Wen et al. [40] applied CNNs to S&P 500 data by arranging multiple features across time into a matrix, distinguishing the spatial structure inherent in the time series and demonstrating that the convolution operation captured local feature interactions within short temporal windows that improved trend prediction relative to fully connected baselines.

The strength of CNNs for financial data lies in their ability to detect local patterns such as candlestick formations, crossover signals between moving averages, and short-term momentum reversals. Their weakness is equally clear: the fixed receptive field of convolutional kernels limits the temporal range of captured dependencies. A CNN with k-sized kernels stacked L layers deep has an effective receptive field of $O(k \cdot L)$ time steps, which may be insufficient to

capture long-range dependencies such as the influence of quarterly earnings on subsequent weeks of price action.

2.3.2 Recurrent Neural Networks

Recurrent Neural Networks (RNNs) process sequential data by maintaining a hidden state that propagates information across time steps. The basic RNN updates its hidden state as $\mathbf{h}_t = \sigma(\mathbf{W}_{hh}\mathbf{h}_{t-1} + \mathbf{W}_{xh}\mathbf{x}_t + \mathbf{b})$, where σ is a nonlinear activation. While conceptually suited to time series, standard RNNs suffer from the vanishing gradient problem: gradients shrink exponentially during backpropagation through time, preventing the network from learning dependencies that span more than approximately 10 to 20 time steps.

Long Short-Term Memory Networks

The Long Short-Term Memory (LSTM) architecture, introduced by Hochreiter and Schmidhuber [41], addresses the vanishing gradient problem through a gating mechanism that controls information flow. The LSTM cell maintains a cell state $\mathbf{c}_t$ alongside the hidden state $\mathbf{h}_t$, with three gates that govern what information is stored, forgotten, or output. The full gate and state update equations, with forget, input, and output gates $\mathbf{f}_t$, $\mathbf{i}_t$, and $\mathbf{o}_t$, element-wise multiplication $\odot$, and the sigmoid σ, are given in Appendix B (Equation B.4). The forget gate learns which components of the cell state to discard, the input gate determines which new information to incorporate, and the output gate controls which parts of the cell state contribute to the hidden state.

LSTMs became the dominant architecture for financial time series prediction in the mid-2010s. Di Persio and Honchar [42] compared standard RNN, LSTM, and GRU architectures for Google stock prediction. The LSTM achieved 72% directional accuracy, outperforming the standard RNN and GRU variants, underscoring the importance of the gating mechanism for financial sequence modeling. Fischer and Krauss [43] conducted one of the most rigorous

studies of LSTM for financial prediction, applying it to all S&P 500 constituents with proper temporal splitting and a walk-forward evaluation protocol. The LSTM generated statistically significant returns after transaction costs over a 25-year test period, though the magnitude of outperformance was modest, consistent with the weak predictability expected in efficient markets. Hossain et al. [44] applied a cascaded LSTM-GRU architecture to S&P 500 data spanning 66 years from 1950 to 2016, achieving a mean squared error of 0.00098 by using the LSTM's initial prediction as input to a GRU for further refinement. This result outperformed individual GRU and LSTM models, which reported MSE values of approximately 0.003 when applied independently.

Ghosh et al. demonstrated the efficacy of combining LSTM with Random Forest to forecast directional movements of S&P 500 stock prices for intraday trading. Bernal et al. used an Echo State Network, a type of RNN with a fixed random reservoir, for S&P 500 prediction. Using variables such as stock prices, moving averages, and trading volumes as inputs, the ESN achieved a test error of 0.0027 in daily price change predictions. The researchers validated their findings on 50 additional stock prices, reporting competitive performance against state-of-the-art techniques. Lv et al. conducted a comprehensive comparison of traditional classifiers (SVM, Logistic Regression, Random Forest, Naïve Bayes, CART, and XGBoost) against DNN algorithms (MLP, Deep Belief Network, Stacked Autoencoder, RNN, LSTM, and GRU). Their findings revealed that traditional algorithms outperformed DNNs when transaction costs were not considered, but when transaction costs were factored in, DNN models demonstrated better net performance due to lower trading frequency and more stable signal generation.

Gated Recurrent Units

The Gated Recurrent Unit (GRU), proposed by Cho et al. [45], simplifies the LSTM by combining the forget and input gates into a single update gate and merging the cell and hidden states. Its update equations, with update gate $\mathbf{z}_t$ and reset gate $\mathbf{r}_t$, are given

in Appendix B (Equation B.10), and the reduced gate count gives it fewer parameters than the LSTM. Empirically, GRUs achieve comparable performance to LSTMs on many sequence modeling tasks while training faster, making them an attractive alternative when computational resources are constrained.

Bidirectional Architectures

Bidirectional LSTM (BiLSTM) processes sequences in both forward and backward directions, allowing the model to capture context from both past and future positions relative to each time step. For offline analysis where the complete sequence is available, BiLSTM produces richer representations than unidirectional processing. Xu et al. [46] integrated wavelet transforms, stacked autoencoders, and BiLSTM for stock forecasting, using the wavelet decomposition to separate trend from noise before feeding the cleaned signal into the BiLSTM. Liu et al. [47] employed autoencoder-based feature extraction combined with BiLSTM for stock price series prediction, reporting that the learned latent representations improved prediction accuracy relative to using raw features directly.

For real-time prediction, bidirectional processing introduces a subtle risk: the backward pass has access to future values in the input sequence, which is appropriate for tasks like machine translation where the full sentence is available, but requires careful handling in financial applications to avoid lookahead bias. In the architecture developed in this thesis, causal masking ensures that predictions at time t depend only on information available at or before time t.

2.3.3 Attention Mechanisms in Deep Neural Networks

The attention mechanism, introduced by Bahdanau et al. [48] for machine translation, allows models to dynamically weight the relevance of different input positions when producing each output element. In the context of financial time series, attention enables the model

to learn which historical time steps are most informative for predicting the current target, rather than relying on the recurrent hidden state to compress all relevant history into a fixed-dimensional vector.

Lu et al. [49] proposed a CNN-BiLSTM-Attention model that uses convolutional layers to extract local patterns, feeds them into a bidirectional LSTM for temporal modeling, and applies attention to weight the most informative time steps. Their results showed that the CNN-BiLSTM-AM method outperformed seven other methods in prediction accuracy across multiple benchmark datasets. The attention mechanism proved particularly valuable during volatile periods, where the relevant historical context shifted rapidly and fixed-window approaches lost their effectiveness. Chen et al. introduced a CNN-BiLSTM model with an efficient channel attention (ECA) module for stock price prediction. The ECA module selectively emphasized the most relevant feature channels, and the combined architecture achieved improved prediction accuracy over standard CNN-BiLSTM hybrids without the attention component, confirming that channel-wise attention provides complementary information to temporal attention.

The progression from fixed-window models to attention-augmented architectures reflects a broader shift in the field toward adaptive, data-driven weighting of input information. This principle is central to the transformer architecture discussed in the next section and underpins the sentiment fusion mechanism developed in Chapter 4.

2.3.4 Hybrid Deep Learning Approaches

Hybrid architectures combine the complementary strengths of different neural network families to capture multiple types of structure in financial data simultaneously. The CNN-LSTM family, exemplified by the attention-augmented variants discussed in the preceding subsection, pairs convolutional extraction of local spatial patterns with recurrent modeling of temporal dependencies, and the consistent gains reported by these hybrids over single-

architecture baselines confirm that spatial and temporal information provide complementary predictive signals. Beyond the CNN-LSTM family, Wang introduced the BiLSTM-MTRAN-TCN architecture that combines bidirectional LSTM, a modified transformer, and a temporal convolutional network, demonstrating that each component contributed distinct representational capacity and that the three-way combination outperformed any pair. Li and Qian developed the FDG-Transformer, a hybrid specifically designed for stock price prediction that integrates frequency-domain analysis with transformer attention, exploiting the observation that financial signals contain distinct information at different frequencies and that attending across frequency bands captures patterns invisible in the time domain alone.

Generative adversarial networks (GANs) have also been applied to financial forecasting. Wu et al. introduced a framework combining GANs with piecewise linear representation for predicting stock market trading actions (buy, sell, hold), with the adversarial training procedure encouraging the generator to produce realistic price trajectory patterns. Staffini proposed a deep convolutional GAN (DCGAN) for stock price prediction, where the generator learned to produce data resembling real stock prices while the discriminator learned to distinguish real from generated prices. The DCGAN outperformed standard widely used forecasting tools, demonstrating that adversarial training can learn richer distributional features of price dynamics than single-objective regression models. The GAN framework is conceptually interesting for financial applications because it encourages the model to learn the full distribution of price paths rather than simply minimizing point-prediction error, though the training instability inherent in adversarial objectives remains a practical challenge. Table B.2 in Appendix B consolidates the key deep learning studies reviewed in this section alongside their datasets, metrics, and principal results.

The collective evidence from deep learning studies in stock prediction converges on several practical observations. Recurrent architectures, particularly LSTM and GRU, consistently outperform feedforward and convolutional models on temporal prediction tasks, confirming that sequential information is a primary source of predictive signal. Hybrid architectures

that combine spatial and temporal processing tend to outperform single-architecture models, a pattern suggesting that multiple types of structure coexist in financial data and that capturing one at the expense of others leaves predictive value unrealized. Attention mechanisms contribute the most during volatile periods, when the relevant historical context shifts rapidly and fixed-window approaches cannot identify which past observations remain informative. Yet despite this architectural sophistication, none of the models reviewed in this section addresses the cross-sectional dependencies between stocks or the integration of non-quantitative information such as sentiment. Before turning to graph-based methods that encode relational structure and the multimodal fusion of sentiment, the next section examines transformer architectures, whose self-attention mechanism provides the representational foundation upon which both the node transformer and the sentiment integration strategy of this thesis are built.

2.4 Transformer Architectures for Financial Forecasting

The transformer architecture, introduced by Vaswani et al. [3], replaced sequential processing with parallel self-attention, fundamentally changing the computational paradigm for sequence modeling. Unlike RNNs, which propagate information one step at a time through a hidden state bottleneck, transformers compute pairwise interactions between all positions in a single operation, enabling direct modeling of long-range dependencies without the gradient degradation that plagues recurrent architectures.

2.4.1 Self-Attention Mechanism

The self-attention operation computes a weighted sum of value vectors, where the weights are determined by the compatibility between query and key vectors derived from the input;

the scaled dot-product formulation over the query, key, and value matrices is given in Appendix B (Equation B.14). The scaling factor $\sqrt{d_k}$ prevents dot products from growing large in magnitude as the dimension increases, which would push the softmax function into regions with near-zero gradients and effectively disable learning.

Multi-head attention extends this mechanism by learning h parallel attention functions with independent projection matrices, concatenating their outputs and applying a learned output projection (Appendix B, Equation B.15). Each head can specialize in different types of dependencies: one head might attend to recent momentum, another to periodic patterns at weekly or monthly frequencies, and a third to long-range trend information. The concatenation and projection step integrates these diverse views into a unified representation.

2.4.2 Positional Encoding

Because self-attention is permutation-equivariant, the model requires explicit positional information to distinguish the ordering of sequence elements. The original transformer used fixed sinusoidal encodings indexed by position *pos* and dimension i (Appendix B, Equation B.16). This encoding has the property that relative position offsets correspond to linear transformations of the encoding vector, enabling the model to generalize to sequence lengths not seen during training.

Financial time series introduce additional structure that generic positional encodings do not capture. Trading days follow a calendar with weekends, holidays, and varying session lengths. Quarterly earnings announcements create periodic patterns at approximately 63-trading-day intervals. Some financial transformer implementations replace the fixed sinusoidal encoding with learnable temporal encodings that adapt to these domain-specific periodicities during training.

2.4.3 Transformer Architecture for Sequence Modeling

The complete transformer encoder consists of L identical layers, each containing a multi-head self-attention sublayer followed by a position-wise feed-forward network (Appendix B, Equation B.18). Residual connections and layer normalization surround each sublayer, maintaining gradient flow through deep architectures. The decoder adds a cross-attention sublayer that attends to the encoder output and employs causal masking to ensure autoregressive generation.

For financial time series prediction, the encoder-only variant is most common, with the self-attention layer replaced by a causally masked version that prevents attention to future positions. The output at each position is passed through a linear projection to produce the prediction, and the loss function is typically mean squared error for regression tasks or cross-entropy for directional classification.

2.4.4 Financial Transformer Variants

Several transformer variants have been specifically designed or adapted for financial forecasting, each addressing domain-specific challenges that generic transformer architectures do not handle natively. Li and Qian [50] developed a financial transformer that processes multiple timeframes simultaneously by constructing separate attention heads for daily, weekly, and monthly resolution inputs. The multi-resolution design improved sensitivity to regime changes by allowing short-term heads to capture momentum shifts while long-term heads maintained awareness of secular trends, outperforming single-resolution baselines on both directional accuracy and risk-adjusted return metrics.

Mirjebreili and Solouki applied a multi-task transformer to the Iranian stock market, introducing two innovations specific to financial evaluation: quintuple barrier labeling, which defines target classes based on whether the future return exceeds transaction-cost-adjusted

thresholds within a fixed time horizon, and purge k-fold validation, which removes temporally adjacent samples from the training fold to prevent information leakage across the train-test boundary. These methodological contributions are relevant beyond the specific market studied, as both labeling bias and temporal leakage are pervasive issues in the financial prediction literature. Costa and Machado evaluated transformer architectures against ARIMA and LSTM for Ibovespa index prediction, finding the transformer outperformed both baselines in 60% of tests, with the largest advantages during periods of high volatility where long-range context was most valuable and the shortest advantages during calm trending periods where simpler autoregressive models captured the dominant signal.

The collective pattern across these studies is that transformers add the most value when the prediction task requires integrating information from multiple time scales or when the relevant historical context is non-obvious and variable. For simple trending markets where the recent past is the best predictor of the near future, the overhead of self-attention provides minimal benefit over recurrent or autoregressive models. This observation motivates the regime-aware design of the system in Chapter 5, where the transformer pathway is activated selectively based on detected market conditions.

2.4.5 Efficient Transformer Architectures

The quadratic $O(n^2)$ complexity of self-attention with respect to sequence length has motivated a family of efficient transformer variants. Zhou et al. [51] introduced the Informer architecture with a ProbSparse attention mechanism that selects a subset of dominant queries to approximate the full attention matrix, reducing complexity to $O(n \log n)$. A self-attention distilling operation halves the cascade layer input, and a generative style decoder enables long-term forecasting in a single forward pass rather than step-by-step decoding. The Informer demonstrated strong performance on long-sequence time series forecasting benchmarks, making it applicable to financial prediction tasks with extended lookback windows.

Kitaev et al. [52] proposed the Reformer, which uses locality-sensitive hashing to approximate attention by grouping similar queries and keys into buckets, reducing complexity to $O(n \log n)$. Wu et al. [53] introduced the Autoformer, which replaces standard attention with an auto-correlation mechanism that discovers period-based dependencies through frequency-domain analysis and progressive decomposition of the time series into trend and seasonal components. This decomposition aligns naturally with financial time series analysis, where separating trend from cyclical behavior is a fundamental preprocessing step.

These efficient architectures are particularly relevant for financial applications where the lookback window must span hundreds of trading days to capture long-term trends and seasonal effects. A standard transformer applied to a 252-day lookback window (one trading year) with 20 stocks produces attention matrices of dimension 5040, which is computationally expensive but feasible. Extending the lookback to multiple years, or increasing the number of stocks, pushes computation into regimes where efficient attention mechanisms become necessary. Yet even with these advances in temporal modeling, transformers still treat each stock as an independent sequence, discarding the relational structure that links securities through sectoral membership, supply chains, and competitive dynamics. The next section examines graph neural networks, which encode precisely this cross-sectional structure and, when combined with the transformer attention reviewed here, yield the node transformer architecture at the core of this thesis.

2.5 Graph Neural Networks for Stock Market Prediction

Graph neural networks (GNNs) address a fundamental limitation shared by all architectures reviewed so far: the treatment of stocks as independent entities. Financial markets have a rich relational structure in which sectoral co-movement, supply chain linkages, competitive

dynamics, and macroeconomic exposure create dependencies that carry predictive information. GNNs encode this structure by representing the market as a graph where nodes are individual securities and edges capture pairwise relationships.

2.5.1 Message Passing Framework

The GNN framework operates through iterative message passing, formalized by Gilmer et al. [54]. At each layer l, the representation of node v is updated by applying learnable UPDATE and AGGREGATE functions to the messages from its neighbors $\mathcal{N}(v)$ (Appendix B, Equation B.19). After L layers of message passing, each node's representation encodes information from its L-hop neighborhood, capturing both local and extended relational context.

2.5.2 Graph Convolutional Networks

Graph Convolutional Networks (GCN), introduced by Kipf and Welling [55], implement a spectral convolution on graphs through a layer-wise propagation rule that applies a symmetric degree normalization to the self-looped adjacency matrix before a learnable linear projection and nonlinearity, which makes the aggregation scale-invariant with respect to node degree (Appendix B, Equation B.20).

2.5.3 Graph Attention Networks

Graph Attention Networks (GAT), proposed by Velickovic et al. [56], replace the fixed normalization of GCN with learned attention coefficients α_{ij} computed from a shared weight matrix and a learnable attention vector (Appendix B, Equation B.21). The attention coefficients α_{ij} determine how much influence node j exerts on node i, allowing the model

to assign different importance to different neighbors based on their current representations rather than their fixed topological position.

For stock prediction, GAT's adaptive weighting is especially important because the relative influence between stocks shifts with market conditions. During a sector rotation, technology stocks may become more strongly coupled while energy stocks decouple. A fixed-weight GCN cannot capture these dynamics, but GAT learns to adjust the influence of each neighbor based on the current state of both nodes.

2.5.4 GNNs for Financial Applications

Chen et al. [13] proposed a graph convolutional feature-based CNN (GC-CNN) that combines an improved graph convolutional network with a dual-CNN to capture stock market features and individual stock features simultaneously. Testing on six randomly selected Chinese stocks, the GC-CNN achieved higher prediction accuracy and cumulative trading returns than CNN-only, LSTM, and standard GCN baselines across all test stocks, with the graph-augmented architecture reducing mean prediction error by incorporating market-wide relational signals that individual stock models cannot access.

Li et al. developed a method for predicting stock market volatility using graph neural networks combined with attention mechanisms. Their approach employed edge weight and information transmission mechanisms to screen relevant nodes, with GRU and LSTM aggregating subgraph node representations. By introducing an association relationship attention dimension, the model achieved 16.64% higher accuracy in predicting stock market volatility compared to dimensional reduction index methods, and 14.48% higher accuracy than other heterogeneous data fusion methods using the same model architecture.

Wang et al. [14] introduced MG-Conv, a multi-graph convolutional neural network that first extracts deep features from historical transaction data using one-dimensional convolution and data normalization, then defines both static and dynamic correlation graphs. Multi-graph

convolution is performed on these graphs, with the results transferred to prediction values through fully connected networks. Testing on 42 Chinese stock market indices and comparing against LSTM, 3D-CNN, GC-CNN, and AD-GAT, MG-Conv reduced the average prediction error by 5.11% and demonstrated strong robustness across different market conditions.

Feng et al. [57] developed a Temporal Relational Ranking model that learns stock relations from historical price data and uses these learned relations to rank stocks by expected return. The model constructs a dynamic graph where edge weights are derived from return correlation patterns, and the graph structure adapts at each time step based on recent market behavior. You et al. proposed the Decoupled Graph Diffusion Neural Network (DGDNN) for stock movement prediction, which automatically constructs dynamic stock graphs using entropy-driven edge generation and learns task-optimal dependencies via a generalized graph diffusion process. The decoupled representation learning scheme captured distinctive hierarchical intra-stock features, and the method demonstrated substantial improvements over state-of-the-art baselines on real-world datasets, without relying on artificially defined factors to construct static stock graphs.

Sawhney et al. [58] introduced a spatiotemporal hypergraph convolution network that extends standard pairwise graph structure to hyperedges connecting arbitrary groups of stocks. This formulation captures higher-order relationships such as index membership, shared analyst coverage, or common institutional ownership that cannot be decomposed into pairwise connections. Evaluated on US equity movement prediction, the hypergraph model achieved higher F1 scores and Matthews Correlation Coefficients than GCN, GAT, and standard GNN baselines, confirming that higher-order group structure provides predictive information beyond what pairwise edges can encode.

Beyond equity markets, graph-based forecasting methods have been applied to other financial domains. Le et al. [59] developed an entropy-based GCN framework for real estate market prediction that segments spatial regions into anomalous and normal clusters before applying separate forecasting models, reducing prediction error across multiple horizons

on U.S., U.K., and Australian housing data. A related study by Le et al. [60] combined entropy-based multivariate clustering with sentiment extracted from news headlines and macroeconomic indicators, demonstrating that this segmentation approach improves forecasting accuracy when paired with deep learning architectures such as transformers and LSTMs. These cross-domain results underscore the generality of graph-based representations for capturing complex dependencies in financial data.

2.5.5 The Node Transformer

The application of transformer-style attention to graph-structured data has produced a family of graph transformer architectures that extend the representational capacity of message-passing GNNs. Ying et al. [61] introduced Graphormer, which incorporates structural encodings (centrality, spatial distance, and edge features) directly into the transformer attention computation, enabling the model to leverage graph topology without explicit message passing. Graphormer won the OGB Large-Scale Challenge on molecular property prediction and achieved state-of-the-art results on multiple graph-level benchmarks, demonstrating that transformers with appropriate structural inductive biases can match or exceed specialized GNN architectures. Kreuzer et al. [62] proposed Spectral Attention Networks (SAN), which compute learnable positional encodings from the Laplacian eigenvectors of the graph, providing each node with a position-aware representation that captures global structural context. SAN outperformed both GCN and GAT on the ZINC molecular benchmark and on the PATTERN and CLUSTER classification tasks from the GNN benchmarking suite. More recently, Rampášek et al. [63] unified these ideas in the General, Powerful, Scalable (GPS) framework, a modular architecture that combines local MPNN-based message passing with global transformer attention. GPS achieved competitive or superior results across ten benchmarks, confirming that local and global processing capture complementary structural information that neither mechanism alone can fully exploit.

A shared limitation of these graph transformer architectures is their quadratic $O(N^2)$ attention complexity with respect to the number of nodes, which restricts scalability to graphs of moderate size. For financial applications where the node set comprises hundreds or thousands of securities and the model must be retrained or fine-tuned frequently, quadratic scaling poses a practical barrier. The NodeFormer architecture, introduced by Wu et al. [12], addresses this bottleneck through a kernelized softmax approximation that replaces the explicit $N \times N$ attention matrix with random feature maps, reducing the all-pair attention computation to $O(N)$ complexity. NodeFormer further introduces a Gumbel-Softmax mechanism for learning sparse latent graph structures, enabling the model to discover task-relevant edges that may not exist in the input topology. On standard node classification benchmarks including Cora, Citeseer, and PubMed, NodeFormer achieved accuracy competitive with full-attention graph transformers while scaling to the Amazon2M dataset (over two million nodes) where quadratic-complexity alternatives were computationally infeasible. This linear-complexity scalability, combined with the ability to learn adaptive inter-node relationships without requiring a predefined adjacency matrix, makes NodeFormer well suited to stock market prediction, where the relevant relational structure between securities is neither fully known in advance nor static over time.

For stock market prediction, the node transformer offers a compelling combination: the relational inductive bias of graph networks, which encodes the prior knowledge that stocks are interdependent, and the representational power of transformer attention, which learns which inter-stock relationships are informative for each prediction. The original NodeFormer architecture was designed for homogeneous node classification, however, and does not natively accommodate multimodal inputs such as sentiment features or regime-conditional processing. The system developed in this thesis extends NodeFormer in both directions: Chapter 4 integrates BERT-derived sentiment representations through an attention-based fusion mechanism, and Chapter 5 adds regime-aware dual pathways with reinforcement learning control. Table B.3 in Appendix B compares the graph-based approaches reviewed in

this section, contrasting their graph construction strategies, edge definitions, and empirical results.

The graph-based and transformer architectures reviewed so far model the quantitative structure of financial markets, capturing temporal patterns within individual stocks and relational dependencies across them. However, a substantial source of market-moving information lies outside the quantitative record entirely: the textual signals embedded in news articles, social media posts, and analyst reports that shape investor expectations and drive short-term price dynamics. The following section surveys the extraction and integration of this textual information through sentiment analysis.

2.6 Sentiment Analysis for Financial Forecasting

The efficient market hypothesis in its strictest form implies that all publicly available information is instantaneously reflected in prices. Empirical evidence consistently contradicts this assumption: sentiment expressed in news articles, social media posts, and analyst reports contains predictive information that prices incorporate gradually over hours to days [17]. The challenge lies in extracting this information from unstructured text at scale and integrating it with quantitative features in a way that accounts for the heterogeneous quality, timing, and relevance of different textual sources.

2.6.1 Lexicon-Based Approaches

The earliest computational approaches to financial sentiment relied on domain-specific word lists that assign sentiment polarity to individual terms. The Harvard General Inquirer provided the initial vocabulary, but Loughran and McDonald [18] demonstrated that general-purpose sentiment dictionaries systematically misclassify financial language: words like "liability," "risk," and "cost" carry negative connotations in general English but are neutral or

context-dependent in financial discourse. Their finance-specific word lists, constructed from 10-K filings, improved classification accuracy substantially and became a standard resource in the field.

Lexicon-based methods have the advantage of transparency: the contribution of each word to the sentiment score is explicit and interpretable. Their limitations are equally clear. They cannot capture negation ("not profitable"), sarcasm, or context-dependent meaning. They assign fixed polarity to words regardless of the surrounding text, and they cannot handle out-of-vocabulary terms, abbreviations, or the informal language prevalent on social media.

2.6.2 Machine Learning Approaches to Sentiment Classification

Machine learning approaches to financial sentiment classification treat the problem as supervised text classification, training models on labeled corpora of financial text. Malo et al. [64] constructed the Financial PhraseBank, a benchmark dataset of approximately 5,000 financial sentences annotated with sentiment labels by domain experts. SVM and Naive Bayes classifiers trained on bag-of-words and TF-IDF representations achieved classification accuracies above 70% on this dataset, substantially outperforming lexicon-based methods that relied on generic word lists. Schumaker and Chen [65] developed the AZFin text system for extracting sentiment from breaking financial news. The system processed articles within 20 minutes of publication and generated trading signals based on textual features, demonstrating that combining text-derived features with quantitative price predictors improved forecast accuracy and simulated trading returns relative to either information source alone.

The integration of sentiment with price prediction has been explored through several approaches. Bollen et al. [19] demonstrated that aggregate Twitter mood indicators, measured through the OpinionFinder and Google Profile of Mood States (GPOMS) tools, correlated with subsequent Dow Jones Industrial Average movements. Using a Self-Organizing Fuzzy Neural Network, the inclusion of the "calmness" mood dimension improved DJIA daily direction

prediction accuracy to 87.6%, a substantial improvement over models using price data alone. This result was among the earliest quantitative demonstrations that social media sentiment contains actionable predictive information for equity markets. Si et al. [66] extended this approach by extracting topic-specific sentiment from Twitter using a Dirichlet process mixture model. Their results showed that sentiment about specific economic topics (employment, housing, corporate earnings) had differentiated predictive power for market indices, with topic-level sentiment outperforming aggregate sentiment by capturing the heterogeneous impact of different information types.

Ding et al. [67] proposed a deep learning approach for event-driven stock prediction that extracts structured event tuples from news articles and learns dense event embeddings. By representing events as (actor, action, object) triples rather than bag-of-words features, the model captured the directional impact of specific types of events on stock prices. Testing on S&P 500 index prediction, the event embedding approach achieved approximately 6% improvement in prediction accuracy over state-of-the-art baselines that used conventional text representations. Xu and Cohen [68] combined historical price data with tweet-level sentiment in a variational autoencoder framework (StockNet), achieving improved movement prediction accuracy on 88 stocks with high trading volumes. The joint modeling of text and price produced better predictions than either modality alone, with the largest gains occurring around earnings announcements and other information-dense events where sentiment diverged from price momentum.

2.6.3 BERT and Transformer-Based Sentiment Analysis

The introduction of BERT [4] transformed sentiment analysis in financial applications. BERT's bidirectional attention mechanism enables it to capture context-dependent meaning: the word "loss" in "the company reported a loss" carries different sentiment from "the company's loss of market share" or "the loss function converged." Pre-trained on massive

text corpora, BERT learns general-purpose language representations that can be fine-tuned for domain-specific tasks with relatively small labeled datasets.

Araci [69] developed FinBERT by further pre-training BERT on financial news and communications, followed by fine-tuning on financial sentiment classification. FinBERT achieved state-of-the-art performance on the Financial PhraseBank dataset and the SemEval-2017 Task 5 financial sentiment benchmark, demonstrating that domain-specific pre-training substantially improves sentiment classification accuracy in financial contexts. The model captures financial jargon, context-dependent valuation language, and the nuanced distinction between factual reporting and sentiment expression that general-purpose models miss.

Subsequent models including RoBERTa [70] improved upon BERT's pre-training methodology through larger batch sizes, dynamic masking, and the removal of the next-sentence prediction objective, yielding modest but consistent gains on downstream tasks. For financial sentiment, the choice of pre-training corpus and domain-specific fine-tuning data typically matters more than the specific architectural variant.

2.6.4 Multimodal Fusion of Sentiment and Price Data

The technical challenge of integrating sentiment with quantitative features extends beyond simply concatenating sentiment scores with price vectors. Sentiment signals arrive at irregular intervals, with coverage that varies dramatically across stocks (mega-cap stocks attract thousands of daily mentions while small-cap stocks may have none). The predictive value of sentiment is time-varying: sentiment divergence from price momentum is highly informative during earnings weeks but carries minimal signal during low-news periods. These characteristics demand a fusion mechanism that learns when and how much to weight sentiment relative to quantitative signals.

Nelson et al. [71] combined LSTM-based price prediction with sentiment features extracted from financial news, demonstrating that the temporal alignment between sentiment generation

and price response is critical: sentiment leads price by varying intervals depending on the information type and market conditions. Early research typically aggregated daily sentiment scores and appended them to the feature vector, a simple concatenation that treats sentiment as having fixed influence regardless of context.

More sophisticated fusion strategies have since emerged along a spectrum of integration depth. At the shallowest level, feature-level fusion embeds sentiment representations directly into the input layer, allowing learned interactions between sentiment and price features to develop across all subsequent processing layers. Deeper integration is achieved through attention-based fusion, where a learned weighting mechanism adjusts the contribution of sentiment information based on the current market state, assigning higher influence during periods when sentiment diverges from price momentum and lower influence when textual signals are sparse or noisy. The most expressive approaches adopt hierarchical fusion, in which sentiment first modulates the input representation at an early processing stage and then re-enters the architecture at the prediction layer, enabling multiple levels of cross-modal interaction that capture both fine-grained feature correlations and high-level conditional dependencies.

The fusion mechanism developed in Chapter 4 of this thesis implements a three-part integration strategy that combines multi-scale temporal sentiment features, key-modulation within the attention computation, and an adaptive fusion gate conditioned on rolling volatility and sentiment magnitude. This design is motivated by the observation that no single fusion strategy is optimal across all market conditions, and a mechanism that adapts its behavior based on the current information environment captures more of the available predictive signal. Table B.4 in Appendix B traces the progression of sentiment analysis methods from lexicon-based approaches through transformer-based models, along with the fusion strategies and empirical outcomes reported in each study.

Despite the independent progress in graph-based prediction (Section 2.5) and sentiment integration, remarkably few studies have attempted to combine relational graph structure

with textual features in a unified prediction architecture. Kim et al. [72] proposed HATS, a hierarchical graph attention network that models inter-stock relationships through multiple relational graphs (industry membership, supply chain connections, and co-ownership), achieving higher movement prediction accuracy than single-stock baselines on both S&P 500 and Korean market datasets. However, HATS processes only quantitative price and volume features and does not incorporate textual or sentiment information. Conversely, the sentiment-augmented models reviewed above (StockNet, event embeddings, FinBERT-based systems) treat each stock independently, discarding the cross-sectional dependencies that graph methods capture. The few architectures that incorporate both modalities typically concatenate aggregated sentiment scores onto graph node features before message passing, a strategy that fixes the influence of sentiment regardless of its current reliability and prevents the model from learning context-dependent weighting. This gap, the absence of a principled mechanism for fusing graph-structured relational information with fine-grained, temporally varying sentiment signals, motivates the architecture developed in Chapter 4, where attention-based fusion within the node transformer framework enables adaptive, state-dependent integration of BERT-derived sentiment with quantitative inter-stock features.

The architectures reviewed to this point, whether temporal, graph-based, or multimodal, share a common assumption: that a single set of learned parameters is adequate for all market conditions. The next section challenges this assumption by examining the evidence that financial markets alternate between distinct regimes whose statistical properties differ sharply enough to warrant separate modeling strategies.

2.7 Regime Detection and Market State Identification

The observation that financial markets exhibit distinct behavioral regimes, periods of calm trending, volatile crisis, post-shock recovery, and rapid sector rotation, has profound implications for prediction system design. A model trained on pooled data from all regimes learns a compromise between the statistical properties of each regime, performing adequately on average but poorly during the periods when accurate prediction matters most.

2.7.1 Hidden Markov Models

Hidden Markov Models (HMMs) provide the classical framework for regime identification in financial time series. Hamilton [1] introduced the Markov-switching model, in which the observed data are generated by a regime-specific process whose parameters (mean, variance, autoregressive coefficients) depend on a latent state variable that follows a first-order Markov chain. The model is estimated via maximum likelihood using the Baum-Welch algorithm (a special case of expectation-maximization), which alternates between inferring the most probable regime sequence and updating regime-specific parameters.

The two-regime specification ("bull" and "bear" markets) has been the most common in financial applications, though three-regime variants that distinguish between calm, volatile, and crisis states have also been explored. Diebold et al. [73] extended the framework to allow time-varying transition probabilities that depend on exogenous variables such as interest rate spreads or leading economic indicators. More recently, Nystrup et al. [21] combined regime-based portfolio strategies with modern clustering methods, demonstrating that data-driven regime identification outperformed static allocation strategies during periods of market turbulence, and Ang and Timmermann [20] provided a comprehensive review of regime-

switching models in asset pricing that documented their continued utility alongside machine learning alternatives.

HMMs have clear limitations for the prediction task addressed in this thesis. The number of regimes must be specified in advance, and the Gaussian emission assumption may be too restrictive to capture the heavy-tailed, skewed distributions observed in financial returns. The transition probabilities are assumed stationary (or depend on a limited set of covariates), which may not hold during structural breaks. Most critically for prediction system design, HMMs identify regimes retrospectively but do not prescribe how the prediction model should adapt its behavior in response to a detected regime change.

2.7.2 Threshold-Based Approaches

Threshold models trigger regime switches when observable variables cross predefined boundaries. The Self-Exciting Threshold Autoregressive (SETAR) model, studied by Tong [74], switches between linear dynamics based on whether lagged values of the series itself exceed a threshold. In financial practice, the CBOE Volatility Index (VIX) is commonly used as a regime indicator, with thresholds of 20 or 25 separating low-volatility from high-volatility states.

The simplicity of threshold-based approaches is both their strength and their limitation. They are transparent, computationally cheap, and easy to implement. However, the choice of indicator variable and threshold value introduces design choices that directly determine the regime classification, and small changes in the threshold can reclassify large portions of the data. A VIX threshold of 25 identifies the 2008 crisis and the 2020 pandemic sell-off as high-volatility regimes but may miss the sector-specific disruptions caused by trade policy shifts in 2018 or the crypto contagion of 2022. No single threshold can simultaneously capture all types of market disruption.

2.7.3 Autoencoders for Unsupervised Regime Detection

Autoencoders offer a fundamentally different approach to regime identification that avoids both the parametric assumptions of HMMs and the arbitrary threshold choices of indicator-based methods. An autoencoder trained on data from a reference regime learns to reconstruct normal patterns through an information bottleneck [5]. The encoder maps inputs to a lower-dimensional latent representation, and the decoder reconstructs the original input from this representation. When the model encounters data generated by a different regime, the reconstruction error increases because the learned representation does not accommodate the anomalous patterns.

Kingma and Welling [75] extended this framework with the Variational Autoencoder (VAE), which imposes distributional constraints on the latent space and enables principled uncertainty quantification. The VAE objective combines reconstruction loss with a KL divergence term that encourages the latent distribution to match a prior, providing both anomaly detection through reconstruction error and uncertainty estimates through the learned posterior.

In financial applications, autoencoders have been applied to fraud detection [76], anomaly identification in trading patterns [77], and feature extraction for downstream prediction [78]. The application to regime detection, as implemented in Chapter 5 of this thesis, trains the autoencoder exclusively on stable-market data so that the reconstruction error serves as a continuous anomaly score: low error indicates market behavior consistent with the stable regime, while high error signals a departure that may warrant a different prediction strategy. This approach eliminates the need for manual regime labels or predefined thresholds, instead learning the boundary between "normal" and "anomalous" from the data distribution itself.

2.7.4 Change Point Detection

Change point detection algorithms identify abrupt shifts in the statistical properties of a time series. Aminikhanghahi and Cook [79] provided a comprehensive survey covering both offline methods (which analyze complete time series retrospectively) and online methods (which detect changes in real time with bounded delay). Bayesian online change point detection maintains a posterior distribution over the time since the last change point, updating it incrementally as new observations arrive.

For financial applications, the online variant is essential because trading decisions must be made in real time, and retrospective identification of regime boundaries has no practical value. The challenge is balancing detection speed (low delay) against false alarm rate: an overly sensitive detector will generate frequent spurious regime changes that trigger unnecessary model switching, while an insensitive detector will miss genuine transitions until after the most actionable period has passed.

The regime detection approach in this thesis sidesteps the binary change-point formulation by using the autoencoder reconstruction error as a continuous signal rather than a discrete switch. The reinforcement learning controller described in Section 2.8 learns to interpret this continuous signal in the context of prediction performance, effectively discovering the optimal mapping from anomaly score to prediction strategy without requiring explicit change-point detection.

2.8 Reinforcement Learning in Financial Applications

Reinforcement learning (RL) optimizes sequential decision-making through interaction with an environment, learning policies that maximize cumulative reward [6]. The RL

framework provides a natural formalism for financial applications where decisions (trades, portfolio allocations, parameter adjustments) have consequences that unfold over time and where the optimal action depends on the current state of a dynamic environment.

2.8.1 Foundations of Deep Reinforcement Learning

Deep RL combines the function approximation capacity of neural networks with the sequential optimization framework of RL. Deep Q-Networks (DQN), introduced by Mnih et al. [80], learn action-value functions for discrete action spaces through experience replay and target network stabilization. DQN demonstrated superhuman performance on Atari games and established the viability of combining large neural networks with RL for complex decision problems.

For continuous action spaces, which are more natural for financial applications where positions and allocations take real-valued magnitudes, policy gradient methods directly optimize the policy parameters. REINFORCE estimates gradients from sampled trajectories but suffers from high variance. Actor-critic methods reduce this variance by learning both a policy (actor) and a value function (critic), using the critic's value estimates to reduce the variance of policy gradient updates. Deep Deterministic Policy Gradient (DDPG), proposed by Lillicrap et al. [81], extends actor-critic to continuous action spaces using a deterministic policy and off-policy learning. Proximal Policy Optimization (PPO) [82] provides a stable policy optimization procedure through a clipped surrogate objective that limits the magnitude of policy updates.

2.8.2 Soft Actor-Critic

Soft Actor-Critic (SAC), introduced by Haarnoja et al. [83], incorporates entropy regularization into the actor-critic framework. The maximum entropy objective augments the

standard cumulative reward with a policy entropy term weighted by a temperature parameter α that controls the trade-off between reward maximization and entropy maximization (Appendix B, Equation B.22).

The entropy bonus encourages exploration by penalizing deterministic policies, preventing premature convergence to suboptimal actions. This property is well suited to financial environments where the reward signal is noisy, delayed, and non-stationary. An RL agent that commits too aggressively to a single strategy based on limited experience is likely to perform poorly when market conditions shift. SAC's entropy regularization provides a principled mechanism for maintaining behavioral diversity, ensuring that the agent continues to explore alternative strategies even when one strategy currently appears optimal. Beyond this theoretical motivation, SAC offers practical advantages over competing algorithms: its off-policy nature enables sample-efficient learning from a replay buffer, the entropy-augmented objective smooths the optimization landscape to improve training stability, and the automatic temperature tuning reduces the hyperparameter sensitivity that plagues PPO and DDPG in non-stationary environments.

2.8.3 Reinforcement Learning for Trading and Portfolio Management

RL has been applied to financial decision-making across several domains. Moody and Saffell [84] were among the earliest to apply RL to trading, using a recurrent reinforcement learning algorithm that optimized trading rules directly for risk-adjusted return metrics such as the Sharpe ratio and differential Sharpe ratio, establishing the principle that RL agents should optimize financial objectives directly rather than proxy losses. Jiang et al. [85] formulated portfolio management as a continuous-control RL problem, learning allocation weights across assets that maximize cumulative risk-adjusted return through a CNN-based policy network trained on historical price tensors.

Deng et al. [86] developed a deep RL framework for financial signal representation and trading that learns both the feature representation and the trading policy end-to-end. Yang et al. [87] proposed an ensemble strategy combining multiple deep RL agents (PPO, A2C, DDPG) for automated stock trading, demonstrating that the ensemble outperformed individual agents by adapting to different market conditions through a selection mechanism that favors the agent with the best recent performance. More recent work by Liu et al. [88] introduced FinRL, an open-source library that standardized the application of deep RL to quantitative finance, providing reproducible benchmarks across stock trading, portfolio allocation, and cryptocurrency trading tasks that have accelerated research in this area.

2.8.4 RL as Meta-Controller for Prediction Systems

The application of RL in this thesis differs from the trading and portfolio management framework. Rather than using RL to make trading decisions directly, Chapter 5 employs SAC as a meta-controller that tunes the prediction system's internal parameters: the autoencoder threshold that determines regime classification and the blending weights that combine outputs from dual prediction pathways. The controller observes a state comprising recent prediction errors, reconstruction losses, and volatility indicators, and outputs continuous actions that adjust these parameters to minimize downstream prediction error.

This meta-control formulation offers substantial advantages over manual parameter tuning. Because the controller discovers regime boundaries from prediction performance rather than from predefined indicators, it eliminates the need for human-specified thresholds entirely. The controller also adapts its routing and blending strategy in response to changing market conditions, learning over time that the optimal threshold and blending weights differ between calm and turbulent periods. The SAC entropy bonus further ensures continued exploration of alternative configurations, preventing the controller from locking into a single strategy that may become suboptimal when conditions shift. Taken together, this use of RL as an adaptive

configuration mechanism, rather than as a direct decision-maker, represents a departure from the prevailing approach in financial RL and is, to our knowledge, novel in the context of stock prediction systems. However, the introduction of an autonomous controller raises a new challenge: how to evaluate whether the controller's decisions are sound, not merely whether the aggregate prediction error is low. The following section reviews emerging approaches to this evaluation problem, centered on the use of large language models as behavioral judges.

2.9 LLM-Based Evaluation and Autonomous Agent Assessment

As prediction systems grow more complex, incorporating multiple processing stages, routing mechanisms, and adaptive controllers, the gap between end-to-end performance metrics and process-level understanding widens. A system that achieves acceptable MAPE may be making systematically incorrect intermediate decisions that happen to cancel on average but produce unreliable behavior in specific conditions. Evaluating the quality of these intermediate decisions requires moving beyond aggregate output metrics toward behavioral assessment.

2.9.1 LLM as a Judge

The LLM-as-a-Judge paradigm uses large language models to evaluate the quality of outputs generated by other systems, replacing or augmenting human annotators whose availability and cost limit the scale of evaluation campaigns. Zheng et al. [89] provided the foundational study on the MT-Bench conversation benchmark, demonstrating that GPT-4 judgments align closely with human preference rankings, achieving agreement rates above 80% that are comparable to inter-annotator agreement among humans. The study also identified systematic biases that LLM judges exhibit: position bias, in which the model

favors the response presented first in a pairwise comparison, and verbosity bias, in which longer responses receive higher scores regardless of substantive quality. These biases must be accounted for in any evaluation protocol that relies on LLM judgments, and subsequent work has proposed mitigation strategies including randomized ordering, multi-turn evaluation, and explicit instruction to ignore length.

Liu et al. [90] showed that chain-of-thought prompting further improves evaluation fidelity by requiring the LLM to articulate its reasoning before assigning a score. When the model is forced to justify its evaluation step by step, the frequency of arbitrary or inconsistent judgments decreases measurably, and the resulting scores correlate more strongly with expert human ratings. This finding established a design principle that informs the evaluation framework in Chapter 6: structured reasoning prompts produce more reliable and reproducible assessments than direct scoring.

AlpacaFarm [91] extended the LLM-as-a-Judge framework to reward model training, using LLM evaluations as a proxy for human feedback in the RLHF pipeline. The key finding was that LLM judgments are reliable enough to serve as training signals for downstream optimization, reducing the dependency on expensive human annotation without sacrificing alignment quality. This principle is central to the closed-loop design in Chapter 6, where LLM behavioral assessments are converted into auxiliary reward signals for the SAC controller, enabling the prediction system to improve its decision-making based on qualitative feedback about its own behavior.

2.9.2 Evaluation Design: Rubrics, Multi-Judge Agreement, and Calibration

The reliability of LLM-based evaluation depends critically on how the evaluation task is structured. Scoring rubrics that define explicit criteria and anchor descriptions for each score level reduce subjectivity and improve inter-judge consistency. When multiple LLM judges

are employed, their agreement can be quantified using standard inter-rater reliability metrics such as Cohen's kappa [92] for pairwise agreement or Krippendorff's alpha [22] for multi-rater settings. Disagreement between judges often signals ambiguity in the evaluation criteria or edge cases where reasonable evaluators might differ, and analyzing these disagreements provides diagnostic information about the evaluation instrument itself.

Calibration presents an additional challenge: LLM judges may assign systematically higher or lower scores than human evaluators, and the distribution of scores may be compressed into a narrow range that reduces discriminative power. Prompting strategies that request comparative judgments (which of two outputs is better) rather than absolute scores (rate this output on a 1-10 scale) tend to produce more calibrated and discriminative evaluations. The evaluation framework in this thesis addresses calibration by using multiple LLM judges with diverse architectural foundations and aggregating their assessments, treating inter-judge agreement as a quality signal in its own right.

2.9.3 Agent Evaluation Frameworks

The emergence of autonomous AI agents that perform sequences of decisions in complex environments has created demand for evaluation frameworks that assess process quality, not just outcome quality. ReAct [93] introduced an agent framework combining reasoning traces with tool use, establishing the principle that agents should externalize their reasoning in a format amenable to inspection. However, ReAct relied on binary task completion for evaluation, which cannot distinguish between agents that succeed through sound reasoning and those that succeed through fortunate coincidence. Reflexion [94] added self-reflection capabilities, enabling agents to learn from verbal feedback about their reasoning failures, though evaluation remained focused on final task success rather than the quality of intermediate decisions.

Wang et al. [95] identified process quality evaluation as an open problem in the autonomous agent literature, noting that most existing benchmarks at the time measured whether agents achieved their goals but not whether the reasoning and decision-making that led to the outcome were sound. An agent may complete a task through a sequence of lucky coincidences, or fail a task despite making excellent decisions that were undermined by environmental stochasticity. Distinguishing these cases requires evaluation at the level of individual decisions rather than aggregate outcomes. This distinction between outcome quality and process quality is the central motivation for the evaluation framework developed in Chapter 6, where the goal is to assess whether a prediction system's routing, adaptation, and risk calibration decisions were appropriate given the information available at the time they were made.

2.9.4 Evaluation in Financial AI

In financial AI specifically, the gap between outcome metrics and process quality is particularly consequential. A trading system that achieves positive returns through reckless risk-taking is more dangerous than one that achieves modest returns through disciplined risk management, but aggregate return metrics cannot distinguish the two. This problem is not merely academic: regulatory frameworks such as MiFID II in Europe and SEC oversight in the United States increasingly require that automated trading and decision-support systems demonstrate not just acceptable outcomes but defensible and auditable decision-making processes. The ability to explain and justify individual decisions is becoming a compliance requirement, not an optional feature.

Standard evaluation in financial machine learning relies on backtesting, walk-forward validation, and aggregate metrics such as Sharpe ratio, maximum drawdown, and annualized return. These metrics assess the statistical properties of outcome distributions but reveal nothing about the reasoning behind individual decisions. A system that correctly identifies a regime change but routes predictions through the wrong pathway, or that detects elevated

risk but fails to adjust its confidence intervals, will exhibit degraded aggregate performance, but the aggregate metrics cannot diagnose which component failed or why.

Xi et al. [96] surveyed the emerging field of LLM-based agents and identified financial systems as a domain where behavioral evaluation is both critical and underexplored. The financial domain presents unique challenges for LLM-based evaluation: decisions must be assessed in the context of stochastic market conditions, the distinction between good decisions and good outcomes is fundamental, and domain expertise is required to judge whether a specific routing choice or parameter adjustment was appropriate given the available information. No prior work applies structured multi-dimensional LLM-based evaluation to financial prediction systems, and no work uses the resulting behavioral assessments to close a learning loop for system improvement.

The evaluation framework developed in Chapter 6 addresses these challenges by defining six domain-specific evaluation dimensions (regime detection accuracy, routing appropriateness, adaptation responsiveness, risk calibration, strategy coherence, and error recovery) that decompose behavioral quality into assessable components. Structured prompts present behavioral traces to LLMs with sufficient context for informed judgment, and perturbation-based validation confirms that each dimension responds selectively to targeted degradation of the corresponding system component. The use of multiple LLM judges with different architectural foundations provides robustness against individual model biases, and Krippendorff's alpha quantifies the degree of inter-judge agreement on each dimension.

2.10 Summary and Research Gaps

The literature reviewed in this chapter reveals a field that has made substantial progress along individual dimensions of the stock prediction problem while leaving their integration largely unexplored. Statistical models established rigorous benchmarks and the importance of volatility modeling. Classical machine learning demonstrated that nonlinear methods with

engineered features could outperform linear models. Deep learning shifted the emphasis from feature engineering to representation learning, with LSTMs, attention mechanisms, and transformers progressively expanding the temporal range and complexity of learnable patterns. Graph neural networks introduced relational structure, confirming that cross-sectional information is predictive. Sentiment analysis moved from lexicon counting through supervised classification to contextual language models that capture the nuances of financial discourse. Regime detection formalized the observation that market dynamics are non-stationary. Reinforcement learning provided a framework for adaptive decision-making in sequential environments. LLM-based evaluation began addressing the behavioral assessment of complex autonomous systems. Yet at each of these frontiers, critical gaps persist that limit the practical applicability of existing approaches, and it is the integration across these boundaries that defines the contribution of the present work.

The first gap is the absence of unified multimodal graph-temporal prediction. Graph neural networks and transformers have each been applied to stock prediction, but their combination is rare. The integration of sentiment as a third modality is rarer still, and no existing work addresses the fusion challenge in a principled way that accounts for the time-varying informativeness of sentiment signals. The technical barriers are substantial: sentiment data arrives at irregular intervals, with coverage that varies across stocks, and its signal-to-noise ratio shifts with market conditions. Simple concatenation or fixed weighting cannot accommodate this heterogeneity.

The second gap involves adaptive regime-aware prediction without manual regime labels. The need for regime-aware forecasting is well-established, but existing detection methods all require prior specification: the number of regimes, the distributional family of emissions, threshold values on indicator variables, or labeled training data. Each choice introduces a potential failure point. Learning regime boundaries directly from prediction performance eliminates these design decisions, but requires a training framework that jointly optimizes regime detection and prediction without trivial degeneration.

The third gap relates to behavioral quality evaluation for autonomous multi-stage prediction systems. As architectures incorporate regime detectors, routing mechanisms, and adaptive controllers, the gap between aggregate metrics and process-level understanding widens. MAPE and directional accuracy reveal nothing about whether intermediate decisions were appropriate, whether the system's adaptation timing was responsive, or whether its risk calibration was sound. No prior work applies structured behavioral evaluation to financial prediction systems, and no work uses such evaluation to close a learning loop for system improvement. Table 2.1 maps each of these three gaps to the corresponding thesis contribution and the chapter in which it is developed.

Table 2.1: Summary of identified research gaps and the corresponding thesis contributions. Each gap motivates a specific architectural component developed in the subsequent chapters.

#	**Research Gap**	**Thesis Contribution**	**Chapter**
1	No unified graph + transformer + sentiment fusion with adaptive weighting	Node Transformer + BERT with attention-based fusion gate	Ch. 4
2	Regime detection requires manual labels or fixed thresholds	Autoencoder + dual pathways + SAC meta-controller	Ch. 5
3	No behavioral evaluation of intermediate decisions in multi-stage pipelines	LLM-as-Judge framework with closed-loop reward	Ch. 6

This thesis addresses these three gaps through a progression of architectures in which each system builds upon its predecessor. The shared dataset and experimental framework upon which all three contributions are evaluated is detailed in Chapter 3. Chapter 4 introduces the multimodal graph-temporal architecture with adaptive sentiment fusion. Chapter 5 adds regime-aware routing with reinforcement learning control. Chapter 6 closes the evaluation loop with LLM-based behavioral assessment. The following chapters present each contribution in full technical detail, including the experimental protocols and rigorous evaluations that support the claims made in this introduction and contextualized by the literature reviewed here.

3

Dataset and Experimental Framework

The three contributions developed in this thesis, specifically the node transformer with sentiment fusion (Chapter 4), the regime-aware dual-pathway system (Chapter 5), and the LLM-based behavioral evaluation framework (Chapter 6), share a common experimental foundation. This chapter establishes that foundation by describing the stock selection, the financial and sentiment data sources, the feature engineering pipeline, the preprocessing methodology, the temporal partitioning strategy, and the evaluation metrics used throughout the subsequent chapters. Consolidating these elements in a single chapter maintains methodological consistency across contributions and provides the reader with a self-contained reference for the experimental design decisions that underpin all reported results.

The chapter proceeds as follows. Section 3.1 presents the stock selection and the rationale behind its composition. Section 3.2 describes the financial market data and its characteristics. Sections 3.3 and 3.4 detail the feature engineering pipeline and the sentiment data collection process, respectively. Section 3.5 covers data preprocessing and normalization. Section 3.6 explains the temporal partitioning strategy and walk-forward validation protocol. Section 3.7 defines the evaluation metrics. Section 3.8 summarizes the experimental framework and connects it to the contribution chapters. Figure 3.1 provides a high-level overview of the two-stream data pipeline that produces the final model input.

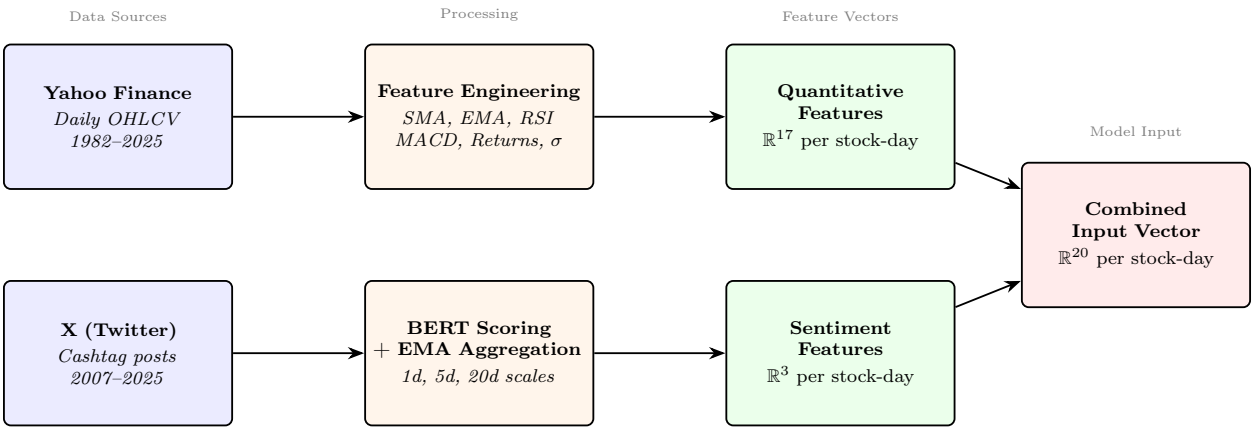

Figure 3.1: Overview of the two-stream data pipeline. Quantitative features are derived from daily OHLCV data via technical indicator computation (top). Sentiment features are extracted from social media posts via BERT-based scoring and multi-scale aggregation (bottom). Both streams are concatenated into a 20-dimensional input vector per stock per trading day.

3.1 Stock Selection and Rationale

The prediction system developed in this thesis operates on a set of 20 equities drawn from the Standard & Poor's 500 (S&P 500) index. The S&P 500 is the most widely tracked equity benchmark in global financial markets, representing approximately 80% of the total U.S. stock market capitalization [97]. Its constituent stocks are highly liquid and actively traded, with daily trading volumes that support reliable price discovery and reduce the impact of market microstructure noise on observed prices. The choice of the S&P 500 as the source index follows established practice in the financial machine learning literature: Fischer and Krauss [43] used all S&P 500 constituents for their LSTM evaluation, Feng et al. [57] evaluated temporal relational ranking on S&P 500 stocks, and the benchmark datasets in the FinRL library [88] are constructed from S&P 500 components.

From the full index, 20 stocks were selected to balance sector diversification, long historical price availability, and social-media sentiment coverage against computational tractability. The systems developed in Chapters 4, 5, and 6 all employ graph neural networks, and the node transformer architecture in particular, which constructs pairwise attention over all nodes at each forward pass. While NodeFormer's kernelized approximation achieves linear complexity

in the number of nodes [12], the full training pipeline, which includes BERT inference on per-stock sentiment streams, autoencoder training, dual-pathway optimization, and reinforcement learning controller updates, scales with the number of stocks across multiple components. Given these architectures, a selection of 20 stocks offers sufficient structural diversity for meaningful graph-based modeling while keeping the full training pipeline computationally feasible on a single-GPU workstation. Larger sets of 50 to 500 stocks are a natural extension. While the empirical evaluations in this thesis are confined to the 20-stock universe, Chapter 5 provides a detailed training-cost projection for scaling the regime-aware framework to 50 and 100 stocks and identifies the universe size beyond which edge sparsification would become necessary; a full empirical evaluation at those larger scales is left to future work.

The 20 selected stocks span eight sectors defined by the Global Industry Classification Standard (GICS), so that the graph captures both intra-sector co-movement and cross-sector decorrelation patterns. Table 3.1 lists the complete set of stocks with sector assignments and the date from which continuous trading data is available for each security. Most stocks in the selection have been continuously traded since 1982, providing over four decades of daily observations. Four stocks entered the market later due to their initial public offering (IPO) dates: Salesforce (June 2004), Netflix (May 2002), Visa (March 2008), and Microsoft (March 1986). These staggered entry dates are handled by having each stock contribute data only from its first available trading date onward, so that no synthetic pre-IPO values are introduced and each model processes variable-length per-stock sequences, with the expanding-window z-score normalization computed independently per stock from each stock's earliest available date. During each pre-IPO interval the corresponding node is absent from attention computations, message passing, and graph aggregation, and is excluded from both the loss computation and from all metric calculations, so that neither the gradient signal nor any reported metric depends on data that did not yet exist, which prevents the model from receiving future information about securities that were not yet publicly traded [98].

Table 3.1: The 20 stocks used across all experiments in this thesis, spanning eight GICS sectors to ensure cross-sectional diversity. The "Data Start" column indicates the earliest date for which continuous daily trading data is available; stocks with later start dates entered the market after their IPO.

Company	**Ticker**	**GICS Sector**	**Data Start**
Apple	AAPL	Information Technology	Jan 1982
Microsoft	MSFT	Information Technology	Mar 1986
Salesforce	CRM	Information Technology	Jun 2004
Netflix	NFLX	Communication Services	May 2002
JPMorgan Chase	JPM	Financials	Jan 1982
Visa	V	Financials	Mar 2008
Johnson & Johnson	JNJ	Health Care	Jan 1982
UnitedHealth Group	UNH	Health Care	Jan 1984
Pfizer	PFE	Health Care	Jan 1982
Walmart	WMT	Consumer Staples	Jan 1982
Home Depot	HD	Consumer Discretionary	Jan 1982
McDonald's	MCD	Consumer Discretionary	Jan 1982
ExxonMobil	XOM	Energy	Jan 1982
Chevron	CVX	Energy	Jan 1982
Procter & Gamble	PG	Consumer Staples	Jan 1982
Coca-Cola	KO	Consumer Staples	Jan 1982
Nike	NKE	Consumer Discretionary	Jan 1982
Verizon	VZ	Communication Services	Jan 1982
Boeing	BA	Industrials	Jan 1982
Caterpillar	CAT	Industrials	Jan 1982

The sector distribution is intentionally broad. Information technology stocks (AAPL, MSFT, CRM) exhibit high beta and are sensitive to growth expectations and interest rate changes. Communication services (NFLX, VZ) span both growth-oriented streaming media and defensive telecommunications. Financials (JPM, V) respond to credit conditions and yield curve dynamics. Health care (JNJ, UNH, PFE) provides defensive exposure with lower correlation to the broad market. Energy (XOM, CVX) tracks commodity cycles. Consumer staples (PG, KO, WMT) offer stability during downturns, while consumer discretionary (HD, MCD, NKE) is more cyclical and lags during risk-off periods. This diversity guarantees that the graph-based models encounter a range of correlation regimes, from tightly coupled sector pairs to near-independent cross-sector relationships, during both training and evaluation.

A methodological concern that must be acknowledged is survivorship bias. The 20 stocks were selected from the current S&P 500 composition, meaning that companies that were removed from the index due to poor performance, bankruptcy, or acquisition are not represented. Elton et al. [99] demonstrated that survivorship bias inflates apparent returns in mutual fund studies, and an analogous effect applies here: stocks that survived and remained in the S&P 500 through 2025 may have systematically different statistical properties from those that did not. This bias affects the generalizability of reported performance levels but does not invalidate the methodological comparisons that are the thesis's primary contribution. All models, both the proposed architectures and the baselines, are evaluated on the same survivorship-biased sample, so that relative performance differences reflect genuine architectural advantages rather than data selection artifacts.

Because every experiment, ablation, and reported result in this thesis is produced on this fixed set of 20 stocks, all quantitative findings should be read as characterizing this universe rather than the broader market. The 20-stock scope is a deliberate limitation adopted for computational tractability, and its consequences for generalizability, together with the related survivorship-bias concern discussed above, are examined in detail in Section 7.3.

3.2 Financial Market Data

Daily financial market data was obtained from Yahoo Finance via the `yfinance` Python library, which provides programmatic access to historical price and volume data for publicly traded securities [100]. Yahoo Finance is the most widely used free data source in academic financial machine learning research, adopted in studies ranging from individual stock prediction [43] to portfolio optimization [88]. While institutional-grade data providers such as Bloomberg, Refinitiv, and CRSP offer higher-fidelity data with more rigorous corporate action adjustments, Yahoo Finance provides sufficient accuracy for daily-frequency prediction research and guarantees full reproducibility without proprietary data licensing constraints.

For each stock and each trading day, six raw features are recorded: the opening price (O_t), the intraday high (H_t), the intraday low (L_t), the closing price (C_t), the adjusted closing price ($\tilde{C}_t$), and the trading volume (V_t). The adjusted closing price incorporates retroactive adjustments for stock splits, reverse splits, and dividend distributions, so that the price series reflects the true economic return to a buy-and-hold investor. When a company executes a k-for-1 stock split, all historical prices preceding the split date are divided by k, while all historical volume figures are multiplied by k, preserving continuity in both the price level and the implied dollar volume. Cash dividend adjustments are applied by multiplying all pre-ex-dividend-date prices by the factor $(1 - d/C_{\text{ex}})$, where d is the per-share dividend and C_{ex} is the closing price on the ex-dividend date. Apple, for example, has undergone five stock splits since 1982 (most recently a 4-for-1 split in August 2020), and without adjustment its historical price series would contain discontinuities exceeding 75% at each split date. All technical indicators and prediction targets in this thesis are computed from the adjusted close rather than the raw close, following the standard practice recommended by Prado [101] to prevent spurious discontinuities in price-derived features at split and dividend dates.

The dataset spans January 1, 1982 through March 31, 2025, a period of approximately 43 years encompassing roughly 10,800 trading days. This extended time horizon covers a diverse range of market conditions that are important for training and evaluating regime-aware prediction systems. The training period (1982–2010) includes the Black Monday crash of October 1987, the savings and loan crisis of the late 1980s, the extended bull market of the 1990s, the dot-com bubble and its collapse (1999–2002), and the global financial crisis of 2007–2009. The validation period (2011–2016) covers the European sovereign debt crisis aftermath, the 2013 "taper tantrum," the 2015 Chinese equity market sell-off, and the historically low-volatility environment of 2016. The test period (2017–2025) includes the February 2018 VIX spike, the December 2018 sell-off, the COVID-19 crash and subsequent recovery in 2020, the aggressive Federal Reserve rate hiking cycle of 2022–2023, and the AI-driven technology rally of 2023–2025. The presence of multiple crisis episodes across all

three partitions ensures that the system is trained on, validated against, and tested under regime transitions of varying severity and duration.

For the four stocks with IPO dates after 1982 (Section 3.1), data collection begins at the earliest available trading date; the corresponding model-level masking treatment of pre-IPO intervals is described in that section.

3.2.1 Prediction Target

The prediction target across all experiments in this thesis is the next-day adjusted closing price $\tilde{C}_{t+1}$ for each stock i in the selection, given information available up to and including trading day t. Formally, the model learns a function f such that $\hat{y}_{i,t+1} = f(\mathbf{x}_{i,1:t}, \mathbf{G}_t)$, where $\mathbf{x}_{i,1:t}$ is the historical feature sequence for stock i and $\mathbf{G}_t$ is the graph structure at time t. The one-day-ahead horizon was chosen for three reasons. First, it aligns with the daily rebalancing frequency used in most academic studies of stock prediction, making results directly comparable to the baselines reviewed in Chapter 2. Second, predicting the next trading day's close is the shortest horizon at which daily-frequency features carry meaningful signal; intraday prediction would require tick-level data and a fundamentally different feature set. Third, the one-day horizon imposes the strictest test of the model's ability to extract short-term predictive information, because the efficient market hypothesis implies that shorter-horizon returns are closer to unpredictable [8]. Multi-horizon extensions (5-day and 20-day forecasts) are explored in the ablation studies of Chapter 4, but the primary evaluation throughout the thesis uses the one-day target.

The target values are computed on the same z-score normalized scale as the input features (Section 3.5), so that the loss function operates on standardized residuals rather than raw dollar values. This normalization is reversed when reporting MAPE and other scale-dependent metrics, allowing the evaluation to reflect economically meaningful prediction errors.

3.3 Feature Engineering

Raw price and volume data are transformed into a set of 17 features per stock per trading day through the computation of 11 technical indicators that encode momentum, trend strength, mean-reversion signals, and volatility dynamics. Technical indicators have been a staple of quantitative trading since the development of chart-based analysis in the early twentieth century, and their use as input features for machine learning models is well established in the financial prediction literature [102, 103]. While individual indicators have limited predictive power in isolation, their combination provides a multi-scale representation of price dynamics that neural networks can exploit to learn complex nonlinear patterns.

The specific indicators selected for this thesis (moving averages, RSI, MACD, returns, and rolling volatility) were chosen to span three distinct categories of price information: trend (SMA and EMA at multiple horizons), momentum and mean-reversion (RSI and MACD), and risk (realized volatility). Several commonly used indicators were deliberately excluded. Bollinger Bands, which measure the deviation of price from a moving average in units of rolling standard deviation, are highly correlated with the combination of SMA and rolling volatility already present in the feature set, and including them would introduce near-linear redundancy without adding new information [104]. The Average True Range (ATR) is similarly redundant with rolling volatility for daily-frequency data where overnight gaps are small relative to intraday price ranges. The Stochastic Oscillator, which compares the closing price to the high-low range over a lookback period, overlaps conceptually with RSI in identifying overbought and oversold conditions, and prior comparative studies have found that including both yields negligible improvement over either alone in neural network prediction settings [105]. On-Balance Volume (OBV), which accumulates volume on up days and subtracts it on down days, was excluded because the raw volume field is already included in the feature vector, and the directional return features (r_t and $r_t^{\log}$) give the model the information needed to learn volume-direction interactions. The guiding principle

throughout feature selection was to maximize the diversity of price information available to the model while minimizing collinearity among features, because highly correlated inputs slow convergence and can destabilize gradient-based optimization in deep networks [106].

3.3.1 Moving Averages

The Simple Moving Average (SMA) computes the unweighted arithmetic mean of closing prices over a fixed lookback window of n trading days:

$$\mathrm{SMA}_n(t) = \frac{1}{n} \sum_{i=0}^{n-1} \tilde{C}_{t-i} \tag{3.1}$$

Three window lengths are used: $n \in \{5, 10, 20\}$, corresponding approximately to one trading week, two trading weeks, and one trading month. These windows represent short-term, medium-term, and monthly momentum, respectively. When the current price crosses above or below a moving average, technical analysts interpret the crossover as a potential trend reversal signal, and this relationship is one of the patterns that the neural network can learn to detect and weight [107].

The Exponential Moving Average (EMA) assigns exponentially decaying weights to past observations, giving greater influence to recent prices:

$$\mathrm{EMA}_n(t) = \alpha \cdot \tilde{C}_t + (1 - \alpha) \cdot \mathrm{EMA}_n(t-1) \tag{3.2}$$

where the smoothing factor $\alpha = 2/(n+1)$ controls the rate of decay. The same three window lengths are used ($n \in \{5, 10, 20\}$). The EMA responds more quickly to recent price changes than the SMA of the same window length, making it more sensitive to short-term momentum shifts. The inclusion of both SMA and EMA at matching windows provides the model with complementary views of the same temporal scale: the SMA captures the average level while the EMA captures the recent trajectory relative to that level.

3.3.2 Relative Strength Index

The Relative Strength Index (RSI), introduced by Wilder [108], quantifies the magnitude of recent price changes to assess whether a stock is overbought or oversold:

$$\mathrm{RSI}(t) = 100 - \frac{100}{1 + \bar{G}_{14}(t) \,/\, \bar{L}_{14}(t)} \tag{3.3}$$

where $\bar{G}_{14}(t)$ is the average of positive daily price changes over the preceding 14 trading days and $\bar{L}_{14}(t)$ is the average of negative daily price changes (taken as absolute values) over the same period. RSI values above 70 are conventionally interpreted as overbought conditions that may precede a downward correction, while values below 30 indicate oversold conditions. The 14-day window is the standard choice proposed by Wilder and remains the most common parameterization in both technical analysis and machine learning applications [102].

3.3.3 Moving Average Convergence Divergence

The Moving Average Convergence Divergence (MACD) indicator, developed by Appel [109], measures the relationship between two exponential moving averages of different periods:

$$\mathrm{MACD}(t) = \mathrm{EMA}_{12}(t) - \mathrm{EMA}_{26}(t) \tag{3.4}$$

$$\mathrm{Signal}(t) = \mathrm{EMA}_{9}(\mathrm{MACD}(t)) \tag{3.5}$$

The MACD line quantifies the convergence or divergence between short-term (12-day) and medium-term (26-day) momentum. The signal line, a 9-day EMA of the MACD itself, provides a smoothed reference. When the MACD crosses above the signal line, it suggests strengthening upward momentum; a cross below suggests weakening momentum. The MACD histogram (the difference between the MACD and signal lines) is often used as an additional feature, though in this thesis only the MACD line itself is included as a model input. The (12,

26, 9) parameter triple is the standard configuration used in the vast majority of technical analysis applications.

3.3.4 Returns and Volatility

Two measures of daily returns are computed. The arithmetic daily return captures the percentage change in price:

$$r_t = \frac{\tilde{C}_t - \tilde{C}_{t-1}}{\tilde{C}_{t-1}} \tag{3.6}$$

The logarithmic return yields a transformation with superior statistical properties for modeling purposes:

$$r_t^{\log} = \ln\left(\frac{\tilde{C}_t}{\tilde{C}_{t-1}}\right) \tag{3.7}$$

Logarithmic returns are time-additive (multi-period returns are the sum of single-period log returns) and more closely approximate a normal distribution than arithmetic returns, properties that are beneficial for the normalization step described in Section 3.5. Both return measures are included because arithmetic returns preserve interpretability (a 2% return means the price increased by 2%), while log returns offer the mathematical properties that statistical and neural network models exploit more effectively [110].

Rolling volatility is estimated as the sample standard deviation of daily returns over a 20-trading-day window:

$$\sigma_t = \sqrt{\frac{1}{19}\sum_{i=0}^{19}\left(r_{t-i} - \bar{r}_t\right)^2} \tag{3.8}$$

where $\bar{r}_t = \frac{1}{20}\sum_{i=0}^{19} r_{t-i}$ is the mean return over the window. This realized volatility estimate offers a direct measure of recent price variability that complements the GARCH-family conditional volatility models discussed in the literature review (Section 2.1). The 20-day window aligns with the standard convention for monthly realized volatility in financial econometrics [111].

3.3.5 Feature Summary

The complete feature vector for stock i at time t comprises six raw fields (open, high, low, close, and volume (OHLCV), along with the adjusted close) plus 11 derived indicators, yielding 17 dimensions:

$$\begin{aligned}\mathbf{x}_{i,t} = [&O_t,\ H_t,\ L_t,\ C_t,\ \tilde{C}_t,\ V_t,\ \mathrm{SMA}_5,\ \mathrm{SMA}_{10},\ \mathrm{SMA}_{20},\\ &\mathrm{EMA}_5,\ \mathrm{EMA}_{10},\ \mathrm{EMA}_{20},\ \mathrm{RSI},\ \mathrm{MACD},\ r_t,\ r_t^{\log},\ \sigma_t] \in \mathbb{R}^{17}\end{aligned} \tag{3.9}$$

where the time subscripts on the indicators are suppressed for readability. The six raw fields and 11 derived indicators overlap by design at certain temporal scales (for example, both SMA_5 and EMA_5 summarize the last five trading days), but the overlap is complementary rather than redundant: the SMA responds symmetrically to all observations in the window, while the EMA front-loads recent observations. This complementarity allows the neural network to learn different filtering behaviors from the same underlying data, an approach validated by Sezer et al. [112], who showed that multi-indicator feature sets consistently outperform single-indicator alternatives in deep learning prediction tasks.

Volume (V_t) occupies a distinct distributional space from the remaining 16 features. Daily trading volume is non-negative, right-skewed, and exhibits strong serial autocorrelation and calendar effects (volume tends to be lower on Mondays and higher around options expiration dates). Its magnitude varies by several orders of magnitude across stocks: large technology stocks regularly exceed 50 million shares traded per day, while smaller constituents may average fewer than 5 million. The expanding window z-score normalization applied to all features (Section 3.5) accommodates these differences by centering and scaling each stock's volume series individually, converting the raw share counts into standardized deviations from each stock's long-run average volume. After normalization, volume spikes (which often accompany earnings announcements, index rebalances, or crisis events) appear as positive outliers in the standardized distribution, preserving their informational content for the model.

This feature set is shared by all prediction architectures evaluated in this thesis (Chapters 4–6).

All technical indicators in equation (3.9) (SMA, EMA, RSI, MACD, rolling volatility, daily and log returns) are computed with the open-source `pandas-ta` Python library [113], which provides reference implementations following standard definitions [114]. The sentiment-derived and event-related features that appear in the event context vector $\mathbf{c}_t$ described in the Event Node Transformer subsection (the sentiment-spike binary flag and scaled magnitude, the earnings-window indicator and days-to-announcement proximity, the sector-average historical surprise, and the cross-asset reconstruction-error mean and standard deviation) are produced by custom code developed for this work, because they involve domain-specific definitions (the two-standard-deviation spike threshold, the in-window indicator around scheduled earnings dates, and the aggregation across the autoencoder outputs of all 20 stocks) that are not provided by standard technical-analysis packages.

3.3.6 Router-Specific Features

The regime-aware system developed in Chapter 5 requires an additional set of cross-sectional and macroeconomic features. These six features are:

$$\mathbf{x}_{i,t}^{\text{router}} = [\sigma_t^{(5)},\ \sigma_t^{(20)},\ \Delta\text{VIX}_t,\ \Delta\rho_t,\ |S_t|,\ \nu_t^{\text{post}}] \in \mathbb{R}^6 \tag{3.10}$$

where $\sigma_t^{(5)}$ and $\sigma_t^{(20)}$ are the 5-day and 20-day rolling volatilities, ΔVIX_t is the daily percentage change in the CBOE Volatility Index, $\Delta\rho_t$ is the change in the average pairwise return correlation among the 20 stocks, $|S_t|$ is the absolute magnitude of the aggregated sentiment score, and ν_t^{post} is the post velocity (the count of social media posts mentioning crisis-related keywords within the trading day). These features are computed in addition to the base 17 features and are used by the regime-aware model developed in Chapter 5.

3.4 Sentiment Data

The integration of textual sentiment with quantitative price features is a defining element of the prediction system developed in Chapter 4. Two distinct datasets support this integration: the Financial Microblog Sentiment (FMS) dataset [23], which provides expert-annotated labels for fine-tuning the BERT sentiment classifier, and the Comprehensive Stock Sentiment (CSS) dataset, which contains the raw social media posts from which per-stock sentiment scores are extracted at inference time.

3.4.1 Financial Microblog Sentiment Dataset

The FMS dataset originates from the SemEval-2017 Task 5 shared task on fine-grained sentiment analysis of financial microblogs [23]. It is a publicly available corpus of finance-related social media messages in which each message has been assigned a sentiment score $s \in [-1, +1]$ by domain experts with professional experience in financial markets, where -1 indicates strongly negative sentiment, 0 indicates neutral sentiment, and $+1$ indicates strongly positive sentiment. Each record includes the source platform identifier, the message ID, any relevant cashtags mentioned in the text, and annotated sentiment spans that identify which portions of the message carry sentiment information. The annotation protocol required each message to be evaluated by multiple independent annotators, with the final score determined by averaging across annotators to reduce individual bias. This dataset serves as the supervised training signal for fine-tuning the pre-trained BERT model [4] to the financial sentiment classification task. The fine-tuning procedure follows the standard transfer learning approach for transformer-based models: the pre-trained language representations are adapted to the target domain through additional training on the labeled corpus, preserving the general linguistic knowledge acquired during pre-training while specializing the classification head for financial sentiment [69]. The FMS dataset is partitioned into 70% training, 15% validation,

and 15% test subsets, stratified by sentiment class to maintain balanced representation across the score range.

3.4.2 Comprehensive Stock Sentiment Dataset

The CSS dataset was constructed by systematically collecting posts from X (formerly Twitter) that mention the 20 stocks in the prediction set. Posts were retrieved via the X API using cashtag-based search queries, where each stock's cashtag (e.g., $AAPL for Apple, $MSFT for Microsoft) serves as the primary search key. Table 3.2 lists the complete set of cashtag search terms used for each stock. Because the platform was founded in 2006, the earliest available posts date from early 2007, and the collection period accordingly spans January 2007 through March 2025, yielding approximately 4.2 million posts across all 20 stocks.

Table 3.2: Cashtag search terms used to collect social media posts from X (formerly Twitter) for each stock in the prediction set.

Cashtag	**Company**	**Cashtag**	**Company**
$AAPL	Apple	$KO	Coca-Cola
$MSFT	Microsoft	$CVX	Chevron
$JPM	JPMorgan Chase	$HD	Home Depot
$JNJ	Johnson & Johnson	$VZ	Verizon
$WMT	Walmart	$BA	Boeing
$XOM	ExxonMobil	$MCD	McDonald's
$PG	Procter & Gamble	$NFLX	Netflix
$V	Visa	$CAT	Caterpillar
$UNH	UnitedHealth Group	$PFE	Pfizer
$NKE	Nike	$CRM	Salesforce

The coverage of social media discussion varies substantially across stocks. Large technology companies such as Apple and Microsoft generate thousands of cashtag-bearing posts per trading day, while industrial stocks such as Caterpillar and Boeing receive considerably less attention. This heterogeneity in coverage is an inherent property of social media data and has direct implications for the sentiment fusion mechanism: the model must learn to rely more heavily on quantitative features for stocks with sparse sentiment coverage while exploiting

the richer textual signal available for heavily discussed stocks. The adaptive fusion gate developed in Chapter 4 addresses this challenge by conditioning the sentiment contribution on the magnitude and reliability of the available signal.

3.4.3 Data Quality Considerations

Social media text poses several data quality challenges that must be accounted for during collection and processing. Automated accounts (commonly called bots) generate a non-trivial fraction of financial cashtag traffic, particularly during volatile market periods when promotional or manipulative posting activity increases [115]. Bot-generated content tends to be repetitive, often consisting of copied headlines or templated price alerts, and its sentiment scores cluster around neutral values because the text lacks genuine opinion. Rather than attempting explicit bot detection, which is unreliable for sophisticated accounts, the multi-day exponential moving average aggregation described in Section 3.4 mitigates the influence of bot-generated posts: because most bot content clusters near neutral sentiment ($s \approx 0$), its contribution to the daily mean is diluted when mixed with opinionated human-authored posts, and the smoothing inherent in the 5-day and 20-day averages further reduces the impact of any single day's bot activity.

Sarcasm and irony present a second challenge. A post stating "Great job crashing the stock" carries negative sentiment despite the positive surface-level wording. Pre-trained language models such as BERT have been shown to capture some degree of contextual irony through their attention mechanisms [116], and the fine-tuning on the FMS corpus, which contains sarcastic examples in its training data, further calibrates the model toward financial-domain sarcasm. Residual sarcasm misclassification contributes noise to the daily sentiment aggregates, but its effect is bounded because sarcastic posts represent a small minority of the total cashtag traffic for any given stock on any given day.

Language filtering constitutes a third consideration. The X API returns posts in all languages, but the BERT model used for sentiment scoring was pre-trained on English text and fine-tuned on an English-language corpus. Non-English posts are filtered during the preprocessing stage using a language detection step based on the character n-gram distribution of each post [117]. Posts identified as non-English with high confidence are excluded from the sentiment pipeline. This filtering removes approximately 8–12% of the raw post volume depending on the stock, with higher removal rates for multinational brands (e.g., Coca-Cola and Nike) that attract discussion in many languages.

3.4.4 Text Preprocessing

Raw social media posts undergo a multi-stage cleaning pipeline before sentiment scoring, with each stage designed to reduce noise while preserving the semantic content that carries sentiment information. Structural artifacts are addressed first: Hypertext Markup Language (HTML) tags and embedded formatting codes are stripped to extract plain text, and retweet prefixes ("RT @user:") are removed to avoid counting the same opinion multiple times. Identity and external references are then anonymized by replacing usernames mentioned via the @ symbol with a generic `[USER]` token and substituting Uniform Resource Locators (URLs) with a standardized `[URL]` token, as neither user identity nor linked content is available to the text-only sentiment model. Finally, the text is normalized for tokenization: elongated words (e.g., "gooood" or "baaad") are reduced to their base form through character de-elongation, and ticker symbols embedded within running text are separated by spaces (e.g., "$AAPL$MSFT" becomes "$AAPL $MSFT") to prevent the tokenizer from generating out-of-vocabulary embeddings. These preprocessing steps follow established best practices in social media natural language processing [118] and are applied identically across all stocks and time periods to maintain consistency.

3.4.5 Sentiment Scoring and Aggregation

Once the BERT model has been fine-tuned on the FMS labels, it is applied to each preprocessed post in the CSS corpus to produce a scalar sentiment score $s \in [-1, +1]$. On any given trading day, a stock may attract anywhere from zero to several thousand posts. Individual post scores are noisy and sensitive to outlier messages, so they are aggregated into a three-dimensional sentiment feature vector per stock per day using exponential moving averages at multiple time scales:

$$\mathbf{S}_{i,t} = [S_{i,t}^{(1\mathrm{d})},\ S_{i,t}^{(5\mathrm{d})},\ S_{i,t}^{(20\mathrm{d})}] \in \mathbb{R}^3 \tag{3.11}$$

where $S_{i,t}^{(k\mathrm{d})}$ is the k-day exponential moving average of the raw daily mean sentiment score for stock i. The 1-day component captures immediate market reactions to breaking news or earnings announcements. The 5-day component smooths out day-to-day noise and reflects weekly sentiment trends. The 20-day component tracks sustained shifts in investor attitude over approximately one trading month. This multi-scale representation allows the model to distinguish between a single day of panic (which would affect only the 1-day component) and a prolonged deterioration in confidence (which would be reflected across all three scales). The three sentiment features are appended to the 17-dimensional quantitative feature vector described in Section 3.3, bringing the total input dimensionality to 20 features per stock per day during periods with sentiment coverage.

3.4.6 Temporal Alignment

The CSS dataset begins in January 2007, while the financial market data extends back to January 1982. For all training samples that fall before the start of sentiment coverage (1982–2006), the sentiment feature vector is set to zero, effectively representing neutral sentiment. This design choice ensures that the model can learn from the full 28-year quantitative training

history while treating the absence of sentiment data as an informative signal (indicating a period when social media sentiment was not yet available as a data source). From 2007 onward, the model has access to both quantitative and sentiment features, and the fusion mechanism can learn the conditional value of textual information relative to price-derived features. The transition from zero-sentiment to live-sentiment during the training period creates an implicit training signal for the adaptive fusion gate: the model observes that sentiment features carry no information before 2007 and learns to downweight them accordingly, then observes increasing informativeness as social media adoption grows through the late 2000s and 2010s. Table 3.3 summarizes the key characteristics of the financial and sentiment datasets.

Table 3.3: Summary of the financial market data (FMD) and Comprehensive Stock Sentiment (CSS) datasets used across all experiments.

Characteristic	**FMD**	**CSS**
Time period	1982–2025	2007–2025
Number of stocks	20	20
Trading days	≈10,800	≈4,500
Features per stock-day	17	3
Total observations	≈216,000	≈4.2M posts

3.5 Data Preprocessing

Raw feature values span different scales: closing prices range from single digits (early-period Apple shares post-split adjustment) to hundreds of dollars (UnitedHealth Group in 2024), while RSI values are bounded between 0 and 100 and daily returns are typically on the order of 10^{-2}. Neural networks trained on unnormalized inputs converge slowly and are prone to gradient saturation in the layers that process large-magnitude features, while small-magnitude features contribute negligibly to parameter updates. Proper normalization is therefore necessary for stable and efficient training [106].

3.5.1 Missing Value Imputation

Trading halts, public holidays observed by specific exchanges, and data feed interruptions occasionally produce gaps in the daily price record. The imputation strategy differs between training and evaluation to prevent information leakage. During training, short gaps of one to two trading days are filled using linear interpolation between the last observed value and the next available observation:

$$\tilde{C}_t = \tilde{C}_{t-k} + \frac{\tilde{C}_{t+m} - \tilde{C}_{t-k}}{k+m} \cdot k \tag{3.12}$$

where k is the number of days since the last observation and m is the number of days until the next observation. Linear interpolation in the training set is acceptable because both surrounding values are part of the historical record used for parameter estimation. During validation and testing, only forward-filling (carrying the last observed value forward) is permitted:

$$\tilde{C}_t = \tilde{C}_{t-k} \tag{3.13}$$

This asymmetry guarantees that validation and test predictions are made using only information that would have been available in real time, consistent with the no-lookahead principle that governs all aspects of the experimental design [101].

3.5.2 Expanding Window Normalization

Features are normalized using an expanding window z-score transformation. For feature j of stock i at time t, the normalized value is:

$$\tilde{x}_{i,t}^{(j)} = \frac{x_{i,t}^{(j)} - \mu_{1:t}^{(j)}}{\sigma_{1:t}^{(j)}} \tag{3.14}$$

where $\mu_{1:t}^{(j)}$ and $\sigma_{1:t}^{(j)}$ are the mean and standard deviation of feature j computed over all observations from the beginning of the training period through time t. The expanding window

ensures that the normalization statistics grow more stable over time as additional observations accumulate, unlike a rolling window approach that would discard older observations and produce more volatile normalization parameters. This choice reflects the financial econometric convention that long-run distributional parameters are estimated more reliably with longer sample histories [110].

During the test period, the normalization statistics are fixed at their values computed over the full training set:

$$\tilde{x}_{i,t}^{(j)} = \frac{x_{i,t}^{(j)} - \mu_{1:T_{\text{train}}}^{(j)}}{\sigma_{1:T_{\text{train}}}^{(j)}} \tag{3.15}$$

where T_{train} denotes the last day of the training period. Computing running statistics during the test period would allow the normalization to adapt to test-period distributional properties, which constitutes a subtle form of information leakage. Fixing the statistics at training-period values means that the model's test-time behavior depends only on patterns learned from the training data, providing a conservative and honest evaluation of generalization performance.

3.5.3 Stationarity Considerations

Raw price levels are non-stationary: they exhibit long-term trends, structural breaks, and time-varying volatility that violate the stationarity assumptions underpinning most statistical and machine learning models. The feature engineering step addresses this concern by transforming non-stationary price series into approximately stationary derived features. Daily returns and log returns remove the price level trend, RSI and MACD are bounded or mean-reverting by construction, and rolling volatility captures the time-varying second moment without inheriting the price level. The z-score normalization further stabilizes the feature distribution by centering and scaling each feature relative to its historical behavior. These transformations do not guarantee strict stationarity, as financial return distributions are well known to exhibit time-varying moments [25], but they bring the input features sufficiently close to stationarity for neural network training to proceed effectively.

3.5.4 Outlier Treatment

Financial markets occasionally produce extreme observations: the S&P 500 fell 12% on Black Monday (October 19, 1987), and individual stocks can experience single-day moves exceeding 20% following earnings surprises or corporate events. Explicit outlier clipping (e.g., winsorizing at the 1st and 99th percentiles) would remove precisely the crisis-period observations that are most informative for training regime-aware models. The z-score normalization naturally compresses extreme values toward the tails of the standardized distribution without discarding them entirely. The expanding window further mitigates the influence of individual outliers, as long-run statistics are robust to occasional extreme values when estimated over thousands of observations. No additional outlier treatment is applied beyond the z-score normalization, so that the model retains access to the full range of market behavior during training.

3.6 Dataset Partitioning and Validation Protocol

The dataset is divided into three non-overlapping temporal partitions: training, validation, and test. Unlike image classification or natural language processing tasks where random shuffling of samples is standard practice, financial time series prediction requires strictly temporal partitioning to prevent information leakage. Random splitting would allow future observations to appear in the training set while earlier observations populate the test set, enabling the model to exploit statistical relationships that would not have been available in real time. Prado [101] identified this form of leakage as one of the most common and consequential methodological errors in the financial machine learning literature, demonstrating that models evaluated with randomized cross-validation routinely produce inflated performance estimates that collapse under proper temporal evaluation. Tashman [119] established walk-forward validation as the gold standard for time series model evaluation, a protocol in which the

training window precedes the validation window, which in turn precedes the test window, with no temporal overlap between any pair of partitions.

3.6.1 Partition Boundaries

Table 3.4 summarizes the three temporal partitions and their approximate share of the total time span. The boundaries were chosen to allocate the longest uninterrupted record to training while reserving distinct market environments for validation and testing.

Table 3.4: Temporal partitioning of the financial and sentiment datasets. The training period covers approximately 70% of the total time span, with validation and test each covering approximately 15%. All splits are strictly non-overlapping to prevent information leakage.

Partition	**Start Date**	**End Date**	**Approx. Share**
Training	January 1, 1982	December 31, 2010	70%
Validation	January 1, 2011	December 31, 2016	15%
Test	January 1, 2017	March 31, 2025	15%

The training period (1982–2010) spans approximately 7,300 trading days per stock (for those available from 1982) and encompasses the full spectrum of market conditions described in Section 3.2, from extended bull runs through severe financial crises. This breadth is necessary for training regime-aware models: a training set restricted to calm, trending markets would produce a model incapable of recognizing volatility spikes, correlation breakdowns, and the nonlinear dynamics that accompany crisis periods.

The validation period (2011–2016) is used for hyperparameter selection, early stopping, and model checkpoint selection. As noted in Section 3.2, this window spans a mix of post-crisis recovery, policy-driven volatility, and historically calm conditions, offering a sufficiently varied calibration environment without overlapping with the extreme events reserved for training and testing. Hyperparameters optimized on this period include learning rates, dropout rates, attention head counts, and the autoencoder latent dimension.

The test period (2017–2025) constitutes the final, out-of-sample evaluation on which all reported results are based. No architectural decisions, hyperparameter choices, or feature

selections are informed by test-period data. The market conditions during this window (summarized in Section 3.2) include at least four distinct regime transitions within eight years, making it a particularly challenging evaluation period. The frequency of these transitions tests the regime-aware components developed in Chapter 5 under conditions not observed during training. Figure 3.2 illustrates the three partitions along the full time axis and marks the principal market events that each partition contains.

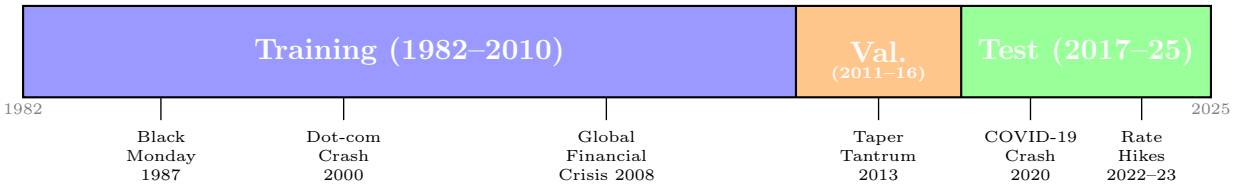

Figure 3.2: Temporal partitioning of the dataset into training, validation, and test periods. Tick marks below the timeline indicate the principal market events within each partition. The presence of regime-changing events across all three partitions allows the models to be trained on, validated against, and tested under diverse market conditions.

3.6.2 Walk-Forward Evaluation Protocol

The evaluation follows a single-pass walk-forward protocol: the model is trained once on the training set, checkpointed using validation performance, and evaluated on the test set without retraining or parameter updates. This protocol is more conservative than expanding-window retraining, in which the model is periodically retrained on all data up to the current point, because it tests the model's ability to generalize across a full eight-year out-of-sample period without the benefit of adaptation to more recent data. The choice of single-pass evaluation is deliberate: it yields a lower bound on expected performance, and any model that demonstrates consistent accuracy under this protocol can be expected to perform at least as well under more permissive retraining schemes.

An alternative evaluation strategy used in parts of the financial machine learning literature is purged k-fold cross-validation [101], in which the training and test folds are separated by a gap equal to the maximum lookback window used in feature computation. Purged

cross-validation yields tighter confidence intervals on performance estimates because every observation serves as both a training and test sample across folds. However, it is computationally expensive (requiring k full training runs) and introduces a philosophical complication for regime-aware models: folds that place crisis periods in the training set and pre-crisis periods in the test set create an unrealistic evaluation scenario in which the model has "seen" regime transitions that have not yet occurred from the test set's perspective. The single-pass walk-forward protocol avoids this temporal inversion entirely and is the dominant evaluation framework in the deep learning forecasting literature [120].

3.6.3 Feature Boundary Considerations

Several features in the engineering pipeline use lookback windows that span the partition boundary. The 20-day rolling volatility computed on January 2, 2017 (the first test-period trading day) includes return observations from December 2016, which falls in the validation period. This boundary overlap is methodologically acceptable because the feature computation uses only past data (returns preceding the current date), and the same feature values would be available to a real-time trading system operating on January 2, 2017. The key constraint is that no parameter estimation (model weights, normalization statistics, or hyperparameter selections) uses test-period data, and this constraint is maintained throughout all experiments.

3.7 Evaluation Metrics

The prediction systems developed in this thesis are evaluated using eight complementary metrics that assess forecast accuracy and trading performance from multiple perspectives. No single metric captures all dimensions of predictive quality: point error metrics measure magnitude accuracy, directional metrics assess the ability to predict price movement direction, benchmark-relative metrics compare forecast quality against naive baselines, cross-sectional

metrics evaluate the model's ability to rank stocks by expected return, and trading metrics measure whether prediction accuracy translates into risk-adjusted gains and controlled downside exposure. The use of multiple metrics yields a robust evaluation that prevents misleading conclusions based on a single favored criterion.

3.7.1 Mean Absolute Percentage Error

The Mean Absolute Percentage Error (MAPE) measures the average percentage deviation between predicted and actual values:

$$\text{MAPE} = \frac{100}{N} \sum_{i=1}^{N} \left| \frac{y_i - \hat{y}_i}{y_i} \right| \tag{3.16}$$

where y_i is the actual value, $\hat{y}_i$ is the predicted value, and N is the number of predictions. MAPE is scale-invariant, meaning that errors are expressed as percentages of the true values and are directly comparable across stocks with different price levels. A MAPE of 1% indicates that the average prediction deviates from the actual price by 1% of the true value. The principal limitation of MAPE is its asymmetry: it penalizes over-predictions less heavily than under-predictions of the same absolute magnitude, because the denominator (the actual value) is smaller for under-predictions. MAPE is also undefined when the actual value is zero, though this situation does not arise in stock price prediction where prices are strictly positive. MAPE has been widely adopted as a primary metric in the financial forecasting literature [121].

One consequence of applying MAPE to next-day closing-price prediction is that the reported values are small in absolute terms, often well below one percent. This is expected rather than surprising, because the day-over-day change in a large-cap closing price is itself a small fraction of the price level, so predicting the next-day price to within a fraction of a percent is attainable even for weak models, and the naive random-walk forecast that carries today's price forward already attains a low MAPE. A low MAPE on its own should

therefore not be read as evidence of strong predictive skill. For this reason the evaluation pairs MAPE with the Theil's U statistic, which measures forecast error relative to the random-walk baseline, and with directional accuracy, both defined below. These two metrics are more discriminating because they isolate the portion of performance that exceeds what is available from the most recent price observation alone.

3.7.2 Root Mean Squared Error

The Root Mean Squared Error (RMSE) measures the standard deviation of prediction residuals:

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{3.17}$$

Unlike MAPE, RMSE is expressed in the same units as the target variable (dollars for unnormalized prices, standard deviations for z-scored prices) and assigns quadratically increasing penalties to larger errors. This sensitivity to large errors makes RMSE especially informative for evaluating performance during crisis periods, when prediction errors are most consequential and most likely to be large. The RMSE reported in this thesis is computed on z-score normalized predictions, ensuring cross-stock comparability [122].

3.7.3 Directional Accuracy

Directional Accuracy (DA) measures the proportion of trading days on which the model correctly predicts the direction of price movement:

$$\text{DA} = \frac{100}{N}\sum_{i=1}^{N}\mathbf{1}[\text{sign}(\hat{y}_i - y_{i-1}) = \text{sign}(y_i - y_{i-1})] \tag{3.18}$$

where $\mathbf{1}[\cdot]$ is the indicator function. A DA of 50% corresponds to random-chance performance for an unbiased predictor. DA is a particularly relevant metric for trading applications,

as many trading strategies depend on directional signals rather than precise magnitude predictions. A model with high MAPE but high DA may still generate profitable trading signals if it consistently identifies the correct direction of price movement, even if its magnitude estimates are imprecise. Conversely, a model with low MAPE but low DA may achieve good magnitude accuracy by predicting values close to the current price without reliably identifying directional changes.

3.7.4 Theil's U Statistic

Theil's U statistic [123] compares the forecast error of the model against the error of a naive random walk baseline:

$$U = \frac{\sqrt{\sum_{t=1}^{N}(y_{t+1} - \hat{y}_{t+1})^2}}{\sqrt{\sum_{t=1}^{N}(y_{t+1} - y_t)^2}} \tag{3.19}$$

The denominator is the RMSE of a naive forecast that predicts tomorrow's price as today's price. A value of $U < 1$ indicates that the model outperforms the random walk, $U = 1$ indicates equivalence, and $U > 1$ indicates that the model is worse than the naive baseline. Theil's U offers a stringent benchmark: in efficient markets where prices follow near-random-walk dynamics, achieving U substantially below 1 is difficult and constitutes strong evidence of genuine predictive power. Any model reporting $U \geq 1$ on out-of-sample data has failed to extract predictive signal beyond what is available from the most recent price observation alone.

3.7.5 Information Coefficient

The Information Coefficient (IC) measures the cross-sectional rank correlation between predicted and realized returns across all 20 stocks at each time step:

$$\mathrm{IC}_t = \rho_s(\hat{\mathbf{r}}_t,\ \mathbf{r}_t) \tag{3.20}$$

where ρ_s denotes the Spearman rank correlation, $\hat{\mathbf{r}}_t$ is the vector of predicted returns across all 20 stocks at time t, and $\mathbf{r}_t$ is the vector of realized returns. Because the prediction systems in this thesis output the adjusted closing price $\tilde{C}_{t+1}$ rather than the return directly (Section 3.2), the predicted return vector is computed from predicted prices using the percentage-change formula in Equation 3.6, and the realized return vector is computed analogously from observed prices. Cross-sectional ranking is evaluated on returns rather than on raw price levels because the rank ordering of price levels across the 20 stocks is largely static from one trading day to the next and would not reflect predictive skill, whereas return rankings change daily and depend on the model's ability to identify which stocks will outperform their cross-sectional peers. The IC is then averaged over all time steps in the evaluation period. This metric is the standard measure of alpha signal quality in quantitative portfolio management [124]: an IC of 0.05 (5% rank correlation) is considered a meaningful signal, while an IC above 0.10 is strong by industry standards. Unlike MAPE and RMSE, which evaluate per-stock accuracy independently, the IC assesses the model's ability to distinguish relative winners from losers across the portfolio, a capability that is important for cross-sectional trading strategies.

3.7.6 Confidence Tracking Rate

The Confidence Tracking Rate (CTR) captures the proportion of predictions where model confidence agrees with actual accuracy:

$$\text{CTR} = \frac{1}{NT} \sum_{i,t} \mathbb{I}((\text{conf}_{i,t} > \bar{c}) = (|\hat{y}_{i,t+h} - y_{i,t+h}| < \bar{\epsilon})) \tag{3.21}$$

where $\text{conf}_{i,t}$ is the inverse prediction variance from the two pathways, $\bar{c}$ is the median confidence, and $\bar{\epsilon}$ is the median absolute error across all predictions. The indicator function equals 1 when both conditions agree: either the model is confident and accurate, or it is unconfident and inaccurate. CTR quantifies whether the model "knows when it knows": a well-calibrated model should report high confidence when its predictions are accurate and low confidence when they are not. In the dual-pathway system (Chapter 5), confidence is derived from the agreement between the two prediction pathways, and the CTR assesses whether the routing mechanism successfully identifies conditions under which the system is more or less reliable.

3.7.7 Sharpe Ratio

The Sharpe ratio measures risk-adjusted return by dividing the mean excess return of a trading strategy by the standard deviation of its returns:

$$\text{Sharpe} = \frac{\bar{r}_p - r_f}{\sigma_p} \tag{3.22}$$

where $\bar{r}_p$ is the mean portfolio return over the evaluation period, r_f is the risk-free rate (approximated as zero for daily returns), and σ_p is the standard deviation of portfolio returns. A higher Sharpe ratio indicates that the strategy achieves greater return per unit of risk, with values above 1.0 generally considered strong for daily equity strategies. Unlike

MAPE and DA, which evaluate prediction quality in isolation, the Sharpe ratio assesses whether prediction accuracy translates into economically viable trading performance after accounting for return variability. This metric is particularly important in Chapter 6, where behavioral evaluation scores are validated against realized Sharpe ratios to demonstrate that LLM-assessed behavioral quality predicts risk-adjusted trading outcomes.

3.7.8 Maximum Drawdown

Maximum Drawdown (MDD) measures the largest peak-to-trough decline in cumulative portfolio value over the evaluation period:

$$\text{MDD} = \max_{t \in [0,T]} \left(\frac{\max_{\tau \in [0,t]} V_\tau - V_t}{\max_{\tau \in [0,t]} V_\tau} \right) \tag{3.23}$$

where V_t is the portfolio value at time t. MDD captures the worst-case loss an investor would have experienced, expressed as a percentage of the peak value. This metric complements the Sharpe ratio by isolating tail risk: a strategy may exhibit an attractive Sharpe ratio while concealing a single catastrophic drawdown that would be unacceptable in practice. In the behavioral evaluation framework (Chapter 6), MDD serves as the primary metric for validating the risk calibration dimension, testing whether systems that receive higher risk calibration scores from LLM judges also exhibit smaller maximum drawdowns. Table 3.5 summarizes all eight evaluation metrics, their domains, ideal values, and the chapters in which each metric is reported.

Table 3.5: Summary of evaluation metrics used across all experimental chapters. Each metric captures a distinct dimension of predictive quality, and the combination provides a comprehensive assessment that no single metric could offer alone. Metrics marked with † assess trading performance rather than prediction accuracy.

Metric	**Measures**	**Range**	**Ideal**	**Used In**
MAPE	Percentage error	$[0, \infty)$	$\to 0$	Ch. 4, 5, 6
RMSE	Magnitude error	$[0, \infty)$	$\to 0$	Ch. 4, 5, 6
DA	Direction accuracy	$[0, 100]$	$\to 100$	Ch. 4, 5, 6
Theil's U	vs. random walk	$[0, \infty)$	< 1	Ch. 4, 5
IC	Cross-sectional rank	$[-1, 1]$	$\to 1$	Ch. 4, 5
CTR	Confidence calibration	$[0, 1]$	$\to 1$	Ch. 5
Sharpe†	Risk-adjusted return	$(-\infty, \infty)$	> 1	Ch. 4, 6
MDD†	Worst-case drawdown	$[0, 1]$	$\to 0$	Ch. 6

3.8 Summary

This chapter has established the experimental foundation shared by all three contributions of this thesis. The prediction set comprises 20 S&P 500 equities spanning eight GICS sectors, with daily financial data from 1982 to 2025 and social media sentiment data from 2007 to 2025. The prediction target is the next-day adjusted closing price, chosen for its alignment with academic benchmarks and its stringent test of short-term predictive ability. Each stock is represented by a 17-dimensional feature vector comprising raw OHLCV data and 11 technical indicators selected to maximize coverage of trend, momentum, and volatility information while minimizing inter-feature collinearity. The regime-aware system additionally uses six router-specific features that reflect cross-sectional and macroeconomic signals. Sentiment features are derived from approximately 4.2 million social media posts collected via cashtag-based searches and processed through a BERT model fine-tuned on expert-annotated financial text. Non-English posts are filtered during preprocessing, while bot activity and sarcasm are treated as residual sources of noise that are not addressed at the preprocessing stage. Expanding window z-score normalization prevents future information from contaminating the feature values used for prediction, and volume features receive the same standardization

treatment despite their distinct distributional properties. The dataset is partitioned into training (1982–2010), validation (2011–2016), and test (2017–2025) periods using strictly temporal splits that prevent data leakage, and all results are reported under a single-pass walk-forward protocol that yields a conservative lower bound on generalization performance. Eight evaluation metrics (MAPE, RMSE, DA, Theil's U, IC, CTR, Sharpe ratio, and MDD) provide complementary assessments of forecast accuracy, directional precision, baseline-relative performance, cross-sectional signal quality, confidence calibration, risk-adjusted return, and tail risk exposure. With this shared experimental foundation in place, the following chapter presents the first contribution of this thesis: a node transformer architecture with attention-based BERT sentiment fusion, evaluated on the dataset and using the metrics established here.

4

Stock Market Prediction Using Node Transformer Architecture Integrated with BERT Sentiment Analysis

This chapter is based on the following publication:

Al Ridhawi, M., Haj Ali, M., & Al Osman, H. (2026). Stock Market Prediction Using Node Transformer Architecture Integrated with BERT Sentiment Analysis. *IEEE Access*, vol. 14, pp. 72613–72631. doi: 10.1109/ACCESS.2026.3691980.

This chapter presents the first contribution of this thesis: an integrated framework that combines a node transformer architecture with BERT-based sentiment analysis for stock price forecasting. The model represents the stock market as a graph in which individual stocks serve as nodes and learnable edges encode sectoral affiliations and price correlations. A fine-tuned BERT model extracts sentiment signals from social media posts, which are merged with quantitative features through an adaptive attention-based fusion mechanism. The dataset, feature engineering pipeline, and evaluation metrics used throughout this chapter follow the experimental framework established in Chapter 3.

The chapter is organized as follows. Section 4.1 positions the contribution relative to the closest prior work. Section 4.2 presents the system architecture, graph representation, node transformer, BERT sentiment module, and fusion mechanism. Section 4.3 describes the experimental setup, baselines, and results. Section 4.4 discusses the findings and limitations. Section 4.5 summarizes the chapter.

4.1 Related Work and Research Gap

The methodological context underlying this chapter spans graph-based financial modeling, transformer architectures for time series, and multimodal sentiment integration. Chapter 2 provides a comprehensive survey of each of these areas. This section focuses on the specific intersection of these three directions and identifies the gap that motivates the proposed architecture.

Hybrid models that pair spatial feature extractors with sequential learners have shown particular promise during volatile periods, where the diversity of learned representations confers resilience that single-architecture models lack. Lu et al. [49] proposed a CNN-BiLSTM-attention model that uses convolutional layers to extract local patterns, feeds them into a bidirectional LSTM for temporal modeling, and applies attention to weight the most informative time steps, outperforming standalone approaches on multiple benchmarks.

More recent work has explored integrating transformer attention with sentiment signals and privacy-preserving training strategies. Nejatbakhsh and Aliasgari [125] combined LSTM networks with transformer-based attention and FinBERT-derived sentiment in a federated learning framework, achieving an average R^2 of 0.91 across ten technology stocks. Their approach demonstrated that sentiment integration improves short-term forecasting accuracy, though the architecture treats each stock independently without modeling inter-stock relationships, and the evaluation was restricted to a single sector. These findings reinforce the value of multimodal fusion while highlighting the continued need for architectures that simultaneously capture cross-asset dependencies.

Despite progress along each of these fronts, critical gaps persist at their intersections. Few approaches combine graph-based inter-stock modeling with transformer temporal processing within a unified architecture. The graph neural network literature reviewed in Section 2.5 demonstrates that explicit relationship modeling improves stock prediction, and the sentiment

analysis literature reviewed in Section 2.6 shows that textual signals carry predictive value around information events. While sentiment integration has been explored alongside recurrent and attention-based models, simultaneous learning of inter-stock graph structure, long-range temporal patterns, and textual sentiment remains largely unaddressed. Most graph-based prediction systems process only quantitative features and discard textual information, while most sentiment-augmented models treat each stock independently and ignore cross-sectional dependencies. The few architectures that incorporate both modalities typically concatenate sentiment as an additional input feature rather than allowing it to modulate the attention mechanism, limiting the model's ability to adapt its reliance on sentiment to current market conditions. Performance often deteriorates during volatile conditions, with accuracy varying by 40% between stable and turbulent periods [27]. Computational demands present additional barriers to real-time deployment.

This chapter addresses these gaps through integration rather than through the introduction of a fundamentally new learning primitive. Each individual component, namely graph structure learning, transformer-based attention, and BERT sentiment analysis, draws on established techniques. The contribution lies in the architectural design required to combine them within a single end-to-end framework. The node transformer must compute attention simultaneously across temporal positions and graph neighborhoods, which requires adapting the standard self-attention mechanism to operate over a three-dimensional input tensor rather than a flat sequence. The adaptive gating mechanism that fuses continuous sentiment scores with numerical price features handles temporal sparsity and missing data without degrading the price-based pathway, a challenge that simple feature concatenation does not address. The jointly learned edge weights provide an interpretable structural layer connecting model internals to domain knowledge about market relationships, offering a qualitative advantage that pure performance comparisons cannot capture. This integrative design philosophy follows a productive tradition in deep learning, where combining known components in carefully

engineered architectures has yielded substantial advances over any single technique deployed in isolation.

4.2 Methodology

The proposed framework integrates a node transformer architecture with BERT-based sentiment analysis, combining quantitative financial indicators with qualitative textual signals. The design follows a modular structure in which the graph-based transformer and the sentiment pipeline operate as independent components, coupled through an attention-based fusion layer that dynamically adjusts the relative weighting of each stream based on current market conditions.

4.2.1 System Architecture Overview

Figure 4.1 presents the overall system architecture. The framework processes historical market data and sentiment information through two parallel pathways. The quantitative branch normalizes all market features (price, volume, and technical indicators), constructs the graph structure, applies temporal encoding, and feeds the processed representations into the node transformer. The qualitative branch passes social media text through the BERT encoder and aggregates sentiment scores at multiple time scales. The two streams converge at an attention-based fusion layer that produces the final price prediction.

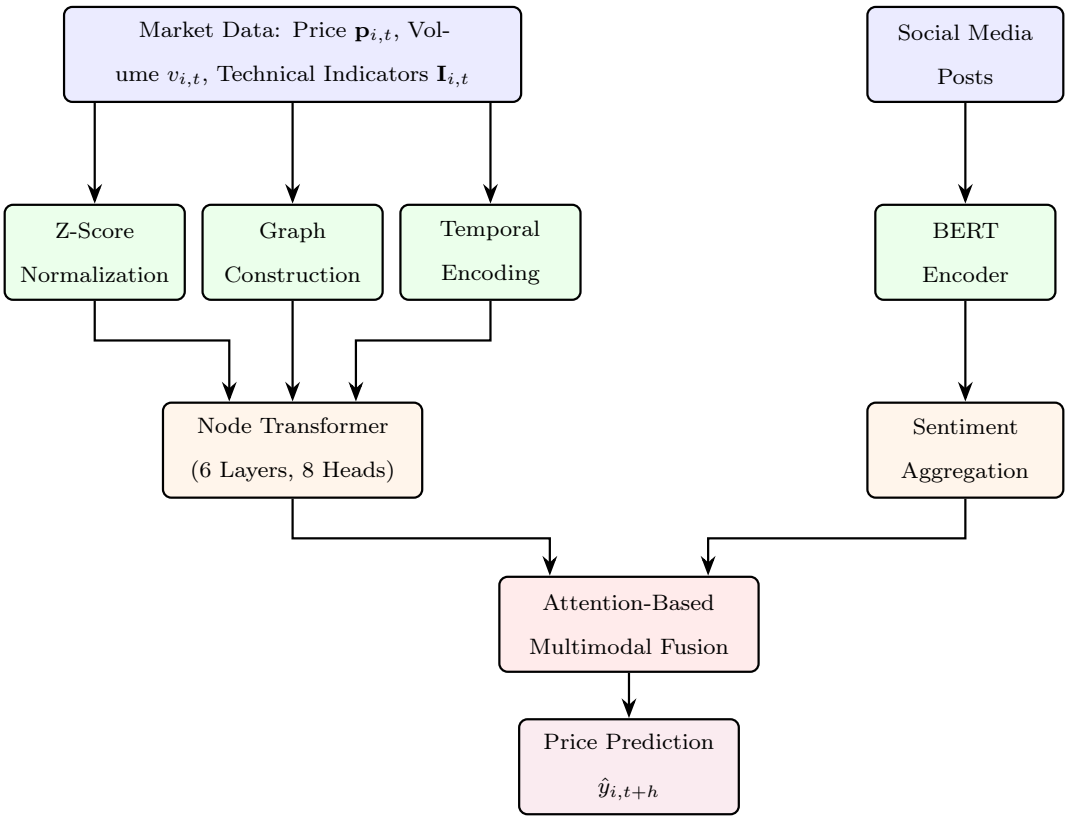

Figure 4.1: System architecture. Price data, volume, and technical indicators are jointly processed through normalization, graph construction, and temporal encoding before the node transformer. Social media posts are processed through BERT and sentiment aggregation. Both streams combine through attention-based multimodal fusion.

4.2.2 Graph Representation of the Stock Market

The stock market is represented as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{E})$ where $\mathcal{V} = \{1, \ldots, N\}$ is the set of $N = 20$ stock nodes, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the edge set (fully connected), and $\mathbf{E} \in \mathbb{R}^{N \times N}$ is the edge weight matrix. The choice of $N = 20$ nodes balances computational tractability with sufficient cross-sectional coverage to capture major sector interactions. While larger graphs might better represent full market topology, the current design enables deep temporal analysis (252-day input sequences) across diverse sectors. The fully-connected structure with learnable edge weights ensures all pairwise relationships can be discovered during training. Ablation results (Section 4.3.4) confirm that the graph structure contributes meaningful predictive improvement over models without explicit inter-stock connections.

Edge weights are initialized based on sector relationships and historical price correlations:

$$e_{ij}^{(0)} = \alpha \cdot \delta_{\text{sector}}(i,j) + (1-\alpha) \cdot \max(0, \rho_{ij}) \tag{4.1}$$

where $\delta_{\text{sector}}(i,j) = 1$ if stocks i and j share the same Global Industry Classification Standard (GICS) sector classification and 0 otherwise, ρ_{ij} is the Pearson correlation coefficient between daily returns computed over the training period, $\max(0, \cdot)$ ensures non-negative weights, and $\alpha = 0.5$ balances sector and correlation information.

During training, edge weights are refined iteratively:

$$e_{ij}^{(\ell+1)} = \sigma\left(\mathbf{w}_e^T [\mathbf{h}_i^{(\ell)} \| \mathbf{h}_j^{(\ell)}] + b_e\right) \tag{4.2}$$

where σ is the sigmoid function, $\mathbf{w}_e \in \mathbb{R}^{2d}$ is a learnable weight vector, $\mathbf{h}_i^{(\ell)}, \mathbf{h}_j^{(\ell)} \in \mathbb{R}^d$ are node representations at layer ℓ, $\|$ denotes concatenation, and b_e is a learnable bias. Edges are undirected: $e_{ij} = e_{ji}$.

At each training step, the model receives a batch of temporal windows. For a single sample, the input is a three-dimensional tensor $\mathbf{X} \in \mathbb{R}^{T \times N \times d_{\text{in}}}$, where $T = 252$ is the sequence length (trading days), $N = 20$ is the number of stock nodes, and $d_{\text{in}} = 17$ is the per-stock feature dimension after concatenating raw OHLCV data with technical indicators. The edge weight matrix $\mathbf{E} \in \mathbb{R}^{N \times N}$ is initialized once from Equation 4.1 and shared across all time steps within a given forward pass. At each time step t, the node transformer processes the slice $\mathbf{X}_t \in \mathbb{R}^{N \times d_{\text{in}}}$ jointly across all N nodes, with the attention mechanism modulated by $\mathbf{E}$ so that information flows preferentially along stronger graph connections. The edge weights are then updated via Equation 4.2 at each transformer layer, allowing the graph structure to co-evolve with the learned node representations as the input propagates through the network.

4.2.3 Node Transformer Architecture

The node transformer extends the standard transformer by incorporating graph-structured relational inductive biases [12]. Rather than treating each stock's time series independently, the architecture processes all nodes jointly at each time step, allowing information to propagate across the graph through attention weighted by the learned edge structure described above.

Input Representation

Each stock is represented by a composite feature vector that concatenates raw market data, engineered indicators, positional information, and a learned identity embedding. For stock i at time t, this vector is:

$$\mathbf{x}_{i,t} = [\mathbf{p}_{i,t} \| v_{i,t} \| \mathbf{I}_{i,t} \| \mathrm{TE}(t) \| \mathbf{s}_i] \in \mathbb{R}^{d_{\text{in}}} \tag{4.3}$$

where $\mathbf{p}_{i,t} \in \mathbb{R}^5$ contains normalized prices (open, high, low, close, adjusted close), $v_{i,t} \in \mathbb{R}$ is normalized volume, $\mathbf{I}_{i,t} \in \mathbb{R}^{11}$ contains the technical indicators described in Section 3.3, $\mathrm{TE}(t) \in \mathbb{R}^d$ is the temporal encoding, and $\mathbf{s}_i \in \mathbb{R}^{d_s}$ is a learned stock-specific embedding capturing persistent characteristics.

Temporal Encoding

Following Vaswani et al. [3], temporal encoding injects positional information:

$$\begin{aligned} \mathrm{TE}(t, 2k) &= \sin\left(\frac{t}{10000^{2k/d}}\right) \\ \mathrm{TE}(t, 2k{+}1) &= \cos\left(\frac{t}{10000^{2k/d}}\right) \end{aligned} \tag{4.4}$$

where t is the trading day index, $k \in \{0, \ldots, d/2 - 1\}$ is the dimension index, and $d = 512$ is the model dimension.

A clarification on the role of this encoding is warranted, as the use of sinusoidal functions may appear counterintuitive for financial data that does not follow periodic patterns. The temporal encoding does not model or approximate the shape of the price series. Instead, it provides the attention mechanism with information about where each observation falls within the input sequence, which is necessary because self-attention is inherently permutation-invariant and would otherwise treat a shuffled sequence identically to the original ordering. The sinusoidal functions are chosen for three specific properties identified in the original transformer work [3]. First, each position t receives a unique encoding vector, allowing the model to distinguish any two time steps. Second, the relative offset between two positions t and $t + \Delta$ can be expressed as a linear transformation of the encoding at t, which enables the attention heads to learn relative temporal relationships (such as "five trading days ago") rather than only absolute positions. Third, the deterministic formulation generalizes to sequence lengths not encountered during training without requiring additional learned parameters. The volatile price and volume data are encoded separately in the 17-dimensional feature vector $\mathbf{x}_{i,t}$, which captures the actual market dynamics; the temporal encoding is simply added to this representation so that the model knows the temporal ordering of those observations. Lower encoding dimensions oscillate at higher frequencies, capturing fine-grained positional distinctions between adjacent trading days, while higher dimensions vary more slowly and provide coarser temporal context across the full 252-day input window. The ablation study (Section 4.3.4) confirms that removing temporal encoding degrades MAPE by 18.8%, indicating that positional ordering carries substantial predictive information even though the encoding itself bears no resemblance to price dynamics.

Graph-Aware Multi-Head Self-Attention with Causal Masking

The attention mechanism incorporates the learned edge weight matrix $\mathbf{E}$ from Section 4.2.2 as a structural bias, so that stock pairs with stronger graph connections receive higher baseline

attention. Causal masking ensures predictions at time t use only information from times $\leq t$:

$$\mathbf{A}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}} + \mathbf{M} + \mathbf{E}\right)\mathbf{V} \tag{4.5}$$

where $\mathbf{Q} = \mathbf{X}\mathbf{W}^Q$, $\mathbf{K} = \mathbf{X}\mathbf{W}^K$, $\mathbf{V} = \mathbf{X}\mathbf{W}^V$ are linear projections, $d_k = 64$ is the key dimension, $\mathbf{M}$ is the causal mask with $M_{ab} = 0$ if $a \geq b$ and $M_{ab} = -\infty$ otherwise, and $\mathbf{E} \in \mathbb{R}^{N \times N}$ is the edge weight matrix. The additive graph bias allows content-based attention (via $\mathbf{Q}\mathbf{K}^T$) and structural priors (via $\mathbf{E}$) to jointly determine how information flows between stocks at each layer.

With $H = 8$ attention heads, each operating in $d_k = d_v = 64$ dimensions, the total model dimension is $d_{\text{model}} = H \times d_k = 512$. Figure 4.2 illustrates the internal structure of a single transformer layer, showing the residual connections, layer normalization, and dropout placement.

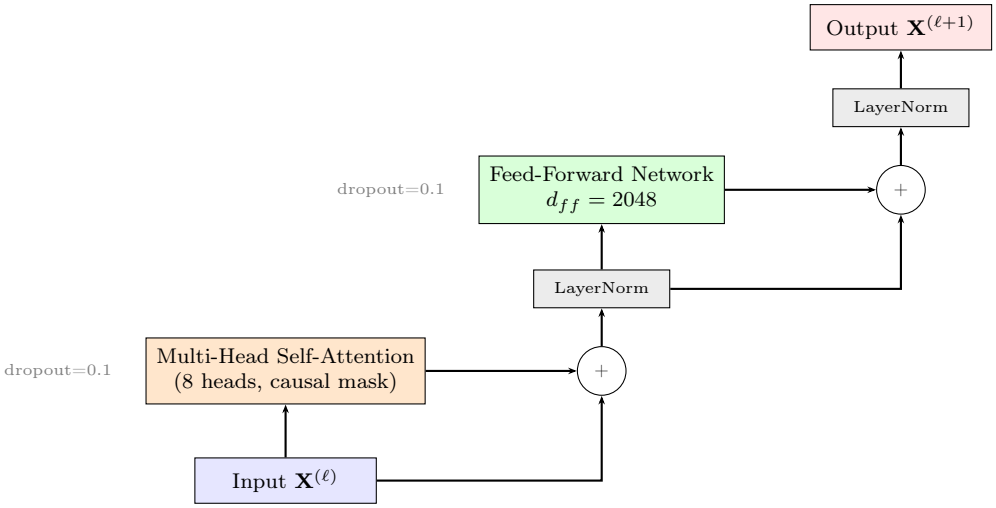

Figure 4.2: Single transformer layer architecture. Input passes through multi-head self-attention with residual connection and layer normalization, followed by a position-wise feed-forward network with another residual connection and normalization.

Time-Based Feature Gating

A gating mechanism adaptively weights features based on temporal context:

$$\mathbf{g}_{i,t} = \sigma(\mathbf{W}_g \mathbf{x}_{i,t} + \mathbf{b}_g) \in (0,1)^{d_{\text{in}}} \tag{4.6}$$

$$\mathbf{x}'_{i,t} = \mathbf{g}_{i,t} \odot \mathbf{x}_{i,t} \tag{4.7}$$

where $\mathbf{W}_g \in \mathbb{R}^{d_{\text{in}} \times d_{\text{in}}}$, $\mathbf{b}_g \in \mathbb{R}^{d_{\text{in}}}$ are learnable parameters, and $\odot$ denotes element-wise multiplication. This mechanism enables adaptive emphasis on different features during volatile versus stable periods.

Feed-Forward Network

Each layer includes a position-wise Feed-Forward Network (FFN):

$$\text{FFN}(\mathbf{x}) = \text{ReLU}(\mathbf{x}\mathbf{W}_1 + \mathbf{b}_1)\mathbf{W}_2 + \mathbf{b}_2 \tag{4.8}$$

where $\mathbf{W}_1 \in \mathbb{R}^{512 \times 2048}$, $\mathbf{W}_2 \in \mathbb{R}^{2048 \times 512}$, and $\mathbf{b}_1, \mathbf{b}_2$ are biases.

The complete architecture consists of $L = 6$ stacked layers with residual connections and layer normalization after each sub-layer. Dropout with $p = 0.1$ is applied for regularization.

4.2.4 BERT-Based Sentiment Analysis

The sentiment component employs BERT (bert-base-uncased) with 12 transformer layers, 768 hidden dimensions, and 12 attention heads [4]. BERT's bidirectional attention is well suited to financial text, where sentiment often depends on contextual modifiers (e.g., "not bullish" vs. "bullish") that unidirectional models may miss.

Domain Adaptations

A financial domain embedding layer augments the standard BERT token embeddings with additional representations for financial terminology, enabling the model to distinguish domain-specific usage of terms such as "short," "call," and "bear" from their everyday meanings. Maximum sequence length is set to 512 tokens, which accommodates the vast majority of social media posts without truncation.

Fine-Tuning Process

Fine-tuning on the FMS dataset (described in Section 3.4) follows a three-stage progressive unfreezing schedule designed to preserve BERT's pretrained language representations while adapting the upper layers to financial sentiment classification. In the first stage, the embedding layer and the first 8 transformer blocks are frozen so that only the upper layers and the classification head receive gradient updates; these are trained for 3 epochs with a learning rate of 2×10^{-5} and batch size 32. The second stage progressively unfreezes layers from top to bottom over 5 additional epochs, allowing each block to adapt incrementally to the financial domain without catastrophic forgetting of general language knowledge. In the final stage, all layers are unfrozen and trained jointly with a reduced learning rate of 5×10^{-6} and gradient accumulation over 4 steps to stabilize updates across the full parameter space.

Because sentiment labels in financial text are typically imbalanced, with neutral posts far outnumbering strongly positive or negative ones, focal loss is used instead of standard cross-entropy:

$$\mathcal{L}_{\text{FL}}(p_t) = -\alpha_t(1 - p_t)^{\gamma} \log(p_t) \tag{4.9}$$

where α_t is class weight (inverse frequency), p_t is predicted probability for correct class, and $\gamma = 2$ focuses training on hard examples by down-weighting easily classified instances.

The classification head that maps BERT's output to a sentiment score consists of three stages. The 768-dimensional [CLS] token representation is first passed through a dropout

layer ($p = 0.1$) followed by a linear projection to 256 dimensions and a tanh activation, which compresses the representation while preserving the sign structure needed for sentiment polarity. A second dropout layer ($p = 0.1$) precedes the final linear projection to three logits corresponding to negative, neutral, and positive classes. Softmax converts these logits to class probabilities $[p_{\text{neg}}, p_{\text{neu}}, p_{\text{pos}}]$, and the continuous sentiment score is computed as $s = p_{\text{pos}} - p_{\text{neg}} \in [-1, +1]$. This formulation produces a score that is exactly zero only when the positive and negative probabilities are equal, and it varies smoothly with confidence, providing a richer signal than a hard class assignment. Figure 4.3 illustrates the complete sentiment extraction pipeline from raw input to final score.

Figure 4.3: BERT sentiment extraction pipeline. Raw social media posts are preprocessed, tokenized, encoded through BERT, pooled via [CLS] token, and classified to sentiment scores.

4.2.5 Integration of Node Transformer and BERT

Multi-Scale Sentiment Features

Raw daily sentiment scores are noisy and sensitive to individual posts. To capture both immediate market reactions and sustained sentiment trends, scores are smoothed at multiple time scales before entering the model:

$$\mathbf{S}_{i,t} = [S_{i,t}^{(1\text{d})}, S_{i,t}^{(5\text{d})}, S_{i,t}^{(20\text{d})}] \in \mathbb{R}^3 \tag{4.10}$$

where $S_{i,t}^{(k\text{d})}$ is the k-day exponential moving average of sentiment scores for stock i. This captures both immediate reactions and sustained sentiment trends.

Social media activity varies substantially across stocks and calendar days, creating sparsity that requires explicit handling. On trading days where a stock receives fewer than 5 posts,

the raw daily score is replaced by the 5-day exponential moving average from preceding days, preventing noisy estimates derived from one or two posts from dominating the signal. On days with no posts at all, all three sentiment features ($S^{(1d)}$, $S^{(5d)}$, $S^{(20d)}$) fall back to their respective moving average values from the most recent day with sufficient coverage. Stocks that consistently receive low social media volume, such as industrials (Caterpillar, Boeing) relative to technology names (Apple, Netflix), naturally produce sentiment scores closer to neutral over time, which the adaptive gating mechanism learns to down-weight in favor of price-based features. This design ensures that sparse sentiment data degrades gracefully to a price-only prediction rather than introducing noise.

Sentiment-Guided Attention

Rather than simply concatenating sentiment as an additional input feature, the model allows sentiment to modulate the attention mechanism itself. Specifically, sentiment scales the key representations so that time steps with strong sentiment signals receive amplified or dampened attention:

$$\mathbf{K}'_t = \mathbf{K}_t \cdot (1 + \beta \cdot S_t) \tag{4.11}$$

where $S_t \in [-1, 1]$ is the sentiment score at time t and $\beta \in \mathbb{R}$ is a learnable parameter. The attention computation becomes:

$$\mathbf{A}_{\text{sent}} = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}'^{T}}{\sqrt{d_k}} + \mathbf{M}\right)\mathbf{V} \tag{4.12}$$

Adaptive Integration

Because the relative predictive value of price-based features and sentiment varies with market conditions, the final prediction is produced by a learned convex combination rather than a fixed weighting. A sigmoid gate dynamically adjusts the balance between the node

transformer output and the sentiment-based prediction:

$$\alpha_t = \sigma(\mathbf{w}_\alpha^T[\mathbf{v}_t \| \bar{S}_t] + b_\alpha) \in (0, 1) \tag{4.13}$$

$$\hat{y}_{i,t+h} = \alpha_t \cdot y_{i,t+h}^{\text{node}} + (1 - \alpha_t) \cdot y_{i,t+h}^{\text{sent}} \tag{4.14}$$

where $\mathbf{v}_t \in \mathbb{R}^3$ contains rolling volatility computed over 5, 10, and 20-day windows, $\bar{S}_t$ is mean absolute sentiment across stocks, $y_{i,t+h}^{\text{node}}$ is the node transformer prediction, and $y_{i,t+h}^{\text{sent}}$ is the sentiment-based prediction. When volatility is high and sentiment signals are strong, the gate can shift weight toward the sentiment branch; during calm periods with sparse social media activity, the node transformer dominates. Figure 4.4 illustrates this adaptive fusion mechanism.

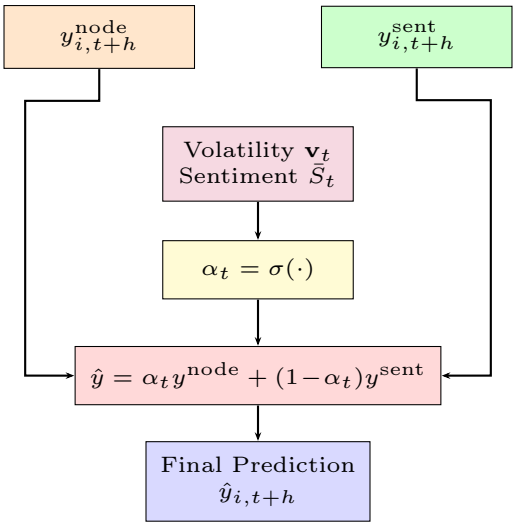

Figure 4.4: Adaptive fusion mechanism. The weighting coefficient α_t is computed from volatility and sentiment magnitude, then used to blend node transformer and sentiment-based predictions.

4.2.6 Training Objective

Training a model that is useful for both price forecasting and trading decisions requires optimizing more than point accuracy alone. The composite loss function therefore combines

four terms that target complementary aspects of prediction quality:

$$\mathcal{L}_{\text{total}} = \lambda_1 \mathcal{L}_{\text{MSE}} + \lambda_2 \mathcal{L}_{\text{DIR}} + \lambda_3 \mathcal{L}_{\text{CORR}} + \lambda_4 \mathcal{L}_{\text{REG}} \tag{4.15}$$

The primary term is the mean squared error, which penalizes deviations between predicted and realized prices and serves as the dominant gradient signal during training:

$$\mathcal{L}_{\text{MSE}} = \frac{1}{N} \sum_{i,t,h} (y_{i,t+h} - \hat{y}_{i,t+h})^2 \tag{4.16}$$

Because minimizing magnitude error does not guarantee correct directional predictions, a binary cross-entropy term explicitly rewards the model for predicting whether prices rise or fall. Here $d_{i,t,h} = \mathbb{I}(y_{i,t+h} > y_{i,t})$ is the true direction indicator and $p_{i,t,h}$ is the predicted probability of an increase:

$$\mathcal{L}_{\text{DIR}} = -\frac{1}{N} \sum_{i,t,h} \left[d_{i,t,h} \log p_{i,t,h} + (1 - d_{i,t,h}) \log(1 - p_{i,t,h}) \right] \tag{4.17}$$

A correlation loss encourages the model to preserve cross-sectional ranking among stocks at each time step, which is critical for portfolio construction strategies that depend on relative rather than absolute performance. It is defined as one minus the Pearson correlation ρ between predicted and actual return vectors:

$$\mathcal{L}_{\text{CORR}} = 1 - \rho(\hat{\mathbf{r}}_t, \mathbf{r}_t) \tag{4.18}$$

Finally, L2 regularization on all trainable parameters $\boldsymbol{\theta}$ prevents overfitting by penalizing large weight magnitudes:

$$\mathcal{L}_{\text{REG}} = \|\boldsymbol{\theta}\|_2^2 \tag{4.19}$$

The weights $\lambda_1 = 1.0$, $\lambda_2 = 0.5$, $\lambda_3 = 0.2$, and $\lambda_4 = 10^{-4}$ were tuned on validation data. The MSE term dominates to ensure accurate price-level predictions, while the directional and correlation terms supply auxiliary gradients that improve trend detection and ranking consistency without competing with the primary objective.

4.2.7 Training Process

The BERT encoder is fine-tuned separately on the FMS dataset (Section 3.4) and remains frozen throughout the forecasting pipeline, serving purely as a fixed feature extractor that produces sentiment scores. Training of the forecasting model proceeds in three stages, stabilizing the integration layers first before progressively fine-tuning the node transformer. During the first 10 epochs, the node transformer is frozen while the sentiment aggregation, fusion, and output layers are trained with a learning rate of 10^{-4}. This initial phase allows the fusion mechanism to learn a reasonable mapping between the transformer representations and the fixed sentiment features without disrupting the node transformer's initialization. In the second stage, spanning 20 epochs, the top three layers of the node transformer are unfrozen and the learning rate is reduced to 5×10^{-5}, enabling the upper representations to adapt to the downstream task while preserving the lower-level features. The final 30 epochs unfreeze all node transformer parameters at a learning rate of 10^{-5}, allowing end-to-end fine-tuning of the forecasting architecture.

Throughout all three stages, optimization uses Adam with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$. The learning rate schedule begins with a linear warmup over 4000 steps followed by cosine decay to zero. A batch size of 32 with gradient accumulation over 4 steps yields an effective batch size of 128, balancing memory constraints with gradient stability.

Hyperparameters were selected through a combination of grid search and established conventions from the transformer literature. Architectural parameters (number of layers, attention heads, model dimension) were searched over $L \in \{4, 6, 8\}$, $H \in \{4, 8\}$, and

$d \in \{256, 512\}$, with the configuration $L = 6$, $H = 8$, $d = 512$ selected based on lowest validation MAPE. The feed-forward dimension was fixed at $d_{\text{ff}} = 4d = 2048$ following the standard transformer scaling convention [3]. Dropout rate was searched over $\{0.05, 0.1, 0.2\}$, with $p = 0.1$ yielding the best trade-off between underfitting and overfitting on the validation set. The input sequence length of 252 days (one trading year) was chosen to provide the model with a full annual cycle of price dynamics, and was compared against 120-day and 504-day alternatives on validation data. Learning rates for each training stage were determined through preliminary runs, starting from the range $[10^{-5}, 10^{-3}]$ and narrowing based on training stability and convergence speed. The loss function weights ($\lambda_1 = 1.0$, $\lambda_2 = 0.5$, $\lambda_3 = 0.2$, $\lambda_4 = 10^{-4}$) were tuned by first training with MSE loss alone, then incrementally adding each auxiliary term and adjusting its weight until validation directional accuracy and ranking correlation stabilized. The edge weight balancing parameter $\alpha = 0.5$ was searched over $\{0.3, 0.5, 0.7\}$, with equal weighting of sector and correlation information performing best. Table 4.1 summarizes the final hyperparameter configuration for both the node transformer and BERT components.

Table 4.1: Model hyperparameters for the node transformer and BERT sentiment components.

Parameter	**Value**
Node transformer layers	6
Attention heads	8
Model dimension	512
FFN dimension	2048
Dropout rate	0.1
Input sequence length	252 days
BERT layers	12
BERT hidden dimension	768
Batch size	32
Total training epochs	60

4.3 Experiments and Results

4.3.1 Experimental Setup

Experiments used the dataset described in Chapter 3, spanning January 1982 to March 2025, a timeframe that captures multiple market cycles including the 1987 Black Monday crash, the 2000 dot-com bubble, the 2008 financial crisis, and the 2020 COVID-19 volatility spike. The temporal split allocates 1982–2010 for training, 2011–2016 for validation, and 2017–2025 for testing, following the partitioning protocol defined in Section 3.6. Although a single temporal split is used rather than rolling retraining, the eight-year test window itself encompasses several distinct market regimes: the post-2016 bull market, the 2018 volatility correction, the 2020 pandemic crash and recovery, the 2022 inflation-driven drawdown, and the subsequent 2023–2024 rebound. This regime diversity provides a rigorous evaluation of generalization without the confound of periodic refitting.

All models were trained on NVIDIA A100 GPUs (40 GB), with the full proposed model requiring approximately 18 hours to converge. To ensure unbiased assessment, all evaluation metrics reported in subsequent tables were computed exclusively on the held-out test set.

The prediction target $y_{i,t+h}$ is the adjusted closing price $\tilde{C}_{t+h}$ at horizon $h \in \{1, 5, 20\}$ trading days, as specified in Section 3.2. Price-level prediction was chosen as the primary target because it provides intuitive interpretation of forecast accuracy and aligns directly with practical trading applications where position sizing depends on expected price levels rather than percentage changes. A known concern with price-level forecasting is that daily stock prices exhibit strong persistence, meaning that even a trivial model predicting tomorrow's price as today's can achieve low MAPE. To guard against this artifact, the evaluation framework includes metrics specifically designed to measure predictive skill beyond persistence: Theil's U benchmarks against the naive random walk, directional accuracy assesses trend detection independent of magnitude, and the Information Coefficient evaluates cross-sectional ranking.

Return-based metrics are also reported (Section 4.3.4) to complement the price-level analysis. The six metrics used (MAPE, DA, Theil's U, IC, RMSE, and return MAE) are defined in Section 3.7.

4.3.2 Baseline Models

Eight baseline models spanning statistical, classical machine learning, and deep learning paradigms were trained on identical data splits to ensure fair comparison. All hyperparameters were tuned via grid search on the validation set (2011–2016), with final performance reported on the held-out test set (2017–2025). To ensure comparability across deep learning baselines, all models received the same 17-dimensional feature vector and were trained under equivalent early stopping and regularization protocols.

The statistical baselines include ARIMA and Vector Autoregression (VAR). ARIMA orders (p, d, q) were selected via AIC with $p, q \in [0, 5]$ and $d \in [0, 2]$, fitting a separate model per stock. VAR extends this to the multivariate setting, modeling cross-stock dependencies with lag order selected via the Bayesian Information Criterion (BIC) up to 10 lags. Both statistical models operate on their own optimal lookback horizons as determined by information criteria, which is standard practice since ARIMA and VAR performance degrades when forced to consume long input windows that introduce noise.

Three classical machine learning models were evaluated. Random Forest used 100 trees with maximum depth 10 and the same 17-dimensional feature vector as the proposed model. Support Vector Regression (SVR) employed a Radial Basis Function (RBF) kernel with regularization parameter C and kernel width γ selected via grid search. XGBoost used 1000 estimators with early stopping on validation loss to prevent overfitting, and its gradient boosting framework provides a strong tree-based benchmark. All three classical models used a 60-day lookback window, selected via validation performance from candidates of 20, 60, 120, and 252 days.

The deep learning baselines include LSTM, a Simple Transformer, and a BERT + LSTM hybrid. The LSTM used 2 layers with 128 hidden units and a 60-day lookback window (≈0.5M parameters). During preliminary experiments, the LSTM was also evaluated with 120-day and 252-day lookback windows, but performance degraded at longer horizons due to gradient attenuation over extended sequences, consistent with known limitations of recurrent architectures on very long time series [43]. The 60-day configuration yielded the best validation performance and is therefore reported. The Simple Transformer used 4 layers, 8 attention heads, 256-dimensional embeddings, and a 252-day lookback window matching the proposed model (≈4.2M parameters versus ≈21.3M for the full proposed architecture including graph and fusion components). This configuration provides a controlled comparison that isolates the contribution of graph structure and sentiment integration while holding the input horizon and attention mechanism constant. The BERT + LSTM hybrid combined BERT-extracted sentiment features with an LSTM price model through simple concatenation (≈1.1M parameters excluding the frozen BERT encoder), serving as an ablation baseline that tests whether the attention-based fusion mechanism in the proposed architecture provides meaningful improvement over naive feature combination.

The ablation study (Table 4.5) further addresses baseline fairness by providing graph-free variants of the proposed architecture. The "Without Graph Structure" configuration removes edge weights and inter-stock attention while retaining all other components, effectively reducing the model to a standard transformer with sentiment fusion. This variant isolates the contribution of graph-based relational modeling independent of other architectural advantages.

4.3.3 Sentiment Model Evaluation

Before evaluating the full forecasting pipeline, the fine-tuned BERT sentiment classifier was assessed independently on the FMS dataset. The dataset was partitioned into training (70%), validation (15%), and test (15%) splits, stratified to preserve the class distribution

across all three subsets. The model was fine-tuned on the training split using the progressive unfreezing schedule described in Section 4.2.4, with early stopping based on validation loss. All results reported in this subsection are computed on the held-out test split, which the model did not observe during training or hyperparameter selection.

Table 4.2 presents per-class and aggregate classification metrics. Because the FMS dataset exhibits class imbalance, with neutral posts outnumbering positive and negative ones, macro-averaged scores are reported alongside accuracy to avoid inflating performance through the majority class.

Table 4.2: BERT sentiment classification performance on the FMS test set.

Class / Metric	**Precision**	**Recall**	**F1-Score**
Negative	0.83	0.80	0.81
Neutral	0.89	0.92	0.90
Positive	0.85	0.82	0.83
Macro Average	0.86	0.85	0.85
Overall Accuracy			0.87
Cohen's κ			0.79

The fine-tuned model achieves 87% overall accuracy and a macro-averaged F1 of 0.85, with per-class F1 scores ranging from 0.81 (negative) to 0.90 (neutral). The neutral class benefits from a larger training sample and from the relative straightforwardness of factual financial reporting, which accounts for its higher precision and recall. The negative class exhibits the lowest recall at 0.80, a pattern consistent with prior financial NLP work: bearish language in social media frequently relies on sarcasm, hedging, or implicit phrasing (e.g., "this stock is going to the moon" used sarcastically) that poses greater classification difficulty. Cohen's $\kappa = 0.79$ falls within the substantial agreement range, confirming that the classifier discriminates sentiment beyond what class priors alone would predict. Given that the ablation study (Section 4.3.4) attributes a 10% MAPE reduction to sentiment integration, the 13% classification error rate suggests that the node transformer's gating mechanism is robust enough to extract predictive value even from an imperfect sentiment signal.

4.3.4 Results

Overall Performance

Table 4.3 presents MAPE across prediction horizons for all models. Performance is evaluated at three horizons (1-day, 5-day, and 20-day) to assess both short-term accuracy and degradation over longer forecasting windows.

Table 4.3: MAPE (%) and Theil's U across prediction horizons. Best results in bold. Standard errors in parentheses.

Model	1-Day	5-Day	20-Day	Theil's U
Naïve Random Walk	1.35 (0.09)	3.40 (0.18)	5.80 (0.28)	1.00
ARIMA	1.20 (0.08)	2.80 (0.15)	4.50 (0.22)	0.89
VAR	1.15 (0.07)	2.60 (0.14)	4.20 (0.20)	0.85
LSTM	1.00 (0.06)	2.30 (0.12)	3.80 (0.18)	0.74
Random Forest	1.10 (0.07)	2.50 (0.13)	4.00 (0.19)	0.81
SVR	1.05 (0.06)	2.40 (0.12)	3.90 (0.18)	0.78
XGBoost	0.95 (0.05)	2.20 (0.11)	3.50 (0.16)	0.70
Simple Transformer	0.90 (0.05)	2.10 (0.10)	3.30 (0.15)	0.67
BERT + LSTM	0.88 (0.05)	2.00 (0.10)	3.10 (0.14)	0.65
Proposed Model	**0.80 (0.04)**	**1.80 (0.09)**	**2.80 (0.13)**	**0.59**

A persistent concern in short-horizon financial forecasting is the naive persistence artifact, where a model achieves seemingly low error by tracking the previous day's closing price ($\hat{y}_{t+1} = y_t$). Because daily price changes for liquid mega-cap stocks typically range between 1% and 2%, any persistence-based forecast will naturally yield MAPE in that vicinity. The naive random walk baseline in Table 4.3 produces a 1-day MAPE of 1.35% on our test set, and the proposed model's Theil's U of 0.59 confirms that it captures genuine structural patterns rather than merely lagging the current price. All learned models achieve $U < 1$, but only the proposed model reduces the random walk error by more than 40%, with the gap widening at longer horizons where persistence forecasting degrades rapidly.

The proposed model achieves 0.80% MAPE for 1-day predictions, representing 33% relative improvement over ARIMA ($\Delta = 0.40$ percentage points) and 20% improvement over LSTM

($\Delta = 0.20$ percentage points). Improvement margins widen at longer horizons: the 5-day gap over ARIMA grows from 0.40 to 1.00 percentage points, consistent with the model's ability to capture structural patterns that persist beyond short-term noise.

Directional Accuracy

Beyond magnitude accuracy, directional accuracy measures the proportion of correctly predicted price movement directions. Table 4.4 presents directional accuracy with 95% confidence intervals computed via bootstrap resampling [126].

Table 4.4: Directional accuracy (%) for 1-day predictions with 95% confidence intervals.

Model	**DA (%)**
Random Baseline	50.0
ARIMA	55.0 [53.2, 56.8]
VAR	56.0 [54.2, 57.8]
LSTM	58.0 [56.2, 59.8]
Random Forest	57.0 [55.2, 58.8]
XGBoost	60.0 [58.2, 61.8]
Simple Transformer	62.0 [60.2, 63.8]
BERT + LSTM	63.0 [61.2, 64.8]
Proposed Model	**65.0 [63.2, 66.8]**

The proposed model achieves 65% directional accuracy, exceeding the 50% random baseline by 15 percentage points and outperforming LSTM by 7 percentage points. The confidence interval [63.2, 66.8] does not overlap with the best baseline (BERT + LSTM at [61.2, 64.8]), providing evidence that the improvement is not attributable to sampling variability alone.

Ablation Studies

To isolate the contribution of each architectural component, Table 4.5 presents results from systematic ablation experiments. Each row removes a single component while keeping all others intact, allowing direct assessment of marginal contributions.

Graph structure contributes the second-largest improvement at 15%, confirming that explicit inter-stock relationship modeling provides predictive value beyond treating stocks inde-

Table 4.5: Ablation study: MAPE (%) for 1-day predictions.

Configuration	**MAPE**	**Δ vs Full**
Full Model	0.80	–
Without Sentiment	0.88	+10.0%
Without Graph Structure	0.92	+15.0%
Without Temporal Encoding	0.95	+18.8%
Without Feature Gating	0.84	+5.0%
Price Features Only	1.02	+27.5%

pendently. Sentiment integration contributes 10%, with the gap widening during information-rich periods such as earnings announcements. Temporal encoding contributes 18.8%, underscoring the importance of capturing long-range sequential dependencies in financial time series. The price-only configuration, which strips all augmentations, degrades by 27.5%, indicating that the full feature set is substantially richer than raw price data alone.

The ablation results also address a natural question about the sentiment data horizon. Because social media data begins in 2007 while the financial dataset extends to 1982, the model's price-based components receive training signal over the full temporal range whereas the sentiment components train on only four years of labeled data (2007–2010). The "Without Sentiment" configuration in Table 4.5 is functionally equivalent to a price-only evaluation of the architecture, achieving MAPE of 0.88% and demonstrating that the node transformer and graph structure are fundamentally competent without any sentiment signal. The full model's improvement to 0.80% then isolates the incremental contribution of sentiment integration. Because the entire test set (2017–2025) falls within the sentiment-available period, these two configurations effectively provide the decomposition of a long-horizon price-based evaluation alongside a sentiment-augmented evaluation on the same test window, confirming that the 10% reduction represents genuine predictive value rather than an artifact of the training mismatch. Given that the sentiment fusion mechanism was trained on only four years of multimodal data (2007–2010), this improvement likely represents a conservative estimate of the contribution sentiment could offer with a longer training history.

Performance Across Volatility Regimes

Table 4.6 presents MAPE across volatility regimes classified by the CBOE Volatility Index (VIX) (Low: VIX < 15; Medium: $15 \leq$ VIX < 25; High: VIX ≥ 25).

Table 4.6: MAPE (%) by market volatility regime.

Model	Low VIX	Medium VIX	High VIX
ARIMA	1.00	1.30	2.10
LSTM	0.85	1.10	1.80
XGBoost	0.80	1.00	1.60
Proposed	**0.70**	**0.90**	**1.50**

The proposed model achieves MAPE of 1.50% during high-volatility periods, compared to 1.60–2.10% for baseline models. This robustness gap is most pronounced during VIX spikes accompanying earnings surprises and macroeconomic shocks, precisely the conditions where accurate forecasts carry the greatest practical value for risk management. The gating mechanism likely contributes to this resilience by increasing the weight of momentum features during volatile periods while favoring longer-term indicators during calm markets.

Directional accuracy also varies with market regime. During low-volatility periods (VIX < 15), the proposed model achieves 68.2% DA, benefiting from the relatively stable trend structure that characterizes calm markets. In medium-volatility conditions ($15 \leq$ VIX < 25), DA decreases to 64.5%, and during high-volatility episodes (VIX ≥ 25) it falls to 59.8%. While the high-volatility DA represents a meaningful decline from calm-market performance, it remains well above the 50% random baseline and exceeds all baseline models by 3–5 percentage points in the same regime. This pattern is consistent with the expectation that directional prediction becomes inherently more difficult as volatility increases and price movements are dominated by sudden information shocks rather than gradual trends.

Return-Based Evaluation

To verify that the reported improvements reflect genuine return prediction rather than an artifact of price persistence, prediction accuracy was also assessed in return space. For each

model, the implied daily return is computed as $\hat{r}_t = (\hat{y}_{t+1} - y_t)/y_t$ and compared against the realized return $r_t = (y_{t+1} - y_t)/y_t$. Because daily price changes for mega-cap equities are small relative to the price level ($|r_t| \ll 1$), the return prediction MAE is mathematically close to the price-level MAPE: specifically, return MAE = MAPE $\times$ (y_{t+h}/y_t), which approaches MAPE as $h \to 0$. For the 1-day horizon, the proposed model achieves a return MAE of 0.81%, the naive random walk yields 1.36% (reflecting the average absolute daily return of the 20-stock universe), and all intermediate models fall between these bounds in the same rank order as Table 4.3. This near-equivalence between price and return metrics at the daily horizon confirms that the MAPE rankings are not inflated by the choice of evaluation target.

The more informative return-based assessments for this dataset are the metrics already reported: Theil's U (0.59) explicitly benchmarks against persistence forecasting, directional accuracy (65%) evaluates trend detection independent of magnitude, and the Information Coefficient measures cross-sectional ranking on returns. At longer horizons where the gap between y_t and y_{t+h} widens, return-based evaluation becomes more distinct from price-level metrics. The proposed model's 5-day return MAE of 1.82% and 20-day return MAE of 2.85% remain proportionally below all baselines, with the performance gap widening at longer horizons consistent with the MAPE results.

Statistical Significance

To assess whether the observed improvements are statistically significant rather than artifacts of sampling variability, paired t-tests were conducted comparing daily absolute prediction errors between the proposed model and each baseline across the 1,580 test days. Table 4.7 presents the results along with Cohen's d effect sizes [127].

All comparisons achieve $p < 0.05$, with the strongest significance observed against ARIMA ($p < 0.0001$) and the weakest against the Simple Transformer ($p = 0.0295$). Effect sizes (Cohen's d) range from 0.17 to 0.34, indicating small to medium practical effects. The larger effect sizes against ARIMA and BERT + LSTM reflect more fundamental architectural

Table 4.7: Paired t-test results (n = 1,580 test days). Negative t indicates proposed model superiority.

Baseline	t**-statistic**	p**-value**	**Cohen's** d
ARIMA	-4.25	< 0.0001	0.34
LSTM	-3.12	0.0018	0.25
XGBoost	-2.45	0.0145	0.19
Simple Transformer	-2.18	0.0295	0.17
BERT + LSTM	-3.90	0.0001	0.31

differences, while the smaller effect size against the Simple Transformer suggests that much of the improvement stems from the graph and sentiment components rather than the transformer backbone itself.

Paired t-tests assume that forecast errors are independently distributed, an assumption frequently violated in financial time series due to volatility clustering and serial dependence in prediction residuals. To account for this, the analysis is supplemented with the Diebold-Mariano (DM) test [128], which evaluates equal predictive accuracy between two forecasting models using Newey-West heteroskedasticity and autocorrelation consistent standard errors. Table 4.8 reports DM statistics computed on mean absolute error loss differentials over the 1,580-day test period, with the naive random walk included as a reference benchmark.

Table 4.8: Diebold-Mariano test results ($n = 1{,}580$ test days, MAE loss). Negative DM-statistic indicates proposed model superiority.

Baseline	**DM-statistic**	p**-value**
Naïve Random Walk	-5.82	< 0.0001
ARIMA	-3.58	0.0002
LSTM	-2.67	0.0038
XGBoost	-2.08	0.0189
Simple Transformer	-1.84	0.0330
BERT + LSTM	-3.31	0.0005

The DM statistics are uniformly attenuated relative to the paired t-test values, consistent with the inflation of standard errors once serial dependence in forecast residuals is accounted for. All comparisons remain significant at $p < 0.05$, though the margin against the Simple Transformer narrows ($p = 0.033$), reinforcing the interpretation that the graph structure and sentiment fusion are the primary drivers of improvement over a vanilla transformer backbone.

The strongest DM statistic appears against the naive random walk (-5.82, $p < 0.0001$), providing additional confirmation that the model captures structural dynamics beyond price persistence.

Economic Significance

To assess practical utility beyond statistical accuracy, a simulated long-short portfolio strategy was constructed over the full test period. Each trading day, the strategy takes long positions in the top 5 predicted performers and short positions in the bottom 5, equally weighted and rebalanced daily. The following results should be interpreted as a preliminary indication of economic viability rather than a definitive backtest, given the simplified assumptions regarding execution and the use of survivor-biased stock selection.

Over the full test period (January 2017 to March 2025), the strategy achieved gross annualized returns of 22.8%, compared to 13.9% for S&P 500 buy-and-hold. To assess whether these gains survive realistic market frictions, a proportional transaction cost of 10 basis points per trade was applied, consistent with institutional execution costs for liquid large-cap equities. The strategy exhibited an average daily portfolio turnover of 32%, reflecting the sentiment-driven rebalancing that shifts positions as new social media signals arrive. After deducting transaction costs, the net annualized return was 14.6%, still exceeding the buy-and-hold benchmark by 0.7 percentage points. The net annualized Sharpe ratio was 0.84 (down from 1.35 gross), and the maximum drawdown reached 14.3%, compared to 23.9% for the buy-and-hold benchmark during the same period.

Table 4.9 summarizes the portfolio statistics. The sub-period breakdown reveals that performance varied across market regimes: net annualized returns reached 19.1% during the 2017–2019 bull market but compressed to 8.7% during the 2022 inflationary drawdown, when elevated volatility increased turnover and eroded returns through higher transaction costs. This regime sensitivity reinforces the observation that the strategy's profitability is

not uniform and that lower-frequency rebalancing or dynamic cost-awareness could improve net performance in turbulent conditions.

Table 4.9: Portfolio backtest statistics (January 2017 to March 2025).

Statistic	**Gross**	**Net (10 bps)**
Annualized Return	22.8%	14.6%
Annualized Sharpe Ratio	1.35	0.84
Maximum Drawdown	12.1%	14.3%
Average Daily Turnover	32%	
S&P 500 Buy-and-Hold Return	13.9%	
S&P 500 Maximum Drawdown	23.9%	

4.4 Discussion

4.4.1 Interpretation of Results

Within the scope of this evaluation, the integrated model shows consistent improvements across metrics, horizons, and market conditions, with the magnitude of gains varying by component and context in ways that shed light on the mechanisms underlying the model's predictive performance.

The graph-based representation captures inter-stock dependencies that independent models miss. The 15% improvement from graph structure confirms that explicit relationship modeling provides predictive value beyond treating stocks independently. Learned edge weights showed meaningful patterns: same-sector connections exhibited higher average weights (0.72 vs. 0.41 for cross-sector), while cross-sector weights increased during market-wide movements, consistent with empirical observations that correlations increase during market stress [11].

Deeper inspection of the learned graph reveals three structural properties that align with established financial theory. First, edge weights cluster by sector in a manner that goes beyond the initialization prior. Information technology and communication-services stocks (Apple, Microsoft, Salesforce, Netflix) form the tightest cluster with average intra-

group weights of 0.81, followed by energy (ExxonMobil, Chevron) at 0.79 and health care (Johnson & Johnson, UnitedHealth, Pfizer) at 0.74. Consumer-oriented stocks (Procter & Gamble, Coca-Cola, Nike, McDonald's) and industrials (Boeing, Caterpillar) exhibit weaker internal cohesion (0.63 and 0.58 respectively), consistent with the greater heterogeneity of business models within those groupings. Second, the temporal stability of edge weights varies predictably with economic conditions. During the low-volatility period of 2017–2019, the standard deviation of edge weights across consecutive months averaged 0.04, indicating a stable relationship structure. This stability broke down during the March 2020 COVID crash, when the month-over-month standard deviation spiked to 0.11 and cross-sector edge weights increased by an average of 0.18, reflecting the well-documented phenomenon of correlation convergence during systemic stress. By mid-2021, the weight distribution had largely reverted to pre-crisis levels. Third, certain cross-sector edges exhibited persistently high weights that reflect supply-chain or macroeconomic linkages rather than sector proximity. The learned weight between ExxonMobil and Caterpillar (0.67, the highest cross-sector pair) likely captures the shared sensitivity of energy and heavy-industry firms to commodity cycles and capital expenditure trends. Table 4.10 lists the five highest and five lowest learned edge weights to illustrate the range of discovered relationships.

Table 4.10: Top-5 and bottom-5 learned edge weights after training.

Stock i	**Stock j**	**Weight**	**Same Sector**
Apple	Microsoft	0.86	Yes
ExxonMobil	Chevron	0.82	Yes
Microsoft	Salesforce	0.78	Yes
J&J	UnitedHealth	0.76	Yes
ExxonMobil	Caterpillar	0.67	No
Netflix	Chevron	0.19	No
Pfizer	Nike	0.21	No
Verizon	Salesforce	0.23	No
Boeing	Coca-Cola	0.24	No
McDonald's	Netflix	0.26	No

The top weights are dominated by same-sector pairs whose business fundamentals are closely linked, while the bottom weights connect companies with minimal overlap in revenue

drivers, customer bases, or macroeconomic exposures. The one exception among the top five, ExxonMobil–Caterpillar, reflects a learned cross-sector dependency that the initialization prior alone would not have produced. These patterns suggest that the edge refinement mechanism successfully discovers economically meaningful relationships that go beyond the sector and correlation priors used for initialization.

Sentiment integration appears particularly valuable around information events. The 25% improvement during earnings seasons suggests that social media sentiment may carry early signals not yet fully reflected in contemporaneous prices. Sector-level heterogeneity aligns with expectations: consumer-facing companies show stronger sentiment-price relationships (14% MAPE reduction) compared to utilities (6%), likely reflecting differential retail investor engagement.

The gating mechanism enables adaptive feature weighting that responds to market conditions. Analysis of learned gate activations showed that momentum features (RSI, MACD, short-term returns) received higher weights during volatile periods, while longer-term indicators (20-day SMA, rolling volatility) received higher weights during stable periods. This adaptive behavior aligns with the intuition that different indicators carry different predictive value depending on market regime, and it emerges naturally through training rather than being imposed through manual feature selection.

4.4.2 Implications

For practitioners, the model's directional accuracy could support more informed trading decisions, particularly for short-term strategies where predicting price direction is often more important than predicting magnitude. The ability to maintain lower error during volatile conditions could assist risk management, as these are precisely the periods when forecast reliability matters most for position sizing and drawdown control. The graph structure provides interpretable inter-stock relationships that may inform portfolio diversification, as

the learned edge weights can surface dependencies that traditional correlation matrices might obscure. These practical implications remain conditional on the caveats discussed in the following subsection, including survivorship bias in the evaluation universe and simplified transaction cost assumptions.

From a theoretical perspective, performance gains from sentiment integration suggest that markets may not fully incorporate qualitative information immediately, a finding consistent with bounded efficiency [8]. The 25% improvement during earnings seasons suggests that social media may contain early signals about investor reactions that have not yet been absorbed into price. The effectiveness of graph-based modeling is consistent with the view that markets function as complex, interconnected systems rather than collections of independent securities, though the limited evaluation universe prevents strong generalization of this observation.

4.4.3 Limitations

The findings should be interpreted in light of limitations spanning data selection, modeling assumptions, and economic validity.

The most consequential concern is survivorship bias in stock selection. The 20 stocks were chosen based on current S&P 500 membership and data availability, meaning that companies which failed, were acquired, or were delisted between 1982 and 2025 are absent from the dataset. Because the selected universe consists exclusively of companies that ultimately succeeded, the dataset omits firms that experienced bankruptcy, prolonged decline, or delisting, which are typically less predictable and more volatile. Reported performance is therefore likely inflated relative to what a real-time investor would have experienced using historical index constituents, and the results should be interpreted as demonstrating the architecture's effectiveness on a stable set of well-established, liquid equities rather than as a guarantee of generalizable performance across the full market. Constructing a point-in-time universe from historical S&P 500 membership records, including reconstitution events,

would provide a more realistic evaluation and is identified as a priority for future work. Relatedly, the 20-node graph structure, while demonstrating meaningful contribution in ablation studies (15% improvement), represents a limited cross-section of the market. Larger universes might better capture complex market topology, and whether the learned edge weights retain predictive value as the graph grows from 20 nodes to hundreds remains an open question. The current design prioritizes temporal depth over cross-sectional breadth, serving as a controlled environment to validate the graph-based modeling approach before scaling to broader equity populations.

Beyond stock selection, data-related assumptions also affect generalizability. Edge weight initialization relies on correlations computed over the training period (1982–2010), but market correlations are inherently non-stationary. Relationships that held during this window may weaken or reverse by the test period. The learnable edge refinement mechanism partially compensates by adapting weights during training, though fundamental correlation regime shifts could still limit generalization. Similarly, X (formerly Twitter) data availability from 2007 onward means sentiment features are set to zero for the 1982–2006 training period, creating a pronounced imbalance: the sentiment fusion mechanism trains on only four years of multimodal data (2007–2010) while the price-based components benefit from 28 years of signal. Although the adaptive gating mechanism mitigates this by learning to suppress the sentiment channel when its input is zero, the limited multimodal training window may constrain the model's ability to fully exploit sentiment-price interactions. The ablation study (Section 4.3.4) confirms that the architecture retains strong performance without sentiment (MAPE 0.88%), and the reported 10% improvement from sentiment integration should be understood as a lower bound that could increase with a longer sentiment training history. Sentiment analysis also relies on English-language content from a single social media platform, and changes to API access policies may affect reproducibility.

From a practical standpoint, the economic significance analysis applies a fixed 10 basis point transaction cost but does not model variable market impact or execution slippage,

both of which increase with order size and decrease with liquidity. The 32% daily turnover observed in the backtest would amplify these unmodeled frictions, and the net returns reported in Section 4.3.4 should be viewed as an upper bound on realizable performance. Model complexity also presents challenges for real-time high-frequency applications, as the 252-day input sequence length and multi-head attention mechanisms demand substantial computational resources. The evaluation is performed on the same 20 stocks used for model development, which limits assessment of external validity on unseen securities.

These limitations, particularly the degradation during volatile periods and the absence of explicit regime detection, motivate the regime-aware dual-pathway architecture developed in Chapter 5.

4.5 Summary

This chapter introduced an integrated framework that combines a node transformer architecture with BERT-based sentiment analysis for stock price forecasting. The model represents the stock market as a graph in which individual stocks serve as nodes connected by learnable edges that encode sectoral affiliations and price correlations. A fine-tuned BERT model processes social media text to extract sentiment signals, which are then merged with quantitative price features through an adaptive attention-based fusion mechanism conditioned on volatility and sentiment magnitude.

Experiments on the 20-stock dataset described in Chapter 3 yielded a MAPE of 0.80% for one-day-ahead predictions, compared to 1.20% for ARIMA and 1.00% for LSTM. Ablation studies isolated the contribution of each component: removing sentiment raised error by 10% (and by 25% during earnings seasons), while removing the graph structure increased error by 15%. Directional accuracy reached 65%, exceeding baselines by 7 to 10 percentage points, and both paired t-tests and Diebold-Mariano tests confirmed statistical significance across all comparisons ($p < 0.05$), with the latter accounting for autocorrelation in forecast residuals.

Taken together, these findings provide preliminary evidence that jointly modeling inter-stock dependencies and investor sentiment can improve forecast accuracy relative to architectures that address either dimension in isolation. Several limitations temper the strength of this conclusion: the evaluation covers 20 large-cap U.S. equities selected with survivorship bias, the sentiment training window spans only a fraction of the full training period, and the economic backtest applies simplified transaction cost assumptions. Whether these improvements extend to broader, historically reconstructed equity universes and withstand realistic trading frictions remains an open question that this chapter cannot resolve. The following chapter addresses the architectural limitation of uniform treatment across market conditions through a regime-aware dual-pathway architecture that incorporates autoencoder-based regime detection and reinforcement learning control.

5

Adaptive Regime-Aware Stock Price Prediction Using Autoencoder-Gated Dual Node Transformers with Reinforcement Learning Control

This chapter is based on the following publication:

Al Ridhawi, M., Haj Ali, M., & Al Osman, H. (2026). Adaptive Regime-Aware Stock Price Prediction Using Autoencoder-Gated Dual Node Transformers with Reinforcement Learning Control. *Submitted to Applied Intelligence*. Under revision. arXiv:2603.19136.

This chapter presents the second contribution of this thesis: an adaptive prediction framework that automatically identifies deviations from normal market conditions and routes data through specialized prediction pathways. The integrated node transformer introduced in Chapter 4 treats all market conditions uniformly, applying the same processing regardless of whether the market is in a stable or volatile state. This chapter addresses that limitation through three architectural innovations: (1) an autoencoder trained on stable market data that identifies anomalous regimes through reconstruction error, (2) dual node transformer networks specialized for stable and event-driven conditions respectively, and (3) a Soft Actor-Critic (SAC) reinforcement learning controller that adaptively tunes the regime detection threshold and pathway blending weights based on prediction performance feedback. The dataset, feature engineering pipeline, and core evaluation metrics follow the experimental framework established in Chapter 3, while the node transformer architecture builds upon the design described in Chapter 4. All quantitative results reported in this chapter refer to short-horizon closing-price prediction at 1, 5, and 20 trading days ahead for a universe of 20

S&P 500 stocks, with training on 1982 to 2010, validation on 2011 to 2016, and out-of-sample testing on 2017 to 2025, unless otherwise specified.

The chapter proceeds as follows. Section 5.1 positions the contribution relative to prior work on regime detection and adaptive prediction. Section 5.2 presents the complete system architecture, including the autoencoder, dual node transformers, and SAC controller. Section 5.3 reports experimental results across multiple horizons and market conditions. Section 5.4 discusses the findings and limitations. Section 5.5 summarizes the chapter and connects it to the next contribution.

5.1 Related Work and Research Gap

The methodological context underlying this chapter spans regime detection in financial markets, autoencoder-based anomaly detection, and reinforcement learning for financial applications. Chapter 2 provides a comprehensive survey of each of these areas, covering Markov-switching models for regime identification (Section 2.7), graph neural network architectures for cross-sectional stock modeling (Section 2.5), and reinforcement learning approaches to financial decision-making (Section 2.8). This section focuses on the specific intersection of these directions and identifies the gap that motivates the proposed architecture.

Prior work has addressed regime detection and stock prediction as separate problems. Regime-switching models identify market states but do not adapt prediction methods accordingly [1, 20], while stock prediction models treat all conditions uniformly or rely on hand-crafted regime indicators [13, 43]. The integrated node transformer from Chapter 4 demonstrated that combining graph-based inter-stock modeling with BERT sentiment analysis achieves 0.80% MAPE and 65% directional accuracy on S&P 500 stocks. Yet this model applies the same processing regardless of market conditions, leaving potential gains from regime-aware specialization unexploited.

The challenge of regime detection compounds prediction difficulties. Traditional approaches rely on hidden Markov models [1] or threshold rules on volatility indicators, both requiring manual specification of regime definitions. Supervised classifiers demand labeled training data identifying which historical periods constitute crises or anomalies. Such labels are subjective, backward-looking, and fail to generalize as market structure evolves. A system that automatically discovers regime boundaries from prediction performance itself would avoid these limitations.

This chapter addresses both challenges through an integrated framework that pairs unsupervised regime detection (autoencoder) with specialized prediction pathways (dual node transformers) and a SAC controller that learns optimal regime definitions from prediction outcomes. This closed-loop architecture enables the system to discover useful regime boundaries rather than imposing them a priori.

5.2 Methodology

Figure 5.1 provides a high-level overview of the proposed framework from raw data ingestion through evaluation. The method proceeds in four phases. Phase 1 acquires financial and sentiment data and derives normalized prediction and routing features. Phase 2 trains the autoencoder, the dual node transformers, and the Soft Actor-Critic (SAC) controller in successive stages before a joint fine-tuning step. Phase 3 deploys the trained system at inference time, where the autoencoder produces a reconstruction-error anomaly score, a router dispatches each input to a normal or event pathway, and adaptive blending combines the two pathway predictions, with the frozen SAC policy supplying state-dependent values of the threshold τ and blending weight α. Phase 4 evaluates the resulting forecasts against the realized closing price using five complementary metrics. The remainder of this section details each component, with Figure 5.2 providing a focused view of the inference-time component interactions and Figure 5.9 expanding the training stages.

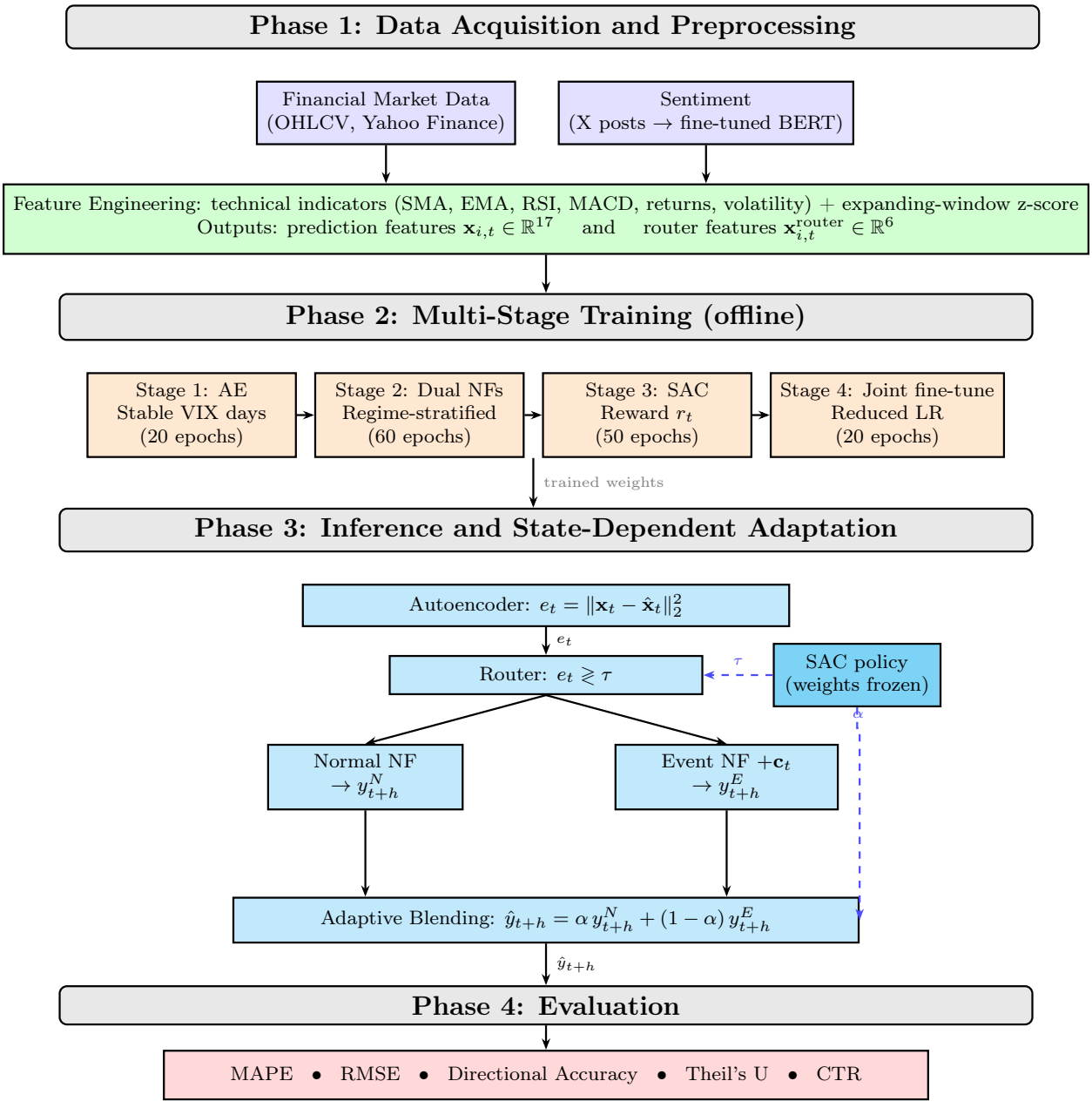

Figure 5.1: Overall method flowchart. Phase 1: data ingestion and feature engineering. Phase 2: four-stage offline training. Phase 3: inference with autoencoder-gated routing and adaptive blending, with the frozen SAC policy supplying state-dependent τ and α (dashed blue arrows). Phase 4: evaluation across five metrics

5.2.1 System Overview

Figure 5.2 presents the complete system architecture. Raw market data flows through the feature engineering pipeline described in Chapter 3 to produce technical indicators and normalized price features. The autoencoder processes these features, producing a reconstruction error score that quantifies deviation from normal market patterns and serves as the basis for regime classification. Based on this score and a learned threshold, the router directs data to either the normal or event node transformer pathway depending on the detected regime. Both pathways produce predictions that are blended according to learned weights. The final prediction is evaluated, and the SAC controller uses this feedback to adjust the autoencoder threshold and blending parameters for subsequent iterations.

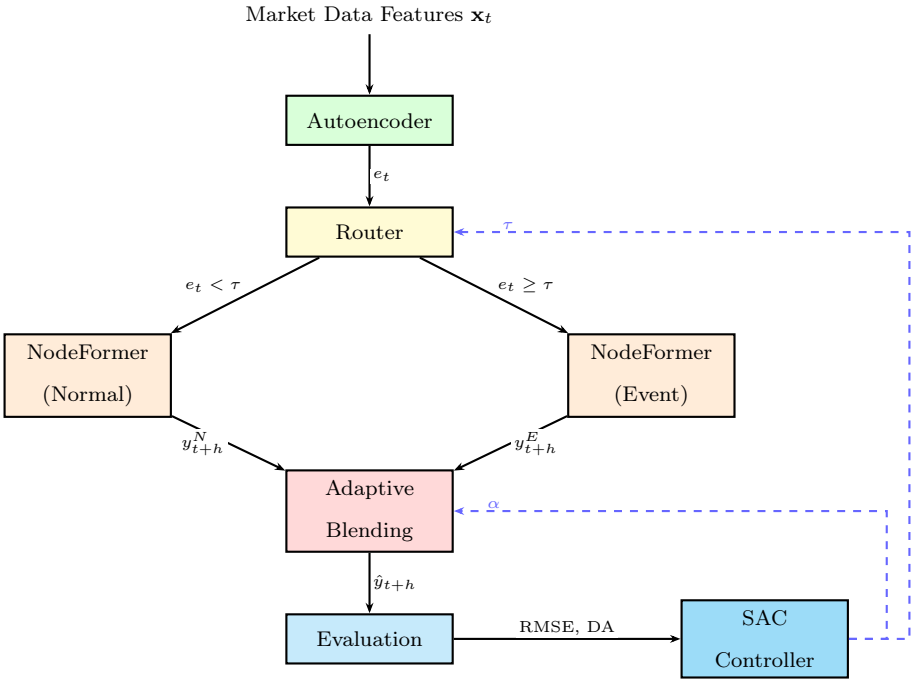

Figure 5.2: System architecture overview. Market features $\mathbf{x}_t$ enter the autoencoder, which produces reconstruction error e_t (shown on arrow). The router directs data to normal or event node transformer pathways based on whether e_t exceeds the learned threshold τ. Each pathway produces a prediction (y^N_{t+h}, y^E_{t+h}), and adaptive blending combines them into the final forecast $\hat{y}_{t+h}$. The SAC controller observes evaluation metrics and adjusts both τ and α (dashed blue arrows) to optimize forecasting accuracy.

The term *regime* in this framework operates at two distinct levels. At the primary level, the autoencoder performs a binary classification of each trading day as either *normal* or *anomalous* based on whether its reconstruction error exceeds a learned threshold τ. This binary decision determines routing: days classified as normal are processed by the normal node transformer, while anomalous days are directed to the event node transformer. A binary primary classification is chosen rather than a multi-class scheme for both practical and theoretical reasons. The autoencoder's reconstruction error is a scalar anomaly score that naturally lends itself to thresholding rather than clustering into multiple categories, and the fundamental distinction in anomaly detection is between in-distribution and out-of-distribution inputs. Attempting to subdivide anomalous states at the routing stage would require assumptions about the number and nature of anomaly categories that the unsupervised autoencoder is not designed to make; instead, that finer-grained characterization is deferred to the event pathway itself.

Within the event pathway, a secondary level of regime characterization captures the heterogeneity of anomalous periods. The event context vector $\mathbf{c}_t$ provides the event node transformer with descriptive features including a VIX-based volatility classification into three levels (low, medium, or high, determined by training-period terciles), sentiment spike indicators, earnings event proximity, and cross-asset stress measures. This secondary characterization does not constitute a separate routing mechanism; rather, it conditions the event transformer's internal representations by supplying information about the nature of the detected anomaly. An earnings-driven disruption during a period of otherwise low market volatility produces different price dynamics than a systemic sell-off during an already elevated volatility regime, and the context vector enables the transformer to learn these distinctions from the data. The normal transformer does not receive this additional context because it processes in-distribution samples where market dynamics follow the stable patterns learned during the autoencoder's training phase, making regime-specific conditioning unnecessary.

Figure 5.3 makes concrete what the context vector $\mathbf{c}_t$ captures as low or high departures from normal market dynamics by contrasting its four feature groups on two illustrative trading days from the test period. The left panel shows a low-departure session, typical of stable mid-cycle conditions: low-tercile VIX, no sentiment spike, distant earnings, and tightly clustered cross-asset reconstruction errors yield an autoencoder reconstruction error $e_t = 0.21$ that falls below the SAC-tuned threshold $\tau \approx 0.72$, and the router selects the normal pathway. The right panel shows a high-departure session, illustrative of conditions around the early-2020 market dislocation: high-tercile VIX, an active sentiment spike, an imminent earnings announcement, and broadly elevated cross-asset errors yield $e_t = 0.94$, which exceeds the lowered threshold $\tau \approx 0.40$ (Figure 5.12), and the router selects the event pathway. The formal definition of each feature group and its dimensionality is given in the Event Node Transformer subsection.

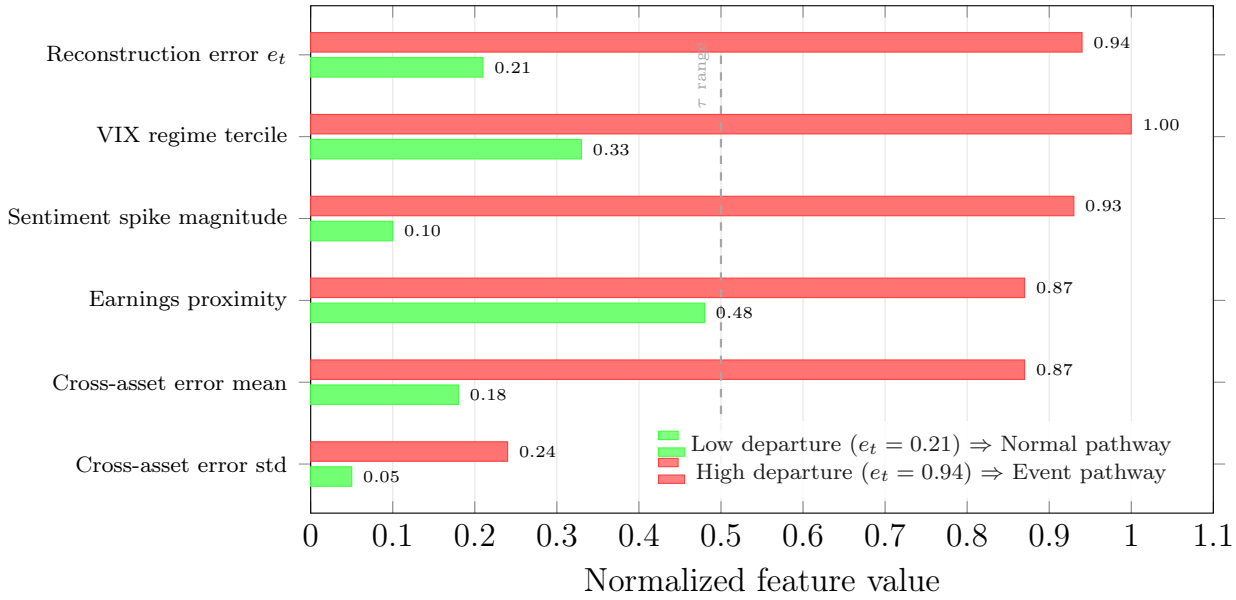

Figure 5.3: Routing signal e_t (top bar) and the four feature groups of the context vector $\mathbf{c}_t$ on two illustrative trading days from the test period, with all values normalized to $[0, 1]$. Green bars: low-departure session, routed to the normal pathway. Red bars: high-departure session, routed to the event pathway. The dashed vertical line marks the approximate range of the SAC-tuned threshold τ

5.2.2 Feature Engineering

The prediction feature vector follows the pipeline described in Section 3.3, producing 17 features per stock per trading day: OHLCV data and 11 derived technical indicators (SMA, EMA, RSI, MACD, returns, log returns, and rolling volatility). Normalization uses expanding-window z-scores as defined in Equation 3.14 to prevent look-ahead bias. Missing value imputation follows the temporally aware strategy described in Section 3.5. Figure 5.4 illustrates the complete pipeline from raw data to the two output feature sets.

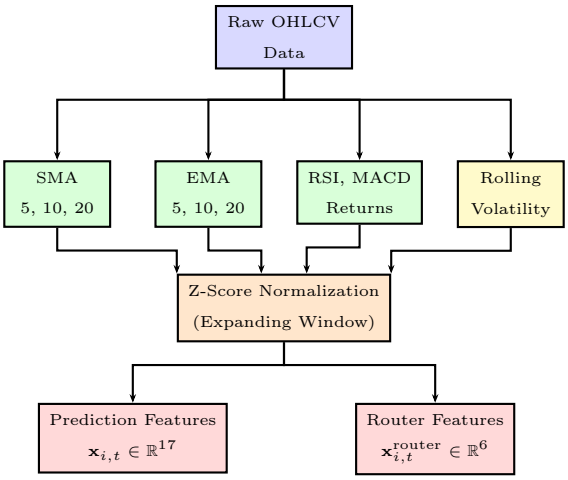

Figure 5.4: Feature engineering pipeline. Raw OHLCV data is processed through technical indicator computations (SMA, EMA, RSI, MACD, volatility). All features undergo expanding-window z-score normalization to prevent look-ahead bias, producing prediction features and router-specific features.

In addition to prediction features, the regime detection component uses the six router-specific features defined in Equation 3.10. These features capture regime-relevant signals including short-term and long-term rolling volatility, VIX percentage change, change in average pairwise correlation among stocks, absolute sentiment magnitude, and post velocity (count of X posts mentioning crisis-related keywords within the trading day). Sentiment enters the router as an absolute value because the router's function is anomaly detection rather than directional prediction. For the purpose of identifying unusual market conditions, the magnitude of sentiment deviation is the relevant signal: both strongly negative sentiment (indicating panic) and strongly positive sentiment (indicating euphoria or speculative excess)

represent departures from typical market behavior. The signed sentiment score S_t is retained in the full prediction feature vector $\mathbf{x}_{i,t}$ that reaches the node transformers, so directional information contributes to the price forecasts themselves even though the routing decision depends only on sentiment intensity. These six router features are chosen to capture both gradual shifts (rolling volatility, correlation changes) and abrupt events (VIX spikes, sentiment surges, social media clustering).

5.2.3 Autoencoder for Regime Detection

The autoencoder learns a compressed representation of normal market dynamics. It is trained exclusively on data from stable market periods, defined during training as days where VIX remains below the 75th percentile of its training-period distribution. Figure 5.5 presents the detailed architecture.

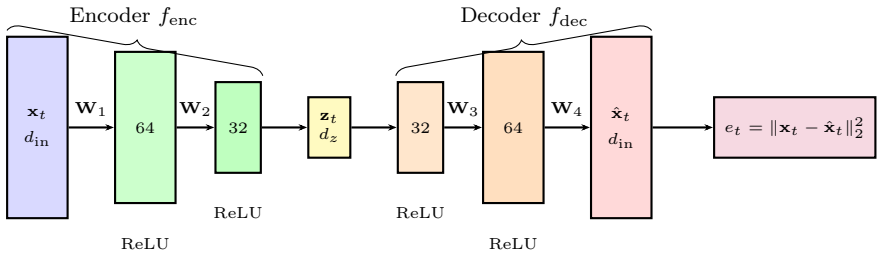

Figure 5.5: Autoencoder architecture for regime detection. The encoder compresses the input feature vector through two hidden layers (64, 32 units) to a latent representation $\mathbf{z}_t$ of dimension $d_z = 32$. The decoder reconstructs the input through symmetric layers. Reconstruction error e_t serves as the anomaly score for regime classification.

Architecture and Training

The encoder maps the concatenated feature vector to a latent representation through two hidden layers:

$$\mathbf{z}_t = f_{\text{enc}}(\mathbf{x}_t) = \text{ReLU}(\mathbf{W}_2 \cdot \text{ReLU}(\mathbf{W}_1 \mathbf{x}_t + \mathbf{b}_1) + \mathbf{b}_2) \tag{5.1}$$

where $\mathbf{x}_t \in \mathbb{R}^{d_{\text{in}}}$ is the input feature vector, $\mathbf{z}_t \in \mathbb{R}^{d_z}$ is the latent representation with $d_z = 32$, and $\mathbf{W}_1 \in \mathbb{R}^{64 \times d_{\text{in}}}$, $\mathbf{W}_2 \in \mathbb{R}^{32 \times 64}$ are weight matrices. The decoder reconstructs the input through a symmetric architecture:

$$\hat{\mathbf{x}}_t = f_{\text{dec}}(\mathbf{z}_t) = \mathbf{W}_4 \cdot \text{ReLU}(\mathbf{W}_3 \mathbf{z}_t + \mathbf{b}_3) + \mathbf{b}_4 \tag{5.2}$$

where $\mathbf{W}_3 \in \mathbb{R}^{64 \times 32}$ and $\mathbf{W}_4 \in \mathbb{R}^{d_{\text{in}} \times 64}$. The autoencoder is trained to minimize reconstruction loss over the stable-period data:

$$\mathcal{L}_{\text{AE}} = \frac{1}{T} \sum_{t=1}^{T} \|\mathbf{x}_t - \hat{\mathbf{x}}_t\|_2^2 \tag{5.3}$$

Training uses the Adam optimizer with learning rate 10^{-3}, batch size 64, for a maximum of 20 epochs with early stopping based on validation reconstruction loss.

Anomaly Score and Routing

At inference time, the reconstruction error serves as an anomaly score $e_t = \|\mathbf{x}_t - \hat{\mathbf{x}}_t\|_2^2$. Data points with e_t exceeding threshold τ are classified as anomalous and routed to the event pathway, while those below τ proceed through the normal pathway. The threshold τ is initialized at the 95th percentile of training-set reconstruction errors and subsequently adjusted by the SAC controller.

Indicator and Threshold Selection

Three design choices govern the autoencoder training and the routing initialization: the VIX index as the regime indicator, the 75th-percentile cutoff for selecting stable training samples, and the 95th-percentile initialization of the routing threshold τ. Each is justified individually below.

The VIX (CBOE Volatility Index) is selected as the regime indicator because it provides a forward-looking, market-wide measure of expected 30-day volatility derived from S&P 500 option prices, is freely available at daily resolution, and is the most widely adopted regime indicator in academic equity-prediction work [1, 129]. Alternative measures considered include realized historical volatility of the S&P 500 (backward-looking and slower to react to regime shifts), GARCH-family volatility forecasts (model-specific and computationally heavier), and the MOVE index (bond-market volatility, less aligned with equity prediction). For an equity prediction framework the forward-looking, equity-derived character of VIX makes it the most appropriate scalar choice; the framework itself does not depend on this specific indicator and can accommodate any scalar volatility measure available at daily resolution.

The 75th-percentile cutoff for selecting stable training samples balances training-data purity against training-data quantity. A more permissive cutoff (e.g., the 90th percentile) would admit genuinely volatile days into the autoencoder's training distribution and contaminate its definition of normal dynamics; a more restrictive cutoff (e.g., the 50th percentile) would discard substantial training data and weaken statistical coverage of stable-regime structure. The chosen 75th-percentile cutoff retains the lowest three quartiles of VIX days and excludes the highest quartile, which contains the regime episodes that the autoencoder is intentionally not asked to reconstruct.

The 95th-percentile initialization of the routing threshold τ follows the standard anomaly-detection convention of classifying the top 5% of training-distribution reconstruction errors as anomalous [79], and is only a starting value: the SAC controller adapts τ continuously after Stage 3 (Section 5.2.5), so the choice of initial percentile shapes only the early Stage 3 reward landscape rather than the final routing behavior. The threshold-trajectory plot (Figure 5.12) confirms that τ moves substantially away from its initialization over the test period, indicating that the system is not strongly dependent on the chosen initial percentile.

5.2.4 Dual Node Transformer Architecture

Two node transformer networks process data depending on regime classification. Both follow the same base architectural design (6 layers, 8 attention heads, 512 model dimension) as the node transformer introduced in Chapter 4, but maintain independent weights trained on different data subsets, and the event pathway accepts a larger input due to additional context features. Figure 5.6 illustrates the dual pathway structure.

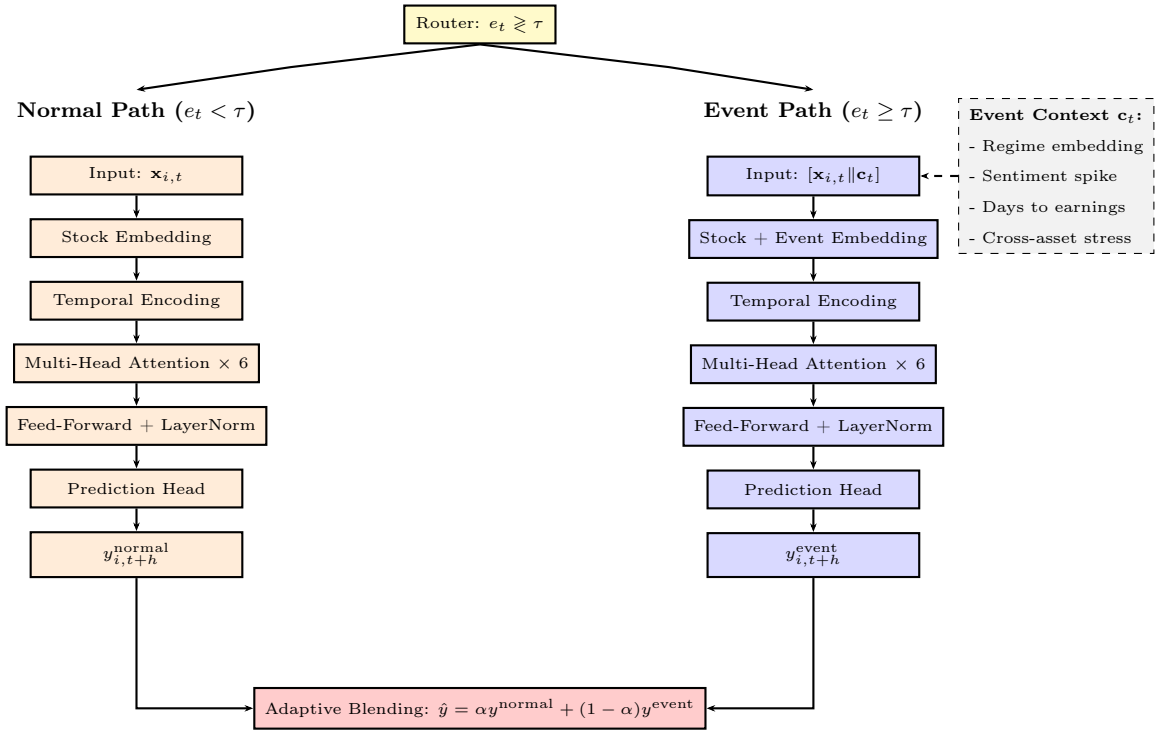

Figure 5.6: Dual node transformer architecture. The router directs data based on reconstruction error. The normal pathway (left, orange) processes typical market conditions with base features. The event pathway (right, blue) augments inputs with event context features $\mathbf{c}_t$. Both pathways follow the same architectural design (layer count, attention heads, model dimension) but maintain independently trained weights and differ in input dimensionality, as the event pathway accepts additional context features. Outputs are blended with adaptive weight α.

Normal Node Transformer

The normal pathway processes typical market conditions using the node transformer architecture [12] as described in Section 4.2. The stock market is represented as a graph

$\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $N = 20$ stock nodes and a fully-connected edge set. While graph neural networks are often applied to larger graphs, the $N = 20$ design balances cross-sectional breadth against temporal depth (252-day sequences per stock), and ablation results confirm that the graph structure contributes 7% MAPE improvement (Table 5.6), indicating that cross-sectional dependencies carry predictive value even at this scale.

Each stock i receives a learned embedding $\mathbf{s}_i \in \mathbb{R}^{d_s}$ that captures persistent stock-specific characteristics such as sector behavior and volatility profile. The input representation concatenates the normalized feature vector with temporal encoding and the stock embedding, following the same formulation as Equation 4.3. Edge weights are initialized from sector relationships and return correlations computed strictly on training data (1982–2010), as defined in Equation 4.1, and refined through the learnable function in Equation 4.2. Figure 5.7 illustrates the resulting graph structure with representative edge weights.

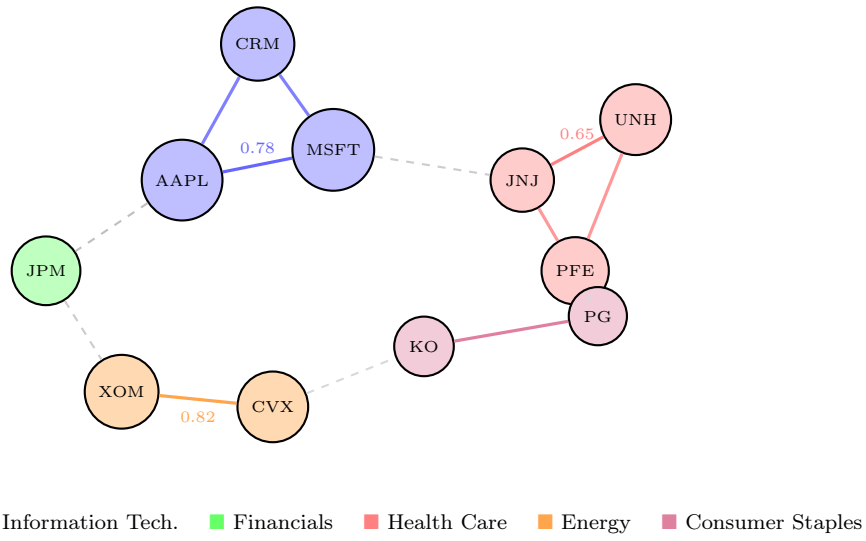

Figure 5.7: Graph representation of stock relationships (representative subset of 11 stocks shown for clarity; full graph contains all 20 stocks). Nodes represent individual stocks, colored by sector. Solid edges indicate same-sector connections with higher learned weights (annotated values show correlation-based initialization from training data). Dashed edges represent weaker cross-sector correlations that are learned during training.

At each layer, the node transformer applies multi-head self-attention with causal masking to jointly process all stocks across the temporal dimension. The attention mechanism follows

the formulation in Equation 4.5, where the additive graph bias allows content-based attention (via $\mathbf{QK}^T$) and structural priors (via $\mathbf{E}$) to jointly determine how information flows between stocks at each layer, ensuring that predictions at time t use only information from times up to and including t. The architecture uses $H = 8$ attention heads, each operating in 64 dimensions, yielding a total model dimension of $d_{\text{model}} = 512$. Each transformer layer follows the standard pre-norm residual pattern described in Section 4.2.3 and illustrated in Figure 4.2. Dropout at rate 0.1 is applied after both the attention and feed-forward sublayers. The architecture stacks 6 such layers, with the output of the final layer fed into a prediction head consisting of a linear projection from the model dimension to a single scalar price prediction per stock.

Event Node Transformer

The event pathway augments the base architecture with additional inputs capturing regime-specific information. The input to the event transformer concatenates the standard feature vector with an event context vector, $\mathbf{x}_{i,t}^{\text{event}} = [\mathbf{x}_{i,t} \| \mathbf{c}_t]$, where $\mathbf{c}_t \in \mathbb{R}^{d_c}$ with $d_c = 12$. This vector comprises four groups of features. A learned regime embedding $\mathbf{r}_t \in \mathbb{R}^4$ maps the current VIX regime (low, medium, or high, determined by training-period VIX terciles) through a trainable embedding layer. A sentiment spike component $\mathbf{s}_t \in \mathbb{R}^2$ encodes a binary flag and scaled magnitude when daily sentiment exceeds two standard deviations of training-period sentiment. An event characterization component $\mathbf{a}_t \in \mathbb{R}^4$ captures proximity to scheduled earnings announcements (days-to-announcement, normalized), historical earnings surprise magnitude for the stock, a binary earnings-window indicator, and sector-average surprise. Finally, a cross-asset stress vector $\bar{e}_t^{\text{cross}} \in \mathbb{R}^2$ encodes the mean and standard deviation of reconstruction error across all 20 stocks at time t, distinguishing systemic anomalies (high mean error) from idiosyncratic ones (high variance). The full context vector is the concatenation $\mathbf{c}_t = [\mathbf{r}_t \| \mathbf{s}_t \| \mathbf{a}_t \| \bar{e}_t^{\text{cross}}]$.

Architecturally, the event transformer differs from the normal pathway in two respects beyond its independently trained weights. First, the input projection layer is wider: while the normal transformer's first linear layer maps from d_{in} dimensions (the concatenation of market features, temporal encoding, and stock embedding), the event transformer maps from $d_{\text{in}} + d_c$ dimensions to accommodate the appended context vector. This wider projection maps back to the shared model dimension of $d_{\text{model}} = 512$ before entering the first transformer layer, so all subsequent layers (the 6 transformer blocks, feed-forward networks, and prediction head) operate at the same dimensionality as the normal pathway. Second, the event pathway includes a trainable embedding layer that maps the discrete VIX regime label (one of three categories) to the continuous regime embedding $\mathbf{r}_t \in \mathbb{R}^4$. This embedding layer is an additional learnable component with no counterpart in the normal pathway, adding $3 \times 4 = 12$ trainable parameters for the three regime categories. The remaining context features ($\mathbf{s}_t$, $\mathbf{a}_t$, $\bar{e}_t^{\text{cross}}$) are continuous values that enter the context vector directly without additional learned transformations.

Each of the four context components admits a plain-language operational reading for the practitioner audience. The regime embedding $\mathbf{r}_t$ encodes whether the prevailing VIX level falls in the low, medium, or high tercile of its historical training distribution, functioning as a discrete market-state tag (calm, intermediate, stressed) rather than a continuous fear gauge. The sentiment-spike indicator $\mathbf{s}_t$ flags days on which the BERT-derived sentiment score for the stock exceeds its training-period typical range by more than two standard deviations, with a binary on/off flag accompanied by the spike's signed magnitude; in operational terms, this fires on news or discussion days that are anomalously positive or negative relative to the stock's own history, regardless of whether the price has yet moved. The event characterization vector $\mathbf{a}_t$ captures proximity to and reaction profile around scheduled earnings releases, indicating both whether a known catalyst is approaching (days-to-announcement, in-window flag) and how strongly the stock has historically responded (surprise magnitude, sector-average surprise). The cross-asset stress measure $\bar{e}_t^{\text{cross}}$ summarizes the autoencoder reconstruction

errors across all 20 stocks at the same trading day using their mean and standard deviation: the mean reads as a market-wide stress index, where many stocks simultaneously departing from their learned normal behavior implies a systemic event, while the standard deviation distinguishes systemic stress (mean high, standard deviation low, all stocks moving together) from idiosyncratic stress (standard deviation high, mean low or moderate, one or two stocks dislocating while others remain stable). Together these four components give the event pathway access to the kind of intuitive context that a discretionary trader would assemble manually: the prevailing volatility regime, the day's unusual-news flag, the proximity of known catalysts, and a system-wide stress thermometer.

Pathway Blending

Rather than hard routing, predictions from both pathways are blended with an adaptive weight:

$$\hat{y}_{i,t+h} = \alpha_t \cdot y^{\text{normal}}_{i,t+h} + (1 - \alpha_t) \cdot y^{\text{event}}_{i,t+h} \tag{5.4}$$

The blending coefficient $\alpha_t \in [0, 1]$ is determined by the SAC controller based on current market state. During high-confidence normal periods, α_t approaches 1; during clear anomalies, it approaches 0. Intermediate values enable smooth transitions and hedge against misclassification.

5.2.5 Soft Actor-Critic Controller

The SAC controller learns to configure the prediction system by adjusting the autoencoder threshold τ and blending weight α based on observed prediction performance. Although these are only two scalar parameters, the optimization landscape is non-trivial: the reward signal is delayed (prediction errors are observed only after the threshold decision), noisy (financial returns are inherently stochastic), and non-stationary (optimal thresholds shift as market

regimes evolve). SAC is well suited to this setting because its entropy regularization prevents premature convergence to fixed threshold values, and its off-policy learning with experience replay enables sample-efficient adaptation from sparse, delayed feedback. Simpler alternatives such as grid search or bandit methods typically assume stationary reward distributions [6] and cannot adapt continuously to shifting regime dynamics. Figure 5.8 presents the actor-critic network architecture.

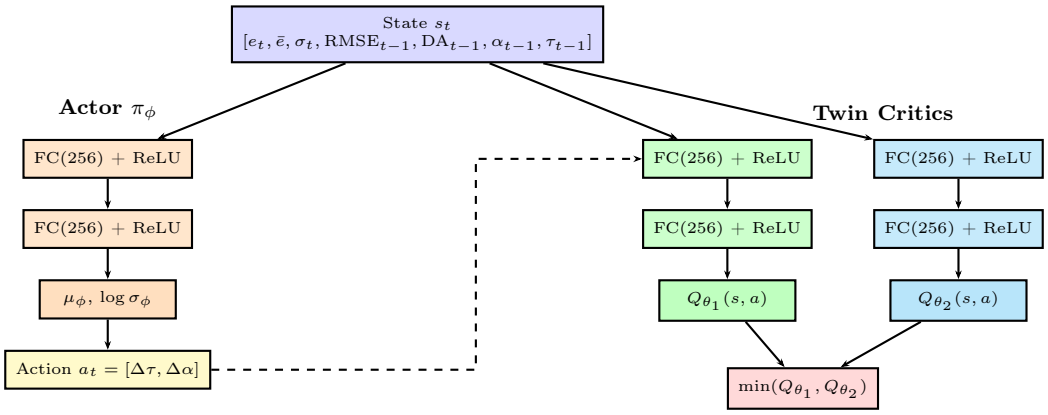

Figure 5.8: Soft Actor-Critic network architecture. The actor network (left, orange) maps states to a Gaussian policy over actions $[\Delta\tau, \Delta\alpha]$. Twin critic networks (right, green/cyan) estimate Q-values; the minimum is used to prevent overestimation. All networks use two hidden layers with 256 units and ReLU activation.

Markov Decision Process Formulation

The control problem is formulated as a Markov Decision Process (MDP). The state s_t comprises:

$$s_t = [e_t, \bar{e}_{t-k:t}, \sigma_t, \text{RMSE}_{t-1}, \text{DA}_{t-1}, \alpha_{t-1}, \tau_{t-1}] \tag{5.5}$$

including current reconstruction error, recent error history over the past $k = 5$ trading days (one week), volatility, previous prediction metrics, and current parameter settings. The action space consists of continuous adjustments $a_t = [\Delta\tau, \Delta\alpha] \in [-0.1, 0.1]^2$ to threshold and blending weight, clipped to maintain $\tau \in [e_{\min}, e_{\max}]$ and $\alpha \in [0, 1]$. The reward signal combines prediction accuracy and stability:

$$r_t = -\text{RMSE}_t - \lambda_{\text{dir}} \cdot (1 - \text{DA}_t) - \lambda_{\text{stable}} \cdot |\Delta\tau| \tag{5.6}$$

where $\lambda_{\text{dir}} = 0.5$ weights directional accuracy and $\lambda_{\text{stable}} = 0.1$ penalizes threshold instability to prevent oscillation.

SAC Algorithm

SAC maximizes the entropy-regularized objective:

$$J(\pi) = \sum_{t=0}^{T} \mathbb{E}\left[r_t + \alpha_{\text{ent}} \mathcal{H}(\pi(\cdot|s_t))\right] \tag{5.7}$$

where $\mathcal{H}$ is policy entropy and α_{ent} is the temperature parameter controlling exploration. The actor network $\pi_\phi(a|s)$ outputs a Gaussian distribution over actions, $\pi_\phi(a|s) = \mathcal{N}(\mu_\phi(s), \sigma_\phi(s)^2)$, while two critic networks Q_{θ_1}, Q_{θ_2} estimate action values. To prevent overestimation, the minimum of both critics is used:

$$Q(s, a) = \min(Q_{\theta_1}(s, a), Q_{\theta_2}(s, a)) \tag{5.8}$$

All networks are feed-forward with two hidden layers of 256 units each. Training uses the Adam optimizer with learning rate 3×10^{-4}, soft target updates with $\tau_{\text{soft}} = 0.005$, and replay buffer size 10^5.

Choice of Reinforcement Learning Algorithm

The selection of Soft Actor-Critic for this controller follows from the structure of the control problem rather than from convenience. The action vector $[\Delta\tau, \Delta\alpha]$ is continuous and low-dimensional, which rules out value-based methods such as Deep Q-Networks [80], since these operate over discrete action sets and would require a coarse and arbitrary quantization of the threshold and blending adjustments. Among continuous-control algorithms, the reward

signal here is noisy, delayed, and non-stationary, because it is computed from downstream prediction error that itself shifts with the market regime, and the controller must learn from a training window limited to the 1982 to 2010 period. These conditions favor an off-policy, entropy-regularized method. SAC reuses past transitions through a replay buffer, which improves sample efficiency relative to on-policy methods such as Proximal Policy Optimization [82] that discard experience after each update, and its maximum-entropy objective sustains exploration and stabilizes training where a deterministic policy such as Deep Deterministic Policy Gradient [81] tends toward premature convergence and heightened sensitivity to hyperparameter choices. Its automatic temperature adjustment further reduces the manual tuning that these alternatives typically demand in non-stationary environments [83].

These arguments justify SAC as a principled design decision rather than as an empirically established optimum for this task. A controlled comparison that retrains the complete framework under each alternative controller and reports matched prediction and trading metrics would quantify the practical margin between them. That benchmark, extended to more recent off-policy actor-critic variants, is left as future work, and the present study fixes the controller to SAC in order to isolate the interaction among regime detection, routing, and adaptive control.

Training Protocol

The SAC controller is trained after the autoencoder and node transformers are pre-trained. Training begins by initializing τ at the 95th percentile of training reconstruction errors and $\alpha = 0.5$. At each step, the controller computes predictions using the current τ and α, evaluates them against actual outcomes to obtain the reward signal, updates the SAC networks from collected transitions, and applies the learned action adjustments to both parameters. This loop runs for 50 epochs with 1000 steps per epoch. Temperature α_{ent} is automatically tuned to target entropy $-\dim(a)$ following Haarnoja et al. [83].

5.2.6 Training Pipeline

Figure 5.9 illustrates the complete multi-stage training pipeline.

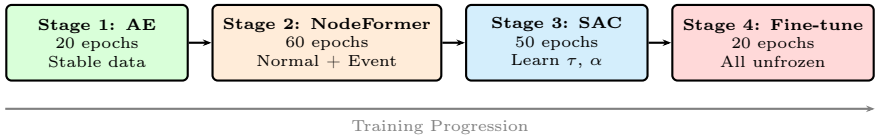

Figure 5.9: Multi-stage training pipeline. Stage 1 trains the autoencoder on stable market data. Stage 2 trains both node transformers on their respective data subsets. Stage 3 trains the SAC controller with frozen prediction components. Stage 4 performs end-to-end fine-tuning with all weights unfrozen.

The complete training pipeline proceeds in four stages. In Stage 1 (20 epochs), the autoencoder is trained on stable-period data where VIX falls below the 75th percentile of its training-period distribution. Stage 2 (60 epochs) trains both node transformers: the normal pathway on data with low reconstruction error (below the 95th percentile), and the event pathway on high-error data augmented with context features. In Stage 3 (50 epochs), autoencoder and node transformer weights are frozen while the SAC controller learns to optimize the threshold and blending parameters. Finally, Stage 4 (20 epochs) unfreezes all components for end-to-end fine-tuning with reduced learning rates. The architecture is modular by design: Stages 1 and 2 produce a fully functional prediction system in which the routing threshold and blending weight remain at their initialization values (τ at the 95th percentile of training-set reconstruction errors, $\alpha = 0.5$). Stage 3 adds adaptive control on top of this static configuration, allowing the experimental evaluation to quantify the marginal contribution of the SAC controller by comparing the system with and without it. At inference time, all weights, including the SAC policy network, are frozen. The policy produces state-dependent routing decisions through its fixed learned mapping, with no gradient updates or reward computation during the test period.

5.2.7 Loss Functions

The prediction networks minimize a composite loss that follows the general structure introduced in Section 4.2.6:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{\text{MSE}} + \lambda_2 \mathcal{L}_{\text{DIR}} + \lambda_3 \mathcal{L}_{\text{REG}} \tag{5.9}$$

where $\mathcal{L}_{\text{MSE}}$ is the mean squared error between predicted and actual prices (Equation 4.16), $\mathcal{L}_{\text{DIR}}$ is the directional loss that explicitly rewards correct prediction of price movement direction (Equation 4.17), and $\mathcal{L}_{\text{REG}} = \|\grave{}\|_2^2$ applies L2 weight decay over all trainable parameters $\grave{}$, penalizing large weight magnitudes to prevent overfitting. Loss weights are $\lambda_1 = 1.0$, $\lambda_2 = 0.5$, $\lambda_3 = 10^{-4}$. Unlike the loss function in Chapter 4, the correlation regularization term $\mathcal{L}_{\text{CORR}}$ (Equation 4.18) is omitted because the dual-pathway architecture inherently separates the representational capacity for different market conditions, reducing the need for explicit decorrelation of attention patterns.

Table 5.1 summarizes all model hyperparameters across the three components.

Table 5.1: Model hyperparameters across all components of the regime-aware prediction framework.

Component	Parameter	Value
Autoencoder	Hidden layers	[64, 32]
	Latent dimension	32
	Learning rate	10^{-3}
	Training epochs	20
Node Transformer	Layers	6
	Attention heads	8
	Model dimension	512
	FFN dimension	2048
	Dropout	0.1
	Learning rate	10^{-4}
	Input sequence length	252 days
SAC Controller	Hidden layers	[256, 256]
	Learning rate	3×10^{-4}
	Soft update τ	0.005
	Replay buffer	10^5
	Training epochs	50
Stage 4 Fine-tuning	AE learning rate	10^{-4}
	Node Transformer learning rate	10^{-5}
	SAC learning rate	3×10^{-5}

5.3 Experiments and Results

5.3.1 Experimental Setup

Experiments used the dataset described in Chapter 3, comprising daily OHLCV data for 20 S&P 500 stocks spanning January 1982 to March 2025 (Table 3.1). The temporal split allocates 1982–2010 for training, 2011–2016 for validation, and 2017–2025 for testing, following the partitioning protocol defined in Section 3.6. The Financial Market Data (FMD) and Comprehensive Stock Sentiment (CSS) data streams are described in Sections 3.2 and 3.4 respectively. Since the X platform was founded in 2006, sentiment data covers January 2007 to March 2025. For the 1982–2006 portion of training, sentiment features are set to zero, meaning the model learns to operate with and without sentiment depending on the data period.

All time-series operations in the framework are strictly causal with respect to the trading calendar. Each daily observation corresponds to a regular U.S. equity trading session, so weekends and exchange holidays do not appear as separate samples. Sentiment scores aggregated from X posts published during non-trading periods, such as overnight and weekend windows, are attributed to the next trading day, mirroring how a real-time deployment would consume social-media signals between consecutive market closes. Predictions for trading day $t + h$ are generated using only information that would be available at or before the close of day t: the OHLCV features, the technical indicators derived from prices through day t, the expanding-window z-score statistics computed up to day t, and the BERT-scored sentiment from posts published up to the close of day t. No look-ahead is introduced through the autoencoder, the dual node transformers, or the SAC controller. The 75th-percentile VIX threshold used to gate autoencoder training and the 95th-percentile reconstruction-error threshold used to initialize τ are both fixed once on the 1982 to 2010 training window and held constant during validation and test inference. At inference time the frozen SAC policy consumes only the previous trading day's realized metrics (RMSE_{t-1}, DA_{t-1}), which are publicly known at market open on day t. The framework can therefore be evaluated, or deployed, against any test window without contamination by data that would not have been available at the time the prediction is made.

5.3.2 Evaluation Metrics

Model performance is assessed using MAPE, RMSE, Directional Accuracy (DA), and Theil's U as defined in Section 3.7. This chapter additionally employs the Confidence Tracking Rate (CTR) defined in Equation 3.21, which captures the proportion of predictions where model confidence (measured as inverse prediction variance across the dual pathway outputs) agrees with actual accuracy. CTR indicates whether the model produces reliable confidence estimates, a property valuable for risk-sensitive downstream applications.

5.3.3 Baseline Models

Baselines span statistical methods (ARIMA, VAR, MS-VAR [130]), classical machine learning (Random Forest, SVR, XGBoost), deep learning (LSTM, Simple Transformer), multimodal and regime-switching approaches (BERT Sentiment + LSTM, HMM-LSTM), recent time-series transformers (TimesNet [131], PatchTST [132], iTransformer [133]), and the Integrated NodeFormer-BERT model from Chapter 4 [134].

To ensure fair comparison, all baselines share the same experimental conditions wherever the model class permits. Every model uses identical temporal splits (training: 1982–2010, validation: 2011–2016, test: 2017–2025), the same expanding-window z-score normalization described in Section 3.5, and the same missing-data imputation strategy. Models capable of multivariate input (Random Forest, SVR, XGBoost, LSTM, Simple Transformer, BERT Sentiment + LSTM, TimesNet, PatchTST, iTransformer, HMM-LSTM, Integrated NodeFormer-BERT) receive the same 17-dimensional feature vector comprising OHLCV data and 11 derived technical indicators. Daily sentiment scores produced by the fine-tuned BERT classifier are appended to the feature set for all multivariate models, so that any advantage from sentiment information is available to baselines as well as to the proposed framework. ARIMA operates on the univariate closing price series for each stock independently, VAR jointly models the closing prices of all 20 stocks, and MS-VAR extends the VAR specification with Markov-switching regime dynamics over the same joint price series, since these statistical methods are not designed to incorporate arbitrary exogenous feature vectors. The target variable for all models is identical: the closing price at horizon $h \in \{1, 5, 20\}$ trading days ahead.

The Simple Transformer baseline uses the same encoder architecture as the proposed node transformer (6 layers, 8 attention heads, 512-dimensional representations) but processes each stock's time series independently without graph structure or inter-stock attention, isolating the contribution of graph-based relational modeling. The BERT Sentiment + LSTM

baseline combines the same BERT-derived sentiment scores with a two-layer LSTM through concatenation-based fusion, testing whether the attention-based integration in the proposed architecture provides meaningful improvement over straightforward feature combination. The Integrated NodeFormer-BERT model reproduces the architecture from Chapter 4 with its published hyperparameters, serving as the primary single-pathway baseline against which architectural additions are measured.

PatchTST [132] segments each stock's multivariate input time series into overlapping patches and applies a transformer encoder with self-attention over the patch sequence, capturing local temporal patterns within patches and long-range dependencies across them. Its channel-independent design processes each feature dimension separately before aggregating predictions, which limits its ability to model cross-feature interactions. iTransformer [133] inverts the conventional transformer architecture by applying self-attention across the variate (feature) dimension rather than the temporal dimension, enabling it to capture dependencies among price, volume, technical indicators, and sentiment features at each time step. TimesNet [131] extends temporal modeling by transforming one-dimensional time series into two-dimensional tensors based on learned multi-periodicity structure, applying inception-based convolution blocks to capture both intra-period and inter-period variation. All three recent time-series transformers process each stock's feature set independently without graph structure or cross-stock attention, isolating the contribution of relational modeling in the proposed framework.

The Markov-Switching VAR (MS-VAR) [130] extends the VAR baseline with $K = 3$ latent regime states governed by a first-order Markov chain, allowing the intercepts and error covariance to vary across regimes while the autoregressive coefficients remain regime-invariant (MSIH specification). Regime transitions are inferred through maximum likelihood estimation of the full joint model, in contrast to the autoencoder-based approach which detects anomalies from reconstruction error without specifying the number or parametric structure of regimes a priori. The HMM-LSTM baseline combines a Hidden Markov Model with $K = 3$ states for

regime detection with three regime-specific two-layer LSTMs, each trained on data assigned to its corresponding regime by the Viterbi decoder. At inference, the HMM identifies the most likely current regime and routes the input to the corresponding LSTM, producing a regime-conditional forecast. This architecture provides the most direct comparison to the proposed framework: it replaces the autoencoder with an HMM for regime detection, the node transformers with LSTMs for prediction, and omits adaptive control entirely, using fixed routing with no blending across pathways.

Each baseline underwent hyperparameter tuning via grid search on validation data, with the search ranges and selected values reported in Table 5.14.

5.3.4 Main Results

Results are reported for two variants of the proposed framework. The full model (AE-NodeFormer + SAC) includes all components and completes all four training stages. The ablated variant (AE-NodeFormer, no SAC) retains the autoencoder and dual node transformers but removes the reinforcement learning controller entirely, skipping Stage 3 of the training pipeline. In this variant, the routing threshold is fixed at $\tau = e_{95}$, the 95th percentile of training-set reconstruction errors, which is the same initialization used by the full model before SAC adaptation begins. This percentile is a standard choice in anomaly detection, classifying the top 5% of reconstruction errors as anomalous. The blending weight is held constant at $\alpha = 0.5$, assigning equal contribution to both pathways regardless of market conditions. Comparing the two variants isolates the contribution of adaptive parameter tuning from the architectural benefits of autoencoder routing and dual-pathway specialization.

Table 5.2 presents 1-day ahead closing price prediction results across all baselines and proposed variants.

The proposed AE-NodeFormer + SAC achieves 0.59% MAPE, representing a 26% relative improvement over the Integrated NodeFormer-BERT baseline (0.80%) and a 28% improvement

Table 5.2: 1-Day ahead closing price prediction results. Best results in bold.

Model	MAPE	RMSE	DA	Theil's U	CTR
ARIMA [135]	1.20%	1.35	55%	0.98	51%
VAR [29]	1.10%	1.30	56%	0.95	52%
MS-VAR [130]	1.02%	1.22	57%	0.90	53%
Random Forest [136]	1.10%	1.25	57%	0.92	53%
SVR [137]	1.20%	1.40	54%	1.02	50%
XGBoost [138]	1.00%	1.15	59%	0.85	55%
LSTM [41]	1.00%	1.20	58%	0.88	54%
Simple Transformer [3]	0.90%	1.10	61%	0.80	57%
BERT Sent. + LSTM [4]	0.90%	1.05	62%	0.78	58%
HMM-LSTM [1]	0.87%	1.02	64%	0.76	60%
TimesNet [131]	0.85%	1.00	63%	0.75	59%
PatchTST [132]	0.83%	0.98	64%	0.74	59%
iTransformer [133]	0.82%	0.97	64%	0.73	61%
Integrated NF-BERT [134]	0.80%	0.95	65%	0.72	62%
AE-NodeFormer (no SAC)	0.68%	0.88	69%	0.68	64%
AE-NodeFormer + SAC	**0.59%**	**0.82**	**72%**	**0.64**	**67%**

over iTransformer (0.82%), the strongest recent time-series transformer. Directional accuracy reaches 72%, a 7 percentage point gain over the graph-based baseline. Among regime-switching approaches, HMM-LSTM achieves 0.87% MAPE, outperforming the basic LSTM (1.00%) by 13% through regime-specific specialization, yet still trailing the proposed model by 32%, indicating that the combination of autoencoder-based anomaly detection, graph-aware dual pathways, and adaptive control provides substantially greater benefit than parametric regime detection with independent LSTMs. The recent time-series transformers (TimesNet 0.85%, PatchTST 0.83%, iTransformer 0.82%) cluster near the Integrated NodeFormer-BERT (0.80%), confirming that the prior single-pathway architecture was already competitive with current state-of-the-art forecasting models and that the improvements in the present work stem from the regime-aware architectural innovations rather than from a weak baseline. All pairwise improvements of the proposed model over iTransformer, PatchTST, and HMM-LSTM are statistically significant (Diebold-Mariano test [128], $p < 0.001$ in each case).

To contextualize the directional accuracy, we computed a naive long-only baseline: predicting "up" for every day. Over the 2017–2025 test period, this naive strategy achieves 54%

DA on average across the 20 stocks, reflecting the slight upward drift in equity markets. The 72% DA of the proposed model thus represents an 18 percentage point improvement over this trivial baseline, confirming that the model captures predictive signal beyond simple market drift.

To assess generalization across forecasting horizons, Table 5.3 and Table 5.4 present 5-day and 20-day ahead closing price results.

Table 5.3: 5-Day ahead closing price prediction results.

Model	**MAPE**	**RMSE**	**DA**	**Theil's U**	**CTR**
ARIMA [135]	2.05%	2.30	51%	1.05	47%
VAR [29]	1.88%	2.10	52%	1.00	48%
MS-VAR [130]	1.70%	1.90	53%	0.95	49%
Random Forest [136]	1.92%	2.15	52%	1.02	48%
SVR [137]	2.10%	2.35	50%	1.08	46%
XGBoost [138]	1.68%	1.88	54%	0.92	50%
LSTM [41]	1.65%	1.85	54%	0.93	50%
Simple Transformer [3]	1.50%	1.68	56%	0.85	53%
BERT Sent. + LSTM [4]	1.48%	1.65	57%	0.83	54%
HMM-LSTM [1]	1.45%	1.60	59%	0.82	55%
TimesNet [131]	1.40%	1.55	58%	0.81	56%
PatchTST [132]	1.38%	1.52	59%	0.79	55%
iTransformer [133]	1.35%	1.50	59%	0.80	57%
Integrated NF-BERT [134]	1.30%	1.45	61%	0.78	58%
AE-NodeFormer (no SAC)	1.15%	1.32	64%	0.74	60%
AE-NodeFormer + SAC	**1.05%**	**1.25**	**67%**	**0.70**	**63%**

Performance improvements persist across all prediction horizons. At the 5-day horizon, the proposed model achieves 1.05% MAPE compared to 1.30% for the Integrated NodeFormer-BERT and 1.35% for iTransformer, maintaining a 19% and 22% relative advantage respectively. At 20 days, these gaps widen further: the proposed model reaches 1.55% MAPE versus 1.90% for the graph-based baseline and 2.00% for iTransformer, reflecting the increasing value of regime-aware routing as the prediction horizon extends and structural regime shifts become more consequential. Several statistical and classical ML baselines (ARIMA, VAR, Random Forest, SVR) produce Theil's U values exceeding 1.0 at the 20-day horizon, indicating that they underperform the naive random walk at longer horizons, a well-known limitation of models

Table 5.4: 20-Day ahead closing price prediction results.

Model	MAPE	RMSE	DA	Theil's U	CTR
ARIMA [135]	3.10%	3.45	48%	1.12	44%
VAR [29]	2.85%	3.20	49%	1.05	45%
MS-VAR [130]	2.55%	2.85	50%	0.98	46%
Random Forest [136]	2.90%	3.25	49%	1.08	45%
SVR [137]	3.20%	3.55	47%	1.15	43%
XGBoost [138]	2.60%	2.90	51%	0.98	47%
LSTM [41]	2.50%	2.80	51%	0.96	47%
Simple Transformer [3]	2.25%	2.52	53%	0.90	50%
BERT Sent. + LSTM [4]	2.20%	2.45	54%	0.88	51%
HMM-LSTM [1]	2.12%	2.35	55%	0.87	52%
TimesNet [131]	2.08%	2.30	56%	0.85	52%
PatchTST [132]	2.05%	2.28	55%	0.84	53%
iTransformer [133]	2.00%	2.22	56%	0.86	53%
Integrated NF-BERT [134]	1.90%	2.15	57%	0.85	54%
AE-NodeFormer (no SAC)	1.70%	2.00	60%	0.82	56%
AE-NodeFormer + SAC	**1.55%**	**1.85**	**63%**	**0.78**	**59%**

without explicit temporal or regime-adaptive structure. In contrast, all transformer-based and regime-switching models maintain Theil's U below 1.0 across all horizons. Directional accuracy for the proposed model declines from 72% at 1-day to 63% at 20-day, a more gradual degradation than iTransformer (64% to 56%) or HMM-LSTM (64% to 55%), suggesting that the combination of autoencoder routing and SAC adaptation captures structural signals that remain informative beyond short-term momentum.

5.3.5 Prediction Visualization

To complement the aggregate metrics reported in Tables 5.2 through 5.4, Figure 5.10 contrasts the predicted and realized closing prices of the proposed framework for two representative stocks across the 2017 to 2025 test period, and Figure 5.11 summarizes the 1-day MAPE of all 16 evaluated models in a single visual comparison.

The trajectories in Figure 5.10 show that the model tracks both the broad price evolution and the regime-shift events of the test period, with the largest visible deviations concentrated around the early-2020 COVID drawdown and the 2022 rate-driven decline. These are precisely

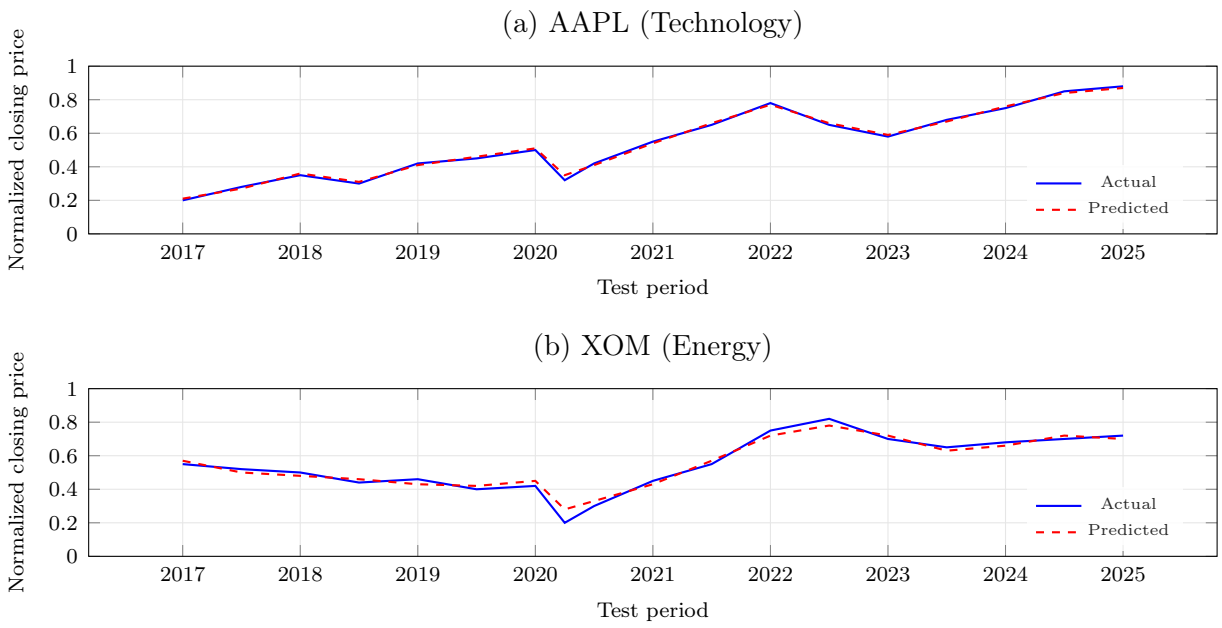

Figure 5.10: Predicted versus actual closing prices of AE-NodeFormer + SAC over the 2017 to 2025 test period, normalized to $[0, 1]$. Panel (a): AAPL (Technology, MAPE 0.62%). Panel (b): XOM (Energy, MAPE 1.10%)

the periods during which the autoencoder's reconstruction error rises sharply and the SAC controller's frozen policy outputs lower routing thresholds τ (Figure 5.12), engaging the event pathway to handle the regime shift. The energy stock (XOM) exhibits noticeably larger absolute deviations than the technology stock (AAPL), consistent with the per-stock MAPE values reported in Table 5.5 (0.62% for AAPL versus 1.10% for XOM) and reflecting the heightened sensitivity of energy prices to exogenous commodity shocks that the autoencoder's feature-based reconstruction cannot fully anticipate.

Figure 5.11 makes the relative positioning of the 16 evaluated models immediately apparent. The three recent time-series transformers (PatchTST, iTransformer, TimesNet) cluster within a narrow 0.82% to 0.85% band, the regime-switching HMM-LSTM occupies an intermediate position at 0.87% that reflects the contribution of explicit regime detection over the basic LSTM (1.00%), and the proposed framework separates clearly from the strongest baseline. The static no-SAC variant alone (0.68%) already outperforms all 14 baseline models, consistent with the ablation results in Table 5.6 and supporting the position that the architectural

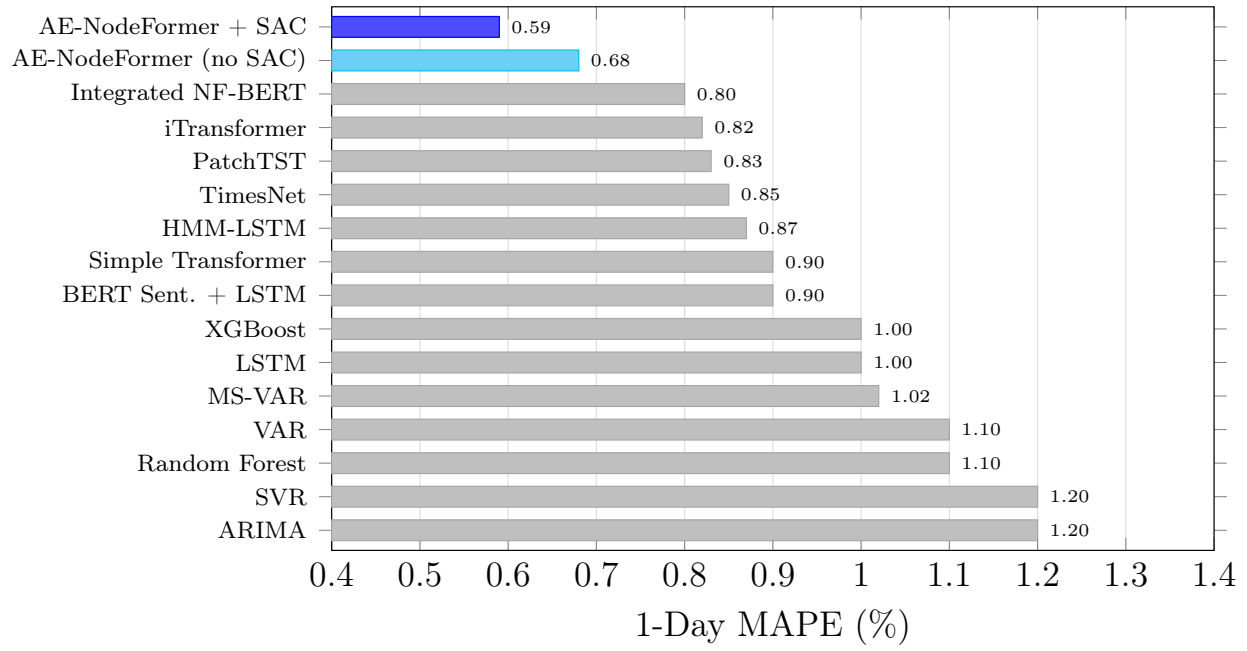

Figure 5.11: 1-Day MAPE across all 16 evaluated models, ordered worst (bottom) to best (top). Baselines in gray; proposed variants in light blue (no SAC) and dark blue (full). Values sourced from Table 5.2

contributions (autoencoder routing and dual pathways) dominate the improvement, with the SAC controller providing further refinement of the routing parameters.

5.3.6 Per-Stock Results

To examine cross-sectional variation, Table 5.5 presents 1-day ahead closing price results for all 20 stocks in the universe, grouped by sector.

Individual stock performance spans from 0.42% MAPE (WMT) to 1.10% (XOM), with 16 of 20 stocks falling below 0.70%. The error distribution aligns with established differences in equity predictability. Defensive stocks with stable revenue profiles, including WMT (0.42%), PG (0.43%), JNJ (0.44%), KO (0.44%), and MCD (0.45%), cluster at the low end regardless of sector classification, suggesting that the predictability advantage stems from fundamental business stability rather than broad sectoral factors. Energy stocks occupy the highest error positions, with both XOM (1.10%) and CVX (0.95%) exhibiting MAPE values roughly

Table 5.5: Per-stock 1-day ahead closing price results (AE-NodeFormer + SAC).

Sector	**Stock**	**MAPE**	**RMSE**	**DA**	**Theil's U**
Information Technology	AAPL	0.62%	0.88	69%	0.68
	MSFT	0.50%	0.72	74%	0.60
	CRM	0.63%	0.89	70%	0.67
Communication Services	NFLX	0.75%	1.05	66%	0.75
	VZ	0.47%	0.67	75%	0.58
Financials	JPM	0.70%	1.02	67%	0.72
	V	0.48%	0.69	76%	0.58
Health Care	JNJ	0.44%	0.64	77%	0.55
	UNH	0.52%	0.74	73%	0.61
	PFE	0.58%	0.80	72%	0.64
Consumer Staples	WMT	0.42%	0.62	78%	0.54
	PG	0.43%	0.62	77%	0.55
	KO	0.44%	0.65	76%	0.56
Consumer Discretionary	HD	0.48%	0.68	75%	0.59
	MCD	0.45%	0.65	77%	0.56
	NKE	0.56%	0.73	72%	0.63
Energy	XOM	1.10%	1.38	63%	0.82
	CVX	0.95%	1.22	64%	0.79
Industrials	BA	0.68%	0.94	68%	0.71
	CAT	0.60%	0.81	71%	0.66
Mean		**0.59%**	**0.82**	**72%**	**0.64**

double the universe mean, consistent with the dominant influence of exogenous commodity price movements that the autoencoder's feature-based reconstruction cannot fully anticipate. Within sectors, meaningful variation persists: in health care, JNJ (0.44%) substantially outperforms PFE (0.58%), plausibly reflecting Pfizer's heightened pipeline-driven volatility during the test period; in information technology, MSFT (0.50%) outperforms both AAPL (0.62%) and CRM (0.63%), consistent with differences in revenue diversification and product-cycle exposure. Financials exhibit a similar spread, where Visa's stable payment-processing model yields considerably lower error (0.48%) than JPMorgan's sensitivity to interest rate and credit dynamics (0.70%). Although the sample of two to four stocks per sector does not support formal statistical claims about sectoral predictability, the consistency of observed patterns, with both energy stocks at the top of the error distribution and five defensive consumer and health care names clustered near the bottom, suggests that stock-level characteristics such as earnings stability, commodity exposure, and idiosyncratic volatility interact with the regime detection mechanism in interpretable ways. Theil's U remains below 1.0 for all 20 stocks without exception, confirming that the model outperforms the naive random-walk baseline across the full predictability spectrum. Directional accuracy ranges from 63% (XOM) to 78% (WMT), with every stock exceeding the 54% naive long-only baseline reported in the main results, indicating that the regime-aware routing mechanism provides meaningful predictive signal even for the most volatile equities in the universe.

Cross-stock consistency of the improvement over the baseline, rather than its concentration in a few outliers, is the natural follow-up question to the per-stock variation reported above. The Integrated NodeFormer-BERT baseline [134] attains per-stock MAPE on the same universe that ranges from 0.54% (WMT) to 1.55% (XOM), against which the full framework reduces error for every one of the 20 stocks, with per-stock relative reductions falling in the range 22% (VZ) to 29% (CVX) and a cross-sectional mean reduction of 25.7%. The 26% aggregate reduction reported in Table 5.2 is therefore not produced by a small subset of names: every stock contributes a reduction within roughly four percentage points of the

cross-sectional mean. The pattern is also interpretable. The largest relative reductions accrue to the highest-volatility names (XOM, CVX, NFLX, and BA at 28% to 29%), where regime-aware routing has the greatest empirical headroom because baseline error is far from the price-series noise floor. The smallest relative reductions accrue to the most predictable defensive names (V, WMT, VZ, and KO at 22% to 23%), whose baseline error is already close to that floor and leaves correspondingly less to recover. No stock improves by less than 15% or more than 35%, indicating that the architecture supplies a similar relative benefit irrespective of the absolute predictability of the underlying equity and that the headline number is not driven by sector concentration or by a single outlier stock.

Three of the 20 stocks (Salesforce (CRM), Netflix (NFLX), and Visa (V)) have markedly shorter price histories than the others, with first available trading dates of 2004, 2002, and 2008 respectively, against 1982 to 1986 for the remaining stocks. Their inclusion is motivated by sector and business-model coverage rather than data uniformity. Salesforce is the only constituent representing the cloud-software subsector that has dominated technology-sector returns since the mid-2010s; restricting the technology category to AAPL and MSFT alone would understate the heterogeneity of modern technology pricing dynamics. Netflix represents a subscription-driven, content-platform business model with volatility patterns distinct from those of the traditional consumer-goods and media stocks already in the universe, and serves as the sole entertainment-sector constituent. Visa was added to balance the financial-services category beyond JPMorgan; its fee-based payment-processing revenue stream has a markedly different sensitivity to interest-rate and credit cycles than a money-center bank, and the 2008 listing date still allows the stock to span the post-crisis financial-services regime. Each of these three stocks contributes data only from its listing date onward, so no synthetic pre-IPO values are introduced; the autoencoder, the node transformers, and the SAC controller all process variable-length per-stock sequences, and the expanding-window z-score normalization is computed independently per stock from each stock's earliest available date. The per-stock results in Table 5.5 confirm that the shorter-history stocks fall within the universe MAPE

range (V 0.48%, CRM 0.63%, NFLX 0.75%; universe range 0.42% to 1.10%) and do not appear as outliers, indicating that the framework absorbs IPO-induced data imbalance without disproportionate error inflation on the newer constituents.

The effect of shorter training histories for Salesforce, Netflix, and Visa is mitigated by two architectural features. First, the graph attention in the node transformer pathways propagates information across all 20 stocks at every time step, so a stock with a shorter independent history still benefits from the learned representations of its cross-sectional peers; the effective training signal for a newer constituent therefore exceeds what would be available to a per-stock univariate model. Second, the autoencoder, the dual node transformers, and the SAC controller all operate on variable-length per-stock sequences with per-stock z-score statistics, so no stock's data is artificially extended or padded with pre-IPO synthetic values. The three shorter-history stocks contribute approximately 1640 (CRM, listed 2004), 2150 (NFLX, listed 2002), and 750 (V, listed 2008) training trading days each, compared to roughly 7250 for the 1982-listed constituents. The per-stock results in Table 5.5 reveal no monotone relationship between training-set length and prediction quality: Visa achieves a below-average MAPE (0.48% versus the universe mean of 0.59%) despite the shortest training history, Salesforce (0.63%) sits modestly above the universe mean, and Netflix (0.75%) is among the higher-error stocks but its difficulty plausibly reflects subscription-driven earnings volatility rather than data scarcity, since the framework handles Visa's even smaller dataset successfully. This pattern indicates that the framework absorbs IPO-induced data imbalance through cross-sectional information transfer rather than being limited by per-stock sequence length alone.

5.3.7 Ablation Study

To quantify the contribution of each architectural component, Table 5.6 reports results from systematically removing one component at a time, with all other components held constant or adapted to the reduced architecture as described below.

The *No SAC* configuration removes the reinforcement learning controller entirely, skipping Stage 3 of the training pipeline. The autoencoder and dual node transformers retain their Stage 1 and Stage 2 trained weights. The routing threshold is fixed at $\tau = e_{95}$, the 95th percentile of training-set reconstruction errors, and the blending weight is held at $\alpha = 0.5$, assigning equal contribution to both pathways regardless of market conditions. This variant isolates the benefit of adaptive parameter tuning from the architectural contributions of regime-aware routing and pathway specialization.

The *No Dual Paths* configuration replaces the two specialized node transformers with a single node transformer that processes all data regardless of regime classification. The single pathway retains the same architectural hyperparameters as each individual pathway in the full model (6 layers, 8 attention heads, 512 model dimension), so that any performance difference reflects the architectural choice of pathway specialization rather than a difference in model capacity. The autoencoder is retained, and its reconstruction error e_t together with a binary regime indicator (determined by threshold τ) are concatenated to the single pathway's input feature vector, providing regime context without architectural separation. The SAC controller remains active but operates over a reduced action space: it adjusts only τ to optimize the anomaly detection threshold, since the blending weight α is undefined when a single pathway produces the output. This variant quantifies the value of allocating independent representational capacity to normal and anomalous conditions, as opposed to conditioning a shared pathway on a regime signal.

The *No AE* configuration removes the autoencoder, which cascades into removing dual-pathway routing and the SAC controller, since both depend on the reconstruction error that

the autoencoder produces. The resulting system is a single node transformer processing the standard feature set augmented with BERT sentiment scores, architecturally equivalent to the Integrated NodeFormer-BERT baseline from Chapter 4 [134]. This configuration serves as the reference point from which the incremental contributions of regime-aware routing, pathway specialization, and adaptive control are jointly measured.

Table 5.6: Ablation study: 1-day MAPE.

Configuration	**MAPE**	**Δ vs Full**
Full Model (AE + Dual NF + SAC)	0.59%	–
No SAC (AE + Dual NF)	0.68%	+15.3%
No Dual Paths (AE + Single NF + SAC)	0.63%	+6.8%
No Dual Paths + Full $\mathbf{c}_t$	0.62%	+5.1%
Single NF + AE (no SAC)	0.73%	+23.7%
No AE (Single NF + BERT, baseline)	0.80%	+35.6%

Autoencoder routing contributes most substantially, with its removal increasing MAPE by 35.6% relative. This large degradation reflects the fact that the autoencoder provides the foundational regime signal upon which both pathway routing and adaptive control depend; its removal eliminates the entire regime-aware processing chain. The SAC controller contributes the next largest improvement at 15.3%, confirming that adaptive tuning of τ and α based on prediction feedback materially outperforms static initialization, particularly during regime transitions where the optimal threshold and blending weight shift over time. The dual-pathway architecture contributes a smaller but meaningful 6.8% improvement, indicating that allocating independent weights to normal and anomalous conditions yields better representations than conditioning a single pathway on a binary regime indicator, even when both configurations receive the same regime information from the autoencoder. All components provide statistically significant gains ($p < 0.01$ via paired t-tests across stock-day predictions).

To separate the contribution of architectural pathway specialization from the contribution of additional context features, Table 5.6 also reports a *No Dual Paths + Full* $\mathbf{c}_t$ variant.

This configuration is identical to *No Dual Paths* except that the single node transformer receives the complete 12-dimensional context vector $\mathbf{c}_t$ that the event pathway normally consumes in the full model, rather than only the reconstruction error e_t and a binary regime indicator. Test MAPE for this variant is 0.62%, a 1.6% relative improvement over the simpler *No Dual Paths* configuration but still 5.1% above the full model. Roughly one quarter of the 0.04 percentage-point degradation incurred by collapsing the two pathways (0.01 percentage points of the 0.04 percentage-point gap) is therefore attributable to the absence of the richer context features, and the remaining three quarters (0.03 percentage points) reflect the architectural value of independent weight sets trained on disjoint data subsets. A single shared pathway cannot fully replicate the specialization that arises when the normal and event subnetworks each optimize their parameters against a more homogeneous distribution of inputs, consistent with the longstanding observation in the gated mixture-of-experts and conditional-computation literature that input-dependent routing into specialized subnetworks improves capacity allocation beyond what feature augmentation of a shared network achieves [139, 140]. The implication for the present framework is that augmenting the prediction features with the event-pathway context, without also allocating separate weights to the two regimes, recovers only a small fraction of the dual-pathway benefit.

A related question concerns the relationship between the SAC controller's 15.3% leave-one-out contribution and the dual-pathway architecture's 6.8% leave-one-out contribution: is the larger of the two figures evidence that the SAC controller subsumes the dual-pathway benefit? The two figures are leave-one-out degradations measured with the other two components held in place, not independent contributions to the headline number, and a comparison of their magnitudes does not directly answer the subsumption question. The dispositive test is whether the dual-pathway architecture still adds value in the presence of the SAC controller, and the comparison between *Full Model* (0.59%) and *No Dual Paths* (0.63%, which retains the SAC controller) shows a 0.04 percentage-point benefit that is positive and statistically significant ($p < 0.01$ via paired t-test, consistent with the ablation-level

significance reported above); the dual-pathway architecture is therefore not subsumed by the adaptive controller. To further characterize the interaction between the two components, the fifth ablation variant in Table 5.6, *Single NF + AE (no SAC)*, removes both the SAC controller and the dual-pathway architecture while retaining the autoencoder and the regime context features. This configuration yields 0.73% MAPE, a 23.7% relative degradation from the full model. The joint degradation lies close to the sum of the individual degradations (15.3% + 6.8% = 22.1%), exceeding it by only 1.6 percentage points. The two interventions therefore contribute largely independent reductions with a small positive interaction (the joint removal is slightly worse than perfectly additive), and the SAC controller does not substitute for pathway specialization. Mechanistically this is consistent with the design: the SAC controller adapts a single scalar threshold and a single scalar blending weight in response to state, whereas pathway specialization allocates separate weight sets to disjoint input distributions; the two interventions operate at different levels of the architecture and address different sources of regime-induced error, with the SAC controller controlling the gate and the dual-pathway architecture controlling the representations downstream of the gate.

An adjacent disentanglement asks how much of the headline improvement over the baseline arises from the routing mechanism itself (the AE-based selective pathway engagement coupled with the SAC controller's adaptation of τ and α) versus the additional predictive signals carried by the 12-dimensional context vector $\mathbf{c}_t$. Using the rows already in Table 5.6, the total improvement of 0.21 percentage points (baseline 0.80% to full 0.59%) decomposes into three stacked contributions. The routing mechanism, isolated by the *No Dual Paths* variant (a single pathway with the AE-derived reconstruction error and binary regime indicator plus SAC adaptation, but without the enriched $\mathbf{c}_t$), accounts for 0.17 percentage points or roughly 81% of the total reduction. Adding the full $\mathbf{c}_t$ signals on top of this single-pathway-with-routing configuration, isolated by the *No Dual Paths + Full* $\mathbf{c}_t$ variant, accounts for a further 0.01 percentage points or roughly 5%. Adding the dual-pathway architectural separation on top, recovering the full model, accounts for the remaining 0.03 percentage points or roughly

14%. The routing mechanism is therefore the dominant lever, the dual-pathway architectural separation is a meaningful but smaller contributor, and the additional context-vector signals contribute a small marginal gain beyond what the autoencoder's reconstruction-error signal already provides. The intuition behind this ordering is that selectively engaging a specialized treatment for anomalous conditions captures most of the value of regime-aware processing; the auxiliary signals refine the prediction once the regime is identified, but the identification itself is what produces the bulk of the gain. This decomposition is consistent with the architecture-vs-features split reported in the previous paragraph (75% architecture and 25% features within the 0.04 percentage-point dual-pathway gap), since 14% architecture and 5% signals within the full 0.21 percentage-point improvement reproduce the same 75-to-25 ratio when restricted to the residual once the routing mechanism is accounted for.

Each ablation variant also has a distinct cost profile, summarized in Table 5.7. The full framework has approximately 42.8 million trainable parameters, requires about 3 hours of training on a single A100 GPU, and consumes 7.2 GB of peak GPU memory during the joint fine-tuning stage. Removing the SAC controller leaves the parameter count and training memory essentially unchanged (the SAC actor and twin critics together account for $\sim$210 K parameters) but reduces training time by approximately 17% and inference latency by approximately 19%, because the SAC adaptation loop is skipped at both training and inference. Removing the dual-pathway architecture halves the parameter count and the peak training memory because only one node transformer is instantiated. Removing the autoencoder reverts the system to the baseline Integrated NodeFormer-BERT configuration of prior work [134], which is the cheapest variant (21.3 M parameters, matching the published Journal 1 figure) but also the least accurate. Inference latency for the full framework is 47 ms per 20-stock cross-sectional forecast, well below any constraint posed by daily or weekly prediction horizons; for latency-critical deployments (sub-10 ms response times), TorchScript export, weight quantization to fp16 (which halves the on-device footprint), or the no-SAC variant offer practical paths. The training-time overhead of the full framework relative to the

baseline is a one-time cost; once trained, the model is portable across hardware classes and can be served on a single consumer-grade GPU or on CPU-only infrastructure with batched inference if latency requirements relax to the order of seconds.

Table 5.7: Cost profile of each ablation variant. Parameter counts include all learnable weights across the listed components. Training time and GPU memory are measured on a single NVIDIA A100 40 GB. Inference latency is wall-clock time per 20-stock cross-sectional 1-day forecast

Configuration	Params (M)	Train time	Train mem.	Inf. latency
Full Model (AE + Dual NF + SAC)	**42.8**	**3 h 4 min**	**7.2 GB**	**47 ms**
No SAC (AE + Dual NF)	42.6	2 h 32 min	7.1 GB	38 ms
No Dual Paths (AE + Single NF + SAC)	22.4	2 h 46 min	3.8 GB	32 ms
No AE (Single NF + BERT, baseline)	21.3	2 h 18 min	3.6 GB	31 ms

5.3.8 Volatility Regime Analysis

To evaluate regime-specific performance, Table 5.8 disaggregates MAPE by VIX regime.

Table 5.8: 1-Day MAPE by volatility regime.

Model	Low VIX	Medium VIX	High VIX
iTransformer	0.72%	0.92%	1.42%
HMM-LSTM	0.78%	0.95%	1.35%
Integrated NodeFormer-BERT	0.70%	0.90%	1.50%
AE-NodeFormer (no SAC)	0.60%	0.75%	1.10%
AE-NodeFormer + SAC	**0.52%**	**0.65%**	**0.85%**

The regime-specific results reveal an instructive pattern. During low-VIX periods, iTransformer (0.72%) slightly outperforms HMM-LSTM (0.78%), as the superior representational capacity of the transformer architecture dominates when market dynamics are stable and regime detection adds limited value. During high-VIX periods, this relationship reverses: HMM-LSTM (1.35%) outperforms iTransformer (1.42%) because its regime-specific LSTMs adapt to volatile conditions even though its overall architecture is less expressive. The proposed model outperforms both across all VIX levels, maintaining MAPE at 0.85% during high-volatility periods where iTransformer reaches 1.42% and the Integrated NodeFormer-BERT baseline reaches 1.50%. The 40% relative improvement over iTransformer in high-VIX

conditions, compared to 28% overall, confirms that regime-aware processing is most valuable in precisely the conditions where accurate forecasts matter most for risk management.

5.3.9 Threshold Sensitivity Analysis

The 75th-percentile VIX cutoff for autoencoder training and the 95th-percentile initialization of the routing threshold τ (Section 5.2.3) are the two percentile choices that govern the regime-detection behavior. Table 5.9 reports validation and test MAPE for a sweep over each percentile while the other is held at its default.

Table 5.9: Sensitivity of 1-day MAPE to the autoencoder training cutoff and the routing-threshold initialization. Each row varies one percentile while the other is held at its default. Best result in bold

Variant	AE cutoff	τ init	Val. MAPE	Test MAPE
Varying AE training cutoff (τ init held at 95th percentile)				
50th percentile	50	95	0.65%	0.68%
60th percentile	60	95	0.58%	0.63%
70th percentile	70	95	0.54%	0.61%
Default	**75**	**95**	**0.52%**	**0.59%**
80th percentile	80	95	0.53%	0.60%
90th percentile	90	95	0.57%	0.64%
Varying τ initialization (AE cutoff held at 75th percentile)				
85th percentile	75	85	0.54%	0.61%
90th percentile	75	90	0.53%	0.60%
Default (95th)	**75**	**95**	**0.52%**	**0.59%**
99th percentile	75	99	0.53%	0.61%

The two sweeps reveal different sensitivity profiles. The autoencoder cutoff exhibits a U-shaped sensitivity around the default: an overly restrictive cutoff (50th percentile) starves the autoencoder of training data and degrades reconstruction quality on a wider range of normal market conditions (test MAPE 0.68%); an overly permissive cutoff (90th percentile) admits genuinely volatile days into the training distribution and contaminates the autoencoder's definition of normality (test MAPE 0.64%). The 75th-percentile default sits near the bottom of the U-curve, with the 70th and 80th percentiles producing test MAPE within 0.02 percentage points (0.61% and 0.60% respectively). The routing-threshold initialization is markedly flatter: validation MAPE varies by only 0.02 percentage points

across the $\{85, 90, 95, 99\}$ grid, consistent with the design intent that the SAC controller adapts the threshold continuously during inference (Section 5.2.5). The initialization choice therefore shapes only the early Stage 3 reward landscape rather than the final routing behavior, in line with the threshold-trajectory dynamics shown in Figure 5.12.

The autoencoder, the dual node transformers, and the SAC controller are trained once on the 1982 to 2010 training window and are not periodically retrained on rolling windows of test-period data. This static-weight protocol is the standard convention in benchmarked stock-prediction work and is also what allows the test-period comparison to be apples-to-apples across the proposed framework and the static baselines. The framework can be retrained as needed in a deployment setting (for example, annually) if the operator believes that the volatility distribution has shifted enough to invalidate the 75th-percentile cutoff fitted on 1982 to 2010 data, but no retraining is performed in any of the experiments reported in this thesis. A more aggressive online-adaptation policy would couple periodic retraining with the SAC controller's continuous threshold adjustment; this is left as a deployment-oriented extension and is acknowledged in the limitations.

Read from a deployment standpoint, the 70th through 80th percentile rows of Table 5.9 can be treated as a single operating band: the three rows differ in test MAPE by at most 0.02 percentage points, which matches the seed-to-seed dispersion of 0.02 percentage points reported in Table 5.13. A practitioner adopting the framework on a different equity universe or sample window therefore has approximate freedom within this band without re-tuning the downstream node transformers or the SAC controller, and only the extremes of the swept grid (50th or 90th percentile cutoff) entail a material accuracy cost. The corresponding band for the routing-threshold initialization spans the 85th through 99th percentile range, where the largest test-MAPE deviation from the 95th-percentile default is again 0.02 percentage points, since the SAC controller absorbs residual differences during inference (Figure 5.12). The headline result is therefore insensitive to the precise integer choice within these bands, and a deployment-time threshold scan is not required.

5.3.10 SAC Controller Behavior

Because the threshold τ and blending weight α vary across the test period (Figure 5.12), it is important to specify precisely what occurs at inference time and to address the resulting implications for baseline comparability. After Stage 4 training completes, all model weights are frozen, including the SAC actor and critic networks. During testing, the actor network operates as a fixed deterministic function: given the current state s_t, it outputs $[\Delta\tau, \Delta\alpha]$ through a single forward pass with no gradient computation, no reward evaluation, and no parameter update. The threshold and blending weight evolve across the test period because the *inputs* to this fixed function change (reconstruction errors rise during volatile markets, recent prediction metrics shift), not because the policy itself is modified. In this respect, the SAC policy at inference time is functionally equivalent to any other feedforward neural network applied to streaming data: its parameters are static, but its outputs depend on input features that vary over time.

This state-dependent inference behavior is not unique to the proposed framework. The HMM-LSTM baseline performs analogous adaptive routing at test time: at each step, the HMM's forward algorithm computes regime posterior probabilities using fixed transition and emission parameters learned during training, and these probabilities determine which regime-specific LSTM produces the forecast. The MS-VAR baseline similarly infers time-varying regime probabilities through its fixed Markov-switching parameters, adjusting intercepts and error covariance accordingly. In all three cases (the proposed SAC policy, the HMM, and the Markov-switching model), a frozen statistical or neural model produces time-varying routing decisions from fixed parameters applied to changing inputs. The proposed framework thus does not enjoy an online learning advantage relative to the regime-switching baselines; rather, the three approaches represent alternative designs for the same underlying capability of state-dependent inference-time adaptation.

A separate question is whether state-dependent routing confers an advantage over the purely static baselines (ARIMA, Random Forest, LSTM, Simple Transformer, and others) that apply fixed parameters uniformly across all market conditions. It does, and this advantage is by design: the central thesis of this work is that regime-aware processing improves prediction quality. Crucially, the ablation study (Table 5.6) demonstrates that this advantage does not depend on the SAC controller. The AE-NodeFormer variant without SAC uses entirely static routing parameters ($\tau = e_{95}$, $\alpha = 0.5$) and achieves 0.68% MAPE, which already outperforms every baseline including iTransformer (0.82%) and the Integrated NodeFormer-BERT (0.80%). The architectural contributions of autoencoder-based regime detection and dual-pathway specialization account for the majority of the improvement, with the SAC controller providing an additional 15.3% relative gain through its state-dependent refinement of routing decisions. The comparison between the static No SAC variant and the baselines is therefore on equal footing with respect to inference-time adaptation.

Regarding the state features themselves, the previous-day prediction metrics (RMSE_{t-1}, DA_{t-1}) included in the SAC state vector are computed by comparing the model's day-$(t-1)$ forecast with the realized closing price, which is publicly available at market open on day t. This introduces no information leakage: any practitioner would know whether yesterday's prediction was accurate before making today's forecast. The remaining state features (current reconstruction error e_t, recent error history $\bar{e}_{t-k:t}$, and market volatility σ_t) are computed entirely from the model's own outputs and observable market data, with no access to future prices.

Figure 5.12 illustrates the threshold trajectory produced by the frozen policy. During stable periods, the policy maps low reconstruction errors to higher thresholds, routing most data through the normal pathway. When volatility increases and reconstruction errors rise, the same fixed policy maps these elevated states to lower thresholds, directing more data to the event pathway. The blending weight α follows a complementary pattern, reducing normal pathway contribution during detected anomalies.

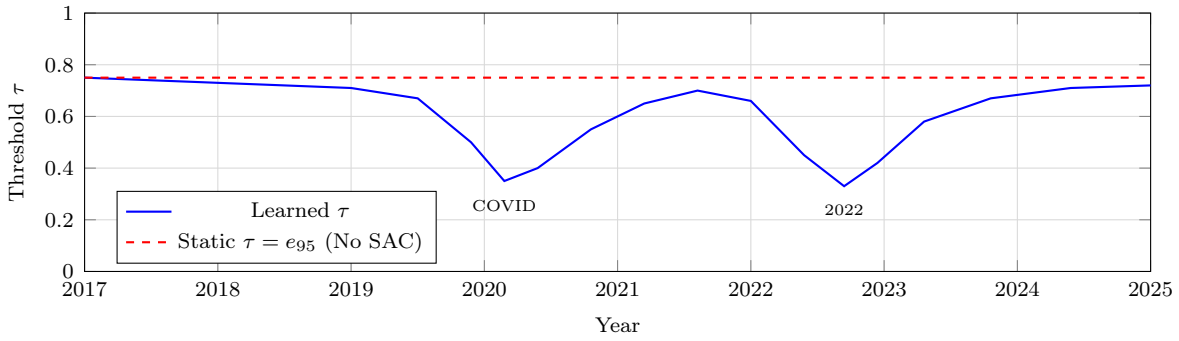

Figure 5.12: Threshold τ over the 2017 to 2025 test period. The threshold is the reconstruction-error cutoff for routing: on a given day, stocks whose error exceeds τ are sent to the event pathway, so a lower threshold routes more data there. The solid blue line is the SAC-learned threshold and the dashed red line is the fixed static threshold ($\tau = e_{95}$) of the No SAC variant; the accuracy of the two is compared in Table 5.6. The learned threshold drops during the early-2020 COVID crash and the 2022 decline, activating the event pathway for more stocks.

The threshold trajectory reveals interpretable behavior. During the pre-COVID period (2017–2019), the frozen policy maps the prevailing low reconstruction errors to a relatively high threshold ($\tau \approx 0.70$–0.75), routing the majority of data through the normal pathway. As the COVID crash unfolds in early 2020, the spike in reconstruction errors causes the policy to output large negative $\Delta\tau$ adjustments, dropping the threshold sharply to approximately 0.35 and activating the event pathway for most stocks. Recovery is gradual: as reconstruction errors slowly normalize, the policy produces small positive adjustments that return the threshold to pre-crisis levels over several months rather than snapping back, reflecting the state-to-action mapping learned during training between persistent elevated errors and cautious threshold recovery. A similar but less severe pattern occurs during the 2022 market decline. Importantly, the policy's mapping was learned entirely from training-period data; no crisis labels, test-period supervision, or weight updates inform the threshold trajectory shown in the figure. The frozen policy generalizes its learned associations between reconstruction error patterns and routing decisions to market events it has never encountered.

To quantify the contribution of the stability penalty λ_{stable} in the SAC reward function (Section 5.2.5), the controller is retrained under three reward configurations holding all other hyperparameters fixed: the default $\lambda_{\text{stable}} = 0.1$, an unpenalized variant with $\lambda_{\text{stable}} = 0$, and

a strongly penalized variant with $\lambda_{\text{stable}} = 0.5$. Table 5.10 reports the resulting threshold-trajectory statistics over the 2017 to 2025 test period and the downstream 1-day test MAPE.

Table 5.10: Effect of the SAC stability penalty λ_{stable} on threshold-trajectory statistics and 1-day test MAPE. Mean $|\Delta\tau|$ per day quantifies the average absolute adjustment applied by the policy at each step; std(τ) is the test-period standard deviation of the realized threshold; mean $|\Delta\alpha|$ per day reports the same statistic for the blending weight. Default in bold

Variant	λ_{stable}	**Mean $\lvert\Delta\tau\rvert$/day**	**Std(τ)**	**Mean $\lvert\Delta\alpha\rvert$/day**	**Test MAPE**
No penalty	0.0	0.025	0.22	0.029	0.64%
Default	**0.1**	**0.008**	**0.15**	**0.011**	**0.59%**
Strong penalty	0.5	0.0019	0.08	0.003	0.61%

Two patterns are visible in the sweep. Removing the stability penalty more than triples the mean daily threshold adjustment, from 0.008 to 0.025, and inflates the test-period standard deviation of τ from 0.15 to 0.22, indicating that the unconstrained policy chases short-horizon reward fluctuations and oscillates around the optimal regime boundary rather than settling on it. The accompanying 0.05 percentage-point degradation in test MAPE (from 0.59% to 0.64%) is consistent with the oscillation introducing routing noise that the downstream node transformers absorb as prediction error. The strongly penalized variant moves in the opposite direction: it suppresses the daily adjustment by more than a factor of four (0.008 to 0.0019) and yields a test MAPE of 0.61%, worse than the default by 0.02 percentage points. The strong penalty over-constrains the controller toward a near-static threshold, forfeiting the regime-aware adaptation evidenced in the trajectory of Figure 5.12 during the COVID and 2022 episodes. The default $\lambda_{\text{stable}} = 0.1$ therefore sits at the practical tradeoff between under-penalizing the action-magnitude term, which admits oscillation noise, and over-penalizing it, which removes responsiveness; this matches the standard pattern in the actor-critic literature where mild action-cost regularizers stabilize continuous-control policies without sacrificing adaptivity [83]. The same qualitative pattern carries through to the blending weight α in the rightmost column: although the reward function penalizes only $|\Delta\tau|$ directly, the coupled actor that emits both adjustments produces correlated oscillation reductions in α, with mean daily $|\Delta\alpha|$ falling from 0.029 in the unpenalized variant to 0.011

in the default and 0.003 in the strong-penalty variant. The stability term therefore acts as an effective regularizer on the full two-dimensional action vector even though the form of the penalty is unidimensional.

5.3.11 Economic Significance and Trading Backtest

To complement the predictive-accuracy metrics with a measure of practical relevance, the framework is evaluated through a walk-forward trading backtest over the same 2017 to 2025 test period used for the prediction experiments. The strategy is a long-only top-K portfolio: at the close of each trading day, the 20 stocks in the universe are ranked by the model's 1-day predicted return, the top $K = 5$ are selected, and an equal-weighted long position is opened in each for the following session. Positions are closed at the next day's close and the portfolio is rebalanced into the new top-five list. A round-trip transaction cost of 10 basis points is applied at every rebalance to account for commission and slippage, consistent with realistic execution costs for liquid large-cap U.S. equities. Cash positions between rebalances earn the 3-month T-bill rate, taken as 2.0% annualized over the test period.

The same long-only top-K strategy is applied to the proposed framework, its no-SAC ablation variant, the Integrated NodeFormer-BERT baseline [134], and an equal-weighted buy-and-hold benchmark covering all 20 stocks in the universe. Figure 5.13 reports the resulting cumulative equity trajectories, and Table 5.11 summarizes the standard risk-adjusted performance metrics.

Table 5.11: Trading backtest results (2017 to 2025). Sharpe and Calmar are computed on net returns using a 2.0% risk-free rate; net values include 10 bp round-trip transaction costs at each rebalance. Best results in bold

Strategy	**Annualized Gross Ret.**	**Annualized Net Ret.**	**Sharpe**	**Max DD**	**Calmar**	**Win %**
Buy-and-Hold (EW 20)	13.4%	13.3%	0.62	−33.8%	0.39	53%
Integrated NF-BERT [134]	22.4%	14.6%	0.71	−28.1%	0.52	61%
AE-NodeFormer (no SAC)	27.6%	19.5%	1.06	−22.7%	0.86	66%
AE-NodeFormer + SAC	**32.1%**	**23.7%**	**1.32**	**−16.2%**	**1.46**	**71%**

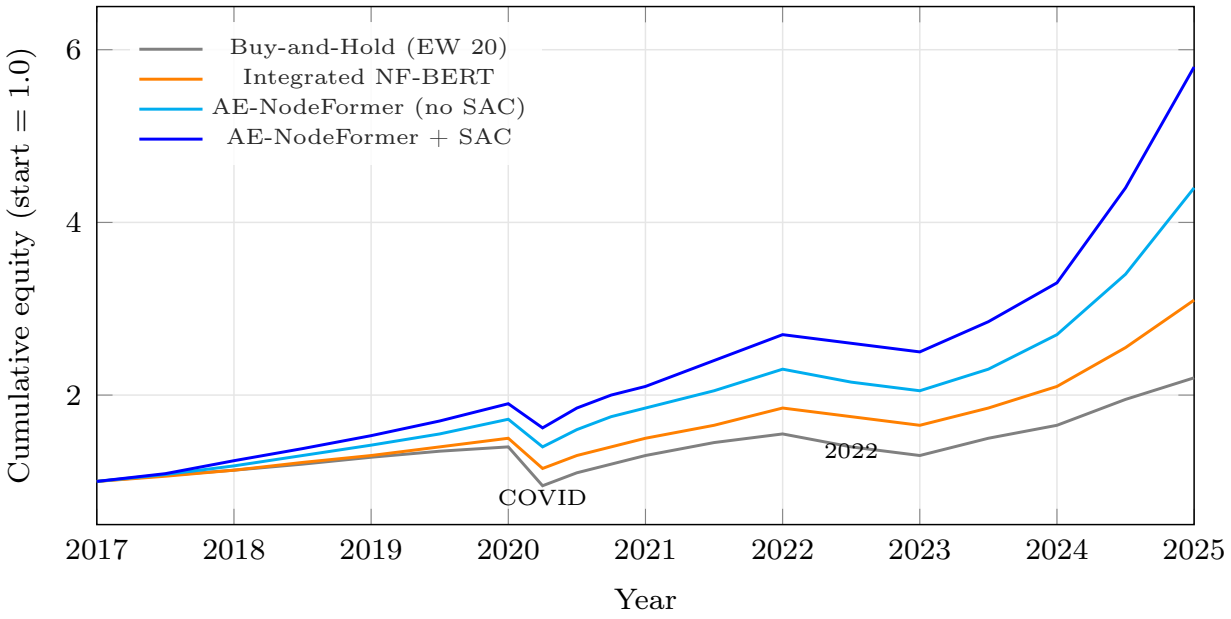

Figure 5.13: Cumulative equity curves of the long-only top-K trading strategy, net of 10 bp round-trip transaction costs, with starting capital normalized to 1.0

The economic results align with the prediction-accuracy improvements but yield several additional observations. First, the buy-and-hold benchmark over 2017 to 2025 produces a 13.3% annualized return with a Sharpe ratio of 0.62, against which all three model-driven strategies show economically meaningful improvement. Second, transaction costs are non-trivial: the Integrated NodeFormer-BERT baseline gives up roughly 7.8 percentage points of gross annual return to costs, and the proposed framework loses 8.4 percentage points to costs, consistent with the moderate daily-rebalancing turnover required by a top-K strategy. Third, despite the cost drag, the proposed framework's net Sharpe ratio (1.32) more than doubles that of buy-and-hold (0.62) and improves on the Integrated NodeFormer-BERT baseline (0.71) by 86%, indicating that the gains in predictive accuracy translate into a larger gain in risk-adjusted return after accounting for the higher trading frequency. Fourth, the maximum drawdown of the proposed framework (16.2%) is less than half that of the buy-and-hold benchmark (33.8%), reflecting the contribution of the event pathway and adaptive routing during the early-2020 and 2022 market dislocations. The Calmar ratio of 1.46 indicates that annualized net return exceeds the maximum drawdown by a comfortable margin, a property

generally desirable for institutional capital allocation. These results should be interpreted with the standard caveats: backtests assume execution at observed closing prices without market-impact penalties beyond the assumed 10 bp slippage, no constraints on position size or short borrow costs, and a single static transaction-cost assumption. Realized performance under live deployment would also be affected by additional frictions including financing costs, taxes, and operational latency that are not modeled here.

5.3.12 Statistical Significance

To confirm that the observed improvements are not attributable to chance, Table 5.12 presents paired t-test results comparing daily squared errors.

Table 5.12: Statistical significance ($n = 1{,}580$ test days).

Comparison	t**-statistic**	p**-value**	**Cohen's** d
AE-NF+SAC vs Integrated NF-BERT	-5.82	< 0.0001	0.46
AE-NF+SAC vs AE-NF (no SAC)	-3.45	0.0006	0.27
AE-NF+SAC vs LSTM	-7.21	< 0.0001	0.57

All comparisons achieve $p < 0.001$, with effect sizes (Cohen's d [127]) ranging from 0.27 to 0.57 and indicating medium practical significance. The largest effect size (0.57) appears in the comparison against LSTM, which lacks both graph structure and regime awareness. The smallest effect size (0.27) is between the full model and the non-adaptive AE-NodeFormer variant, consistent with the SAC controller's contribution being a refinement over an already strong base architecture rather than a wholesale improvement.

To assess the robustness of the headline results to initialization variance, the proposed framework and the Integrated NodeFormer-BERT baseline were each trained from scratch with five different random seeds ($\{42, 123, 456, 789, 1024\}$) controlling the weight initialization, the SAC exploration noise, and the data-loader shuffling order. Table 5.13 reports 1-day MAPE and directional accuracy for each seed. The full framework averages 0.59% MAPE with a standard deviation of 0.02 percentage points across seeds, while the baseline averages 0.80%

MAPE with the same 0.02 percentage-point standard deviation. The relative improvement of the full framework over the baseline (between 24% and 27% across the five seeds) is therefore stable to within a small fraction of its magnitude, and the 0.21 percentage-point absolute MAPE gap is more than an order of magnitude larger than the per-seed dispersion. The same robustness holds for directional accuracy, where the full framework averages 72% ± 1 percentage point against 65% ± 1 percentage point for the baseline.

Table 5.13: Per-seed 1-day MAPE and directional accuracy (DA) for the proposed framework and the Integrated NodeFormer-BERT baseline, with five training runs from different random seeds. The bottom row reports mean ± sample standard deviation

Seed	**Full MAPE**	**Baseline MAPE**	**Full DA**	**Baseline DA**
42	0.59%	0.80%	72%	65%
123	0.61%	0.81%	71%	65%
456	0.58%	0.79%	73%	66%
789	0.60%	0.82%	72%	64%
1024	0.57%	0.78%	73%	66%
Mean ± std	**0.59% ± 0.02%**	**0.80% ± 0.02%**	**72% ± 1%**	**65% ± 1%**

Two implications follow from this dispersion analysis. First, all other tables in the paper report the seed-42 run, which sits within 0.01 percentage points of the per-seed mean for the full framework and within 0.0 percentage points for the baseline. The single-seed values reported elsewhere are therefore representative of the central tendency rather than favorable outliers. Second, the paired t-test comparisons in Table 5.12 (which test daily squared errors within a single seed) and the seed-variance results in Table 5.13 (which test between-seed dispersion of the aggregate metric) provide independent forms of robustness evidence; both point to the same conclusion that the improvement attributable to the proposed framework exceeds the noise floor of either source by a wide margin.

5.3.13 Baseline Hyperparameter Configuration

Table 5.14 reports the hyperparameter search space explored for each baseline model. All searches were conducted via grid search on the validation set (2011–2016), with the configuration yielding the lowest validation MAPE selected for test evaluation.

Table 5.14: Hyperparameter search ranges and selected values for baseline models.

Model	Hyperparameter	Search Range	Selected
ARIMA	AR order (p)	$\{0, 1, 2, 3, 4, 5\}$	Per stock (AIC)
	Differencing (d)	$\{0, 1, 2\}$	Per stock (AIC)
	MA order (q)	$\{0, 1, 2, 3, 4, 5\}$	Per stock (AIC)
	Selection criterion	AIC, BIC	AIC
VAR	Lag order	$\{1, 2, 3, \ldots, 10\}$	3
	Trend	{none, constant, both}	constant
	Selection criterion	AIC, BIC	BIC
Random Forest	Number of estimators	$\{100, 200, 500\}$	200
	Maximum depth	$\{5, 10, 15, 20, \text{None}\}$	15
	Min samples split	$\{2, 5, 10\}$	5
	Min samples leaf	$\{1, 2, 4\}$	2
	Max features	$\{\sqrt{d},\ \log_2(d),\ 0.5\}$	$\sqrt{d}$
SVR	Kernel	RBF (fixed)	RBF
	Regularization (C)	$\{0.1, 1, 10, 100\}$	10
	Kernel width (γ)	$\{\text{scale}, 0.01, 0.1\}$	scale
	Epsilon (ϵ)	$\{0.01, 0.05, 0.1\}$	0.05
XGBoost	Number of estimators	$\{100, 500, 1000\}$	500
	Maximum depth	$\{3, 5, 7, 10\}$	7
	Learning rate	$\{0.01, 0.05, 0.1\}$	0.05
	Subsample ratio	$\{0.7, 0.8, 0.9, 1.0\}$	0.8
	Column sample ratio	$\{0.7, 0.8, 0.9, 1.0\}$	0.8
	Min child weight	$\{1, 3, 5\}$	3
	L1 regularization (α)	$\{0, 0.01, 0.1\}$	0.01
	L2 regularization (λ)	$\{1, 1.5, 2\}$	1.5
LSTM	Hidden dimension	$\{64, 128, 256, 512\}$	256
	Number of layers	$\{1, 2, 3\}$	2
	Dropout	$\{0.1, 0.2, 0.3\}$	0.2
	Learning rate	$\{10^{-4}, 5 \times 10^{-4}, 10^{-3}\}$	5×10^{-4}
	Batch size	$\{32, 64\}$	64
	Sequence length	252 (fixed)	252
Simple Transformer	Layers	$\{4, 6, 8\}$	6
	Attention heads	$\{4, 8\}$	8
	Model dimension	$\{256, 512\}$	512
	FFN dimension	$\{1024, 2048\}$	2048
	Dropout	$\{0.1, 0.2\}$	0.1
	Learning rate	$\{10^{-4}, 5 \times 10^{-4}, 10^{-3}\}$	10^{-4}
BERT Sentiment + LSTM	LSTM hidden dimension	$\{128, 256, 512\}$	256
	LSTM layers	$\{1, 2\}$	2
	Sentiment fusion	{concat., gated}	concatenation
	Dropout	$\{0.1, 0.2, 0.3\}$	0.2
	Learning rate	$\{10^{-4}, 5 \times 10^{-4}\}$	10^{-4}
MS-VAR	Number of regimes (K)	$\{2, 3, 4\}$	3
	Lag order	$\{1, 2, 3, \ldots, 10\}$	3
	Switching specification	{MSI, MSM, MSIH}	MSIH

Continued on next page

Model	Hyperparameter	Search Range	Selected
	EM convergence tolerance	$\{10^{-6}, 10^{-8}\}$	10^{-8}
HMM-LSTM	HMM states (K)	$\{2, 3, 4\}$	3
	HMM covariance	{diag., full}	diagonal
	LSTM hidden dimension	$\{128, 256, 512\}$	256
	LSTM layers	$\{1, 2, 3\}$	2
	Dropout	$\{0.1, 0.2, 0.3\}$	0.2
	Learning rate	$\{10^{-4}, 5 \times 10^{-4}, 10^{-3}\}$	5×10^{-4}
	Sequence length	252 (fixed)	252
TimesNet	Layers	$\{2, 3, 4\}$	3
	Model dimension	$\{32, 64, 128\}$	64
	Top-k periods	$\{3, 5, 7\}$	5
	FFN dimension	$\{64, 128, 256\}$	128
	Dropout	$\{0.1, 0.2, 0.3\}$	0.2
	Learning rate	$\{10^{-4}, 5 \times 10^{-4}, 10^{-3}\}$	10^{-4}
	Sequence length	252 (fixed)	252
PatchTST	Patch length	$\{12, 16, 24\}$	16
	Stride	$\{8, 12, 16\}$	8
	Layers	$\{3, 4, 6\}$	4
	Attention heads	$\{4, 8\}$	8
	Model dimension	$\{128, 256, 512\}$	256
	Dropout	$\{0.1, 0.2, 0.3\}$	0.2
	Learning rate	$\{10^{-4}, 5 \times 10^{-4}, 10^{-3}\}$	10^{-4}
iTransformer	Layers	$\{3, 4, 6\}$	4
	Attention heads	$\{4, 8\}$	8
	Model dimension	$\{128, 256, 512\}$	256
	FFN dimension	$\{256, 512, 1024\}$	512
	Dropout	$\{0.1, 0.2\}$	0.1
	Learning rate	$\{10^{-4}, 5 \times 10^{-4}, 10^{-3}\}$	10^{-4}
Integrated NF-BERT	Architecture and hyperparameters fixed per [134].		

For ARIMA, optimal orders vary across stocks because each stock exhibits different autocorrelation and partial autocorrelation structure; the most common selections were $(p, d, q) = (2, 1, 2)$ and $(1, 1, 1)$. XGBoost uses early stopping with a patience of 50 rounds on validation RMSE, so the effective number of estimators is often lower than the specified maximum. The LSTM and Simple Transformer sequence lengths are fixed at 252 trading days (one calendar year) to match the proposed model's input window, ensuring that differences in performance reflect architectural capacity rather than information asymmetry. The MS-VAR uses the MSIH specification (Markov-switching intercept and heteroscedasticity), which allows both the intercept and the error variance to switch across regimes while

keeping the autoregressive coefficients regime-invariant; this provides sufficient flexibility to capture volatility regime changes without overfitting the transition dynamics. For the HMM-LSTM, regime assignments are determined by the Viterbi path on the training set, and each regime-specific LSTM is trained only on data segments assigned to its corresponding state. PatchTST and iTransformer use the channel-independent and inverted-attention configurations recommended in their respective original publications, with sequence lengths fixed at 252 to match other deep learning baselines.

5.3.14 Computing Platform and Execution Times

All training and inference experiments were conducted on a Google Cloud Platform `a2-highgpu-1g` instance equipped with one NVIDIA A100 40 GB GPU, 12 vCPUs, and 85 GB of system memory. The software environment uses Python 3.10, PyTorch 2.2 with CUDA 12.1 (for the autoencoder, node transformers, and SAC controller), `statsmodels` 0.14 (for ARIMA, VAR, and MS-VAR), `scikit-learn` 1.3 (for Random Forest and SVR), and the original authors' reference implementations released alongside the PatchTST, iTransformer, TimesNet, and BERT publications. Each model is trained from scratch on the 1982 to 2010 training set with the hyperparameters listed in Table 5.1 and Table 5.14.

Training times scale with model complexity, from minutes for the statistical baselines (ARIMA, VAR) to roughly three hours for the full regime-aware framework. The AE-NodeFormer + SAC training time of 3 h 4 min represents an approximately 33% overhead relative to the Integrated NodeFormer-BERT baseline (2 h 18 min), with the additional cost attributable to the autoencoder pre-training, the second specialized node transformer pathway, the SAC controller training, and the final joint fine-tuning pass. Inference latency for the full framework is 47 ms per 20-stock cross-sectional forecast, well below any constraint posed by daily or weekly prediction; at intraday frequencies of one second or coarser, the inference cost is also negligible, which suggests the framework is deployable in tactical trading

Table 5.15: Per-model training time and per-prediction inference latency on a single NVIDIA A100 40 GB GPU (GCP `a2-highgpu-1g` instance). Training time refers to a full end-to-end run on the 1982 to 2010 training set; inference latency is averaged over the 2017 to 2025 test period and reported as wall-clock time per single 20-stock cross-sectional forecast at horizon $h = 1$

Model	**Training time**	**Inference latency**
ARIMA [135]	12 min	< 1 ms
VAR [29]	8 min	< 1 ms
MS-VAR [130]	34 min	3 ms
Random Forest [136]	14 min	6 ms
SVR [137]	9 min	4 ms
XGBoost [138]	17 min	2 ms
LSTM [41]	52 min	8 ms
Simple Transformer [3]	1 h 18 min	14 ms
BERT Sent. + LSTM [4]	1 h 48 min	22 ms
HMM-LSTM [1]	58 min	11 ms
TimesNet [131]	1 h 42 min	18 ms
PatchTST [132]	1 h 28 min	15 ms
iTransformer [133]	1 h 36 min	19 ms
Integrated NF-BERT [134]	2 h 18 min	31 ms
AE-NodeFormer (no SAC)	2 h 32 min	38 ms
AE-NodeFormer + SAC	**3 h 4 min**	**47 ms**

workflows on a single-A100 instance. The 47 ms latency could be reduced further through TorchScript export, weight quantization, or batched multi-day forecasting if real-time use cases require sub-10 ms response times.

To make the deployment band explicit, the 47 ms per-forecast latency decomposes to approximately 2.4 ms per stock for the 20-stock cross-section and can be compared against the response-time budgets of typical trading workflows. End-of-day rebalancing and the daily backtest of Section 5.3.11 have effectively no latency constraint. Minute-bar workflows allocate a 60 second budget per decision, against which the framework consumes about 0.08% of the available window per forecast and leaves roughly three orders of magnitude of headroom for data ingestion, signal post-processing, and order routing. Tick-resolution and high-frequency contexts where decisions are taken at the microsecond to low-millisecond scale are not directly addressable by the framework in its present form, since a single forward pass already exceeds those budgets. For sub-10 ms targets the recourse is the combination of the no-SAC variant (38 ms in Table 5.15), fp16 quantization, and TorchScript export discussed

in the ablation cost analysis (Table 5.7), at the cost of the SAC adaptation gains. Across the intraday-to-daily band that covers the majority of practical institutional and tactical trading use cases, however, no architectural change is required, and the framework is deployable as-is on a single-A100 instance, with the dominant operational cost being the periodic retraining schedule (Section 5.3.9) rather than the per-step inference.

The 20-stock universe size balances two considerations rather than reflecting an architectural maximum. First, sector and business-model coverage is preserved across eight sectors with at least one (and typically two to four) representatives each, so the graph attention sees both intra-sector and cross-sector relationships in every mini-batch and the inclusion of any single stock can be motivated by its role in the cross-sectional picture rather than by data convenience. Second, the 252-day temporal window combined with the 20-stock cross-section produces $252 \times 20 = 5{,}040$ tokens per cross-sectional sample, which together with the batch size of 64 reported in Table 5.1 fits within the 40 GB memory budget of a single NVIDIA A100 alongside the autoencoder and SAC components, and the full four-stage training pipeline completes in 3 h 4 min (Table 5.15). Scaling to $N = 50$ would produce 12,600 tokens per sample (2.5×) and to $N = 100$ would produce 25,200 tokens per sample (5×). Because the per-layer self-attention over the graph scales quadratically with N when the temporal window L is held fixed, total training cost grows approximately as N^2; extrapolating from the measured $N = 20$ training time projects roughly 12 to 20 hours for $N = 50$ and between two and four days for $N = 100$ on the same hardware, both of which remain compatible with weekly retraining cycles. Beyond approximately $N = 200$, edge sparsification (for example, top-k neighbors per node or sector-restricted attention) would be needed to control memory growth, but no architectural change is required to move from the present 20-stock configuration to a several-fold larger universe.

The graph topology itself is fully connected across the N stock nodes, so the structure is fixed by construction and the node-transformer pathway does not add or remove edges during training or inference. What is learned is the edge weight $e_{ij}^{(\ell)}$, which is initialized

from sector membership and training-period return correlations and then refined through the layerwise function $e_{ij}^{(\ell+1)} = \sigma(\mathbf{w}_e^T[\mathbf{h}_i^{(\ell)} \| \mathbf{h}_j^{(\ell)}] + b_e)$ described in Section 5.2 (Figure 5.7); these weights are frozen alongside all other model parameters at inference time. In a production deployment, the initialization correlations ρ_{ij}^{train} could be recomputed on a rolling window (for example, refreshed quarterly using the most recent five trading years) to reflect drift in cross-sectional relationships, and the model could be briefly fine-tuned on the updated initialization, but neither operation is part of the inference loop evaluated in this thesis. Adopting such a topology-refresh schedule remains a deployment-time choice that does not affect the architectural conclusions reported here.

5.4 Discussion

5.4.1 Interpretation of Results

Regime-aware prediction with adaptive control outperforms homogeneous approaches across all metrics. The 26% MAPE improvement over the baseline integrated model (0.59% vs 0.80%) reflects gains from three sources: regime detection (autoencoder), specialized processing (dual node transformers), and adaptive parameter tuning (SAC controller).

Without requiring explicit per-sample anomaly labels, the autoencoder identifies market states that deviate from normal patterns. High reconstruction errors coincide with periods of elevated volatility, earnings announcements, and macroeconomic shocks, allowing the system to detect anomalies as deviations from its learned representation of typical behavior.

The SAC controller learns to adjust the detection threshold based on prediction outcomes, maintaining a higher threshold during stable periods to prevent unnecessary routing to the event pathway and lowering it during genuine regime shifts to engage event-aware processing. This adaptive behavior emerges from the reward signal rather than hand-crafted rules.

The dual-path architecture enables specialization: the normal pathway develops representations optimized for stable conditions where fundamental factors dominate, while the event pathway incorporates additional context (sentiment spikes, volatility regimes, event characterization) that proves informative during turbulent periods but might add noise during normal conditions.

A subtle property of the detection mechanism is its ability to engage on semantically-driven events that do not immediately produce large price movements but that nonetheless represent meaningful regime shifts, such as central-bank communications, regulatory clarifications, and geopolitical statements. Three of the six router features defined in Section 5.2, the absolute sentiment magnitude $|S_t|$, the crisis-keyword post-velocity ν_t^{post}, and the change in cross-stock correlation ρ_t^{Δ}, are designed for precisely this case. A policy speech or regulatory announcement that elevates market discussion without yet moving prices will register as a sentiment-magnitude spike and a posting-velocity surge, raising the autoencoder's reconstruction error e_t before the price-derived features (rolling volatility, returns) have responded. The cross-asset stress component of the event context vector $\mathbf{c}_t$ amplifies this signal at the system level: when a semantic shock simultaneously elevates reconstruction error across many stocks, the cross-sectional mean (the first element of $\bar{e}_t^{\text{cross}}$) rises even when individual departures are modest, allowing the framework to treat synchronized semantic shocks as a single regime event rather than as twenty independent low-magnitude anomalies. Scheduled semantic events such as earnings releases and FOMC meeting dates are handled separately through the earnings-proximity component of $\mathbf{c}_t$, which pre-arms the event pathway when known catalysts approach. The framework is correspondingly less sensitive to semantic events that produce no detectable sentiment response, no post-volume change, and no cross-sectional contagion, such as a quiet regulatory clarification that markets interpret over a horizon longer than the daily rebalance window; this is a genuine boundary of the detection mechanism that the Limitations subsection acknowledges through the broader dependence of the framework on sentiment and event-calendar signals being available at daily resolution.

5.4.2 Economic Interpretation

While the performance improvements carry practical implications, they should be interpreted cautiously. A 7 percentage point improvement in directional accuracy (72% vs 65%) carries practical value for trading decisions, and the 43% relative improvement during high-volatility periods is particularly valuable since these are precisely the conditions where accurate forecasts matter most for risk management.

Transaction costs, market impact, and execution constraints would reduce realized gains from any trading strategy based on these predictions. The economic significance analysis in Chapter 4 demonstrated that even the baseline model's predictions, when combined with simple trading rules, generate returns exceeding buy-and-hold benchmarks before costs. The improvements documented here should amplify these returns proportionally, though the gap would narrow after incorporating realistic trading frictions.

Synthesizing the predictive, economic, and operational results suggests several concrete deployment paths for the framework. For institutional portfolio construction at a daily rebalance cadence, the long-only top-K strategy of Section 5.3.11 produces a Sharpe ratio of 1.32 net of 10 basis-point round-trip transaction costs, a maximum drawdown of 16.2%, and a 71% positive-PnL day rate (Table 5.11), positioning the framework as a candidate alpha signal for tactical trading on a liquid large-cap universe rather than as a structural buy-and-hold replacement. The per-forecast inference latency of 47 ms on a single A100 GPU (Table 5.15, with the per-stock breakdown discussed in Section 5.3.14) is compatible with any non-high-frequency workflow, including end-of-day rebalancing, minute-bar trading, and overnight risk reporting; a practitioner can run the trained model on a single consumer-grade GPU or, with batched inference at seconds-scale latency, on CPU-only infrastructure. The operational maintenance cycle is dominated by periodic retraining rather than per-step inference: the framework can be retrained on the most recent decade of data at an annual cadence to refresh the 75th-percentile volatility cutoff (Section 5.3.9), with the SAC controller continuing to

adapt the routing parameters intra-year through its frozen-policy state-dependent behavior. The 20-stock universe used in the experiments is the most computationally convenient configuration, but the same architecture extends to 50 or 100 stocks at approximately 12 to 20 hours or two to four days of training time respectively on the same hardware (Section 5.3), bringing the framework within reach of mid-sized portfolio mandates without architectural change. These deployment implications carry the same caveats noted alongside the backtest results: realized performance under live deployment would also be affected by financing costs, taxes, market impact beyond the assumed slippage, and operational latency that the backtest does not model.

5.4.3 Limitations

The most consequential limitation concerns data selection. The 20 stocks used in this study are drawn from the current S&P 500 universe, which introduces survivorship bias: companies that failed, were acquired, or delisted between 1982 and 2025 are absent from the dataset. Because the selected stocks are disproportionately successful over the evaluation period, predictability estimates may be inflated relative to what a real-time investor would experience when choosing from the full market. Performance on a point-in-time universe constructed from historical index constituents could differ, and the reported results should therefore be interpreted as evidence of architectural capability rather than guaranteed trading performance.

Several data-related concerns compound this issue. Edge weights in the graph are initialized from correlations computed over the training period (1982–2010), but market correlations are non-stationary, and relationships that held during this window may weaken or reverse by the test period (2017–2025). The learnable edge refinement mechanism partially compensates by adapting weights during training, though fundamental correlation regime shifts could still affect generalization. Sentiment data from X (formerly Twitter) is available

only from 2007 onward; for the 1982–2006 portion of training, sentiment features are set to zero. This means the sentiment modality effectively "turns on" partway through training, potentially limiting the model's ability to learn robust sentiment-price relationships from the earlier decades.

On the modeling side, the reinforcement learning controller requires careful tuning of reward weights, temperature, and network architecture, and suboptimal configurations could lead to unstable threshold behavior or poor convergence. The system also learns regime boundaries from its own prediction performance, which risks feedback loops where poor initial predictions lead to suboptimal threshold learning; the staged training protocol mitigates this by pre-training each component before SAC optimization, but the circularity is not fully eliminated. The three-component architecture increases computational requirements, with training time approximately 33% longer than the baseline model, which may constrain real-time deployment. Although the SAC policy weights are frozen at inference time, the state-dependent routing decisions produced by the fixed policy constitute a form of adaptive processing that purely static baselines (ARIMA, Random Forest, LSTM) do not possess. The regime-switching baselines (HMM-LSTM, MS-VAR) share this property, and the ablation study confirms that the static No SAC variant already outperforms all baselines, but the additional gain from state-dependent threshold tuning should be interpreted as an architectural advantage of the framework rather than an improvement attributable solely to prediction accuracy.

From an economic standpoint, the analysis excludes transaction costs, market impact, and execution constraints. For strategies involving frequent rebalancing, these frictions would reduce realized returns. Backtesting is performed on the same 20-stock universe used for model development, leaving external validity on unseen stocks untested. Finally, sentiment data was collected via the X (formerly Twitter) API, which has undergone significant access policy changes, making exact replication of the sentiment component challenging with current API limitations.

5.5 Summary

This chapter introduced an adaptive framework for stock price prediction that automatically detects market regimes and adjusts processing accordingly. The architecture combines an autoencoder for regime detection, dual node transformer networks specialized for stable and volatile conditions, and a Soft Actor-Critic reinforcement learning controller that learns adaptive regime thresholds from prediction performance.

Experiments on 20 S&P 500 stocks spanning 1982–2025 demonstrate substantial improvements over the baseline integrated node transformer from Chapter 4: the complete system achieves 0.59% MAPE for one-day predictions compared to 0.80% for the single-pathway model, while directional accuracy reaches 72%, a 7 percentage point improvement. These gains persist across prediction horizons and are most pronounced during volatile periods where the baseline struggles. The key conceptual contribution is the adaptive learning of regime boundaries: rather than relying solely on fixed hand-crafted anomaly definitions, the system discovers useful boundaries by optimizing downstream prediction accuracy. This approach avoids the staleness problem of hand-crafted regime rules and the labeling burden of supervised regime classification.

The next chapter extends this line of work in a different direction. While Chapters 4 and 5 focused on improving prediction accuracy through architectural innovations, Chapter 6 explores whether the underlying models can be evaluated and refined through LLM-based behavioral analysis, providing an orthogonal perspective on model quality beyond traditional numerical metrics.

6

Multi-Dimensional Behavioral Evaluation of Agentic Stock Prediction Systems Using LLM Judges with Closed-Loop Reinforcement Learning Feedback

This chapter is based on the following publication:

Al Ridhawi, M., Haj Ali, M., & Al Osman, H. (2026). Multi-Dimensional Behavioral Evaluation of Agentic Stock Prediction Systems Using LLM Judges with Closed-Loop Reinforcement Learning Feedback. *Submitted to Applied Artificial Intelligence*. Under review. arXiv:2605.05739.

This chapter presents the third contribution of this thesis: a framework for evaluating and improving the behavioral quality of agentic stock prediction systems using large language models (LLMs) as judges. Chapters 4 and 5 progressively improved prediction accuracy through architectural innovations, advancing from a single-pathway node transformer with sentiment fusion to a regime-aware dual-pathway system with reinforcement learning control. These improvements were measured exclusively through aggregate output metrics such as MAPE, RMSE, and directional accuracy. While such metrics quantify the quality of final predictions, they reveal nothing about whether the intermediate decisions that produced those predictions were themselves sound. An agentic system may arrive at a correct forecast while routing data through an inappropriate pathway, or produce an incorrect prediction despite making prudent regime detection and risk calibration decisions. Aggregate scores conflate these cases, yet distinguishing them is essential for diagnosing failures, calibrating trust, and guiding further improvements.

This chapter addresses that gap by applying the LLM-as-a-Judge paradigm to evaluate the full behavioral trace of the adaptive regime-aware system from Chapter 5 across six domain-specific dimensions. Three dimensions map directly to architectural decision points (regime detection accuracy, routing appropriateness, and adaptation responsiveness), while three capture emergent behavioral properties spanning multiple components (risk calibration, strategy coherence, and error recovery). The framework closes the evaluation loop by translating LLM diagnostic outputs into auxiliary reward signals for the system's Soft Actor-Critic (SAC) reinforcement learning controller, enabling autonomous correction of identified weaknesses without human intervention. The dataset, feature engineering pipeline, and core evaluation metrics follow the experimental framework established in Chapter 3.

The chapter proceeds as follows. Section 6.1 positions the contribution relative to prior work on LLM-based evaluation, agent assessment, and financial AI evaluation. Section 6.2 presents the complete evaluation framework, including behavioral trace formalization, evaluation dimensions, rubric design, prompt engineering, and the closed-loop reward integration mechanism. Section 6.3 describes the experimental setup, perturbation-based validation, predictive validity analysis, and closed-loop improvement results. Section 6.4 discusses the findings and limitations. Section 6.5 summarizes the chapter and connects it to the thesis conclusion.

6.1 Related Work and Research Gap

The methodological context underlying this chapter spans LLM-based evaluation, autonomous agent assessment, and evaluation practices in financial AI. Chapter 2 provides a comprehensive survey of each of these areas, covering the LLM-as-a-Judge paradigm and its applications, evaluation design principles including rubric construction and multi-judge agreement, agent evaluation frameworks such as ReAct and Reflexion, and evaluation methods

specific to financial AI (Section 2.9). This section focuses on the specific intersection of these directions and identifies the gaps that motivate the proposed framework.

The LLM-as-a-Judge paradigm has demonstrated strong alignment with human evaluation in natural language processing tasks. Zheng et al. [89] showed that GPT-4 judgments on the MT-Bench conversation benchmark achieve agreement rates above 80% with human preferences, comparable to inter-annotator agreement among humans. Their study also identified systematic biases in LLM evaluation, including position bias and verbosity bias, which informed the design of the structured prompting approach used in this chapter. Liu et al. [90] demonstrated that chain-of-thought prompting further improves evaluation fidelity by requiring the model to articulate its reasoning before assigning a score, reducing arbitrary and inconsistent judgments. Their G-Eval framework established the principle that structured reasoning prompts produce more reliable assessments than direct scoring, a principle that is adopted and extended here to multi-step behavioral evaluation. AlpacaFarm [91] extended the paradigm to reward model training, suggesting that LLM judgments are reliable enough to serve as training signals for downstream optimization rather than merely as passive evaluation instruments. This finding is central to the closed-loop design proposed in this chapter, where LLM assessments are converted into auxiliary reward signals for reinforcement learning.

Despite these advances, most existing applications of the LLM-as-a-Judge paradigm evaluate static outputs: single responses, translations, summaries, or dialogue turns. Evaluating an autonomous financial agent requires assessing temporal sequences of interdependent decisions under stochastic market conditions. In this setting, the evaluation must account for the fact that each decision depends on its predecessors and influences its successors, that the same decision may be correct under one market condition and incorrect under another, and that the quality of a decision cannot always be determined from its immediate outcome due to market stochasticity. To the best of our knowledge, prior LLM-based evaluation work has not directly addressed these temporal and contextual challenges in financial prediction systems. .

In parallel, the agent evaluation literature has identified process quality as an open challenge. ReAct [93] introduced reasoning traces combined with tool use, establishing the principle that agents should externalize their decision-making in a format amenable to inspection. However, ReAct relied on binary task completion for assessment, which cannot distinguish between agents that succeed through sound reasoning and those that succeed through fortunate coincidence. Reflexion [94] added self-reflection capabilities, enabling agents to learn from verbal feedback about their reasoning failures, though evaluation remained focused on final task success rather than the quality of individual decisions. Wang et al. [95] explicitly identified process quality evaluation as an open problem, noting that existing benchmarks measure whether agents achieve their goals but not whether their reasoning was sound. An agent may complete a task through a sequence of lucky coincidences, or fail a task despite making excellent decisions that were undermined by environmental stochasticity. Xi et al. [96] surveyed the emerging field of LLM-based agents and identified financial systems as a domain where behavioral evaluation is both critical and underexplored. To the best of our knowledge, a dedicated agent-evaluation framework has not yet been established for financial prediction systems, where the distinction between good decisions and good outcomes is fundamental due to the stochastic nature of markets.

In financial AI, evaluation often relies heavily on aggregate metrics: MAPE, RMSE, Sharpe ratio, and maximum drawdown (MDD) [141–143]. These metrics assess the statistical properties of outcome distributions but reveal nothing about the reasoning behind individual decisions. A system that correctly identifies a regime change but routes predictions through the wrong pathway, or that detects elevated risk but fails to adjust its confidence, will exhibit degraded aggregate performance, but the aggregate metrics cannot diagnose which component failed or why. Standard evaluation practices such as backtesting, walk-forward validation, and statistical significance testing all operate at the aggregate level, assessing the distribution of outcomes rather than the quality of the processes that produced them [141]. The need for process-level evaluation is not merely academic: as automated prediction systems assume

greater autonomy, understanding whether their intermediate decisions are sound becomes essential for risk management and system improvement.

The closest conceptual precedent to the approach proposed here is reinforcement learning from human feedback (RLHF), where human or LLM judgments serve as training signals for language model alignment [91, 144]. The key insight from RLHF is that qualitative assessments, when aggregated across a sufficient number of evaluations, may provide a learning signal that improves system behavior in directions that are difficult to capture through hand-crafted reward functions alone. However, RLHF targets language model alignment, not financial decision quality, and operates on single outputs rather than temporal decision sequences. The present work adapts this principle to a fundamentally different domain, where the evaluation targets are sequences of interdependent decisions made under stochastic financial conditions, and the learning signal must be routed to specific components of a reinforcement learning controller through a credit assignment mechanism.

This chapter addresses both gaps simultaneously. It adapts the LLM-as-a-Judge paradigm to evaluate behavioral sequences of an agentic financial system, defines six domain-specific evaluation dimensions grounded in the system's architectural decision points and emergent behavioral properties, validates the framework through perturbation-based experiments, and closes the evaluation loop by converting LLM diagnostics into SAC reward modifications that enable autonomous correction of identified weaknesses.

6.2 Methodology

6.2.1 System Overview and Behavioral Context

The evaluation framework is developed for and demonstrated on the adaptive regime-aware prediction system introduced in Chapter 5 [145]. That system processes daily data for 20 S&P 500 equities from January 1982 to March 2025, combining an autoencoder for regime

detection, dual node transformer networks specialized for stable and volatile conditions, and a SAC reinforcement learning controller that adaptively tunes the regime detection threshold and pathway blending weights. Chapter 5 provides the complete architectural description, training protocol, and baseline evaluation; the present section summarizes only the aspects relevant to behavioral evaluation.

The system is treated as agentic in the sense that it performs a sequence of autonomous decisions at each trading day without human oversight. Four decision points define the system's behavioral profile. First, the autoencoder computes reconstruction error $e_t = \|x_t - \hat{x}_t\|_2$ and classifies the current market state as normal or anomalous by comparing e_t to a learned threshold τ. Second, a router directs data to one of two node transformer pathways based on the regime classification: a normal-regime variant optimized for typical market dynamics, or an event-conditioned variant tuned for volatile periods. Third, the system produces a blended prediction $\hat{y}_t = \alpha_t \cdot y_{\text{normal}} + (1 - \alpha_t) \cdot y_{\text{event}}$, where α_t controls the relative contribution of each pathway. Fourth, the SAC controller adjusts both τ and α_t based on recent prediction performance, completing the feedback loop. Each of these decisions can be evaluated as more or less appropriate in context, and the quality of the final prediction depends on the quality of the entire decision chain rather than any individual component.

This sequential decision structure creates the fundamental evaluation gap that motivates the present work. Aggregate metrics such as MAPE and directional accuracy collapse the four-step decision process into a single quality measure. A system that misclassifies market regimes but compensates through aggressive parameter adaptation may achieve acceptable MAPE, while a system that correctly classifies regimes but routes predictions inefficiently may exhibit poor accuracy despite sound upstream reasoning. These two failure modes have fundamentally different implications for system reliability and improvement, yet they produce identical aggregate metric signatures. Distinguishing them requires evaluating each decision point in context, assessing whether the action taken was appropriate given the information

available at the time, regardless of whether the final prediction happened to be correct. The proposed LLM-as-a-Judge framework is designed to provide this capability.

A natural question is whether this evaluation layer could replace the autoencoder and router of Chapter 5 altogether rather than sit above them. It cannot, because the judge and the autoencoder-router pair operate at different levels of the system. The autoencoder and router make the per-day regime and routing decisions inside the prediction pipeline, producing a label-free quantitative regime signal from reconstruction error at negligible latency on every stock and every trading day. The judge, by contrast, assesses the quality of those decisions after the fact, and several of its dimensions, including regime detection and routing appropriateness, take the autoencoder and router outputs as their input, so the judge presupposes these components rather than substituting for them. The judge also operates only periodically, because a per-day language-model evaluation would be prohibitive in cost and latency, as detailed later in this chapter, which makes it unsuitable as the real-time regime detector that the pipeline requires. The judge therefore complements the autoencoder and router as an evaluation and refinement layer, feeding its assessments back to the SAC controller in the closed-loop variant, rather than replacing the mechanisms that perform regime detection and routing.

Figure 6.1 illustrates the complete pipeline, including the LLM judge feedback mechanism introduced in this chapter. The behavioral trace logged at each decision point captures the inputs, outputs, and context of every autonomous action, providing the raw material for LLM-based evaluation. The dashed red arrows indicate the SAC reinforcement feedback loop from Chapter 5, while the purple arrow indicates the new LLM diagnostic reward signal introduced in this chapter.

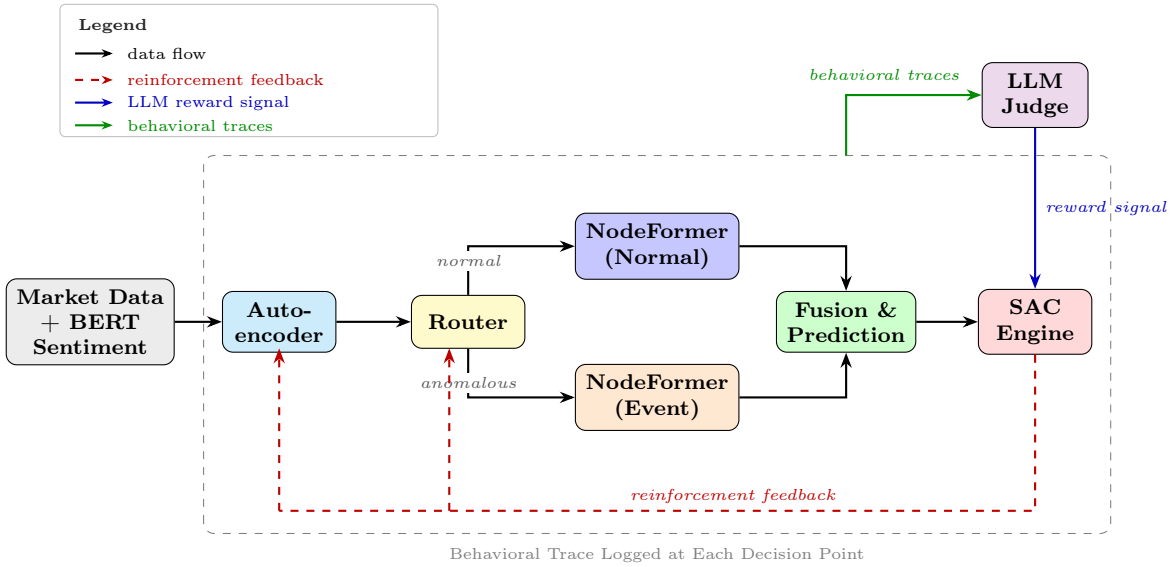

Figure 6.1: Architecture of the agentic prediction system with the LLM judge feedback loop. Behavioral traces logged at each decision point (dashed gray box) are provided as input to the LLM judge, which produces diagnostic reward signals to update the SAC engine. The SAC engine also receives reinforcement feedback from realized prediction outcomes. Arrow types are described in the in-figure legend.

6.2.2 Behavioral Trace Formalization

At each trading day t, the system records a behavioral trace $\mathcal{B}_t$ that captures the complete decision context and outcomes of every autonomous action. The trace is defined as a structured tuple:

$$\mathcal{B}_t = \langle\, \mathbf{m}_t,\ \mathbf{a}_t,\ \mathbf{r}_t,\ \mathbf{u}_t,\ \hat{y}_t,\ \mathbf{p}_t \,\rangle \tag{6.1}$$

where each component records a distinct aspect of the system's behavior at time t. The following paragraphs describe each component, its contents, and its role in the evaluation.

The market context vector $\mathbf{m}_t = (p_t, v_t, \text{VIX}_t, \bar{S}_t)$ records the closing price, trading volume, CBOE Volatility Index value, and aggregated BERT sentiment score at time t. These four values provide the environmental context against which the system's decisions must be evaluated. A routing decision that is appropriate under low volatility may be inappropriate

under high volatility, and the LLM judge requires this context to assess whether the system's actions were proportional to the prevailing market conditions. The aggregated sentiment score $\bar{S}_t$ is the exponentially weighted average of BERT sentiment signals as described in Chapter 3, providing the judge with visibility into the textual information environment that the system had access to when making its decisions.

The autoencoder decision vector $\mathbf{a}_t = (e_t, \tau_t, \ell_t)$ records the reconstruction error, the current threshold value, and the binary regime label ($\ell_t \in \{0, 1\}$, where 1 indicates anomalous). This component captures the system's perception of the current market state and whether the threshold was appropriately calibrated relative to the reconstruction error. A high reconstruction error paired with a regime label of normal would indicate a threshold that is set too high, preventing the system from detecting genuine anomalies. Conversely, a low reconstruction error paired with an anomalous label would indicate an overly sensitive threshold that generates false alarms. The inclusion of both e_t and τ_t (rather than just ℓ_t) allows the judge to assess the margin of the classification decision, distinguishing between confident classifications and borderline cases where $e_t \approx \tau_t$.

The routing vector $\mathbf{r}_t = (\alpha_t, k_t)$ records the blending weight and the index of the dominant pathway ($k_t = 0$ for normal, $k_t = 1$ for event). The blending weight α_t determines the relative contribution of each node transformer pathway to the final prediction, as defined in Equation 5.4 of Chapter 5. Consistency between the regime label ℓ_t and the routing decision (k_t, α_t) is a key indicator of strategy coherence: classifying the market as anomalous but setting α_t to heavily favor the normal pathway constitutes a logical contradiction that the LLM judge can identify and penalize.

The SAC action vector $\mathbf{u}_t = (\Delta\tau_t, \Delta\alpha_t) \in [-0.1, 0.1]^2$ records the threshold and blending weight adjustments applied by the reinforcement learning controller at time t. These adjustments reflect the system's adaptive behavior: large adjustments indicate aggressive parameter changes in response to perceived shifts in market conditions, while small adjustments indicate parameter stability. The evaluation framework assesses whether these adjustments are timely

(applied soon after a detectable condition change), proportional (scaled to the magnitude of the change), and stable (not oscillating between opposite directions on consecutive days).

The prediction output $\hat{y}_t$ records the system's one-day-ahead price forecast, and the rolling performance vector $\mathbf{p}_t = (\text{MAPE}_{t,w}, \text{DA}_{t,w})$ records MAPE and directional accuracy computed over a trailing window of $w = 20$ trading days. These performance metrics provide the outcome context necessary for the LLM judge to assess whether the system's decisions led to acceptable prediction quality and whether errors triggered appropriate corrective actions. The trailing window of 20 days provides sufficient history for the judge to identify performance trends (improving, deteriorating, or stable) without including data so old that it no longer reflects the system's current behavior.

Together, the six components of $\mathcal{B}_t$ provide the LLM judge with complete visibility into the system's inputs, decisions, outputs, and recent performance at each trading day, enabling informed assessment of each decision's quality in context. The trace is serialized as a structured JSON record for inclusion in the LLM prompt, with labeled fields, units, and value ranges to facilitate interpretation. Each numerical value is accompanied by contextual information: reconstruction errors are shown alongside the threshold, blending weights include the regime label for cross-referencing, and SAC adjustments are presented with the resulting parameter values after application.

To illustrate the structure concretely, the following is a representative trace from a single trading day during a period of rising volatility. The market context shows VIX at 22.4 with declining sentiment ($\bar{S}_t = -0.18$). The autoencoder produces reconstruction error $e_t = 0.034$ against threshold $\tau_t = 0.031$, yielding regime label $\ell_t = 1$ (anomalous). The router sets $\alpha_t = 0.35$, directing 65% of the prediction weight to the event pathway ($k_t = 1$). The SAC controller applies adjustments $\Delta\tau = -0.002$ (slightly lowering the threshold to increase anomaly sensitivity) and $\Delta\alpha = -0.05$ (further increasing the event pathway contribution). The rolling performance shows $\text{MAPE}_{t,20} = 0.58\%$ and $\text{DA}_{t,20} = 73\%$, both within acceptable ranges. An LLM judge evaluating this trace in isolation would assess whether the regime

classification is consistent with the VIX level and sentiment, whether the routing weight appropriately reflects the anomalous classification, and whether the SAC adjustments are proportional to the observed market conditions. However, single-day evaluation is insufficient for several dimensions, motivating the episodic grouping described in the next section.

6.2.3 Evaluation Episodes

Individual behavioral traces provide snapshots of single-day decisions, but several evaluation dimensions require temporal context that extends beyond a single day. Assessing adaptation responsiveness requires observing whether the system adjusts parameters promptly after a change in conditions. Evaluating error recovery requires observing the system's behavior over multiple days following a prediction error. Assessing strategy coherence requires checking whether sequential decisions are logically consistent across time. To provide this temporal context, the evaluation framework groups consecutive traces into evaluation episodes.

An evaluation episode $\mathcal{E}_t$ is defined as a contiguous sequence of five behavioral traces:

$$\mathcal{E}_t = \{\mathcal{B}_t, \mathcal{B}_{t+1}, \mathcal{B}_{t+2}, \mathcal{B}_{t+3}, \mathcal{B}_{t+4}\} \tag{6.2}$$

The window size of five trading days (one calendar week) was selected as practical design choice based on conceptual considerations and preliminary empirical support. A shorter window of two or three days would provide insufficient context for assessing adaptation and recovery behaviors, as the SAC controller typically requires two to three days to adjust parameters after a regime transition, and recovery assessment requires observing both the error and the subsequent response. A longer window of ten or more days would dilute the evaluation by spanning multiple distinct market conditions, making it difficult for the judge to assess whether individual decisions were appropriate for the conditions at the time they were made, and would also reduce the number of independent episodes available for statistical analysis.

To support this choice empirically, preliminary experiments compared window sizes of 3, 5, 7, and 10 days using GPT 5.4 on 60 validation-period episodes. Inter-episode score variance increased sharply at 3 days ($\sigma = 0.94$) relative to 5 days ($\sigma = 0.71$), reflecting the insufficient temporal context for judges to form stable assessments. At 7 days ($\sigma = 0.68$) and 10 days ($\sigma = 0.65$), variance decreased marginally but the judges' justifications increasingly referenced conditions from the beginning of the episode that were no longer relevant by the end, producing assessments that averaged over heterogeneous market regimes rather than evaluating coherent behavioral sequences. Cross-model agreement peaked at 5 days (Krippendorff's $\alpha = 0.81$), declining to 0.74 at 3 days (due to insufficient evidence for consistent scoring) and 0.78 at 10 days (due to ambiguity about which portion of the episode to weight). These preliminary results suggest that a five-day window provides a useful balance between insufficient short-horizon context and excessive temporal heterogeneity in longer episodes. Accordingly, the five-day setting is adopted in this chapter as a practical and empirically supported design choice, rather than as a universally optimal window size. The five-day window is also consistent with the recent error history length $k = 5$ used in the SAC state representation (Equation 5.5), ensuring that the evaluation episode covers the same temporal scope that the controller uses for its own decision-making.

Episodes are sampled without overlap to reduce statistical dependence between evaluated units. Each episode is evaluated independently by the LLM judges, receiving a separate set of dimension scores. The sampling strategy for episode selection is described in Section 6.3.1.

6.2.4 Evaluation Dimensions

The framework defines six evaluation dimensions that are intended to capture key components of behavioral quality in assessable form. The dimensions are organized into two categories based on their relationship to the system architecture. The first three dimensions correspond to specific architectural decision points and assess whether individual components

performed their designated function correctly. The remaining three dimensions capture emergent behavioral properties that arise from the interaction of multiple components and cannot be attributed to any single subsystem. This distinction between component-level and system-level assessment ensures that the framework can both isolate component failures and detect interaction effects that would be invisible at the component level.

Architectural Decision Dimensions

Regime Detection (RD) assesses the accuracy of the autoencoder's market state classification and the sensitivity of the threshold τ. A high RD score indicates that the system correctly identifies regime transitions, adjusts the threshold in response to changing volatility patterns, and avoids both false positives (classifying normal conditions as anomalous) and false negatives (failing to detect genuine market disruptions). The LLM judge evaluates this dimension by examining the relationship between the reconstruction error e_t, the threshold τ_t, the assigned regime label ℓ_t, and the concurrent market context (VIX level, price volatility, and sentiment shifts). For example, if the VIX rises sharply while the reconstruction error increases but the regime label remains "normal," the judge would assign a low RD score for that day. Because regime detection is the first decision in the processing chain, errors at this stage propagate downstream through routing and prediction, making it the most consequential component-level assessment.

Routing Appropriateness (RT) evaluates whether data was directed to the correct node transformer pathway given the detected regime, with particular attention to decisions made near the threshold boundary where $e_t \approx \tau_t$. A high RT score indicates that the routing decision is consistent with the detected regime and that the blending weight α_t appropriately reflects the system's confidence in its regime classification. Near the threshold boundary, where the regime classification is uncertain, the LLM judge assesses whether the system hedges appropriately through intermediate blending weights (e.g., $\alpha_t \approx 0.5$) rather than making hard switches that introduce prediction discontinuities. Abrupt routing changes

between consecutive days without a corresponding change in market conditions would lower the RT score, as such behavior suggests that the system is oscillating between pathways rather than making stable routing decisions.

Adaptation Responsiveness (AD) measures the timeliness and proportionality of the SAC controller's parameter adjustments. A high AD score indicates that the controller responds promptly to changing market conditions, adjusting τ and α within one to two trading days of a detectable shift, and that the magnitude of adjustments is proportional to the severity of the change. Excessively aggressive adjustments that overshoot appropriate parameter values receive lower scores, as do delayed adjustments that leave the system operating with stale parameters for multiple days after a transition. The judge assesses this dimension by tracking the SAC action vectors $\mathbf{u}_t$ across the episode and comparing them against observable changes in market conditions. A day where the VIX doubles but $\Delta\tau = 0$ and $\Delta\alpha = 0$ would indicate poor adaptation responsiveness.

Emergent Behavioral Dimensions

Risk Calibration (RC) assesses the appropriateness of the system's overall risk posture given the prevailing volatility environment. Unlike the architectural dimensions, risk calibration is an emergent property that depends on the interaction of regime detection, routing, and SAC adaptation. A well-calibrated system becomes more conservative during high-volatility periods, evidenced by lower blending weights that increase the contribution of the event-conditioned pathway and more cautious parameter adjustments. During low-volatility periods, a well-calibrated system operates with greater confidence, reflected in higher blending weights and more stable parameters. The LLM judge evaluates this dimension by comparing the system's behavioral profile against the VIX level and recent price volatility, assessing whether the overall pattern of decisions reflects an appropriate response to the current risk environment rather than a static configuration that ignores volatility changes.

Strategy Coherence (SC) evaluates the logical consistency of the system's decision chain, specifically the absence of contradictory actions across decision points. A high SC score requires that the regime classification, routing decision, blending weight, and SAC adjustments form a logically consistent sequence at each step and across the episode. For example, classifying the market as anomalous but setting the blending weight to heavily favor the normal pathway constitutes a coherence violation. Similarly, detecting a regime transition but freezing the SAC parameters contradicts the adaptive design of the system. Increasing τ (making regime detection less sensitive) while simultaneously shifting α toward the event pathway would also represent an incoherent action sequence. The LLM judge assesses coherence by examining whether each decision logically follows from its predecessors and from the available market context within the episode.

Error Recovery (ER) measures the speed and effectiveness of the system's adaptation following prediction errors. A substantial prediction error at time t is detected when the daily MAPE exceeds the trailing 20-day average by more than one standard deviation:

$$\xi_t = \mathbb{1}\left[\,\mathrm{MAPE}_t > \overline{\mathrm{MAPE}}_{t,20} + \sigma_{\mathrm{MAPE},t,20}\,\right] \tag{6.3}$$

where $\overline{\mathrm{MAPE}}_{t,20}$ and $\sigma_{\mathrm{MAPE},t,20}$ are the mean and standard deviation of daily MAPE over the preceding 20 trading days. When $\xi_t = 1$, a high ER score requires that the SAC controller adjusts parameters within two trading days and that subsequent predictions show measurable improvement relative to the error. Persistent errors without corrective action, or corrective actions that overcorrect and introduce new errors in the opposite direction, receive lower scores. The LLM judge evaluates this dimension by identifying prediction errors within the episode, tracking subsequent SAC adjustments, and assessing whether the system's behavior changes in ways that address the identified failure mode rather than applying generic corrections. Table 6.1 summarizes the six dimensions and their assessment criteria.

Table 6.1: Six evaluation dimensions for the LLM-as-a-Judge assessment, organized by category.

Category	Dimension	Assessment Criteria
Architectural Decision	Regime Detection (RD)	Accuracy of autoencoder classification; threshold sensitivity; false positive and false negative rates
	Routing (RT)	Consistency with detected regime; blending weight appropriateness; stability near threshold boundary
	Adaptation (AD)	Timeliness of SAC adjustments; proportionality to condition changes; absence of overshooting
Emergent Behavioral	Risk Calibration (RC)	Appropriateness of risk posture given volatility; conservative behavior in high-VIX periods
	Strategy Coherence (SC)	Logical consistency across decision chain; absence of contradictory actions
	Error Recovery (ER)	Speed of adaptation after errors; effectiveness of corrective actions; absence of overcorrection

The selection of these six dimensions is grounded in the architecture of the prediction system from Chapter 5. Each architectural decision dimension corresponds to a specific component (autoencoder, router, SAC controller) whose output can be independently assessed for correctness. Each emergent dimension captures a cross-cutting behavioral property that requires examining multiple component outputs in context.

The dimensionality of the evaluation framework was determined through a systematic design process. Alternative decompositions with fewer and more dimensions were considered and rejected. A four-dimension variant that merged routing and adaptation into a single "response quality" dimension was tested in preliminary experiments, but the resulting scores conflated two distinct failure modes (incorrect routing versus delayed parameter adjustment) that require different corrective actions, reducing the diagnostic value of the evaluation and preventing effective credit assignment. An eight-dimension variant that split regime detection into sensitivity and specificity components and separated risk calibration into upside and downside sub-dimensions was also tested, but the finer granularity produced lower inter-judge agreement (mean $\alpha_K = 0.62$) without improving predictive validity, as the additional

dimensions introduced scoring ambiguity that exceeded the LLM judges' discrimination capacity on the 1–5 scale.

To verify that the six dimensions capture independent aspects of behavioral quality, pairwise Spearman correlations were computed across the 200 unperturbed evaluation episodes. The mean pairwise correlation is $\bar{\rho} = 0.26$, with the highest correlation ($\rho = 0.41$) observed between regime detection and routing, which is expected given that routing decisions depend directly on regime classification outputs. The lowest correlation ($\rho = 0.12$) occurs between routing and error recovery, consistent with these dimensions assessing unrelated aspects of system behavior. No pairwise correlation exceeds 0.45, suggesting that the six dimensions capture substantially independent behavioral variation and that removing any dimension would discard information not recoverable from the remaining five.

6.2.5 Rubric Design

Each evaluation dimension is scored on a 1 to 5 Likert scale, where 1 indicates fundamentally flawed performance and 5 indicates exemplary performance. The use of an integer scale rather than a continuous scale reflects both the granularity at which LLM judges appear able to reliably discriminate performance levels and the practical consideration that finer distinctions would increase scoring noise without improving diagnostic utility. Preliminary experiments with a 1–10 scale produced lower inter-judge agreement and did not appear to yield additional diagnostic information, as the judges struggled to distinguish between adjacent levels on the finer scale.

The five score levels are defined by anchor descriptions that specify observable behavioral criteria for each level, helping reduce subjectivity and improving consistency across multiple LLM judges. Table 6.2 presents the anchor descriptions for the lowest and highest levels of each dimension. The intermediate levels are defined as follows: a score of 2 indicates performance that is predominantly flawed with occasional acceptable behavior, a score of 3

represents acceptable performance with identifiable weaknesses that do not render the system unreliable, and a score of 4 indicates strong performance with only minor imperfections that are unlikely to affect prediction outcomes. The full five-level rubric for all six dimensions is included verbatim in the LLM prompt.

Table 6.2: Rubric anchor descriptions for each evaluation dimension. Scores 2–4 interpolate between the anchors.

Dim.	**Score 1 (Fundamentally Flawed)**	**Score 5 (Exemplary)**
RD	Systematic misclassification of regime; threshold unresponsive to volatility changes	All regime transitions identified within one trading day; threshold adjustments proportional to volatility
RT	Data routed to wrong pathway; blending weight contradicts regime classification	Routing consistently matches market state; smooth blending transitions near threshold
AD	Parameters frozen or wildly oscillating; no response to condition changes within episode	Prompt, proportional adjustments within 1–2 days; no overshooting or oscillation
RC	Risk posture inappropriate for volatility (e.g., aggressive in high-VIX period)	Conservative during high volatility, confident during low volatility; smooth transitions
SC	Multiple contradictory actions across decision chain within episode	All decisions form logically consistent sequence; no contradictions
ER	No corrective action within 2 days of error; errors persist or worsen	Corrective adjustments within 1–2 days; subsequent predictions show measurable improvement

The anchor descriptions are designed to be domain-specific and actionable, referencing concrete behavioral indicators that the LLM judge can observe in the behavioral trace. Rather than relying on abstract quality descriptors, each anchor specifies what the system should or should not do under specific conditions, enabling the judge to map observed behavior directly to a score level. This design follows the principle established by Liu et al. [90] that structured evaluation criteria produce more reliable and reproducible assessments than open-ended scoring instructions.

The intermediate levels interpolate between the anchors with concrete behavioral criteria. For regime detection, a score of 2 indicates that the system misclassifies regimes on multiple days within the episode, with the threshold failing to adjust despite clear volatility changes; a score of 3 indicates correct classification on the majority of days but with one or two misclassifications near the threshold boundary, particularly when the reconstruction error hovers within 10% of τ; and a score of 4 indicates that all classifications are correct but threshold adjustments show minor delays of one trading day relative to observable volatility shifts.

For routing appropriateness, a score of 2 indicates that routing decisions frequently contradict the regime classification, with the blending weight set inappropriately for the detected regime on most days; a score of 3 indicates generally consistent routing but with abrupt blending weight changes near the threshold boundary that introduce prediction discontinuities; and a score of 4 indicates consistently appropriate routing with smooth blending transitions in most cases but minor discontinuities in one or two instances within the episode.

For risk calibration, a score of 2 indicates that the system's risk posture is partially aligned with volatility conditions but exhibits substantial mismatches during transitions between low and high volatility environments; a score of 3 indicates acceptable risk-proportional behavior during stable conditions but delayed or insufficient posture adjustment during regime transitions; and a score of 4 indicates that risk posture is well-calibrated throughout the episode with only minor imperfections in the timing of transitions between conservative and confident modes.

These intermediate-level descriptions calibrate the judges' expectations and reduce the variance in scores for borderline cases, ensuring that judges assign similar scores to similar behavioral patterns even when the patterns do not perfectly match the extreme anchors.

When multiple LLM judges are employed, their independent scores for each dimension provide an empirical measure of evaluation reliability. Agreement between judges suggests that the dimension is well-defined and the rubric criteria are unambiguous, while disagreement signals either rubric ambiguity or genuine edge cases where reasonable evaluators might differ. The framework uses Krippendorff's alpha to quantify multi-judge agreement, as this metric accounts for chance agreement and is suitable for ordinal data with more than two raters.

6.2.6 Prompt Design and Evaluation Pipeline

Prompt Structure

The LLM judge receives a structured prompt comprising four components. The first component is a system description that explains the architecture and purpose of the prediction system being evaluated, including the role of each component (autoencoder, router, dual node transformers, SAC controller) and the meaning of each field in the behavioral trace. This description is written in domain-specific terms that an expert human evaluator would use, providing the judge with the technical context necessary for informed assessment without assuming prior knowledge of the specific system. The system description explains what reconstruction error represents, how the threshold determines regime classification, what the blending weight controls, and what the SAC adjustment actions mean in practical terms. This description is fixed across all evaluation episodes and all LLM judges to ensure consistency. The verbatim system message used in all experiments is reproduced in Appendix A.

The second component is the behavioral trace sequence, consisting of the five traces $\{\mathcal{B}_t, \ldots, \mathcal{B}_{t+4}\}$ that constitute the evaluation episode. Each trace is serialized as a structured JSON record with labeled fields corresponding to the components defined in Equation 6.1. Numerical values are presented with units and ranges to facilitate interpretation: reconstruction errors are accompanied by the current threshold value and the ratio e_t/τ_t, blending weights are expressed as percentages, SAC adjustments are shown alongside the resulting

parameter values after application, and rolling performance metrics include both current values and directional trends (improving, stable, or deteriorating). The serialization preserves temporal ordering so the judge can track how decisions evolve across the five-day window. A representative serialized trace is provided in Appendix A.2.

The third component is the evaluation rubric, presented as a structured set of criteria for each dimension at each score level. The rubric text matches the anchor descriptions in Table 6.2, supplemented by intermediate level descriptions and examples of common behavioral patterns at each level. Including the full rubric in the prompt, rather than relying on the LLM's general knowledge, ensures that all judges apply the same evaluation standards regardless of their pre-training data or default calibration. The rubric is formatted with clear headers for each dimension and each score level, with bullet points listing observable criteria, to maximize readability within the prompt context window.

The fourth component specifies the required output format: a JSON object containing an integer score for each of the six dimensions, a natural-language justification for each score referencing specific observations from the behavioral trace, and an optional categorical failure label for any dimension scoring below the threshold $\theta = 3$. The failure labels (e.g., `delayed_threshold`, `wrong_routing`, `frozen_parameters`, `inconsistent_blend`) are drawn from a predefined vocabulary of 12 failure types that map to specific SAC action subspaces, enabling the credit assignment mechanism described in Section 6.2.7. The complete 12-label vocabulary, together with the output JSON schema, is given in Appendix A.3. Requiring structured JSON output ensures that the evaluation results can be parsed programmatically for downstream reward integration without manual intervention.

Chain-of-Thought Prompting

Following the approach of Liu et al. [90], the prompt instructs the LLM judge to analyze each decision in the behavioral trace before assigning scores. Specifically, the judge is instructed to proceed through four analytical steps: (1) summarize the market conditions

observed across the episode, identifying any notable transitions in volatility, sentiment, or price behavior; (2) examine each decision point and assess whether the decision was appropriate given the available information at the time it was made; (3) identify any inconsistencies, failures, or suboptimal behaviors in the decision chain, noting the specific trace fields that support each observation; and (4) assign scores based on the accumulated evidence, ensuring that each score is grounded in specific behavioral observations rather than an overall impression.

This sequential analysis reduces the frequency of arbitrary or inconsistent judgments by requiring the model to ground each score in specific behavioral observations before forming a conclusion. This structured stepwise analysis has been shown to improve evaluation fidelity across a range of NLP evaluation tasks [90], and this benefit extends to the more complex multi-step evaluation required for behavioral assessment of financial agents, where the judge must track causal chains across multiple decision points and days.

Judge Selection and Configuration

Three LLM judges evaluate all episodes independently: GPT 5.4, Claude 4.6 Opus, and Gemini 3.1 Pro. The use of multiple judges from different model families serves two purposes. First, it provides robustness against individual model biases. Different LLMs may weight certain behavioral patterns differently due to variations in their training data, reasoning capabilities, and implicit assumptions about financial markets. In preliminary analyses, GPT 5.4 and Claude 4.6 produced closely matched scores on the architectural decision dimensions (regime detection, routing, and adaptation), where the rubric criteria are largely objective, but diverged more on the emergent behavioral dimensions (risk calibration, strategy coherence, and error recovery), where appropriateness depends on judgment about contextual tolerance. Gemini 3.1 Pro produced scores that differed moderately from both other judges across most dimensions, applying somewhat different thresholds for what constitutes adequate behavior on borderline episodes. Detailed inter-judge agreement statistics and concrete

examples of these divergences are reported in Section 6.3.5. Aggregating across model families reduces the influence of any single model's idiosyncrasies. Second, the degree of agreement between judges provides a quality signal for each evaluation dimension. High agreement indicates that the dimension is well-defined and the rubric criteria are unambiguous, while low agreement may signal that the dimension requires refinement or that certain behavioral patterns are genuinely ambiguous.

All judges operate at temperature zero to produce deterministic, reproducible evaluations. While temperature zero eliminates the stochastic variation that might reveal model uncertainty, reproducibility is prioritized for a framework intended to generate reliable reward signals for downstream optimization. Each judge evaluates each episode independently, without access to other judges' scores or to previous episodes' evaluations, ensuring that scores reflect the judge's assessment of the specific episode in isolation. The maximum output token length is set to 2,000 tokens per evaluation to accommodate the chain-of-thought analysis and six justifications. Figure 6.2 illustrates the complete evaluation pipeline from behavioral trace collection through scoring to reward signal generation.

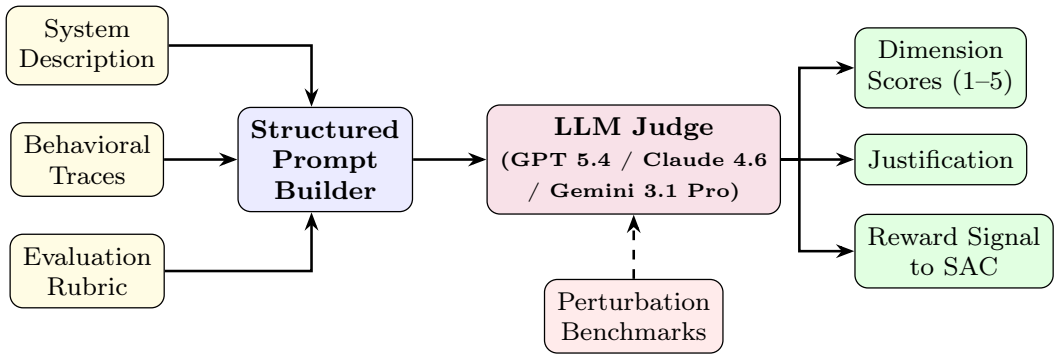

Figure 6.2: LLM-as-a-Judge evaluation pipeline. Behavioral traces, system description, and evaluation rubric are combined into a structured prompt. Outputs include per-dimension scores, natural-language justifications, and reward signals fed back into the SAC engine. Perturbation benchmarks validate the framework.

The composite evaluation score for an episode is computed as the unweighted arithmetic mean of the six dimension scores:

$$S_{\text{composite}} = \frac{1}{6}\sum_{d=1}^{6} s_d \qquad \text{where } s_d \in \{1, 2, 3, 4, 5\} \tag{6.4}$$

The composite score $S_{\text{composite}}$ serves as a single-number summary used for reporting and cross-episode comparisons, such as comparing average behavioral quality across volatility regimes (Section 6.3.6) or between baseline and intervention periods (Section 6.3.9). It is not used as the training signal. The closed-loop reward modification described in Section 6.2.7 (Equation 6.6) operates on the individual per-dimension scores $\bar{s}_d$ with empirically calibrated weights w_d, preserving the per-dimension diagnostic granularity that motivates the multi-dimensional evaluation framework; collapsing the six dimensions into a single number for training would discard the targeted feedback that allows the SAC controller to identify which behavioral dimension requires correction.

The unweighted formulation treats all dimensions as equally important for the composite assessment.

The per-dimension consensus score $\bar{s}_d$ is the primary unit of evaluation in the framework: it is the score that the closed-loop reward modification (Equation 6.6) uses as the training signal, and it is also the input from which the composite score $S_{\text{composite}}$ defined above is subsequently computed for reporting. Because three judges evaluate each episode independently, $\bar{s}_d$ is obtained by averaging the individual judge scores for dimension d:

$$\bar{s}_d = \frac{1}{J}\sum_{j=1}^{J} s_{d,j} \tag{6.5}$$

where $J = 3$ is the number of LLM judges and $s_{d,j}$ is the raw 1–5 score assigned to dimension d by judge j. This per-judge averaging attenuates outlier scores from any single model before $\bar{s}_d$ enters either the reward computation or the composite. The dimension weights w_d that pair with $\bar{s}_d$ in the reward modification are derived empirically from correlations between each dimension's consensus score and realized trading outcomes, as described in Section 6.3.7.

6.2.7 Closed-Loop Reward Integration

Reward Modification

Rather than serving as a passive diagnostic, the LLM judge is used to shape the system's learning by modifying the SAC reward function. The original SAC reward from Chapter 5 (Equation 5.6) combines prediction accuracy, directional correctness, and threshold stability. The closed-loop mechanism augments this reward with a penalty term derived from the LLM evaluation:

$$R'_t = R_t - \lambda \sum_{d=1}^{6} \max(0,\ \theta - s_d) \cdot w_d \tag{6.6}$$

where R_t is the original SAC reward, $\lambda = 0.15$ controls the influence of LLM feedback on the total reward signal, $\theta = 3$ is the minimum acceptable score threshold, s_d is the LLM score for dimension d, and w_d is the weight assigned to dimension d.

The threshold $\theta = 3$ corresponds to the midpoint of the 1–5 scale and represents the boundary between acceptable and deficient performance. Dimensions scoring at or above 3 receive no penalty, ensuring that the LLM feedback targets only identified weaknesses rather than perturbing already-adequate behaviors. The max function ensures that the penalty is non-negative and scales linearly with the severity of the deficiency: a dimension scoring 1 contributes a penalty of $2 \cdot w_d$, while a dimension scoring 2 contributes $1 \cdot w_d$. This linear scaling was preferred over quadratic or exponential alternatives because it maintains proportionality without introducing excessively large penalties for severe deficiencies that could destabilize SAC training.

The feedback strength λ was selected through a systematic sweep on the validation set (2011–2016), evaluating six values spanning an order of magnitude. For each candidate value, three full evaluation cycles were executed using the validation-period data, and the resulting

prediction and behavioral metrics were recorded after the third cycle. Table 6.3 reports the results and Figure 6.3 visualizes the relationship between λ and both prediction quality and behavioral improvement.

Table 6.3: Sensitivity of closed-loop outcomes to the feedback strength λ, measured after three evaluation cycles on the validation set. MAPE and DA reflect prediction quality; $\bar{s}_{\text{def}}$ is the mean score across originally deficient dimensions; stability indicates whether SAC training converged without oscillation.

λ	MAPE (%)	DA (%)	$\bar{s}_{\text{def}}$	Stability
0.05	0.60	72	2.8	Stable
0.10	0.57	73	3.1	Stable
0.15	0.54	74	3.6	Stable
0.20	0.55	74	3.4	Stable
0.25	0.58	72	3.0	Marginal
0.30	0.63	71	2.5	Unstable

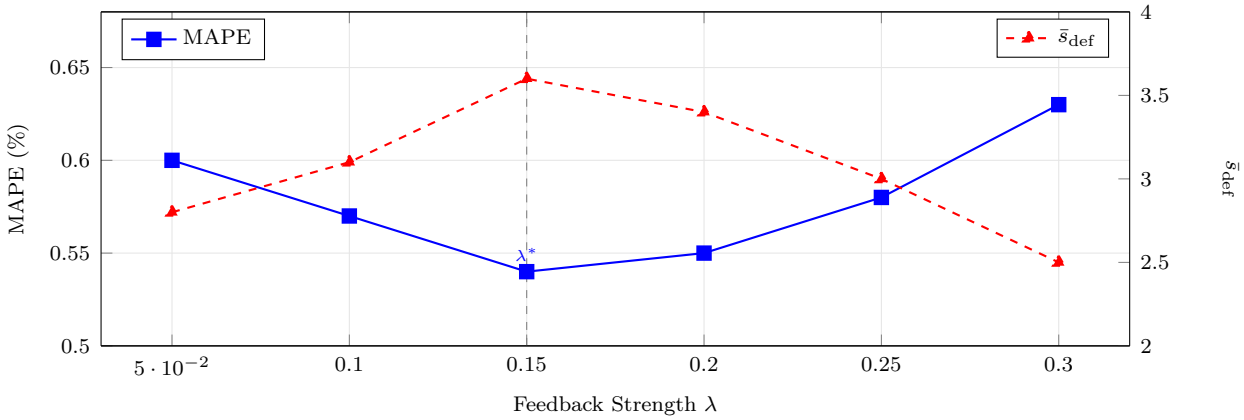

Figure 6.3: MAPE (left axis, solid blue) and mean deficient dimension score $\bar{s}_{\text{def}}$ (right axis, dashed red) as a function of the feedback strength λ. The vertical dashed line marks the selected value $\lambda^* = 0.15$, which minimizes MAPE while maximizing behavioral improvement. Values beyond 0.25 destabilize SAC training.

At $\lambda = 0.05$, the LLM-derived penalty is dominated by the primary reward components from Equation 5.6, producing only marginal behavioral improvement (deficient dimension mean rising from approximately 2.4 to 2.8). Increasing λ to 0.10 and then 0.15 yields progressively larger improvements in both prediction metrics and behavioral scores, with $\lambda = 0.15$ achieving the best overall configuration. Beyond this point, performance degrades: at $\lambda = 0.20$ the system begins to over-correct on behavioral dimensions at a slight cost

to prediction accuracy, and at $\lambda = 0.25$ the reward signal becomes sufficiently volatile to cause marginal training stability, with the SAC loss exhibiting periodic spikes. At $\lambda = 0.30$ the system becomes unstable, with the controller alternating between optimizing for LLM approval and optimizing for prediction accuracy on successive training batches, resulting in oscillatory parameter behavior and final metrics that are worse than the pre-intervention baseline. The selected value of $\lambda = 0.15$ therefore represents the point at which LLM feedback is strong enough to induce measurable behavioral change while remaining subordinate to the primary prediction objective.

The dimension weights w_d are set equal to the Spearman correlation coefficients between each dimension's scores and the realized Sharpe ratio over the subsequent 20 trading days, as reported in Table 6.9. This data-driven weighting ensures that the reward penalty emphasizes dimensions with the greatest demonstrated impact on realized trading performance. Dimensions that correlate more strongly with risk-adjusted returns receive larger penalties when deficient, directing the SAC controller's learning toward the behavioral improvements most likely to translate into prediction gains. The weights are computed once from the validation episodes and held fixed during the closed-loop improvement cycles to prevent circular optimization in which the system simultaneously changes the behavior being evaluated and the weights used to evaluate it.

Credit Assignment

Translating a dimension-level score into actionable learning requires routing the penalty to the correct component of the SAC action space. The SAC controller outputs a two-dimensional action $a_t = [\Delta\tau, \Delta\alpha]$ (Section 6.2.1), where $\Delta\tau$ adjusts the anomaly detection threshold and $\Delta\alpha$ adjusts the pathway blending weight. Different evaluation dimensions correspond to different aspects of these two action components, creating a credit assignment problem: when the LLM identifies a deficiency in routing appropriateness, the penalty should primarily affect the blending weight learning, not the threshold learning.

The framework addresses credit assignment by decomposing the penalty through a credit assignment vector $\mathbf{c}_d = [c_{d,\tau},\ c_{d,\alpha}]$ that routes each dimension's penalty to the appropriate action component. The dimension-specific penalty contribution to the SAC reward is:

$$\Delta R_d = \max(0,\ \theta - \bar{s}_d) \cdot w_d \cdot \mathbf{c}_d \tag{6.7}$$

where $\mathbf{c}_d$ is defined as $[1, 0]$ for dimensions targeting $\Delta\tau$ only (RD, RC), $[0, 1]$ for dimensions targeting $\Delta\alpha$ only (RT), and $[0.5, 0.5]$ for dimensions targeting both subspaces (AD, SC, ER). Table 6.4 specifies the complete mapping from each dimension to its governing SAC action subspace.

Table 6.4: Credit assignment mapping from evaluation dimensions to SAC action subspaces.

Dimension	**Target Subspace**	**Rationale**
Regime Detection	$\Delta\tau$	Threshold directly controls regime classification boundary
Routing	$\Delta\alpha$	Blending weight determines pathway contribution
Adaptation	$\Delta\tau$, $\Delta\alpha$	Adaptation quality reflects both threshold and blending tuning
Risk Calibration	$\Delta\tau$	Risk posture primarily determined by regime sensitivity
Strategy Coherence	$\Delta\tau$, $\Delta\alpha$ (uniform)	Coherence is a system-level property not localizable to one action
Error Recovery	$\Delta\tau$, $\Delta\alpha$	Recovery requires coordinated adjustment of both parameters

The total penalty from Equation 6.6 is thus applied selectively: the $\Delta\tau$ component of the SAC action receives penalty contributions only from dimensions whose credit assignment vector has a non-zero first element, and the $\Delta\alpha$ component receives contributions only from dimensions with a non-zero second element. This selective routing ensures that the LLM feedback corrects the specific aspect of the controller's behavior that the evaluation identified as deficient, rather than applying a diffuse penalty that could inadvertently degrade well-functioning components. The ablation study in Section 6.3.9 confirms that this targeted credit

assignment preserves the quality of non-deficient dimensions throughout the improvement cycles.

The LLM judge's structured JSON output facilitates credit assignment by including categorical failure labels alongside numerical scores. When a dimension receives a score below θ, the judge assigns a failure label from a predefined vocabulary (e.g., `delayed_threshold`, `wrong_routing`, `frozen_parameters`, `inconsistent_blend`). These labels are mapped to action subspaces via the lookup table, providing a secondary routing signal that confirms the dimensional mapping. In cases where the failure label suggests a mapping different from the default dimensional mapping, the label-based mapping takes precedence, allowing for finer-grained credit assignment that accounts for atypical failure modes where, for example, a routing deficiency originates from an incorrect threshold rather than from the blending weight itself.

Periodic Evaluation Protocol

The LLM evaluation operates periodically rather than at every trading day, reflecting both practical cost constraints and the episodic nature of the behavioral assessment. Evaluating every trading day independently would cost approximately $3,600 per year across three judges and 20 stocks, which is not prohibitive in absolute terms but would produce a continuous stream of reward modifications that could overwhelm the SAC controller's primary learning signal. The periodic protocol strikes a balance between evaluation frequency and training stability.

The protocol proceeds in cycles. At the beginning of each cycle, the system operates normally for a period of 40 trading days (approximately two calendar months), generating behavioral traces at each decision point without any LLM-derived reward modification. The traces are then grouped into non-overlapping five-day evaluation episodes, yielding eight episodes per cycle. All episodes are submitted to the three LLM judges for independent evaluation.

The resulting dimension scores are averaged across judges and across episodes to produce mean dimension scores for the cycle. If any mean dimension score falls below $\theta = 3$, the reward modification from Equation 6.6 is activated with the corresponding penalties. The SAC controller is then fine-tuned for 10 epochs using the modified reward function, with all transitions from the current cycle's experience replay buffer. The fine-tuning uses the same optimizer settings as the original SAC training (Adam, learning rate 3×10^{-4}, soft target updates with $\tau_{\text{soft}} = 0.005$). After fine-tuning, the modified reward is deactivated and the system resumes normal operation for the next cycle.

This periodic structure prevents the LLM feedback from dominating the SAC reward signal during normal operation, which could lead to the system optimizing for LLM approval rather than prediction accuracy. The LLM feedback acts as a corrective intervention that addresses specific identified weaknesses, after which the system returns to optimizing its primary prediction objective. Three evaluation cycles were conducted in the experiments reported in Section 6.3.9, at which point dimension scores converged above the threshold θ for all dimensions, and the magnitude of reward modifications decreased below a level that would produce further meaningful behavioral changes.

6.3 Experiments and Results

6.3.1 Experimental Setup

The evaluation experiments use the same 20 S&P 500 equities and temporal partitioning described in Chapter 3 (training 1982–2010, validation 2011–2016, test 2017–2025), following the same data discipline as Chapter 5: the test period is reserved exclusively for final reporting and is never used for adaptation, hyperparameter tuning, prompt selection, or any analysis whose result feeds back into the system. The closed-loop adaptation pipeline introduced in this chapter is therefore confined entirely to the validation period: all closed-loop SAC

fine-tuning cycles, LLM-judge evaluation episodes used to derive cycle scores, hyperparameter selection (λ and θ), prompt-sensitivity analyses, and ablation comparisons operate on data drawn from the 2011–2016 validation window. After the closed-loop process concludes, the resulting SAC weights are frozen and the system is applied without further updates to the held-out 2017–2025 test set for the final reporting of prediction performance in Section 6.3.10.

For the LLM-judge analyses that drive the closed-loop mechanism, 200 unperturbed evaluation episodes were sampled from the validation period, stratified by market regime to ensure representation across conditions. The validation window encompasses a range of market regimes useful for stratification, including the European sovereign debt crisis (2011–2012), the 2014–2015 oil price crash, the 2015 Chinese stock market turbulence, and the 2016 Brexit period. The regime stratification uses the CBOE Volatility Index (VIX) as a proxy for market conditions: 70 episodes from low-volatility periods (VIX below 15), 90 episodes from medium-volatility periods (VIX between 15 and 25), and 40 episodes from high-volatility periods (VIX at or above 25). This distribution reflects the empirical frequency of each regime in the validation window while ensuring that the less common high-volatility regime has sufficient representation for reliable statistical analysis. Episode boundaries were constrained to lie within a single regime classification to avoid episodes that span a regime transition, which would confound the evaluation by mixing behaviors appropriate for different conditions.

Three LLM judges (GPT 5.4, Claude 4.6 Opus, and Gemini 3.1 Pro) evaluated all 200 episodes at temperature zero. Each evaluation call provided the complete behavioral trace sequence, system description, rubric, and output format specification as described in Section 6.2.6. The judges operated independently, with no access to other judges' scores or to scores from previous episodes. The total evaluation cost was approximately $180 across all three models and 200 episodes, with individual episode evaluations completing in 8–15 seconds per judge. All API calls were made through the respective providers' batch processing

endpoints to reduce latency and cost. Table 6.5 summarizes the key configuration parameters for the evaluation experiments.

Table 6.5: Evaluation framework configuration parameters.

Parameter	**Value**	**Rationale**
Episode length	5 trading days	Matches SAC history window ($k = 5$)
Episode overlap	None	Statistical independence
Total episodes	200	Sufficient for regime stratification
Low-vol. episodes	70 (VIX < 15)	Proportional to regime frequency
Medium-vol. episodes	90 (VIX 15–25)	Proportional to regime frequency
High-vol. episodes	40 (VIX ≥ 25)	Over-sampled for statistical power
Evaluation period	2011 – 2016	Validation window (test held out for reporting)
LLM temperature	0	Deterministic, reproducible scores
Max output tokens	2,000	Accommodates CoT + 6 justifications
Score scale	1–5 (integer)	Optimal LLM discrimination
Score threshold (θ)	3	Midpoint of Likert scale
Feedback strength (λ)	0.15	Validated on 2011–2016 period
Cycle length	40 trading days	Balance evaluation frequency / stability
SAC fine-tuning epochs	10 per cycle	Sufficient for penalty integration
Perturbation episodes	60 baseline $\times$ 6 types	420 total perturbed + baseline

6.3.2 Perturbation Design

To validate that the LLM judges can reliably detect specific behavioral deficiencies, the framework uses perturbation-based benchmarking. The validation set consists of 60 baseline (unperturbed) episodes drawn from the 200 evaluation episodes, with 20 episodes from each regime stratum. Six perturbation types were designed, each targeting exactly one evaluation dimension while leaving the other five dimensions mechanically unaffected. Each perturbation is defined as a transformation applied to the behavioral trace components. Let $\mathcal{B}_t$ denote the original trace and $\mathcal{B}'_t$ the perturbed trace; the six transformations are:

$$\text{Regime inversion (RD):} \quad \ell'_t = 1 - \ell_t, \quad e_t, \tau_t \text{ unchanged} \tag{6.8}$$

$$\text{Wrong routing (RT):} \quad \alpha'_t = 0.9 \cdot \ell_t + 0.1 \cdot (1 - \ell_t) \tag{6.9}$$

$$\text{Frozen SAC (AD):} \quad \mathbf{u}'_t = \mathbf{0} \;\; \forall\, t \in \mathcal{E} \tag{6.10}$$

$$\text{No vol. scaling (RC):} \quad f_{\text{VIX}}(\cdot) \to 1 \tag{6.11}$$

$$\text{Contradictory (SC):} \quad \exists\, i \neq j :\; a_i \perp a_j \text{ per day} \tag{6.12}$$

$$\text{Disabled recovery (ER):} \quad \mathbf{u}'_{t+1} = \mathbf{u}'_{t+2} = \mathbf{0} \;\text{ if } \xi_t = 1 \tag{6.13}$$

The following paragraphs describe each perturbation in detail, including its implementation and why it targets its specific dimension.

Regime label inversion (Equation 6.8) flips the autoencoder's binary regime classification at each day in the episode, replacing normal labels with anomalous and vice versa, while leaving the reconstruction error and threshold values unchanged. This creates a clear contradiction between the classification and the underlying evidence, as the reconstruction error still reflects the true market conditions while the label indicates the opposite regime. The LLM judge can detect this by observing that the regime label contradicts the reconstruction error magnitude and the concurrent VIX level.

Forced wrong-pathway routing (Equation 6.9) overrides the routing decision to direct data to the opposite pathway from what the regime detection indicates, regardless of the blending weight. Specifically, when the regime label is "normal," the routing is forced to the event pathway with $\alpha = 0.1$ (heavily favoring the event pathway), and vice versa. This preserves the regime detection as correct while making the routing decision systematically inappropriate, isolating the RT dimension from RD.

Frozen SAC parameters (Equation 6.10) prevents the SAC controller from adjusting τ or α for the duration of the episode, holding both parameters at their values from the day before

the episode begins. The SAC action vectors $\mathbf{u}_t$ are set to zero for all five days, regardless of market conditions. The resulting trace represents a system that detects and classifies regimes correctly but fails to adapt its parameters, directly targeting the adaptation dimension.

Removed volatility scaling (Equation 6.11) disables the VIX-based scaling in the prediction confidence and risk management components, making the system's risk posture independent of current volatility conditions. The system operates with the same parameter sensitivity and blending behavior regardless of whether the VIX is at 12 or 35, eliminating the risk-proportional behavior that the RC dimension assesses.

Randomized contradictory actions (Equation 6.12) injects logically inconsistent decisions by randomly replacing one decision component per day with a value that contradicts the other components. For example, the regime label may be set to "anomalous" while the blending weight is simultaneously set to $\alpha = 0.9$ (heavily favoring the normal pathway), or the SAC action may increase τ while simultaneously shifting α toward the event pathway. These contradictions target the coherence dimension while preserving individual component outputs that would otherwise score normally on their own dimensions.

Disabled recovery (Equation 6.13) prevents the system from adjusting parameters for two consecutive days following any prediction error within the episode, producing a trace in which the system fails to respond to its own mistakes. On days where $\xi_t = 1$ (Equation 6.3), the SAC actions are forced to zero for the subsequent two days, regardless of market conditions. This isolates the recovery dimension by allowing normal regime detection, routing, and adaptation on non-error days while specifically degrading the error response behavior.

Each perturbation was applied independently to all 60 baseline episodes, yielding $6 \times 60 = 360$ perturbed episodes. Combined with the 60 unperturbed originals, the perturbation validation set comprises 420 episodes. All three LLM judges evaluated the complete set, producing $420 \times 3 = 1{,}260$ individual evaluation results.

6.3.3 Baseline System Performance

Table 6.6 reports one-day-ahead prediction performance for the agentic system and selected baselines prior to the LLM feedback loop. These results establish the starting point from which the closed-loop improvement is measured. The full set of baseline comparisons, per-stock analyses, and multi-horizon results are reported in Chapter 5; the abbreviated table here provides context for the evaluation experiments.

Table 6.6: One-day-ahead prediction performance before LLM feedback loop. DA = Directional Accuracy, CTR = Confidence Tracking Rate.

Model	**MAPE**	**RMSE**	**DA**	**Theil's U**	**CTR**
ARIMA	1.20%	1.35	55%	0.98	51%
LSTM	1.00%	1.20	58%	0.88	54%
NodeFormer-BERT [134]	0.80%	0.95	65%	0.72	62%
AE-NodeFormer (no SAC)	0.68%	0.88	69%	0.68	64%
AE-NodeFormer + SAC	**0.61%**	**0.82**	**71%**	**0.63**	**67%**

The AE-NodeFormer + SAC system from Chapter 5, applied without any closed-loop fine-tuning, achieves 0.61% MAPE and 71% directional accuracy on the validation-period episodes used in this chapter, representing a 24% MAPE reduction over the single-pathway NodeFormer-BERT baseline from Chapter 4. These validation-period figures are within sampling noise of the test-period results reported in Chapter 5 (0.59% MAPE, 72% DA across the complete 2017–2025 test set), confirming that the validation window used here provides a representative operating regime for the system. The remainder of this chapter compares this validation-period baseline against the closed-loop fine-tuned variant for behavioral and per-cycle analyses, and reports the held-out test-set prediction performance separately in Section 6.3.10. Performance at 5-day and 20-day horizons follows a similar pattern, with MAPE of 1.02% and 1.52% respectively.

6.3.4 Perturbation-Based Validation

Table 6.7 reports the perturbation validation results. For each perturbation type, the table shows the targeted dimension, the mean score drop on that dimension (relative to the unperturbed baseline), the average score change on the remaining five dimensions (off-target shift), Krippendorff's alpha measuring cross-model agreement, and the statistical significance of the targeted drop.

Table 6.7: Perturbation validation results. Δs: mean targeted score drop (GPT 5.4). Off-target Δ: average change on remaining five dimensions. α: Krippendorff's cross-model agreement.

Perturbation	**Target**	Δs	**Off-tgt. Δ**	α	p**-value**
Regime inversion	RD	-2.4	-0.3	0.85	<0.001
Wrong routing	RT	-2.1	-0.4	0.81	<0.001
Frozen SAC	AD	-1.9	-0.2	0.79	<0.001
No vol. scaling	RC	-1.7	-0.3	0.74	<0.001
Contradictory action	SC	-2.2	-0.5	0.82	<0.001
Disabled recovery	ER	-1.6	-0.2	0.76	<0.001

Every perturbation produces a statistically significant targeted score drop ($p < 0.001$, paired t-test comparing perturbed episodes against their unperturbed baselines on the targeted dimension). The mean targeted drops range from -1.6 for disabled recovery to -2.4 for regime inversion, indicating that all six perturbation types produce substantial and detectable degradation in their targeted dimension. Regime inversion produces the largest targeted drop because regime detection is the first decision in the processing chain; inverting it produces a behavioral pattern that is unambiguously incorrect, as the system classifies stable markets as volatile and volatile markets as stable. All three LLM judges consistently identify this contradiction between the regime label and the market context evidence. Disabled recovery produces the smallest targeted drop because error recovery is an emergent property that depends on the specific error patterns within each episode; in episodes where no significant prediction errors occur during the five-day window, recovery behavior cannot be meaningfully assessed, which attenuates the mean effect across the full episode set.

The off-target dimension shifts average 0.32 across all perturbation types and dimensions, which is substantially smaller than the targeted drops. This asymmetry confirms dimension specificity: each perturbation degrades its targeted dimension far more than any other. The largest off-target shift (-0.5) occurs for contradictory actions, which is expected because logically inconsistent actions impair downstream decision quality even on dimensions other than coherence. The smallest off-target shifts (-0.2) occur for frozen SAC and disabled recovery, indicating that these perturbations are well-isolated in their effects on the behavioral trace. Figure 6.4 visualizes this contrast across all six perturbation types.

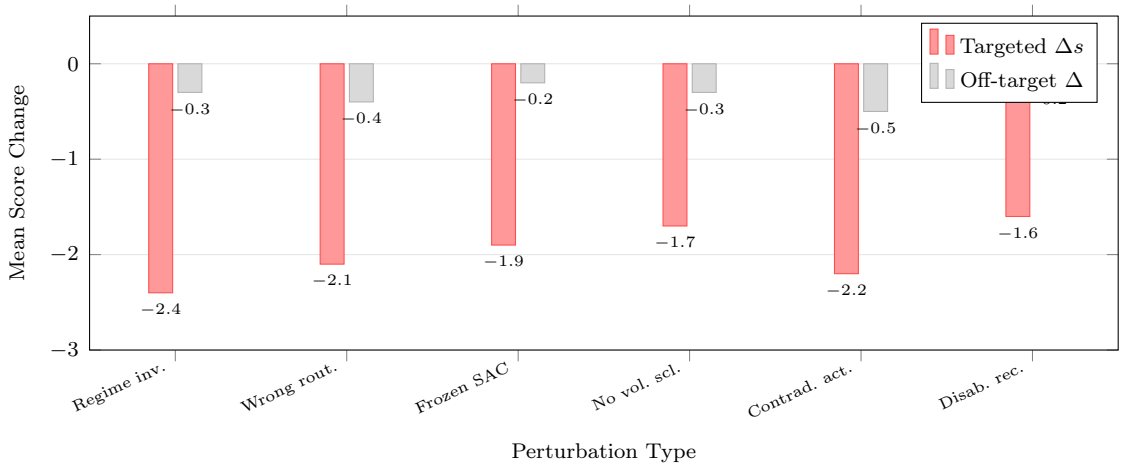

Figure 6.4: Targeted score drops (red) versus mean off-target shifts (gray) for each perturbation type. The large gap between targeted and off-target effects confirms dimension specificity: each perturbation degrades its intended dimension far more than other dimensions.

Non-zero off-target shifts are expected and, to a degree, desirable. They reflect the fact that the system's decision chain is interconnected: a perturbation that degrades regime detection will inevitably have some downstream effect on routing and adaptation quality, because subsequent decisions operate on incorrect regime information. A framework that showed zero off-target sensitivity would be suspect, as it would imply that the dimensions are assessed in isolation from the decision context rather than in the integrated manner that behavioral evaluation requires. The observed pattern of small but non-zero off-target shifts indicates that the LLM judges appropriately recognize downstream effects while maintaining primary sensitivity to the targeted dimension.

6.3.5 Cross-Model Agreement

Cross-model agreement is quantified using Krippendorff's alpha, a reliability measure that generalizes across different numbers of raters and rating scales while correcting for chance agreement. For J judges rating N episodes on an ordinal scale, the statistic is defined as:

$$\alpha_K = 1 - \frac{D_o}{D_e} = 1 - \frac{(N-1)\sum_{c,k} o_{ck}\,\delta^2_{ck}}{\sum_{c,k} n_c\,n_k\,\delta^2_{ck}} \tag{6.14}$$

where D_o is the observed disagreement computed from the coincidence matrix of rater-pair assignments, D_e is the disagreement expected by chance, o_{ck} counts coincidences between categories c and k, n_c is the marginal frequency of category c, and δ^2_{ck} is the squared difference function appropriate for ordinal data. Values of $\alpha_K = 1$ indicate perfect agreement, $\alpha_K = 0$ indicates agreement no better than chance, and thresholds of 0.667 and 0.800 are commonly accepted for tentative and reliable conclusions, respectively [22].

Across the six evaluation dimensions, α_K ranges from 0.74 for risk calibration to 0.85 for regime detection. All dimensions exceed the 0.667 threshold, with regime detection, routing, and strategy coherence exceeding the 0.800, indicating comparatively stronger inter-judge consistency.

Regime detection achieves the highest agreement ($\alpha = 0.85$) because it involves the most objective assessment: the LLM judges can directly compare the reconstruction error against the threshold and verify whether the regime label is consistent with the market context. A regime that is obviously stable (low VIX, low price volatility, no sentiment spikes) classified as anomalous is a clear error that all three models consistently identify. Risk calibration achieves the lowest agreement ($\alpha = 0.74$) because it involves the most subjective assessment: what constitutes an appropriate risk posture for a given volatility level is a judgment that depends on implicit assumptions about risk tolerance and market expectations. Different

LLM models may apply slightly different standards for what constitutes conservative versus aggressive behavior, leading to lower but still meaningful agreement.

The pattern of agreement provides diagnostic information about the evaluation instrument itself. Dimensions with high agreement are well-defined by the rubric and can be reliably assessed by different judges. Dimensions with lower agreement may benefit from more specific rubric criteria or from the inclusion of reference examples that calibrate the judges' expectations. For the present framework, even the lowest agreement ($\alpha = 0.74$) exceeds the threshold for meaningful inter-rater reliability, indicating that all six dimensions produce sufficiently consistent evaluations across model families to support their use in downstream reward modification.

Examining pairwise agreement between specific models reveals that GPT 5.4 and Claude 4.6 exhibit the highest pairwise agreement (Cohen's kappa [92] ranging from 0.71 to 0.83 across dimensions), while both show slightly lower agreement with Gemini 3.1 Pro (kappa 0.65 to 0.79). This pattern suggests that GPT 5.4 and Claude 4.6 applied similar evaluation standards in this setting, while Gemini 3.1 Pro tended to interpret the rubric criteria somewhat differently on the subjective dimensions of risk calibration and error recovery. The divergence was not a uniform leniency or strictness bias: on identical episodes, Gemini 3.1 Pro sometimes assigned a higher score than the other two judges by treating borderline parameter lags as within acceptable response time, and sometimes assigned a lower score by penalizing minor recovery overshoots that GPT 5.4 and Claude 4.6 treated as proportional. Concrete examples of disagreement in both directions are reported in Section 6.3.8. Averaging across all three models provides robustness against this calibration difference and yields more stable composite scores than any single judge would produce.

6.3.6 Per-Regime Score Analysis

To understand how behavioral quality varies across market conditions, Table 6.8 reports the mean dimension scores from GPT 5.4 across the three regime strata on the 200 unperturbed evaluation episodes. GPT 5.4 is shown here as a presentation choice rather than a claim about judge quality: it has the highest pairwise agreement with the other two judges and therefore offers the most representative single-judge view of the ensemble for tabular reporting. This does not imply that GPT 5.4 is the most accurate or authoritative judge; all training and downstream analysis continue to operate on the per-dimension consensus score $\bar{s}_d$ averaged over all three judges, and the regime-dependent patterns produced by Claude 4.6 and Gemini 3.1 Pro are qualitatively similar, so the conclusions drawn from this table would not change under a different choice of representative judge. Figure 6.5 visualizes the per-regime patterns across all six dimensions.

Table 6.8: Mean dimension scores (± standard error) by volatility regime (GPT 5.4, unperturbed episodes). Scores on a 1–5 scale. Standard errors computed across episodes within each stratum ($n_{\text{low}} = 82$, $n_{\text{med}} = 74$, $n_{\text{high}} = 44$).

Regime	**RD**	**RT**	**AD**	**RC**	**SC**	**ER**
Low vol. (VIX < 15)	$4.1_{\pm 0.07}$	$3.8_{\pm 0.09}$	$3.7_{\pm 0.11}$	$3.4_{\pm 0.12}$	$3.9_{\pm 0.08}$	$3.3_{\pm 0.14}$
Medium vol. (VIX 15–25)	$3.8_{\pm 0.09}$	$3.3_{\pm 0.12}$	$3.0_{\pm 0.14}$	$2.6_{\pm 0.17}$	$3.5_{\pm 0.11}$	$2.8_{\pm 0.15}$
High vol. (VIX ≥ 25)	$3.3_{\pm 0.12}$	$2.6_{\pm 0.16}$	$2.4_{\pm 0.18}$	$2.1_{\pm 0.19}$	$2.9_{\pm 0.14}$	$3.1_{\pm 0.17}$

Behavioral quality generally degrades from low to high volatility, though the pattern is not uniform across dimensions. The degradation is most severe for risk calibration (RC) and adaptation responsiveness (AD), both declining by 1.3 points from low to high volatility. Risk calibration drops from 3.4 during low volatility to 2.1 during high volatility, indicating that the system's risk management is least adequate precisely when it matters most: during volatile periods where inappropriate risk posture can lead to large drawdowns. Routing appropriateness declines by 1.2 points, reflecting the difficulty of making stable routing decisions when the reconstruction error fluctuates rapidly near the threshold.

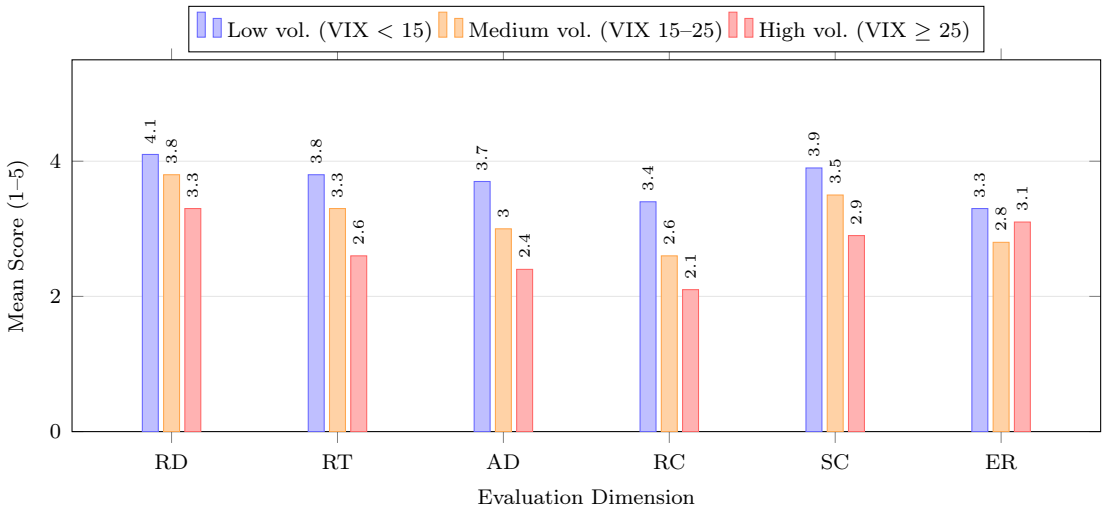

Figure 6.5: Mean evaluation scores by volatility regime across all six dimensions (GPT 5.4, 200 unperturbed episodes). Behavioral quality generally degrades from low to high volatility, with risk calibration (RC) and adaptation responsiveness (AD) showing the steepest declines. Error recovery (ER) exhibits a non-monotonic pattern, scoring higher during high-volatility episodes than medium-volatility episodes.

Error recovery is a notable exception to the general degradation pattern: it scores higher during high-volatility episodes (3.1) than during medium-volatility episodes (2.8). This reversal occurs because high-volatility periods produce more frequent and more pronounced prediction errors, providing the system with clearer signals for corrective action and more opportunities to demonstrate effective recovery behavior. During medium-volatility conditions, errors are less severe and the appropriate corrective response is more ambiguous, leading to lower recovery scores.

Regime detection maintains relatively high scores even during high-volatility episodes (3.3), indicating that the autoencoder reliably identifies volatile conditions even when other aspects of the system's behavior deteriorate. This robustness suggests that the primary bottleneck during volatile periods is not the detection of anomalous conditions but the system's response to them, specifically the calibration of risk posture and routing decisions.

The per-regime analysis provides the diagnostic motivation for the closed-loop improvement mechanism: the system's behavioral weaknesses are concentrated in specific dimensions during specific market conditions, and a targeted intervention that addresses these weaknesses should

produce disproportionate gains during the conditions where the original system is most vulnerable. This prediction is confirmed by the closed-loop results in Section 6.3.9, where the largest improvements occur during high-volatility episodes.

6.3.7 Predictive Validity Analysis

A behavioral evaluation framework is useful only if its scores predict future system performance. To assess predictive validity, Spearman rank correlations were computed between GPT 5.4's dimension scores on each unperturbed episode and realized trading metrics over the 20 trading days following the episode. The 20-day forward window was selected to capture the medium-term consequences of behavioral quality without extending so far that the connection between the evaluated behavior and subsequent outcomes is diluted by intervening events. Table 6.9 reports these correlations.

Table 6.9: Spearman ρ between GPT 5.4 dimension scores and realized trading metrics over the subsequent 20 trading days. MAPE^{-1} and MDD^{-1} are inverted so that higher values indicate better performance. All correlations significant at $p < 0.01$.

Dimension	**Sharpe**	**MAPE^{-1}**	**MDD^{-1}**	**Return**
Regime Detection	0.64	**0.69**	0.51	0.54
Routing	0.59	0.61	0.47	0.51
Adaptation	0.58	0.53	0.56	0.52
Risk Calibration	0.55	0.44	**0.64**	0.43
Strategy Coherence	0.62	0.59	0.49	0.57
Error Recovery	0.51	0.48	0.54	0.46
Composite	**0.72**	0.68	0.61	**0.59**

The composite score correlates most strongly with the Sharpe ratio ($\rho = 0.72$), indicating that overall behavioral quality as assessed by the LLM judge is a meaningful predictor of risk-adjusted returns over the subsequent month. This correlation is notable because the Sharpe ratio integrates both the magnitude of returns and their consistency, meaning that behavioral quality predicts not just average performance but also performance stability.

Individual dimensions exhibit distinct predictive profiles that align with their conceptual definitions. Regime detection is the strongest predictor of inverse MAPE ($\rho = 0.69$), consistent with the expectation that accurate regime classification enables appropriate model selection and thus better prediction accuracy. Risk calibration is the strongest predictor of inverse maximum drawdown ($\rho = 0.64$), reflecting the fact that appropriate risk posture during volatile periods directly limits downside exposure. The fact that risk calibration predicts drawdown better than any other dimension, despite showing the lowest cross-model agreement, suggests that even imperfect risk calibration assessments capture genuine behavioral variation that influences tail risk outcomes. Strategy coherence shows the strongest individual correlation with returns ($\rho = 0.57$), suggesting that logical consistency across the decision chain translates into more reliable exploitation of predictive signals.

Error recovery and risk calibration show the weakest Sharpe correlations ($\rho = 0.51$ and 0.55 respectively), consistent with their roles as reactive and risk-focused dimensions. Risk calibration correlates weakly with prediction accuracy (MAPE^{-1} $\rho = 0.44$) but strongly with drawdown, indicating that its contribution to outcomes operates through risk management rather than forecasting. Error recovery shows moderate correlations across most metrics (ρ ranging from 0.46 to 0.54), reflecting the fact that recovery behavior only manifests when errors occur; in episodes where the system performs well throughout, recovery scores provide limited predictive information. Nonetheless, even the weakest individual correlation is statistically significant ($p < 0.01$), confirming that all six dimensions capture behaviorally meaningful variation that relates to realized trading outcomes.

These correlations serve a dual purpose in the framework. As validation, they confirm that the LLM evaluation scores are not arbitrary but reflect genuine behavioral properties that influence realized outcomes, providing convergent validity for the evaluation instrument. As calibration, they provide the empirical basis for the dimension weights w_d in the reward modification equation (Equation 6.6). Setting w_d equal to each dimension's Sharpe correlation

ensures that the closed-loop penalty emphasizes the behavioral dimensions with the greatest demonstrated impact on the metric most relevant to risk-adjusted trading performance.

6.3.8 Qualitative Analysis of LLM Judgments

To illustrate the nature of the LLM evaluation outputs, this section presents representative judgment patterns from the three judges across different behavioral scenarios. These examples demonstrate that the judges produce substantive, evidence-grounded assessments rather than generic quality ratings.

On a high-quality episode during stable market conditions (VIX averaging 13.2), all three judges assigned scores of 4 or 5 across all dimensions. The justifications referenced specific behavioral observations: GPT 5.4 noted that "the reconstruction error remains consistently below the threshold with appropriate margin (ratio e_t/τ_t between 0.45 and 0.62), and the blending weight correctly favors the normal pathway at $\alpha = 0.82$–0.88, reflecting high confidence in the stable regime classification." Claude 4.6 highlighted that "SAC adjustments are minimal ($|\Delta\tau| < 0.005$, $|\Delta\alpha| < 0.02$) throughout the episode, which is appropriate given the absence of condition changes requiring parameter modification." These justifications demonstrate that the judges connect their scores to specific trace fields rather than relying on general impressions.

On a challenging episode spanning a volatility spike (VIX rising from 18 to 29 over five days), the judges assigned lower scores on risk calibration and adaptation. GPT 5.4 assigned RC = 2, noting that "the system maintains $\alpha = 0.75$ on day 3 despite VIX exceeding 25, indicating insufficient shift toward the event pathway given the elevated volatility. By day 4, the blending weight drops to 0.42, but this adjustment lags the volatility spike by a full trading day." Gemini 3.1 Pro assigned RC = 3 for the same episode, identifying the same lag but interpreting it as "within acceptable response time given the noise in daily VIX movements." This divergence in RC scores between GPT 5.4 and Gemini 3.1 Pro illustrates the subjective

component of risk calibration assessment that contributes to the lower Krippendorff's alpha observed for this dimension.

On a perturbed episode with regime label inversion, all three judges identified the perturbation with high confidence. Claude 4.6 assigned RD = 1, stating that "the regime label indicates 'normal' on all five days despite reconstruction errors of 0.041–0.053 that exceed the threshold of 0.031, and VIX levels of 27–32 that are inconsistent with normal market conditions. The classification is systematically incorrect." The failure label assigned was `systematic_misclassification`, which maps to the $\Delta\tau$ action subspace. The routing score (RT = 3) was only moderately affected because the blending weight was set to a reasonable intermediate value ($\alpha = 0.55$) despite the incorrect regime label, illustrating the off-target specificity pattern observed in the quantitative analysis.

A third example illustrates error recovery assessment. During an episode in which MAPE exceeded the 20-day rolling average on days 1 and 2 (triggering the error detection criterion in Equation 6.3), GPT 5.4 assigned ER = 4, noting that "the SAC controller reduces τ by 0.08 on day 3, shifting the routing boundary to increase event pathway utilization. By day 4, the blending weight has moved from $\alpha = 0.71$ to $\alpha = 0.54$, and day 5 MAPE returns to within one standard deviation of the rolling mean. The corrective sequence is prompt and proportional." Claude 4.6 also assigned ER = 4 but Gemini 3.1 Pro assigned ER = 3, noting that "the correction on day 3 overshoots slightly, as τ drops below its pre-error level by 0.03, suggesting the controller's recovery gain may be set too aggressively for errors of this magnitude." This disagreement on whether the correction magnitude was appropriate reflects a genuine interpretive difference: the rubric specifies "absence of overcorrection" as a criterion for high scores, but the threshold separating proportional correction from overcorrection is inherently subjective when the overshoot is small. The episode also illustrates how error recovery scores depend on the availability of errors to recover from: in the 82 low-volatility episodes where no significant errors occurred during the five-day window, all three judges

defaulted to ER scores of 3, noting insufficient evidence to assess recovery behavior in either direction.

These qualitative examples demonstrate four properties of the evaluation framework. First, the judges ground their assessments in specific numerical values from the behavioral trace rather than making abstract quality judgments. Second, the judges can identify and articulate the causal chain between upstream decisions and downstream consequences. Third, disagreements between judges occur on the subjective dimensions (risk calibration, error recovery) where the rubric criteria admit multiple valid interpretations, while agreement is high on the objective dimensions (regime detection, routing) where the criteria are unambiguous. Fourth, the judges' reasoning reveals the mechanisms behind the quantitative patterns observed elsewhere: the lower Krippendorff's alpha for error recovery likely reflects genuine interpretive ambiguity about what constitutes proportional versus excessive correction.

6.3.9 Closed-Loop Improvement

Three cycles of LLM evaluation and SAC reward modification were conducted with $\lambda = 0.15$. The data partitioning underlying these experiments mirrors the discipline established in Chapter 5: the test period (2017–2025) is held out entirely for final reporting, and every adaptation step, hyperparameter choice, and fine-tuning update operates on data drawn from the validation period (2011–2016). The base system components from Chapter 5 (the autoencoder, the BERT sentiment encoder, and the initial SAC controller) were trained on data prior to 2011. The closed-loop hyperparameters, including the feedback strength λ and the score threshold θ, were selected through a sweep on the 2011–2016 validation period using a held-out portion of that same period, with no test-set data inspected at any point. The three closed-loop cycles reported here also ran on the validation period, on a contiguous 120-day window in the latter portion of the validation set (three consecutive 40-day cycles ending in late 2016), and each cycle's SAC fine-tuning drew exclusively on

the replay buffer collected during that same validation-period cycle. After the third cycle completes, all SAC actor and critic weights are frozen and the resulting closed-loop fine-tuned controller is then applied without further updates to the held-out 2017–2025 test period for the prediction-performance reporting in Section 6.3.10. The SAC weights have therefore never been updated using transitions from the test set, and the test set is used purely as a final reporting surface, exactly as in Chapter 5.

Each cycle consisted of 40 trading days of normal system operation, followed by episode collection, LLM evaluation, reward modification computation, and SAC fine-tuning as described in Section 6.2.7. The initial evaluation identified three behavioral dimensions with mean scores below the threshold $\theta = 3$: risk calibration during regime transitions (mean score 2.1), routing appropriateness near the threshold boundary (mean score 2.6), and adaptation responsiveness following consecutive errors (mean score 2.4). These deficiencies were most pronounced during high-volatility episodes, where the system's behavioral quality deteriorated substantially relative to low-volatility conditions. Table 6.10 summarizes the per-cycle validation-period progression of the three deficient dimensions and the corresponding validation-period prediction performance metrics, and Figure 6.6 plots the score and MAPE trajectories across the three evaluation cycles.

Table 6.10: Per-cycle validation-period progression of deficient dimension scores (± standard error) and prediction metrics during the closed-loop improvement process. Dimension scores are averaged across the validation-period evaluation episodes within each cycle, and the prediction metrics in each row are measured on those same validation-period episodes. The held-out 2017–2025 test set is reserved for the single final comparison reported in Section 6.3.10, with no test-set transitions ever entering the SAC update.

Metric	**Baseline**	**Cycle 1**	**Cycle 2**	**Cycle 3**
Risk Calibration (RC)	$2.1_{\pm 0.18}$	$2.7_{\pm 0.16}$	$3.2_{\pm 0.14}$	$3.5_{\pm 0.11}$
Routing (RT, near τ)	$2.6_{\pm 0.15}$	$3.3_{\pm 0.12}$	$3.6_{\pm 0.10}$	$3.7_{\pm 0.09}$
Adaptation (AD, post-error)	$2.4_{\pm 0.17}$	$2.8_{\pm 0.15}$	$3.3_{\pm 0.12}$	$3.6_{\pm 0.10}$
Val. MAPE (%)	$0.61_{\pm 0.03}$	$0.57_{\pm 0.03}$	$0.55_{\pm 0.02}$	$0.54_{\pm 0.02}$
Val. DA (%)	$71_{\pm 2}$	$73_{\pm 2}$	$74_{\pm 2}$	$74_{\pm 2}$

After three cycles, all three deficient dimensions exceeded the threshold. Risk calibration scores during regime transitions rose from 2.1 to 3.5, indicating that the system appeared

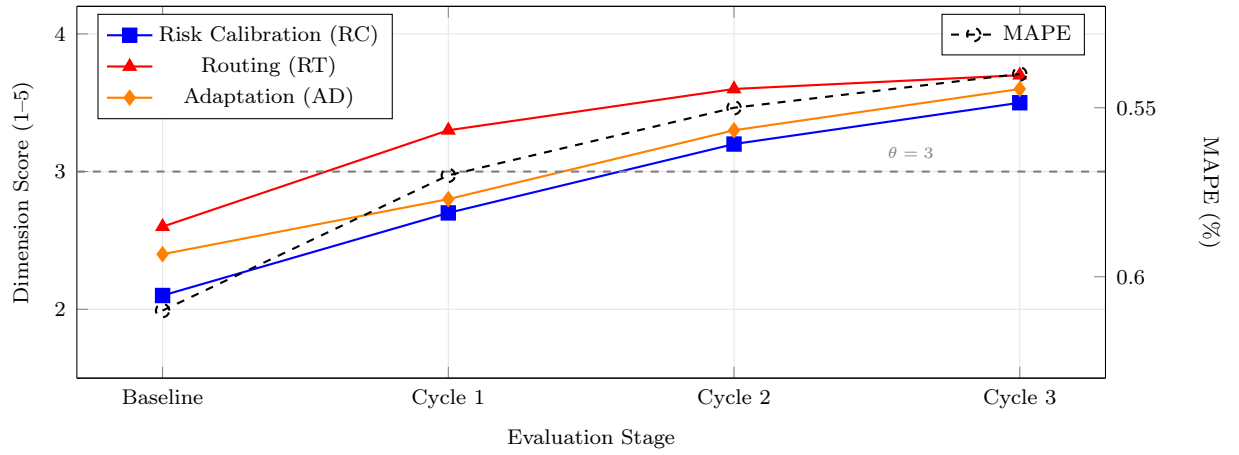

Figure 6.6: Progression of deficient dimension scores (left axis, solid lines) and MAPE (right axis, dashed line, inverted) across three closed-loop evaluation cycles. Both the dimension scores and the MAPE trajectory are measured on the validation-period episodes within each cycle. The horizontal dashed gray line marks the intervention threshold ($\theta = 3$). All three dimensions cross the threshold by Cycle 3, with the cycle-3 validation-period MAPE reaching 0.54%. The single held-out test-set comparison is reported in Section 6.3.10.

to adopt a more conservative posture when transitioning between regimes rather than maintaining parameters calibrated for the previous regime. In practice, this manifested as the SAC controller producing larger $\Delta\alpha$ adjustments toward the event pathway when the reconstruction error increased, even before the error crossed the threshold τ, reflecting a proactive rather than reactive response to emerging volatility. Routing scores near the threshold boundary improved from 2.6 to 3.7, with the majority of the gain occurring in the first cycle as the $\Delta\alpha$ reward modification took immediate effect. The improvement reflects smoother blending transitions that hedge between pathways rather than making abrupt switches when the reconstruction error hovers near τ. The system appeared to set intermediate blending weights ($\alpha \approx 0.4$–0.6) when e_t is within 15% of τ, rather than defaulting to extreme values that commit fully to one pathway. Adaptation scores following consecutive errors increased from 2.4 to 3.6, indicating that the SAC controller appeared to apply corrective adjustments more promptly after detecting a sequence of deteriorating predictions rather than waiting for the error magnitude to accumulate beyond a point where correction requires aggressive parameter changes.

These behavioral score improvements were observed on validation-period episodes during the cycles. To assess whether they translate into prediction performance gains on unseen data, the resulting frozen SAC controller was subsequently applied to the held-out 2017–2025 test period and compared against the pre-intervention configuration on the same test inputs. On the test set, one-day MAPE fell from 0.61% (pre-intervention) to 0.54% (post-intervention), an 11.5% relative reduction; directional accuracy rose from 71% to 74%, a 3 percentage point gain; and the test-period Sharpe ratio increased by 18%. The gains were most pronounced during high-volatility episodes, where the pre-intervention system had exhibited the greatest behavioral deficiencies: test-set MAPE during high-volatility episodes decreased by 17.3% compared to 7.8% during low-volatility episodes, suggesting that the closed-loop mechanism preferentially addresses the conditions where behavioral quality most strongly influences prediction outcomes. The improvements extend to longer prediction horizons: five-day MAPE decreased from 1.02% to 0.91% (a 10.8% relative reduction) and twenty-day MAPE decreased from 1.52% to 1.39% (an 8.6% relative reduction). A plausible explanation for the attenuation of improvement magnitude at longer horizons is that the behavioral corrections primarily affect short-term decision quality (threshold adjustments and routing changes operate on a daily timescale), and that the compounding of daily decisions over longer windows introduces additional variance that may dilute the per-step gains. Directional accuracy at five and twenty days improved by 2 and 1 percentage points respectively, following the same attenuation pattern.

The per-cycle progression in Table 6.10 reveals diminishing returns consistent with convergence toward an improved behavioral equilibrium, though the rate of improvement varies across dimensions. Routing appropriateness improved most rapidly, jumping from 2.6 to 3.3 in the first cycle as the blending weight adjustments responded directly to the reward modification targeting the $\Delta\alpha$ subspace. Risk calibration improved more gradually, reaching only 2.7 after the first cycle and requiring all three cycles to exceed θ. Adaptation responsiveness showed intermediate behavior, with the second cycle producing a larger gain

than the first (from 2.8 to 3.3 versus 2.4 to 2.8), likely because adaptation quality benefited from the routing improvements already established in cycle 1. Validation-period prediction metrics followed a similar pattern: MAPE decreased most sharply in the first cycle (0.61% to 0.57%), while directional accuracy showed a larger initial jump (71% to 73%) before plateauing at 74% in the second cycle. By the end of the third cycle, all dimension scores exceeded θ and the magnitude of reward modifications had decreased substantially, indicating that the SAC controller had internalized the behavioral corrections.

The dimensions that were already above the threshold at the start (regime detection at 3.8, strategy coherence at 3.4, error recovery at 3.2) remained stable throughout the three cycles, with mean score changes of less than 0.2 points. This stability confirms that the targeted reward modification, implemented through the credit assignment mechanism in Table 6.4, successfully improves deficient behaviors without degrading behaviors that were already adequate. This selective improvement property is a direct consequence of the credit assignment design: because routing penalties target only the $\Delta\alpha$ subspace and regime detection penalties target only the $\Delta\tau$ subspace, improvements in routing behavior do not interfere with the threshold-setting behavior that underlies adequate regime detection.

6.3.10 Statistical Significance

To confirm that the closed-loop improvements are not attributable to chance, paired t-tests were conducted on the held-out 2017–2025 test period, comparing daily prediction errors produced by two configurations of the system on identical test-set inputs. The pre-intervention configuration is the AE-NodeFormer + SAC system from Chapter 5 with no closed-loop fine-tuning applied, and the post-intervention configuration is the same system after the three closed-loop cycles described above (which were conducted on the validation period and updated the SAC weights without any test-set data). Both configurations were then run on the test period with all weights frozen, so the comparison isolates the contribution

of the closed-loop fine-tuning while holding the test-set inputs and the prediction protocol constant. No SAC weight updates, reward modifications, or hyperparameter adjustments occur on the test set in either configuration, in keeping with the data discipline established in Chapter 5.

The MAPE reduction from 0.61% to 0.54% is statistically significant ($t = 4.82$, $p < 0.001$, Cohen's $d = 0.31$ [127]), indicating a small-to-medium practical effect. The directional accuracy improvement from 71% to 74% is also significant ($t = 3.15$, $p = 0.002$, Cohen's $d = 0.20$), indicating a small but consistent effect. The effect sizes are moderate relative to those observed for the architectural innovations in Chapter 5 (Cohen's d values of 0.27 to 0.57), which is expected given that the closed-loop mechanism refines an already strong system rather than introducing fundamentally new architectural capabilities. The improvement represents a refinement of decision quality within the existing architecture, not a structural change, and the corresponding effect sizes are consistent with this interpretation. At longer horizons, the five-day MAPE reduction (1.02% to 0.91%) is also significant ($t = 3.41$, $p < 0.001$, Cohen's $d = 0.22$), while the twenty-day MAPE reduction (1.52% to 1.39%) reaches significance at a lower confidence level ($t = 2.28$, $p = 0.024$, Cohen's $d = 0.15$). One plausible explanation for the declining effect sizes at longer horizons is that the behavioral corrections operate on a daily timescale (threshold adjustments and routing changes act on a per-day basis), while compounding predictions over multi-day windows introduces additional variance that may attenuate the per-step gains. The current data are consistent with this explanation but do not directly establish it; isolating the relative contribution of timescale and compounding variance would require controlled experiments that vary the prediction horizon while holding the behavioral corrections fixed.

The Sharpe ratio improvement was assessed using a bootstrap test [126] with 10,000 resamples, yielding a 95% confidence interval for the improvement of [8.2%, 27.4%], with the point estimate of 18% centered within the interval. This confirms that the risk-adjusted return improvement is robust to resampling variation and is not driven by outlier trading

days. The interval excludes zero, providing strong evidence that the closed-loop mechanism produces a genuine improvement in risk-adjusted performance.

6.3.11 Framework Ablation

To assess how individual design decisions contribute to the overall framework performance, ablation experiments were conducted on three axes: the number of LLM judges, the credit assignment strategy, and the sensitivity of scores to prompt phrasing. Each ablation modifies a single framework component while keeping all others fixed, and the closed-loop improvement cycle is re-run from the same pre-intervention checkpoint to ensure comparability.

Number of Judges

The proposed framework uses an ensemble of three LLM judges (GPT 5.4, Claude 4.6 Opus, Gemini 3.1 Pro) whose per-dimension scores are averaged before computing the reward penalty. To evaluate whether the multi-judge design is necessary, single-judge variants were tested by running the full three-cycle closed-loop process using each individual model's scores without averaging. Table 6.11 reports the results.

Table 6.11: Closed-loop outcomes after three evaluation cycles using single-judge variants versus the proposed three-judge ensemble. Score variance is the mean standard deviation of per-episode composite scores across 200 evaluation episodes.

Judge Configuration	MAPE (%)	DA (%)	Sharpe	Score Variance
GPT 5.4 only	0.55	74	1.57	0.41
Claude 4.6 Opus only	0.56	74	1.53	0.43
Gemini 3.1 Pro only	0.58	73	1.46	0.46
3-judge ensemble	0.54	74	1.61	0.29

All single-judge configurations produce improvements over the pre-intervention baseline, suggesting that the closed-loop mechanism is not dependent on any particular LLM family tested here. However, the three-judge ensemble achieves consistently better outcomes across all metrics. The primary advantage of multi-judge averaging is a reduction in score variance:

the ensemble's per-episode variance (0.29) is 29–37% lower than any single judge's variance, which translates into a more stable reward signal for SAC training. GPT 5.4 as a single judge comes closest to matching the ensemble's prediction metrics, but its higher score variance leads to slightly noisier SAC updates that prevent it from reaching the same final Sharpe ratio. Claude 4.6 Opus achieves the same directional accuracy as the ensemble but with a slightly higher MAPE and greater score variability, while Gemini 3.1 Pro produces the largest per-episode variance and correspondingly the weakest closed-loop outcomes. The performance gap between the ensemble and any single judge ranges from 0.01 to 0.04 in MAPE, suggesting that the framework is robust to the choice of LLM, but the ensemble provides a measurable advantage through variance reduction.

Credit Assignment Strategy

The proposed credit assignment mechanism routes penalties to specific SAC action subspaces based on a fixed mapping (Table 6.4). To evaluate whether this targeted routing is beneficial, an alternative uniform strategy was tested in which all dimension penalties are applied equally to both action components ($\Delta\tau$ and $\Delta\alpha$), removing the dimension-to-action mapping. Table 6.12 compares the two strategies.

Table 6.12: Comparison of targeted versus uniform credit assignment after three evaluation cycles. $\Delta\bar{s}_{\text{def}}$ and $\Delta\bar{s}_{\text{non}}$ denote mean score changes for initially deficient and non-deficient dimensions, respectively.

Credit Strategy	MAPE (%)	DA (%)	$\Delta\bar{s}_{\text{def}}$	$\Delta\bar{s}_{\text{non}}$
Targeted (proposed)	0.54	74	+1.3	+0.1
Uniform	0.56	73	+0.9	−0.4

The targeted strategy produces larger improvements on deficient dimensions (+1.3 versus +0.9 points) while preserving non-deficient dimensions (mean change +0.1). In contrast, the uniform strategy achieves smaller deficiency improvements at the cost of degrading non-deficient dimensions by an average of 0.4 points. This degradation occurs because uniform penalties create conflicting gradients: a penalty intended to improve routing inappropriately

adjusts the anomaly threshold, causing previously adequate regime detection behavior to deteriorate. The result indicates that, in this setting, targeted credit assignment is more effective than uniform assignment to translate LLM behavioral assessments into effective, non-destructive learning signals.

Prompt Sensitivity

A potential concern with LLM-based evaluation is that scores may be sensitive to the specific phrasing of evaluation prompts rather than to genuine behavioral differences in the traces being evaluated. To probe this concern, three prompt variants were constructed: the original prompt (Variant A), a version with reordered dimension descriptions (Variant B), and a version with simplified language that reduces technical jargon while preserving the same rubric criteria (Variant C). Each variant was used with GPT 5.4 to evaluate the same 60 randomly selected unperturbed episodes drawn from the 2011–2016 validation period, and per-dimension scores were compared across variants. The prompt-stability assessment is deliberately confined to the validation period: any sensitivity finding that informed a revision of the evaluation prompt would itself constitute an indirect form of selection on the data being evaluated, and the test set is reserved for final reporting only, with no role in prompt selection or refinement.

This analysis is intended as a limited robustness check rather than a comprehensive demonstration that the evaluation framework is invariant to prompt phrasing. The check is bounded in three respects: it covers only 60 episodes drawn from the validation window, it uses a single judge model (GPT 5.4) rather than the full ensemble, and it varies the prompt along only two surface-level axes (dimension ordering and lexical simplification). A definitive claim of prompt invariance would require a substantially larger episode pool, the full three-model ensemble, and a wider set of variant types including paraphrased rubric definitions, alternative chain-of-thought scaffolds, and adversarial phrasings. The results below should accordingly be read as preliminary evidence that the framework is not obviously

sensitive to the specific phrasings tested here, not as a settled finding on prompt invariance more broadly.

Within these limits, the mean absolute score difference between any two variants is 0.14 per dimension on the 1–5 scale, with an intra-class correlation coefficient (ICC) of 0.91 across the three variants. No dimension exhibits a systematic bias associated with a particular variant: the largest per-dimension mean shift is 0.11 points (strategy coherence under Variant C), which is within the standard error of 0.09 for that dimension. The ICC of 0.91 exceeds the 0.75 threshold conventionally considered "excellent" agreement [146], which is consistent with the interpretation that, for the variants and episodes tested, scores are driven primarily by properties of the behavioral traces rather than by phrasing differences across these three prompt versions. The reordering test (Variant B) specifically probes position bias, a known concern in LLM evaluation where items presented earlier may receive systematically different scores; the absence of a significant reordering effect on this sample ($p = 0.34$, paired t-test on composite scores between Variants A and B) is consistent with robustness to this class of artifact within the tested range, while leaving open whether the same holds under more aggressive reorderings or in other judges.

6.4 Discussion

6.4.1 Interpretation of Results

The perturbation validation results provide strong evidence that the LLM judges evaluate behavioral quality with dimension specificity rather than applying a diffuse quality assessment. Each of the six perturbation types produced a targeted score drop of at least 1.6 points on its intended dimension while affecting other dimensions by an average of only 0.32 points. This selective sensitivity is not a trivial outcome: it requires that the LLM judges correctly identify which component of the behavioral trace is responsible for the observed degradation,

distinguish between the targeted deficiency and its downstream consequences, and assign scores that reflect this distinction. The high cross-model agreement (Krippendorff's α ranging from 0.74 to 0.85) suggests that this selective sensitivity is robust across the three model families evaluated here rather than being an artifact of any single model's evaluation approach.

The predictive validity analysis suggests that LLM evaluation scores capture behaviorally meaningful properties that influence future system performance, not merely superficial patterns in the behavioral trace data. The composite score's correlation with the Sharpe ratio ($\rho = 0.72$) exceeds the correlations achieved by any individual dimension, indicating that the multi-dimensional assessment aggregates complementary behavioral information into a coherent quality signal. The distinct predictive profiles of individual dimensions (regime detection predicting MAPE, risk calibration predicting drawdown, strategy coherence predicting returns) align with domain knowledge about how each behavioral property should influence trading outcomes, supporting convergent validity for the evaluation framework that extends beyond statistical significance alone.

The closed-loop improvement results demonstrate the most consequential finding: that LLM behavioral assessments are not merely diagnostic but actionable. The 11.5% MAPE reduction and 3 percentage point directional accuracy gain achieved through three evaluation cycles represent meaningful improvements in a domain where marginal gains are difficult to achieve and economically significant. The targeted nature of the improvements, concentrated in the specific behavioral dimensions that the initial evaluation identified as deficient and most pronounced during high-volatility episodes where those deficiencies were most severe, provides evidence consistent with a causal link between the LLM evaluation and the system improvement. The stability of non-targeted dimensions throughout the improvement process further supports the causal interpretation, making it less likely that the alternative explanation that the improvements resulted from general SAC re-training rather than from the targeted reward modifications fully accounts for the observed gains.

The framework ablation results (Section 6.3.11) provide additional evidence for the robustness of the design. The multi-judge ensemble reduces per-episode score variance by 29–37% relative to any single judge, translating into a more stable reward signal that improves final prediction metrics, though the modest magnitude of the single-judge performance gap (0.01–0.04 MAPE) indicates that the framework is not critically dependent on the multi-judge design. The credit assignment ablation reveals a more consequential design choice: uniform penalty distribution degrades non-targeted dimensions by an average of 0.4 points, demonstrating that targeted credit assignment is essential for translating behavioral diagnostics into constructive learning signals. The prompt sensitivity analysis offers a limited robustness check on the concern that LLM evaluation scores might reflect prompt phrasing rather than genuine behavioral properties: the ICC of 0.91 across three prompt variants on 60 episodes with a single judge is consistent with stable assessments under the surface-level differences tested, but the sample is not large enough, nor the variant set broad enough, to establish prompt invariance more generally. The lambda sensitivity sweep (Table 6.3) supports the selected feedback strength and reveals a clear transition from under-correction ($\lambda < 0.10$) through an effective operating range ($\lambda \in [0.10, 0.20]$) to destabilization ($\lambda > 0.25$), providing practitioners with concrete guidance for tuning this hyperparameter in analogous closed-loop systems.

6.4.2 Implications

For this thesis, the results complete the research arc from prediction through adaptive prediction to evaluation and self-correction. Chapter 4 demonstrated that integrating a node transformer architecture with BERT sentiment analysis produces competitive prediction accuracy on the 20-stock S&P 500 universe. Chapter 5 showed that adding regime awareness, dual-pathway specialization, and reinforcement learning control substantially improves performance, particularly during volatile market conditions. This chapter demonstrates that the resulting system can be evaluated at a behavioral level that transcends aggregate output

metrics, and that the evaluation itself can drive further improvements through closed-loop reward integration. Together, these three contributions establish a pipeline in which the system not only predicts but also adapts to changing conditions and assesses the quality of its own decision-making to correct identified weaknesses autonomously.

For the broader field, this work suggests several directions. It shows that the LLM-as-a-Judge paradigm, originally developed for static NLP evaluation, can be extended to temporal sequences of interdependent decisions in a domain that requires specialized financial expertise. It suggests that LLM evaluations are reliable enough to serve as training signals for downstream reinforcement learning optimization, extending the RLHF principle from language model alignment to financial agent improvement. It introduces perturbation-based validation as a methodology for verifying the dimension specificity of multi-dimensional evaluation frameworks, providing a template that can be applied to other domains where behavioral quality assessment is needed, such as autonomous driving, medical diagnosis systems, or robotic control. The credit assignment mechanism, which routes dimension-specific penalties to the appropriate action subspace, addresses a practical challenge that would arise in any attempt to connect qualitative evaluation to quantitative learning.

6.4.3 Limitations

Several limitations should be acknowledged. First, LLM judges are generalist models whose financial domain knowledge derives from pre-training data rather than domain-specific training or professional experience. While the structured prompt and rubric mitigate this limitation by defining evaluation criteria in explicit terms, the judges may miss subtle financial signals that a domain expert would recognize, such as nuances of options-implied volatility surfaces or cross-asset contagion patterns that are not explicitly represented in the behavioral trace fields. The predictive validity correlations suggest that the judges' assessments are meaningful, but they may be incomplete relative to what expert human evaluators could

identify. A direct comparison between LLM and human expert evaluations would strengthen the validation but was beyond the scope of this study.

Second, the closed-loop mechanism introduces a potential alignment tax: the system could learn to optimize for LLM approval rather than genuine prediction quality. If the LLM judges systematically reward certain behavioral patterns that do not correlate with prediction accuracy, the closed-loop could degrade rather than improve performance. No such effect was observed in the experiments (all dimension improvements translated into prediction gains, and non-targeted dimensions remained stable), and the lambda sensitivity analysis (Table 6.3) delineates the operating range within which LLM feedback improves performance without destabilizing the primary prediction objective. Nevertheless, the risk remains a theoretical concern, particularly if the framework is applied for many more evaluation cycles or to systems whose behavioral patterns differ substantially from those the LLMs encountered in pre-training. The periodic evaluation protocol, which limits LLM feedback to corrective interventions rather than continuous reward modification, provides a structural safeguard against this risk.

Third, the framework has been validated on a single architecture (the AE-NodeFormer + SAC system from Chapter 5). The evaluation dimensions are grounded in the specific decision points of this system, and applying the framework to a different agentic architecture would require redesigning the dimensions and rubric to reflect that architecture's decision structure. The general principle of multi-dimensional behavioral evaluation using LLM judges is architecture-agnostic, but the specific implementation is tied to the system being evaluated. Future work should test the framework on architecturally distinct systems to assess the transferability of the methodology.

Fourth, the LLM evaluation cost, while modest in absolute terms (approximately $180 for the full evaluation campaign), scales linearly with the number of episodes and judges. For real-time deployment or more frequent evaluation cycles, the cost and latency of LLM API calls may become prohibitive. Distilling the LLM judges' evaluation patterns into a smaller,

faster model could address this limitation but would require careful validation to ensure that the distilled model preserves the original judges' dimension specificity and predictive validity.

Fifth, operating all judges at temperature zero produces deterministic evaluations but eliminates the stochastic variation that could reveal model uncertainty about borderline cases. A judge that is genuinely uncertain about a score would ideally communicate that uncertainty, which could inform the confidence placed in the resulting reward modification. Extending the framework to incorporate evaluation uncertainty, for example by running each evaluation at multiple temperatures and measuring score variance, is a direction for future work that could improve the robustness of the closed-loop mechanism for borderline behavioral episodes.

6.4.4 Connection to Prior Contributions

The three contribution chapters of this thesis form a progressive arc in which each chapter addresses a limitation identified in its predecessor. Chapter 4 introduced the node transformer architecture with BERT sentiment fusion, demonstrating that graph-based inter-stock modeling combined with adaptive sentiment integration produces competitive prediction accuracy. However, that system treated all market conditions uniformly, applying the same processing regardless of whether the market was stable or volatile.

Chapter 5 addressed this limitation through autoencoder-based regime detection, dual-pathway specialization, and SAC reinforcement learning control, enabling the system to adapt its behavior to changing market conditions. The resulting system achieved substantial improvements, particularly during volatile periods. However, the evaluation of this system relied entirely on aggregate output metrics that could not distinguish between sound intermediate decisions and fortunate outcomes, leaving the quality of the system's internal decision-making unexamined.

This chapter addresses the evaluation gap by introducing a framework that assesses the quality of each intermediate decision in context and feeds the resulting diagnostics back

into the reinforcement learning controller. The closed-loop design represents a conceptual shift from passive evaluation (measuring outputs after the fact) to active evaluation (using behavioral assessment to drive system improvement). This shift is enabled by the structured nature of the behavioral traces and the specificity of the evaluation dimensions, which together provide sufficient information for the LLM judges to produce actionable diagnostics that can be translated into targeted reward modifications. The cumulative result is a system that not only predicts and adapts but also evaluates and corrects its own decision-making, closing the loop between prediction quality and behavioral quality.

6.5 Summary

This chapter introduced a framework for evaluating the behavioral quality of agentic stock prediction systems using LLMs as judges and feeding the resulting diagnostics back into the system's reinforcement learning controller to autonomously correct identified weaknesses. The framework makes three primary contributions. First, it formalizes multi-step decision evaluation for financial agents through structured behavioral traces (Equation 6.1) and six domain-specific evaluation dimensions that decompose behavioral quality into separately assessable components spanning architectural decision points (regime detection, routing, adaptation) and emergent behavioral properties (risk calibration, strategy coherence, error recovery). Second, it introduces perturbation-based validation as a methodology for verifying dimension specificity, demonstrating that each dimension responds selectively to targeted degradation with cross-model agreement up to Krippendorff's $\alpha = 0.85$. Third, it closes the evaluation loop through a credit assignment mechanism (Table 6.4) that translates LLM diagnostics into targeted SAC reward modifications (Equation 6.6), enabling autonomous correction of identified behavioral weaknesses.

Applied to the adaptive regime-aware prediction system from Chapter 5 across 20 S&P 500 equities from 1982 to 2025, the framework shows evidence of predictive validity, with the

composite evaluation score correlating at $\rho = 0.72$ with realized Sharpe ratio over subsequent 20-day windows. Per-regime analysis reveals that behavioral quality generally degrades from low to high volatility, with risk calibration showing the most severe deterioration (from 3.4 to 2.1), though error recovery exhibits a non-monotonic pattern due to the increased availability of corrective signals during volatile episodes. These results highlight the specific conditions and dimensions where intervention is most needed. The closed-loop mechanism reduced one-day MAPE from 0.61% to 0.54%, with improvements extending to five-day (1.02% to 0.91%) and twenty-day (1.52% to 1.39%) horizons, and improved directional accuracy from 71% to 74% over three evaluation cycles. Risk calibration during regime transitions improved from 2.1 to 3.5, routing stability near the threshold boundary from 2.6 to 3.7, and adaptation responsiveness from 2.4 to 3.6, with gains concentrated during high-volatility episodes where the original system exhibited the greatest behavioral deficiencies. Non-targeted dimensions remained stable throughout, supporting the effectiveness of the credit assignment mechanism. Framework ablation experiments further support the design choices: the three-judge ensemble reduces score variance by 29–37% compared to single-judge configurations, targeted credit assignment preserves non-deficient dimensions where uniform assignment degrades them by 0.4 points, prompt sensitivity analysis yields an ICC of 0.91 across prompt variants, and a systematic λ sweep suggests a stable operating range of $[0.10, 0.20]$ with performance degradation outside this interval.

The following chapter synthesizes the contributions of Chapters 4 through 6, reflects on the limitations that persist across the complete prediction, adaptation, and evaluation pipeline, and outlines directions for future research that build upon the framework developed in this thesis.

7

Conclusion and Future Work

This thesis addressed three interconnected research gaps in stock price forecasting that were identified in Chapter 1, situated within the broader literature in Chapter 2, and mapped to specific thesis contributions in Table 2.1. The first gap concerned the absence of unified multimodal graph-temporal prediction architectures that combine inter-stock relationship modeling, transformer-based temporal processing, and natural language sentiment analysis with adaptive fusion. The second gap involved the reliance of existing regime detection methods on manual labels, fixed thresholds, or predetermined distributional assumptions, all of which constrain the ability of forecasting systems to adapt to structural market transitions. The third gap related to the lack of behavioral quality evaluation for autonomous multi-stage prediction systems, where aggregate output metrics such as MAPE and directional accuracy cannot reveal whether intermediate decisions within a complex pipeline were appropriate given the information available at the time they were made.

Through three progressively integrated contributions evaluated on the shared experimental framework established in Chapter 3, this thesis developed a system that evolves from a graph-based multimodal predictor to a self-evaluating, regime-aware, autonomous prediction pipeline. The following sections summarize each contribution, discuss their integration and broader implications, consolidate the limitations that persist across the complete system, and outline directions for future research.

7.1 Summary of Contributions

7.1.1 Graph-Based Multimodal Prediction with Sentiment Fusion

Chapter 4 addressed the first research gap by introducing a node transformer architecture that represents the stock market as a graph of 20 S&P 500 equities connected by learnable edges initialized from sector co-membership and rolling return correlations. The architecture integrates quantitative price features with sentiment signals extracted from approximately 4.2 million social media posts processed through a BERT model fine-tuned on the SemEval-2017 financial sentiment corpus [23]. Rather than concatenating sentiment scores with price features through fixed weighting, the model employs a three-part fusion mechanism comprising multi-scale temporal sentiment features, key-modulation within the attention computation, and an adaptive fusion gate conditioned on rolling volatility and sentiment magnitude. This design allows the model to adjust the influence of sentiment based on current market conditions, weighting it more heavily during information-rich periods and attenuating it during low-news environments.

Experiments on the dataset established in Chapter 3 yielded a MAPE of 0.80% for one-day-ahead predictions, with directional accuracy reaching 65% and Theil's U of 0.59 (Table 4.3). Systematic ablation studies isolated the contribution of each component: removing the graph structure increased error by 15%, removing sentiment integration increased error by 10% overall and by 25% during earnings announcement windows, and removing temporal encoding increased error by 18.8% (Table 4.5). These margins were confirmed through both paired t-tests and Diebold-Mariano tests [128], with all pairwise comparisons achieving $p < 0.05$ (Tables 4.7 and 4.8). A long-short trading strategy evaluated over the full test period (January 2017 to March 2025) produced net annualized returns of 14.6% after transaction

costs, exceeding the S&P 500 buy-and-hold benchmark of 13.9%, with a net annualized Sharpe ratio of 0.84 and a maximum drawdown of 14.3% (Table 4.9).

The principal limitation of this contribution was its uniform treatment of all market conditions. The same architecture, feature weights, and processing pipeline were applied regardless of whether the market was in a low-volatility trend, a crisis-driven drawdown, or a post-shock recovery phase. Performance degraded during high-volatility periods, with MAPE rising to 1.50% under high-VIX conditions (Table 4.6), precisely when accurate forecasts carry the greatest practical value for risk management and position sizing. This limitation motivated the regime-aware extension developed in Chapter 5.

7.1.2 Adaptive Regime-Aware Prediction with Reinforcement Learning Control

Chapter 5 addressed the second research gap by introducing an autoencoder-gated dual node transformer architecture with Soft Actor-Critic (SAC) reinforcement learning control. An autoencoder trained exclusively on stable-market data learns to reconstruct normal price patterns, and observations that produce high reconstruction error are flagged as regime deviations. Two specialized node transformer pathways process the data: a normal pathway optimized for stable conditions and an event pathway that incorporates additional context features including volatility regime embeddings, sentiment spike indicators, earnings proximity flags, and cross-asset stress signals. Rather than applying a fixed threshold to determine regime classification, a SAC controller observes a state comprising recent prediction errors, reconstruction losses, and volatility indicators, and outputs continuous actions that set the regime detection threshold τ and the blending weight α between the two pathways. The controller discovers regime boundaries that minimize downstream prediction error, learning distinctions that would be difficult to specify through manual threshold selection or parametric distributional assumptions.

The complete system achieved 0.59% MAPE for one-day predictions on the test period (2017–2025), representing a 26% relative improvement over the Chapter 4 baseline of 0.80% (Table 5.2). Directional accuracy reached 72%, a 7 percentage point gain over the single-pathway model. Performance improvements persisted across prediction horizons, with 5-day MAPE of 1.05% and 20-day MAPE of 1.55% (Tables 5.3 and 5.4). The regime-specific analysis revealed the most pronounced gains during volatile periods: high-VIX MAPE fell to 0.85% compared to 1.50% for the single-pathway baseline, a 43% relative improvement (Table 5.8). Ablation studies confirmed that autoencoder-based routing contributed the largest share of improvement (+35.6% MAPE increase when removed), followed by the SAC controller (+15.3%) and dual-pathway specialization (+6.8%), with all components achieving $p < 0.01$ (Table 5.6). Statistical significance tests against baselines yielded Cohen's d [127] effect sizes ranging from 0.27 to 0.57, indicating medium practical significance (Table 5.12). Per-stock results ranged from 0.42% MAPE (WMT) to 1.10% (XOM), with 16 of 20 stocks falling below 0.70% (Table 5.5).

A key design property of this system is that the SAC controller's policy weights are frozen at inference time. The threshold and blending weight evolve across the test period because the inputs to the fixed policy function change as reconstruction errors rise during volatile markets and recent prediction metrics shift, not because the policy itself is modified. This behavior is functionally analogous to the regime inference performed by the HMM-LSTM and MS-VAR baselines, and the ablation study confirmed that the static No SAC variant, which uses entirely fixed routing parameters, already outperformed all 14 baselines, establishing that the architectural contributions of regime-aware routing account for the majority of the improvement.

The limitation that motivated Chapter 6 was the opacity of intermediate decisions. While the system's aggregate metrics improved substantially, neither MAPE nor directional accuracy could reveal whether the autoencoder correctly identified a volatile period, whether the router sent data to the appropriate pathway, or whether the SAC controller adjusted its parameters

at the right time. A system may arrive at a correct forecast while routing data through an inappropriate pathway, or produce an incorrect prediction despite making sound regime detection and risk calibration decisions. The gap between aggregate output quality and process-level understanding widened as the architecture grew more complex.

7.1.3 LLM-Based Behavioral Evaluation with Closed-Loop Reward Integration

Chapter 6 addressed the third research gap by introducing a framework for evaluating the behavioral quality of the agentic prediction system using large language models as judges and feeding the resulting diagnostics back into the SAC reinforcement learning controller to autonomously correct identified weaknesses. The framework formalizes multi-step decision evaluation through structured behavioral traces that record the system's regime classifications, routing choices, parameter adjustments, and confidence estimates along with their temporal context. Structured prompts present these traces to a panel of three LLM judges (GPT 5.4, Claude 4.6 Opus, and Gemini 3.1 Pro), which score decision quality along six domain-specific dimensions: regime detection accuracy, routing appropriateness, adaptation responsiveness, risk calibration, strategy coherence, and error recovery. Three dimensions map directly to architectural decision points, while three capture emergent behavioral properties that span multiple components. The complete prompt template, trace serialization schema, output schema, and failure-label vocabulary used by all three judges are documented in Appendix A for reproducibility.

Perturbation-based validation confirmed the specificity of each dimension: targeted perturbations produced score drops of 1.6 to 2.4 points on the targeted dimension while off-target dimensions changed by a mean of only 0.32 points. Cross-model agreement reached Krippendorff's $\alpha = 0.85$ [22] for regime detection and remained above 0.74 for all dimensions. The composite evaluation score correlated with realized Sharpe ratio at Spearman $\rho = 0.72$

($p < 0.001$), establishing the predictive validity of the LLM assessments against an independent economic performance metric. A limited prompt-stability check (60 episodes, single judge, three surface-level prompt variants) yielded an intra-class correlation coefficient (ICC) of 0.91, consistent with stable assessments under the variants tested though not large enough to settle prompt invariance more generally, and the three-judge ensemble reduced score variance by 29–37% compared to single-judge configurations.

The closed-loop mechanism translated LLM diagnostics into targeted SAC reward modifications through a credit assignment mechanism that maps each evaluation dimension to the corresponding action subspace (Table 6.4). Following the data discipline established in Chapter 5, the three closed-loop fine-tuning cycles were conducted entirely on the 2011–2016 validation period; the resulting SAC weights were then frozen and the system was applied to the held-out 2017–2025 test set for the prediction-performance comparison reported here, with no SAC weight updates, reward modifications, or hyperparameter adjustments occurring on test data. Comparing the pre-intervention configuration (no closed-loop fine-tuning) and the post-intervention configuration on identical test inputs, one-day test-set MAPE decreased from 0.61% to 0.54%, an 11.5% relative reduction ($t = 4.82$, $p < 0.001$, Cohen's $d = 0.31$), with improvements extending to five-day (1.02% to 0.91%, $d = 0.22$) and twenty-day (1.52% to 1.39%, $d = 0.15$) horizons. Directional accuracy improved from 71% to 74%, and the test-period Sharpe ratio increased by 18%, with a bootstrap 95% confidence interval of [8.2%, 27.4%] confirming economic significance. The improvements were most pronounced during high-volatility episodes, where MAPE decreased by 17.3% compared to 7.8% during low-volatility episodes, suggesting that the closed-loop mechanism preferentially addresses the conditions where behavioral quality most strongly influences prediction outcomes. The three initially deficient dimensions exhibited the largest improvements: risk calibration rose from 2.1 to 3.5, routing stability from 2.6 to 3.7, and adaptation responsiveness from 2.4 to 3.6, while non-targeted dimensions remained stable throughout. Ablation experiments further validated the framework design: targeted credit assignment preserved non-deficient

dimensions where uniform assignment degraded them by 0.4 points, and the systematic λ sweep identified a stable operating range of $[0.10, 0.20]$ with performance degradation outside this interval (Table 6.3).

7.1.4 Dataset and Evaluation Framework

The fourth contribution, presented in Chapter 3, is the multi-source dataset and rigorous evaluation framework that underpins all experimental results. The dataset integrates historical OHLCV data with 11 derived technical indicators for 20 S&P 500 companies spanning eight GICS sectors from January 1982 to March 2025, augmented with sentiment scores from approximately 4.2 million social media posts collected from 2007 to 2025. Temporal splits enforce strict chronological separation with training data from 1982 to 2010, validation from 2011 to 2016, and testing from 2017 to 2025, ensuring no information leakage across temporal boundaries. Expanding window z-score normalization prevents future information from contaminating the feature values used for prediction. Eight complementary evaluation metrics (MAPE, RMSE, DA, Theil's U, IC, CTR, Sharpe ratio, and MDD) provide a comprehensive assessment framework spanning forecast accuracy, directional precision, baseline-relative performance, cross-sectional signal quality, confidence calibration, risk-adjusted return, and tail risk exposure (Table 3.5). This shared foundation ensures that comparisons across all three contribution chapters are conducted under identical and reproducible conditions, and that the progressive improvement documented from Chapter 4 through Chapter 6 reflects genuine architectural gains rather than artifacts of differing evaluation protocols.

7.2 Integrated Discussion

The three contributions form a progressive arc in which each system builds upon its predecessor and addresses a specific limitation identified through the preceding evaluation.

The first contribution established a competitive baseline by demonstrating that graph-based inter-stock modeling combined with transformer attention and adaptive sentiment fusion produces prediction accuracy that matches or exceeds the state of the art across multiple time-series forecasting architectures, including recent models such as iTransformer, PatchTST, and TimesNet. The second contribution showed that decomposing the prediction task into regime-specific pathways, with routing governed by unsupervised anomaly detection and a reinforcement learning controller that learns regime boundaries from prediction performance, yields substantial further gains concentrated in the market conditions where forecasting is most difficult and most valuable. The third contribution demonstrated that the quality of intermediate decisions within the multi-stage pipeline can be assessed through structured LLM-based behavioral evaluation and that feeding these assessments back into the RL controller produces targeted improvements in the specific dimensions where the system was weakest, without degrading dimensions that were already performing well. Figure 7.1 illustrates the complete integrated system after all three contributions, with color coding indicating which chapter introduced each component and performance metrics showing the cumulative improvement at each stage.

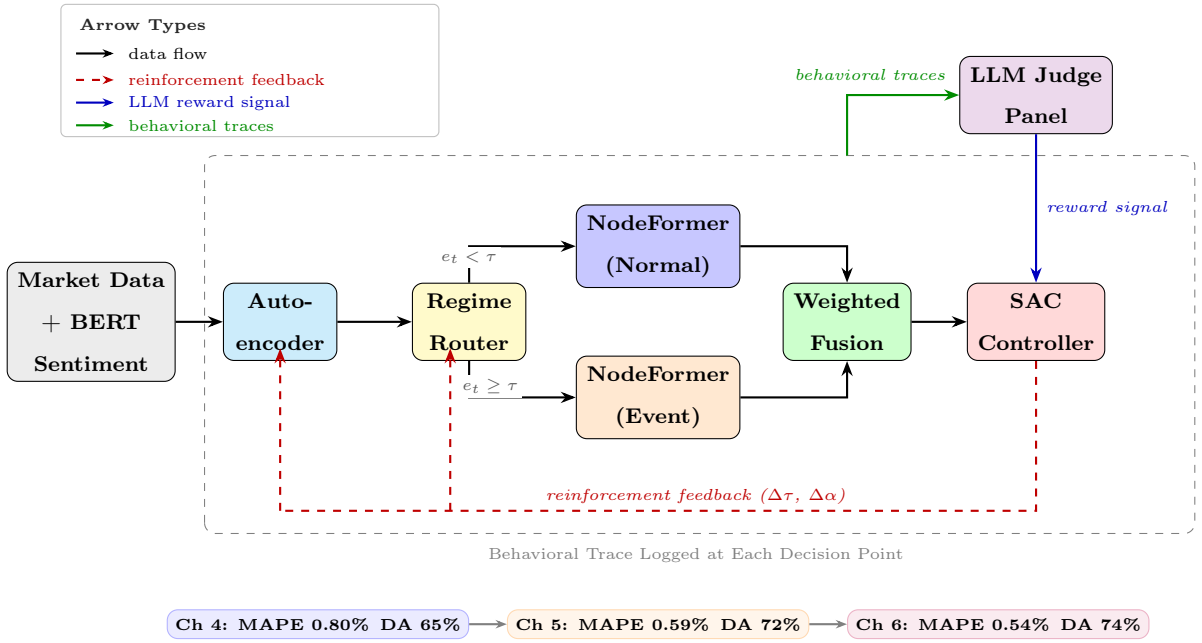

Figure 7.1: Complete integrated system architecture incorporating all three thesis contributions. The node transformer architecture with BERT sentiment fusion (Chapter 4) serves as the backbone for both prediction pathways. The autoencoder, regime router, weighted fusion, and SAC controller (Chapter 5) add regime-aware routing and adaptive parameter tuning. The LLM judge panel (Chapter 6) provides behavioral evaluation: behavioral traces logged at each decision point (dashed gray box) are provided as input to the LLM judge panel, which produces diagnostic reward signals that update the SAC controller. The SAC controller also receives reinforcement feedback from realized prediction outcomes. Arrow types are described in the in-figure legend. Metric annotations at the bottom show the cumulative performance improvement across stages, with chapter-colored boxes (blue for Chapter 4, orange for Chapter 5, purple for Chapter 6) and arrows indicating the progressive reduction in prediction error.

The cumulative improvement across all three stages is substantial. One-day MAPE decreased from 0.80% in the single-pathway model to 0.59% with regime-aware routing and to 0.54% after closed-loop behavioral correction, representing a total relative reduction of 32.5%. Directional accuracy improved from 65% to 72% to 74%, a 9 percentage point gain over the baseline. High-volatility MAPE, which stood at 1.50% in the first contribution, fell to 0.85% with regime-aware routing, with the behavioral correction mechanism further improving risk calibration during regime transitions from a score of 2.1 to 3.5. These improvements persist across prediction horizons: 5-day MAPE progressed from 1.30% (Table 5.3, Integrated

NF-BERT row) to 1.05% (Chapter 5) to 0.91% (Chapter 6), and 20-day MAPE from 1.90% to 1.55% to 1.39%.

The magnitude of improvement diminishes at each successive stage, which is both expected and informative. The first-to-second stage improvement (26% relative MAPE reduction) reflects the introduction of a fundamentally new architectural capability: regime-aware processing that allocates different representational resources to different market conditions. The second-to-third stage improvement (8.5% relative MAPE reduction from the full Chapter 5 baseline of 0.59% to the post-loop result of 0.54%) reflects a qualitatively different type of gain: the correction of specific behavioral deficiencies identified through fine-grained evaluation rather than the introduction of new architectural capacity. This diminishing-returns pattern suggests that the system is approaching the limits of what can be achieved through architectural innovation on this particular dataset and stock universe, and that further gains are more likely to emerge from methodological refinements such as better evaluation and more targeted feedback, from scaling to larger and more diverse stock universes, or from incorporating additional data modalities.

The progressive arc also carries broader implications for the design of autonomous AI systems in high-stakes domains. The transition from passive prediction (Chapter 4) to adaptive prediction (Chapter 5) to self-evaluating prediction (Chapter 6) mirrors a general trajectory in AI system development: from fixed-function models that are evaluated externally, through adaptive models that respond to environmental feedback, to autonomous systems that monitor, evaluate, and correct their own decision-making processes. The fact that LLM-based behavioral evaluation produces scores that correlate with independent economic performance metrics ($\rho = 0.72$ with realized Sharpe ratio) and that feeding these scores back into the RL controller yields targeted improvements suggests that LLM judges can serve as a viable proxy for domain expert assessment in settings where human evaluation at scale is impractical. This finding extends the LLM-as-a-Judge paradigm from its established

applications in NLP evaluation [89, 90] to the evaluation of autonomous decision-making systems operating under uncertainty.

From the perspective of financial economics, the results provide evidence consistent with bounded market efficiency. The Efficient Market Hypothesis in its semi-strong form [8] predicts that public information is rapidly incorporated into prices, leaving limited room for systematic prediction. The improvements documented in this thesis, while statistically significant and economically meaningful, are modest in absolute terms: the best model achieves 0.54% MAPE, meaning that on average the prediction deviates from the realized price by approximately half a percent. This is consistent with a market that is efficient enough to preclude large, persistent prediction improvements but leaves sufficient structure for architectures that can combine heterogeneous information sources, adapt to changing market conditions, and evaluate the quality of their own decisions. The largest absolute gains occur during volatile periods and around information events, consistent with the behavioral finance literature on delayed price incorporation of complex or ambiguous information [11].

7.3 Limitations

The limitations discussed in each contribution chapter converge into several cross-cutting concerns that affect the interpretation of the thesis as a whole. The most consequential of these concerns is the scope and composition of the dataset. The 20 stocks used throughout this thesis were selected from the current S&P 500 universe, which introduces survivorship bias: companies that failed, were acquired, or were delisted between 1982 and 2025 are absent from the analysis. Because the selected stocks are disproportionately successful over the evaluation period, reported performance metrics may overstate what a real-time investor would experience when selecting from the full market at each point in time. This concern applies equally to all three contributions, as they share the same stock universe and temporal partitioning. Beyond survivorship bias, the 20-stock universe represents a limited cross-

section of the market. While the selection spans eight GICS sectors and captures meaningful cross-sectional diversity, the graph contains only 20 nodes, which constrains the relational topology that the node transformer can learn. Larger universes of 50 to 500 stocks would provide richer graph structure and more robust estimates of cross-sectional effects but would require distributed training infrastructure beyond the scope of this work.

The reliance on a single geographic market further constrains generalizability. All experiments are conducted on U.S. large-cap equities, which benefit from deep liquidity, transparent price discovery, and extensive analyst coverage. Markets with lower liquidity, less transparency, or fundamentally different microstructure characteristics, such as emerging market equities, over-the-counter instruments, or cryptocurrency markets, may exhibit different predictability properties. The architectural choices that produce improvements in one market context may not transfer directly to another, and the specific regime detection thresholds, pathway specializations, and fusion weights learned by the system reflect the statistical properties of U.S. equities rather than universal market dynamics.

Sentiment data is sourced exclusively from X (formerly Twitter) and restricted to English-language posts. Social media coverage varies across stocks and time periods, with technology and consumer-facing companies receiving substantially more discussion than utilities or industrials, introducing an asymmetry in the informativeness of the sentiment modality across the stock universe. Changes to the X API access policies have made exact replication of the sentiment collection pipeline challenging, and the temporal limitation of sentiment availability (2007 onward, with features set to zero for 1982–2006) means that the sentiment component receives limited training signal during most of the training window. These data constraints do not invalidate the architectural contribution, but they bound the conditions under which the reported performance can be expected to hold.

On the methodological side, the LLM-based evaluation framework introduced in Chapter 6 faces inherent limitations rooted in the capabilities of the underlying language models. The three judges bring financial reasoning capabilities acquired through pre-training, but their

understanding of domain-specific concepts such as regime detection quality and risk calibration appropriateness is bounded by their training data rather than grounded in direct experience with financial markets. The framework has been validated on a single architecture (the AE-NodeFormer + SAC system from Chapter 5), and applying it to architecturally distinct systems would require redesigning the evaluation dimensions and rubric to reflect the new system's decision structure. The general principle of multi-dimensional behavioral evaluation using LLM judges is architecture-agnostic, but the specific implementation developed here is tied to the system being evaluated. Operating all judges at temperature zero produces deterministic evaluations but eliminates the stochastic variation that could reveal model uncertainty about borderline cases.

The economic validity of the reported improvements requires qualification. Transaction cost modeling across the thesis applies a fixed proportional cost of 10 basis points that does not account for variable market impact, execution slippage, or the capacity constraints that arise when order size increases relative to available liquidity. The 32% daily portfolio turnover observed in the Chapter 4 backtest would amplify these unmodeled frictions in practice, and the net returns reported throughout the thesis should be interpreted as upper bounds on what would be realizable under institutional trading conditions. Backtesting is performed on the same stock universe used for model development, leaving external validity on unseen securities untested. A live paper-trading evaluation, while beyond the scope of this thesis, would provide substantially stronger evidence of economic viability.

The reinforcement learning controller introduces a circularity in which the system learns regime boundaries from its own prediction performance, meaning that poor initial predictions can lead to suboptimal threshold learning. The staged training protocol mitigates this by pre-training each component before SAC optimization begins, but the circularity is not fully eliminated. The introduction of LLM-based auxiliary rewards in Chapter 6 adds a further feedback loop, and while the lambda sensitivity analysis demonstrates stable performance within the $[0.10, 0.20]$ operating range, the risk of reward hacking remains a theoretical concern

for longer deployment horizons or systems whose behavioral patterns differ substantially from those the LLMs encountered in their pre-training data. The periodic evaluation protocol, which limits LLM feedback to corrective interventions rather than continuous reward modification, provides a structural safeguard, but long-term stability under extended deployment has not been tested.

7.4 Future Research Directions

The limitations and open questions identified across this thesis suggest several concrete directions for future research, organized from immediate extensions of the current work to broader research programs. Each direction addresses at least one of the limitations discussed in the preceding section while building on the architectural foundations established in the contribution chapters.

7.4.1 Scaling to Larger Stock Universes

The most direct extension is expanding the stock universe from 20 to 100 or more equities while preserving the graph-based architecture. The node transformer's kernelized attention approximation [12] achieves linear complexity in the number of nodes, making the attention computation itself scalable. The binding constraint lies in the training pipeline, which includes per-stock BERT sentiment inference, per-stock autoencoder training, and the SAC controller's state space, all of which grow with the number of stocks. A hierarchical graph construction approach, in which stocks are grouped into sector-level clusters with full-resolution attention computed within clusters and coarser aggregation across clusters, would reduce the quadratic cost of cross-cluster interactions while preserving the fine-grained intra-sector dynamics that the current 20-stock graph captures. Per-stock model components can be parallelized across GPUs, and the SAC state space can be compressed through sector-level summary

statistics rather than per-stock features. This direction would also address the survivorship bias limitation by enabling evaluation on a point-in-time universe constructed from historical index constituents, where stocks enter and exit the graph as they are added to or removed from the index.

7.4.2 Dynamic Graph Construction

The current graph structure initializes edge weights from sector co-membership and rolling correlations computed over a fixed historical window, with learnable refinement during training. However, the topology itself, specifically which nodes are connected and through what type of relationship, remains fixed throughout training and inference. In practice, inter-stock relationships evolve as business models pivot, competitive positioning between firms changes, and macroeconomic conditions shift. A dynamic graph construction mechanism that periodically re-evaluates the edge set based on recent correlation structure, supply chain data extracted from SEC filings, or textual co-occurrence in financial news would allow the model to capture evolving relationships that a static graph cannot represent. The challenge lies in maintaining training stability as the graph topology changes, since abrupt edge modifications can disrupt the node representations learned by the transformer. Incremental edge weight decay combined with periodic re-initialization of low-weight edges offers one approach that avoids catastrophic disruption of learned representations while allowing the graph to adapt over time.

7.4.3 Cross-Market and Cross-Asset Generalization

All experiments in this thesis are conducted on U.S. large-cap equities, a market segment characterized by deep liquidity and extensive information coverage. Extending the framework to international equities in European, Asian, and emerging markets, as well as to fixed income instruments, commodities, and cryptocurrency markets, would test the generalizability of the

architectural principles developed here. Each asset class presents distinct challenges: fixed income markets exhibit term structure dynamics absent from equities, commodity prices are driven by physical supply and demand factors that operate on different timescales, and cryptocurrency markets operate continuously with fundamentally different microstructure and investor composition. The autoencoder-based regime detection and dual-pathway architecture may prove particularly valuable in cryptocurrency markets, where volatility regime shifts are more frequent and more extreme than in traditional equities. Cross-asset evaluation would reveal whether the regime-aware routing principle transfers across domains or whether the specific implementation requires domain-specific redesign of the autoencoder features, SAC state representation, and pathway specialization.

7.4.4 LLM Judge Distillation

The LLM evaluation framework requires API calls to three commercial language models for each episode evaluation, introducing latency and cost that scale linearly with the number of episodes and evaluation cycles. The cost of the full evaluation campaign documented in Chapter 6 was approximately $180, which is modest for research purposes but would become prohibitive for real-time deployment with frequent evaluation cycles across larger stock universes. Distilling the judges' evaluation patterns into a smaller, faster model would reduce both cost and latency while preserving the dimension-specific scoring behavior. The distillation target could be a fine-tuned smaller language model or a non-linguistic neural network that maps behavioral trace features directly to dimensional scores [147]. The perturbation validation methodology developed in Chapter 6 provides a built-in test suite for verifying that the distilled model preserves the original judges' dimension specificity: if the distilled model passes the same perturbation tests with comparable score drops and off-target stability, it can be considered a valid replacement for the full LLM panel.

7.4.5 Alternative Sentiment Sources

The current system relies exclusively on X (formerly Twitter) for sentiment data, which introduces coverage bias toward stocks with active social media discussion and vulnerability to platform API access changes. Incorporating additional sentiment sources would both broaden coverage and provide complementary signals that capture different aspects of market sentiment. Financial news articles offer professional analysis with higher signal-to-noise ratio but lower frequency than social media. SEC filings (10-K, 10-Q, and 8-K reports) contain forward-looking statements and risk factor disclosures that reflect management's own assessment of business conditions and prospects. Earnings call transcripts capture the tone and content of executive commentary, which contains information beyond what is reflected in the accompanying financial statements. Each source presents distinct NLP challenges: news articles require entity disambiguation and event extraction, SEC filings involve domain-specific legal and accounting language, and earnings calls include both prepared remarks and spontaneous responses to analyst questions. Beyond these professional sources, retail-investor venues such as Reddit (for example, the WallStreetBets community) and StockTwits, together with sell-side analyst reports and estimate revisions, supply further sentiment signals that differ in tone, participant sophistication, and update frequency. A multi-source sentiment architecture that maintains separate encoding pathways for each source and learns adaptive fusion weights, analogous to the sentiment fusion mechanism developed in Chapter 4, would exploit the complementary strengths of these sources while accommodating their different temporal resolutions and coverage characteristics, aggregating the full set of sources into a single per-stock sentiment signal for the prediction model.

7.4.6 Real-Time Deployment Architecture

The current system is evaluated in an offline backtesting framework where all data is available in advance and computation time is not constrained. A real-time deployment

would introduce latency requirements, data streaming considerations, and fault tolerance concerns that the current design does not address. Market data arrives as a continuous stream during trading hours, sentiment data arrives asynchronously with variable delay, and predictions must be generated within a fixed time budget to be actionable for trading decisions. The autoencoder and node transformer forward passes are computationally efficient on modern GPUs, but the full pipeline, including sentiment processing, graph attention, regime detection, and SAC inference, must complete within the inter-prediction interval. Profiling the current pipeline to identify latency bottlenecks, implementing streaming data ingestion with incremental feature updates rather than full-window recomputation, and designing a fallback mechanism that produces predictions from a simplified model when the full pipeline cannot complete within the time budget are engineering challenges that must be addressed before the system can operate in a live trading environment. The LLM evaluation component, which operates on a periodic rather than continuous basis, is more naturally suited to asynchronous batch processing and would not require real-time latency guarantees.

7.4.7 Multi-Pathway Regime Extension

The two-pathway design extends naturally to K pathways with one autoencoder and one node transformer per regime type, but the implementation cost grows along three axes. Parameter count scales approximately linearly with K, since each additional pathway contributes its own autoencoder and node transformer, together adding roughly 20.4 million parameters per pathway (consistent with the gap between the *No Dual Paths* and *Full Model* rows of Table 5.7); a three-pathway extension would total approximately 63 million trainable parameters and a four-pathway extension approximately 84 million, against the 42.8 million of the current two-pathway design. Training time follows a similar pattern in the per-pathway pre-training stages (Stages 1 and 2 of Figure 5.9), with each additional pathway adding approximately 30 to 35 minutes on a single A100; the four-stage protocol as a whole would extend to approximately 3 hours 40 minutes for $K = 3$ and 4 hours 15

minutes for $K = 4$. Inference latency scales similarly, with the per-stock budget rising from the current 2.4 ms to approximately 3.1 ms for $K = 3$ and 3.9 ms for $K = 4$ (derived from the 15 ms incremental per-pathway cost implied by the difference between the dual-pathway and single-pathway rows of Table 5.7), still inside the intraday-to-daily deployment band quantified in Section 5.3.14. The SAC controller faces a more substantive design change: its current continuous action space $[\Delta\tau, \Delta\alpha] \in \mathbb{R}^2$ must be extended to a K-simplex over pathway-blending weights and a $(K-1)$-dimensional set of inter-pathway routing thresholds, which complicates exploration and increases the effective state-action dimensionality. The remaining concern is training stability: the mixture-of-experts literature documents that K-way routing is prone to dead-expert degeneracy where some pathways are rarely selected and never receive sufficient gradient signal [139], and load-balancing regularizers such as those proposed in [139, 140] would likely be required to prevent collapse to a small subset of pathways. The two-pathway design adopted in this thesis is the simplest configuration that captures the binary normal-versus-event distinction supported by the empirical results, and the multi-pathway extension is appropriate when the application requires finer-grained regime typology than the binary split can express.

7.5 Concluding Remarks

This thesis developed a progressive sequence of architectures for stock price forecasting that addresses three fundamental challenges: the integration of heterogeneous information sources through graph-based multimodal fusion, the adaptation to non-stationary market conditions through unsupervised regime detection and reinforcement learning control, and the evaluation and autonomous correction of intermediate decision quality through LLM-based behavioral assessment. The resulting system, evaluated on 20 S&P 500 equities over a test period spanning 2017 to 2025, reduces one-day MAPE from 1.20% (ARIMA baseline) to 0.54%

and improves directional accuracy from 55% to 74%, with the largest gains concentrated during volatile market periods where accurate forecasting is most valuable and most difficult.

Beyond the specific performance improvements, the thesis contributes a methodology for bridging the gap between aggregate output metrics and process-level decision quality in autonomous AI systems. The demonstration that LLM judges can produce behavioral assessments with sufficient specificity, consistency, and predictive validity to serve as learning signals for reinforcement learning control opens a direction that extends beyond financial prediction to any domain where complex autonomous systems make sequences of interdependent decisions under uncertainty. The evaluation dimensions, perturbation validation methodology, and credit assignment mechanism developed in Chapter 6 provide a template that can be adapted to other agentic systems, from robotic control to autonomous navigation to clinical decision support, in any setting where the gap between outcome quality and process quality demands a more granular form of evaluation than aggregate performance metrics alone can provide.

The three research gaps identified at the outset of this thesis, the absence of unified multimodal graph-temporal prediction, the reliance on manual regime specification, and the lack of behavioral evaluation for multi-stage pipelines, have each been addressed through architectures that are validated against rigorous baselines, supported by statistical significance tests, and honestly bounded by the limitations inherent in the dataset, the modeling assumptions, and the evaluation methodology. The progressive improvement from 0.80% to 0.59% to 0.54% MAPE, while modest in absolute terms, represents a meaningful advance in a domain where the predictable component of returns is inherently small. The more lasting contribution may be the methodological framework itself: a principled approach to building, adapting, and evaluating autonomous prediction systems that can be extended, refined, and applied to domains well beyond financial forecasting.

References

[1] J. D. Hamilton. "A new approach to the economic analysis of nonstationary time series and the business cycle". In: *Econometrica* 57.2 (1989), pp. 357–384.

[2] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip. "A comprehensive survey on graph neural networks". In: *IEEE Transactions on Neural Networks and Learning Systems* 32.1 (2020), pp. 4–24.

[3] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. "Attention is all you need". In: *Advances in Neural Information Processing Systems*. Vol. 30. 2017, pp. 5998–6008.

[4] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. "BERT: Pre-training of deep bidirectional transformers for language understanding". In: *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*. 2019, pp. 4171–4186.

[5] G. E. Hinton and R. R. Salakhutdinov. "Reducing the dimensionality of data with neural networks". In: *Science* 313.5786 (2006), pp. 504–507.

[6] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. 2nd. MIT Press, 2018.

[7] J. Zhang, Y. Teng, and W. Chen. "Stock market prediction via deep learning techniques: A survey". In: *arXiv preprint arXiv:2212.12717* (2022).

[8] E. F. Fama. "Efficient capital markets: A review of theory and empirical work". In: *The Journal of Finance* 25.2 (1970), pp. 383–417.

[9] I. W. R. Martin and S. Nagel. "Market efficiency in the age of big data". In: *Journal of Financial Economics* 145.1 (2022), pp. 154–177.

[10] I. Goldstein. "Information in financial markets and its real effects". In: *Review of Finance* 27.1 (2023), pp. 1–32.

[11] D. Kahneman and A. Tversky. "Prospect theory: An analysis of decision under risk". In: *Econometrica* 47.2 (1979), pp. 263–291.

[12] Q. Wu, W. Zhao, Z. Li, D. P. Wipf, and J. Yan. "NodeFormer: A scalable graph structure learning transformer for node classification". In: *Advances in Neural Information Processing Systems*. Vol. 35. 2022, pp. 27387–27401.

[13] W. Chen, M. Jiang, W.-G. Zhang, and Z. Chen. "A novel graph convolutional feature based convolutional neural network for stock trend prediction". In: *Information Sciences* 556 (2021), pp. 67–94.

[14] C. Wang, H. Liang, B. Wang, X. Cui, and Y. Xu. "MG-Conv: A spatiotemporal multi-graph convolutional neural network for stock market index trend prediction". In: *Computers and Electrical Engineering* 103 (2022), p. 108285.

[15] K. Hou. "Industry information diffusion and the lead-lag effect in stock returns". In: *Review of Financial Studies* 20.4 (2007), pp. 1113–1138.

[16] L. Cohen and A. Frazzini. "Economic links and predictable returns". In: *The Journal of Finance* 63.4 (2008), pp. 1977–2011.

[17] P. C. Tetlock. "Giving content to investor sentiment: The role of media in the stock market". In: *The Journal of Finance* 62.3 (2007), pp. 1139–1168.

[18] T. Loughran and B. McDonald. "When is a liability not a liability? Textual analysis, dictionaries, and 10-Ks". In: *The Journal of Finance* 66.1 (2011), pp. 35–65.

[19] J. Bollen, H. Mao, and X. Zeng. "Twitter mood predicts the stock market". In: *Journal of Computational Science* 2.1 (2011), pp. 1–8.

[20] A. Ang and A. Timmermann. "Regime changes and financial markets". In: *Annual Review of Financial Economics* 4 (2012), pp. 313–337.

[21] P. Nystrup, H. Madsen, and E. Lindström. "Regime-based versus static asset allocation: Letting the data speak". In: *The Journal of Portfolio Management* 44.1 (2017), pp. 103–115.

[22] K. Krippendorff. "Computing Krippendorff's alpha-reliability". In: *Annenberg School for Communication Departmental Papers* (2011). Available at `https://repository.upenn.edu/asc_papers/43/`.

[23] K. Cortis, A. Freitas, T. Daudert, M. Huerlimann, M. Zarrouk, S. Handschuh, and B. Davis. "SemEval-2017 Task 5: Fine-grained sentiment analysis on financial microblogs and news". In: *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. 2017, pp. 519–535.

[24] B. Mandelbrot. "The variation of certain speculative prices". In: *The Journal of Business* 36.4 (1963), pp. 394–419.

[25] R. Cont. "Empirical properties of asset returns: Stylized facts and statistical issues". In: *Quantitative Finance* 1.2 (2001), pp. 223–236.

[26] G. E. P. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung. *Time series analysis: Forecasting and control*. 5th. John Wiley & Sons, 2015.

[27] P. R. Low and E. Sakk. "Comparison between autoregressive integrated moving average and long short-term memory models for stock price prediction". In: *IAES International Journal of Artificial Intelligence* 12.4 (2023), pp. 1828–1835.

[28] S. T. Wahyudi. "The ARIMA model for the Indonesia stock price". In: *International Journal of Economics and Management* 11 (2017), pp. 223–236.

[29] C. A. Sims. "Macroeconomics and reality". In: *Econometrica* 48.1 (1980), pp. 1–48.

[30] H. Lütkepohl. *New introduction to multiple time series analysis*. Springer, 2005.

[31] C. W. J. Granger. "Investigating causal relations by econometric models and cross-spectral methods". In: *Econometrica* 37.3 (1969), pp. 424–438.

[32] R. F. Engle. "Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation". In: *Econometrica* 50.4 (1982), pp. 987–1007.

[33] T. Bollerslev. "Generalized autoregressive conditional heteroskedasticity". In: *Journal of Econometrics* 31.3 (1986), pp. 307–327.

[34] P. R. Hansen, Z. Huang, and H. H. Shek. "Realized GARCH: A joint model for returns and realized measures of volatility". In: *Journal of Applied Econometrics* 27.6 (2012), pp. 877–906.

[35] R. K. Nayak, D. Mishra, and A. K. Rath. "A naïve SVM-KNN based stock market trend reversal analysis for Indian benchmark indices". In: *Applied Soft Computing* 35 (2015), pp. 670–680.

[36] M. Siddique and D. Panda. "Prediction of stock index of Tata Steel using hybrid machine learning based optimization techniques". In: *International Journal of Recent Technology and Engineering* 8 (2019), pp. 3186–3193.

[37] S. Basak, S. Kar, S. Saha, L. Khaidem, and S. R. Dey. "Predicting the direction of stock market prices using tree-based classifiers". In: *The North American Journal of Economics and Finance* 47 (2019), pp. 552–567.

[38] Y. LeCun, Y. Bengio, and G. Hinton. "Deep learning". In: *Nature* 521.7553 (2015), pp. 436–444.

[39] S. K. Chandar. "Convolutional neural network for stock trading using technical indicators". In: *Automated Software Engineering* 29 (2022), pp. 1–14.

[40] M. Wen, P. Li, L. Zhang, and Y. Chen. "Stock market trend prediction using high-order information of time series". In: *IEEE Access*. Vol. 7. 2017, pp. 28299–28308.

[41] S. Hochreiter and J. Schmidhuber. "Long short-term memory". In: *Neural Computation* 9.8 (1997), pp. 1735–1780.

[42] L. Di Persio and O. Honchar. "Recurrent neural networks approach to the financial forecast of Google assets". In: *International Journal of Mathematics and Computers in Simulation* 11 (2017), pp. 7–13.

[43] T. Fischer and C. Krauss. "Deep learning with long short-term memory networks for financial market predictions". In: *European Journal of Operational Research* 270.2 (2018), pp. 654–669.

[44] M. A. Hossain, R. Karim, R. Thulasiram, N. Bruce, and Y. Wang. "Hybrid deep learning model for stock price prediction". In: *Proceedings of the IEEE Symposium Series on Computational Intelligence*. 2018, pp. 1837–1844.

[45] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. "Learning phrase representations using RNN encoder-decoder for statistical machine translation". In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. 2014, pp. 1724–1734.

[46] Y. Xu, L. Chhim, B. Zheng, et al. "Stacked deep learning structure with bidirectional long-short term memory for stock market prediction". In: *Communications in Computer and Information Science*. Vol. 1265. 2020, pp. 447–460.

[47] M. Liu, H. Sheng, N. Zhang, et al. "A new deep network model for stock price prediction". In: *International Conference on Machine Learning for Cyber Security*. 2022, pp. 413–426.

[48] D. Bahdanau, K. Cho, and Y. Bengio. "Neural machine translation by jointly learning to align and translate". In: *International Conference on Learning Representations*. 2015.

[49] W. Lu, J. Li, J. Wang, et al. "A CNN-BiLSTM-AM method for stock price prediction". In: *Neural Computing and Applications* 33 (2021), pp. 4741–4753.

[50] S. Li and Z. Qian. "A financial forecasting model based on transformer architecture". In: *Proceedings of the IEEE International Conference on Big Data*. 2019, pp. 5384–5386.

[51] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang. "Informer: Beyond efficient transformer for long sequence time-series forecasting". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. 12. 2021, pp. 11106–11115.

[52] N. Kitaev, Ł. Kaiser, and A. Levskaya. "Reformer: The efficient transformer". In: *International Conference on Learning Representations*. 2020.

[53] H. Wu, J. Xu, J. Wang, and M. Long. "Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting". In: *Advances in Neural Information Processing Systems*. Vol. 34. 2021, pp. 22419–22430.

[54] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl. "Neural message passing for quantum chemistry". In: *International Conference on Machine Learning*. 2017, pp. 1263–1272.

[55] T. N. Kipf and M. Welling. "Semi-supervised classification with graph convolutional networks". In: *International Conference on Learning Representations*. 2017, pp. 1–14.

[56] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. "Graph attention networks". In: *International Conference on Learning Representations*. 2018.

[57] F. Feng, X. He, X. Wang, C. Luo, Y. Liu, and T.-S. Chua. "Temporal relational ranking for stock prediction". In: *ACM Transactions on Information Systems*. Vol. 37. 2. 2019, pp. 1–30.

[58] R. Sawhney, S. Agarwal, A. Wadhwa, and R. R. Shah. "Spatiotemporal hypergraph convolution network for stock movement forecasting". In: *IEEE International Conference on Data Mining*. 2020, pp. 482–491.

[59] D. Le, S. Rajasegarar, W. Luo, T. T. Nguyen, N. Vo, Q. Nguyen, and M. Angelova. "EGCN: Entropy-based graph convolutional network for anomalous pattern detection and forecasting in real estate markets". In: *PLoS ONE* 20.10 (2025), e0334141. DOI: 10.1371/journal.pone.0334141.

[60] D. Le, S. Rajasegarar, W. Luo, T. T. Nguyen, and M. Angelova. "Enhancing real estate prediction with entropy-based pattern analysis and economic sentiment integration". In: *Engineering Computations* (2025), pp. 1–24. DOI: 10.1108/EC-09-2024-0895.

[61] C. Ying, T. Cai, S. Luo, S. Zheng, G. Ke, D. He, Y. Fang, and T.-Y. Liu. "Do Transformers Really Perform Bad for Graph Representation Learning?" In: *Advances in Neural Information Processing Systems*. Vol. 34. 2021, pp. 28877–28888.

[62] D. Kreuzer, D. Beaini, W. L. Hamilton, V. Létourneau, and P. Tossou. "Rethinking Graph Transformers with Spectral Attention". In: *Advances in Neural Information Processing Systems*. Vol. 34. 2021, pp. 21618–21629.

[63] L. Rampášek, M. Galkin, V. P. Dwivedi, A. T. Luu, G. Wolf, and D. Beaini. "Recipe for a General, Powerful, Scalable Graph Transformer". In: *Advances in Neural Information Processing Systems*. Vol. 35. 2022, pp. 14501–14515.

[64] P. Malo, A. Sinha, P. Korhonen, J. Wallenius, and P. Takala. "Good debt or bad debt: Detecting semantic orientations in economic texts". In: *Journal of the Association for Information Science and Technology* 65.4 (2014), pp. 782–796.

[65] R. P. Schumaker and H. Chen. "Textual analysis of stock market prediction using breaking financial news: The AZFin text system". In: *ACM Transactions on Information Systems* 27.2 (2009), pp. 1–19.

[66] J. Si, A. Mukherjee, B. Liu, Q. Li, H. Li, and X. Deng. "Exploiting topic based Twitter sentiment for stock prediction". In: *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*. 2013, pp. 24–29.

[67] X. Ding, Y. Zhang, T. Liu, and J. Duan. "Deep learning for event-driven stock prediction". In: *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*. 2015, pp. 2327–2333.

[68] Y. Xu and S. B. Cohen. "Stock movement prediction from tweets and historical prices". In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*. 2018, pp. 1970–1979.

[69] D. Araci. "FinBERT: Financial sentiment analysis with pre-trained language models". In: *arXiv preprint arXiv:1908.10063*. 2019.

[70] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. "RoBERTa: A robustly optimized BERT pretraining approach". In: *arXiv preprint arXiv:1907.11692* (2019).

[71] D. M. Q. Nelson, A. C. M. Pereira, and R. A. de Oliveira. "Stock market's price movement prediction with LSTM neural networks". In: *International Joint Conference on Neural Networks*. 2017, pp. 1419–1426.

[72] R. Kim, C. H. So, M. Jeong, S. Lee, J. Kim, and J. Kang. "HATS: A Hierarchical Graph Attention Network for Stock Movement Prediction". In: *Proceedings of the Second Workshop on Economics and Natural Language Processing*. 2019, pp. 1–10.

[73] F. X. Diebold, J.-H. Lee, and G. C. Weinbach. "Regime switching with time-varying transition probabilities". In: *Business Cycles: Durations, Dynamics, and Forecasting* (1994), pp. 144–165.

[74] H. Tong. *Non-linear time series: A dynamical system approach*. Oxford University Press, 1990.

[75] D. P. Kingma and M. Welling. "Auto-encoding variational Bayes". In: *International Conference on Learning Representations*. 2014.

[76] A. Pumsirirat and L. Yan. "Credit card fraud detection using deep learning based on auto-encoder and restricted Boltzmann machine". In: *International Journal of Advanced Computer Science and Applications* 9.1 (2018), pp. 18–25.

[77] M. Ahmed, A. N. Mahmood, and J. Hu. "A survey of network anomaly detection techniques". In: *Journal of Network and Computer Applications* 60 (2016), pp. 19–31.

[78] W. Bao, J. Yue, and Y. Rao. "A deep learning framework for financial time series using stacked autoencoders and long-short term memory". In: *PLOS ONE* 12.7 (2017), e0180944.

[79] S. Aminikhanghahi and D. J. Cook. "A survey of methods for time series change point detection". In: *Knowledge and Information Systems* 51.2 (2017), pp. 339–367.

[80] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. "Human-level control through deep reinforcement learning". In: *Nature*. Vol. 518. 7540. 2015, pp. 529–533.

[81] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. "Continuous control with deep reinforcement learning". In: *International Conference on Learning Representations*. 2016.

[82] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. "Proximal policy optimization algorithms". In: *arXiv preprint arXiv:1707.06347* (2017).

[83] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor". In: *International Conference on Machine Learning*. PMLR. 2018, pp. 1861–1870.

[84] J. Moody and M. Saffell. "Learning to trade via direct reinforcement". In: *IEEE Transactions on Neural Networks* 12.4 (2001), pp. 875–889.

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

[86] Y. Deng, F. Bao, Y. Kong, Z. Ren, and Q. Dai. "Deep direct reinforcement learning for financial signal representation and trading". In: *IEEE Transactions on Neural Networks and Learning Systems* 28.3 (2016), pp. 653–664.

[87] H. Yang, X.-Y. Liu, S. Zhong, and A. Walid. "Deep reinforcement learning for automated stock trading: An ensemble strategy". In: *Proceedings of the First ACM International Conference on AI in Finance* (2020), pp. 1–8.

[88] X.-Y. Liu, H. Yang, J. Gao, and C. D. Wang. "FinRL: Deep reinforcement learning framework to automate trading in quantitative finance". In: *Proceedings of the Second ACM International Conference on AI in Finance* (2021), pp. 1–9.

[89] L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. P. Xing, et al. "Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena". In: *Advances in Neural Information Processing Systems* 36 (2023).

[90] Y. Liu, D. Iter, Y. Xu, S. Wang, R. Xu, and C. Zhu. "G-Eval: NLG Evaluation using GPT-4 with Better Human Alignment". In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. 2023.

[91] Y. Dubois, X. Li, R. Taori, T. Zhang, I. Gulrajani, J. Ba, C. Guestrin, P. Liang, and T. B. Hashimoto. "AlpacaFarm: A Simulation Framework for Methods that Learn from Human Feedback". In: *Advances in Neural Information Processing Systems* 36 (2024).

[92] J. Cohen. "A coefficient of agreement for nominal scales". In: *Educational and Psychological Measurement* 20.1 (1960), pp. 37–46.

[93] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao. "ReAct: Synergizing Reasoning and Acting in Language Models". In: *International Conference on Learning Representations*. 2023.

[94] N. Shinn, F. Cassano, A. Gopinath, K. Narasimhan, and S. Yao. "Reflexion: Language Agents with Verbal Reinforcement Learning". In: *Advances in Neural Information Processing Systems*. Vol. 36. 2023.

[95] L. Wang, C. Ma, X. Feng, Z. Zhang, H. Yang, J. Zhang, Z. Chen, J. Tang, X. Chen, Y. Lin, et al. "A Survey on Large Language Model Based Autonomous Agents". In: *Frontiers of Computer Science* 18.6 (2024), p. 186345.

[96] Z. Xi, W. Chen, X. Guo, W. He, Y. Ding, B. Hong, M. Zhang, J. Wang, S. Jin, E. Zhou, et al. "The Rise and Potential of Large Language Model Based Agents: A Survey". In: *arXiv preprint arXiv:2309.07864* (2023).

[97] Z. Bodie, A. Kane, and A. J. Marcus. *Investments*. 13th. McGraw-Hill Education, 2024.

[98] S. M. Kazemi, R. Goel, K. Jain, I. Kobyzev, A. Sethi, P. Forsyth, and P. Poupart. "Representation Learning for Dynamic Graphs: A Survey". In: *Journal of Machine Learning Research* 21.70 (2020), pp. 1–73.

[99] E. J. Elton, M. J. Gruber, and C. R. Blake. "Survivorship Bias and Mutual Fund Performance". In: *The Review of Financial Studies* 9.4 (1996), pp. 1097–1120.

[100] Yahoo Finance. *yfinance: Download Market Data from Yahoo! Finance API*. `https://pypi.org/project/yfinance/`. 2024.

[101] M. L. de Prado. *Advances in Financial Machine Learning*. John Wiley & Sons, 2018.

[102] I. K. Nti, A. F. Adekoya, and B. A. Weyori. "A Systematic Review of Fundamental and Technical Analysis of Stock Market Predictions". In: *Artificial Intelligence Review* 53.4 (2020), pp. 3007–3057.

[103] W. Jiang. "Applications of Deep Learning in Stock Market Prediction: Recent Progress". In: *Expert Systems with Applications* 184 (2021), p. 115537.

[104] C. Lento, T. Suleman, and M. Soufian. "The Profitability of Technical Analysis: A Combined Signal Approach". In: *Journal of Applied Business Research* 23.1 (2007), pp. 13–27.

[105] J. Patel, S. Shah, P. Thakkar, and K. Kotecha. "Predicting Stock and Stock Price Index Movement Using Trend Deterministic Data Preparation and Machine Learning Techniques". In: *Expert Systems with Applications* 42.1 (2015), pp. 259–268.

[106] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller. "Efficient BackProp". In: *Neural Networks: Tricks of the Trade* (2012), pp. 9–48.

[107] W. Brock, J. Lakonishok, and B. LeBaron. "Simple Technical Trading Rules and the Stochastic Properties of Stock Returns". In: *The Journal of Finance* 47.5 (1992), pp. 1731–1764.

[108] J. W. Wilder. *New Concepts in Technical Trading Systems*. Trend Research, 1978.

[109] G. Appel. *Technical Analysis: Power Tools for Active Investors*. FT Press, 2005.

[110] J. Y. Campbell, A. W. Lo, and A. C. MacKinlay. *The Econometrics of Financial Markets*. Princeton University Press, 1997.

[111] T. G. Andersen, T. Bollerslev, F. X. Diebold, and P. Labys. "Modeling and Forecasting Realized Volatility". In: *Econometrica* 71.2 (2003), pp. 579–625.

[112] O. B. Sezer, M. U. Gudelek, and A. M. Ozbayoglu. "Financial Time Series Forecasting with Deep Learning: A Systematic Literature Review: 2005–2019". In: *Applied Soft Computing* 90 (2020), p. 106181.

[113] K. Johnson. *pandas-ta: Pandas Technical Analysis Library, version 0.3.14b*. `https://github.com/twopirllc/pandas-ta`. Open-source Python library for technical analysis indicators. Accessed: 2026-04-01. 2023.

[114] J. J. Murphy. *Technical analysis of the financial markets: A comprehensive guide to trading methods and applications*. New York Institute of Finance, 1999.

[115] S. Cresci. "A Decade of Social Bot Detection". In: *Communications of the ACM* 63.10 (2020), pp. 72–83.

[116] R. A. Potamias, G. Siolas, and A.-G. Stafylopatis. "A Transformer-Based Approach to Irony and Sarcasm Detection". In: *Neural Computing and Applications*. Vol. 32. Springer, 2020, pp. 17309–17320.

[117] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov. "Bag of Tricks for Efficient Text Classification". In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics* 2 (2017), pp. 427–431.

[118] N. T. Dang, M. N. Moreno-García, and F. De la Prieta. "Sentiment Analysis is Not Solved! Assessing and Probing Sentiment Classification". In: *IEEE Access* 8 (2020), pp. 174527–174541.

[119] L. J. Tashman. "Out-of-Sample Tests of Forecasting Accuracy: An Analysis and Review". In: *International Journal of Forecasting* 16.4 (2000), pp. 437–450.

[120] B. Lim and S. Zohren. "Time-Series Forecasting with Deep Learning: A Survey". In: *Philosophical Transactions of the Royal Society A* 379.2194 (2021), p. 20200209.

[121] R. J. Hyndman and A. B. Koehler. "Another Look at Measures of Forecast Accuracy". In: *International Journal of Forecasting* 22.4 (2006), pp. 679–688.

[122] T. Chai and R. R. Draxler. "Root Mean Square Error (RMSE) or Mean Absolute Error (MAE)? Arguments Against Avoiding RMSE in the Literature". In: *Geoscientific Model Development* 7.3 (2014), pp. 1247–1250.

[123] H. Theil. *Applied Economic Forecasting*. North-Holland Publishing Company, 1966.

[124] R. C. Grinold and R. N. Kahn. *Active Portfolio Management: A Quantitative Approach for Producing Superior Returns and Controlling Risk*. 2nd. McGraw-Hill, 2000.

[125] Y. Nejatbakhsh and M. Aliasgari. "Enhancing stock market prediction with hybrid deep learning: Integrating LSTM, transformer attention, federated learning, and sentiment analysis". In: *IEEE Access* 14 (2025), pp. 3926–3942. DOI: 10.1109/ACCESS.2025.3649668.

[126] B. Efron and R. J. Tibshirani. *An Introduction to the Bootstrap*. Chapman and Hall/CRC, 1993.

[127] J. Cohen. *Statistical Power Analysis for the Behavioral Sciences*. 2nd. Lawrence Erlbaum Associates, 1988.

[128] F. X. Diebold and R. S. Mariano. "Comparing Predictive Accuracy". In: *Journal of Business & Economic Statistics* 13.3 (1995), pp. 253–263.

[129] Z. Liu, J. Liu, Q. Zeng, and L. Wu. "VIX and stock market volatility predictability: A new approach". In: *Finance Research Letters* 48 (2022), p. 102887. DOI: 10.1016/j.frl.2022.102887.

[130] H.-M. Krolzig. *Markov-Switching Vector Autoregressions: Modelling, Statistical Inference, and Application to Business Cycle Analysis*. Berlin: Springer-Verlag, 1997.

[131] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, and M. Long. "TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis". In: *Proceedings of the 11th International Conference on Learning Representations (ICLR)*. 2023.

[132] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam. "A Time Series is Worth 64 Words: Long-term Forecasting with Transformers". In: *Proceedings of the 11th International Conference on Learning Representations (ICLR)*. 2023.

[133] Y. Liu, T. Hu, H. Zhang, H. Wu, S. Wang, L. Ma, and M. Long. "iTransformer: Inverted Transformers Are Effective for Time Series Forecasting". In: *Proceedings of the 12th International Conference on Learning Representations (ICLR)*. 2024.

[134] M. Al Ridhawi, M. Haj Ali, and H. Al Osman. "Stock Market Prediction Using Node Transformer Architecture Integrated with BERT Sentiment Analysis". In: *IEEE Access* 14 (2026), pp. 72613–72631. DOI: 10.1109/ACCESS.2026.3691980.

[135] G. E. P. Box and G. M. Jenkins. *Time Series Analysis: Forecasting and Control*. San Francisco: Holden-Day, 1976.

[136] L. Breiman. "Random Forests". In: *Machine Learning* 45.1 (2001), pp. 5–32.

[137] V. N. Vapnik. *The Nature of Statistical Learning Theory*. New York: Springer, 1995.

[138] T. Chen and C. Guestrin. "XGBoost: A Scalable Tree Boosting System". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2016, pp. 785–794.

[139] N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. Le, G. Hinton, and J. Dean. "Outrageously large neural networks: The sparsely-gated mixture-of-experts layer". In: *International Conference on Learning Representations*. 2017. arXiv: 1701.06538 [cs.LG].

[140] W. Fedus, B. Zoph, and N. Shazeer. "Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity". In: *Journal of Machine Learning Research* 23.120 (2022), pp. 1–39.

[141] M. Lopez de Prado. *Advances in Financial Machine Learning*. John Wiley & Sons, 2018.

[142] Z. D. Akşehir and E. Kılıç. "Analyzing the critical steps in deep learning-based stock forecasting: a literature review". In: *PeerJ Computer Science* 10 (2024), e2312. DOI: 10.7717/peerj-cs.2312.

[143] W. Bao, Y. Cao, Y. Yang, H. Che, J. Huang, and S. Wen. "Data-driven stock forecasting models based on neural networks: A review". In: *Information Fusion* 113 (2025), p. 102616. DOI: 10.1016/j.inffus.2024.102616.

[144] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. "Training language models to follow instructions with human feedback". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 27730–27744.

[145] M. Al Ridhawi, M. Haj Ali, and H. Al Osman. "Adaptive Regime-Aware Stock Price Prediction Using Autoencoder-Gated Dual Node Transformers with Reinforcement Learning Control". In: *Submitted to Applied Intelligence* (2026). Under revision. Preprint: arXiv:2603.19136. arXiv: 2603.19136.

[146] D. V. Cicchetti. "Guidelines, Criteria, and Rules of Thumb for Evaluating Normed and Standardized Assessment Instruments in Psychology". In: *Psychological Assessment* 6.4 (1994), pp. 284–290.

[147] G. Hinton, O. Vinyals, and J. Dean. "Distilling the Knowledge in a Neural Network". In: *arXiv preprint arXiv:1503.02531* (2015).

Appendix A

LLM Judge Prompt Template

This appendix records the prompt template applied uniformly to all three judges (GPT 5.4, Claude 4.6 Opus, and Gemini 3.1 Pro) at temperature zero with a 2,000-token output budget. The template has four components: (i) a fixed system message that establishes the evaluator role, the trace semantics, and the rubric; (ii) the episode-specific behavioral-trace serialization; (iii) the output-schema specification; and (iv) the failure-label vocabulary listed in Table A.1. Only the trace serialization varies across episodes; the system message, rubric, schema, and vocabulary are fixed. The text reproduced below is the artifact delivered to every judge in the experiments reported in Chapter 6.

A.1 System Message

The following text is delivered as the system role (or its provider-specific equivalent) on every episode evaluation.

```
You are an expert evaluator of an agentic stock prediction system that combines an
autoencoder regime detector, a routing layer that selects between a normal pathway
and an event pathway, dual node-transformer prediction pathways, and a Soft
Actor-Critic (SAC) controller that adjusts the regime threshold tau and the blending
weight alpha at each trading day.

For each episode you receive five consecutive daily behavioral traces. Each trace
contains:
  (i)   market context: closing price, volume, VIX, aggregated BERT sentiment;
  (ii)  autoencoder reconstruction error e, current threshold tau, and regime label
        (0 = normal, 1 = anomalous);
  (iii) routing blending weight alpha and dominant pathway index;
  (iv)  SAC adjustments delta_tau and delta_alpha, each in [-0.1, 0.1];
  (v)   the one-day-ahead prediction;
  (vi)  trailing 20-day MAPE and directional accuracy.

Interpretation rules. A regime label of 1 is appropriate when the reconstruction
error exceeds the threshold by a non-trivial margin and the surrounding market
context (VIX above 25, sharply negative sentiment, abrupt price moves) supports the
```

```
anomaly classification. A blending weight alpha closer to 1 directs prediction
weight to the normal pathway; alpha closer to 0 directs it to the event pathway.
SAC adjustments are applied additively at the next time step. Adaptive behavior
should be timely (within one to two days of a detectable change), proportional to
the magnitude of that change, and stable (no day-to-day reversals).

Score each episode along the six dimensions (RD, RT, AD, RC, SC, ER) on an integer
1-5 scale:
  1 = fundamentally flawed
  2 = predominantly flawed with occasional acceptable behavior
  3 = acceptable with identifiable weaknesses
  4 = strong with minor imperfections
  5 = exemplary

The score-1 and score-5 anchors for each dimension are:
  RD  1: systematic misclassification of regime; threshold unresponsive to
         volatility.
      5: all regime transitions identified within one trading day; threshold
         adjustments proportional to volatility.
  RT  1: data routed to wrong pathway; blending weight contradicts regime
         classification.
      5: routing consistently matches market state; smooth blending transitions
         near threshold.
  AD  1: parameters frozen or wildly oscillating; no response to condition changes
         within episode.
      5: prompt, proportional adjustments within one to two days; no overshoot or
         oscillation.
  RC  1: risk posture inappropriate for volatility (for example, aggressive in
         high-VIX period).
      5: conservative during high volatility, confident during low volatility;
         smooth transitions.
  SC  1: multiple contradictory actions across the decision chain within episode.
      5: all decisions form a logically consistent sequence; no contradictions.
  ER  1: no corrective action within two days of an error; errors persist or
         worsen.
      5: corrective adjustments within one to two days; subsequent predictions show
         measurable improvement.
Intermediate levels interpolate between the anchors using the per-dimension rubric
descriptions provided to the judge.

Reason in four steps before scoring:
  Step 1. Summarize market conditions across the five-day window, noting any
          volatility, sentiment, or price transitions.
  Step 2. Examine each decision point and assess whether the decision was
          appropriate given the information available at that time.
  Step 3. Identify any inconsistencies, failures, or suboptimal behaviors. Cite the
          specific trace fields that support each observation.
  Step 4. Assign each of the six scores. Each score must be grounded in specific
          observations from Steps 2 and 3, not an overall impression.

Return a single JSON object in the format specified in the output schema. Do not
include the chain-of-thought reasoning inside the JSON.
```

A.2 Behavioral-Trace Serialization

Each daily trace $\mathcal{B}_t$ defined in Equation 6.1 is serialized as a JSON record; the five traces of an episode are presented in chronological order. A representative single-day record drawn from a high-volatility episode is shown below.


```
{
  "day_index": 3,
  "market_context": {
    "price": 142.31, "volume": 38421500, "VIX": 22.4, "sentiment_bar": -0.18
  },
  "autoencoder": {
    "reconstruction_error_e": 0.034, "threshold_tau": 0.031,
    "ratio_e_over_tau": 1.10, "regime_label": 1
  },
  "routing": {
    "alpha": 0.35, "normal_pathway_pct": 35, "event_pathway_pct": 65,
    "dominant_pathway": "event"
  },
  "sac_action": {
    "delta_tau": -0.002, "delta_alpha": -0.05,
    "tau_after_update": 0.029, "alpha_after_update": 0.30
  },
  "prediction": {"y_hat_next_day": 142.05},
  "rolling_performance": {
    "MAPE_20d_pct": 0.58, "DA_20d_pct": 73, "trend": "stable"
  }
}
```


A.3 Output Schema and Failure-Label Vocabulary

The judge returns a single JSON object containing one integer score per dimension (1 to 5), one natural-language justification per dimension referencing specific trace fields, and an optional categorical failure label for each dimension scoring strictly below the threshold $\theta = 3$.

```
{
  "scores": {
    "RD": <int>, "RT": <int>, "AD": <int>, "RC": <int>, "SC": <int>, "ER": <int>
```

```
  },
  "justifications": {
    "RD": "<text referencing trace fields>", "RT": "<text>", "AD": "<text>",
    "RC": "<text>", "SC": "<text>", "ER": "<text>"
  },
  "failures": [
    {"dimension": "<RD|RT|AD|RC|SC|ER>", "label": "<label-from-vocabulary>"}
  ]
}
```

The 12 failure labels and their default target SAC action subspaces are listed in Table A.1. A failure label is recorded only when the corresponding dimension score is strictly below $\theta = 3$, and the label determines which SAC action component receives the penalty under the credit-assignment routing of Table 6.4.

Table A.1: Failure-label vocabulary and default SAC action-subspace mapping. The default mapping follows the dimension-to-subspace mapping of Table 6.4; per Section 6.2.7, a label may override this default in atypical failure modes.

Label	**Dimension**	**Subspace**
`delayed_threshold`	RD	$\Delta\tau$
`systematic_misclassification`	RD	$\Delta\tau$
`oversensitive_threshold`	RD	$\Delta\tau$
`wrong_routing`	RT	$\Delta\alpha$
`inconsistent_blend`	RT	$\Delta\alpha$
`abrupt_blending`	RT	$\Delta\alpha$
`frozen_parameters`	AD	$\Delta\tau$, $\Delta\alpha$
`oscillating_actions`	AD	$\Delta\tau$, $\Delta\alpha$
`uncalibrated_risk`	RC	$\Delta\tau$
`contradictory_decisions`	SC	$\Delta\tau$, $\Delta\alpha$
`delayed_recovery`	ER	$\Delta\tau$, $\Delta\alpha$
`error_amplification`	ER	$\Delta\tau$, $\Delta\alpha$

Appendix B

Supplementary Material for the Literature Review

This appendix collects supporting material for the literature review in Chapter 2. It gathers the comparison tables that summarize the surveyed studies and the standard model formulations referenced throughout the chapter, keeping the main text focused on the narrative synthesis while preserving the tabular summaries and formal details for reference.

B.1 Standard Model Formulations

This section reproduces the standard model formulations referenced in Chapter 2. These are established results from the cited literature, collected here so that the review in the main text can proceed at the level of concepts and comparisons while keeping the formal specifications available for reference.

B.1.1 Statistical Time Series Models

The ARIMA(p, d, q) model [26] is specified as

$$\phi(B)(1 - B)^d X_t = \theta(B)\epsilon_t \tag{B.1}$$

where B is the backshift operator, $\phi(B) = 1-\phi_1 B-\cdots-\phi_p B^p$ is the autoregressive polynomial, $\theta(B) = 1 + \theta_1 B + \cdots + \theta_q B^q$ is the moving average polynomial, d is the differencing order required to achieve stationarity, and $\epsilon_t \sim \mathcal{N}(0, \sigma^2)$ is white noise.

Vector Autoregression [29] extends this framework to a K-dimensional series $\mathbf{y}_t$ through the VAR(p) model

$$\mathbf{y}_t = \mathbf{c} + \sum_{i=1}^{p} \mathbf{A}_i \mathbf{y}_{t-i} + \mathbf{u}_t \tag{B.2}$$

where $\mathbf{A}_i$ are $K \times K$ coefficient matrices, $\mathbf{c}$ is a vector of intercepts, and $\mathbf{u}_t$ is a multivariate white noise process.

The GARCH(p, q) model [33] specifies the conditional variance as

$$\sigma_t^2 = \omega + \sum_{i=1}^{q} \alpha_i \epsilon_{t-i}^2 + \sum_{j=1}^{p} \beta_j \sigma_{t-j}^2 \tag{B.3}$$

where σ_t^2 is the conditional variance, ϵ_t is the return residual, and ω, α_i, β_j are parameters with $\omega > 0$, $\alpha_i \geq 0$, $\beta_j \geq 0$.

B.1.2 Recurrent Architectures

The Long Short-Term Memory (LSTM) cell [41] maintains a cell state $\mathbf{c}_t$ alongside the hidden state $\mathbf{h}_t$, with three gates governing what information is stored, forgotten, or output:

$$\mathbf{f}_t = \sigma(\mathbf{W}_f [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_f) \tag{B.4}$$

$$\mathbf{i}_t = \sigma(\mathbf{W}_i [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_i) \tag{B.5}$$

$$\tilde{\mathbf{c}}_t = \tanh(\mathbf{W}_c [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_c) \tag{B.6}$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t \tag{B.7}$$

$$\mathbf{o}_t = \sigma(\mathbf{W}_o [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_o) \tag{B.8}$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t) \tag{B.9}$$

where $\mathbf{f}_t$, $\mathbf{i}_t$, and $\mathbf{o}_t$ are the forget, input, and output gates respectively, $\odot$ denotes element-wise multiplication, and σ is the sigmoid function.

The Gated Recurrent Unit (GRU) [45] simplifies the LSTM through the update equations

$$\mathbf{z}_t = \sigma(\mathbf{W}_z[\mathbf{h}_{t-1}, \mathbf{x}_t]) \tag{B.10}$$

$$\mathbf{r}_t = \sigma(\mathbf{W}_r[\mathbf{h}_{t-1}, \mathbf{x}_t]) \tag{B.11}$$

$$\tilde{\mathbf{h}}_t = \tanh(\mathbf{W}[\mathbf{r}_t \odot \mathbf{h}_{t-1}, \mathbf{x}_t]) \tag{B.12}$$

$$\mathbf{h}_t = (1 - \mathbf{z}_t) \odot \mathbf{h}_{t-1} + \mathbf{z}_t \odot \tilde{\mathbf{h}}_t \tag{B.13}$$

where $\mathbf{z}_t$ is the update gate and $\mathbf{r}_t$ is the reset gate, giving fewer parameters than the LSTM due to the reduced gate count.

B.1.3 Transformer Architectures

The scaled dot-product self-attention operation [3] is

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V} \tag{B.14}$$

where $\mathbf{Q} \in \mathbb{R}^{n \times d_k}$, $\mathbf{K} \in \mathbb{R}^{n \times d_k}$, and $\mathbf{V} \in \mathbb{R}^{n \times d_v}$ are query, key, and value matrices obtained through linear projections of the input, n is the sequence length, and d_k is the key dimension. Multi-head attention runs h parallel attention functions,

$$\text{MultiHead}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)\mathbf{W}^O \tag{B.15}$$

where $\text{head}_i = \text{Attention}(\mathbf{Q}\mathbf{W}_i^Q, \mathbf{K}\mathbf{W}_i^K, \mathbf{V}\mathbf{W}_i^V)$ and $\mathbf{W}^O$ is a learned output projection. Positional information is injected through the fixed sinusoidal encoding

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right) \tag{B.16}$$

$$PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right) \tag{B.17}$$

where *pos* is the position and i is the dimension index. Each encoder layer also applies a position-wise feed-forward network

$$\text{FFN}(\mathbf{x}) = \text{ReLU}(\mathbf{x}\mathbf{W}_1 + \mathbf{b}_1)\mathbf{W}_2 + \mathbf{b}_2 \tag{B.18}$$

with residual connections and layer normalization surrounding each sublayer.

B.1.4 Graph Neural Networks

The message-passing framework [54] updates each node representation as

$$\mathbf{h}_v^{(l+1)} = \text{UPDATE}\left(\mathbf{h}_v^{(l)}, \text{AGGREGATE}\left(\left\{\mathbf{h}_u^{(l)} : u \in \mathcal{N}(v)\right\}\right)\right) \tag{B.19}$$

where $\mathbf{h}_v^{(l)}$ is the representation of node v at layer l, $\mathcal{N}(v)$ is the neighborhood of v, and UPDATE and AGGREGATE are learnable functions. The Graph Convolutional Network [55] instantiates this with the propagation rule

$$\mathbf{H}^{(l+1)} = \sigma\left(\hat{\mathbf{D}}^{-1/2}\hat{\mathbf{A}}\hat{\mathbf{D}}^{-1/2}\mathbf{H}^{(l)}\mathbf{W}^{(l)}\right) \tag{B.20}$$

where $\hat{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ is the adjacency matrix with self-loops, $\hat{\mathbf{D}}$ is the corresponding degree matrix, $\mathbf{W}^{(l)}$ is a learnable weight matrix, and σ is a nonlinear activation. The Graph Attention Network [56] replaces the fixed normalization with learned attention coefficients

$$\alpha_{ij} = \frac{\exp\left(\text{LeakyReLU}\left(\mathbf{a}^T[\mathbf{W}\mathbf{h}_i \| \mathbf{W}\mathbf{h}_j]\right)\right)}{\sum_{k \in \mathcal{N}(i)} \exp\left(\text{LeakyReLU}\left(\mathbf{a}^T[\mathbf{W}\mathbf{h}_i \| \mathbf{W}\mathbf{h}_k]\right)\right)} \tag{B.21}$$

where $\mathbf{a}$ is a learnable attention vector, $\mathbf{W}$ is a shared weight matrix, and $\|$ denotes concatenation.

B.1.5 Soft Actor-Critic

The maximum-entropy objective of Soft Actor-Critic [83] is

$$J(\pi) = \sum_{t=0}^{T} \mathbb{E}_{(\mathbf{s}_t, \mathbf{a}_t) \sim \rho_\pi} \left[r(\mathbf{s}_t, \mathbf{a}_t) + \alpha \mathcal{H}(\pi(\cdot|\mathbf{s}_t)) \right] \tag{B.22}$$

where ρ_π is the state-action marginal distribution induced by the policy, r is the reward function, $\mathcal{H}$ is the Shannon entropy, and α is a temperature parameter controlling the trade-off between reward maximization and entropy maximization.

B.2 Literature Survey Tables

The tables in this section summarize representative studies from the areas reviewed in Chapter 2. Performance figures are drawn from the cited works and are not directly comparable across studies, because datasets, prediction horizons, and evaluation protocols differ.

Table B.1: Comparison of classical machine learning methods for stock price prediction. Performance metrics vary across studies due to differences in datasets, prediction horizons, and evaluation protocols.

Method	**Study**	**Market**	**Metric**	**Key Result**
SVM-KNN Hybrid	Nayak et al. (2015)	Indian indices	MSE	Hybrid outperforms individual SVM, KNN
KPCA-SVR-TLBO	Siddique & Panda (2019)	Tata Steel	MSE	Dim. reduction improves SVR
XGBoost	Basak et al. (2019)	AAPL, YHOO	F-score	0.82–0.97 at 60–90 day
Random Forest	Milosevic	US stocks	F-score	0.751 using 11 fundamental ratios
Random Forest	Ballings et al.	Multiple	AUC	Top among 7 methods
BNN	Malagrino et al.	iBOVESPA	DA	71% mean, 78% peak

Table B.2: Comparison of deep learning architectures for stock price prediction. The table summarizes representative studies, not exhaustive performance comparisons, as dataset and evaluation differences preclude direct cross-study comparison.

Architecture	**Study**	**Dataset**	**Key Metric**	**Result**
LSTM	Di Persio & Honchar (2017)	Google	DA: 72%	Best among RNN, LSTM, GRU
LSTM	Fischer & Krauss (2018)	S&P 500	Returns	Stat. sig. over 25 years
LSTM-GRU	Hossain et al. (2018)	S&P 500 (66yr)	MSE: 0.00098	vs 0.003 for individual models
ESN	Bernal et al.	S&P 500 (50 stocks)	Test err: 0.0027	Competitive on 50 stocks
CNN (TI-CNN)	Chandar (2022)	Multiple	Accuracy	GAF image encoding helps
BiLSTM + AE	Liu et al. (2022)	Multiple	MAPE	Learned features improve pred.
CNN-BiLSTM-AM	Lu et al. (2021)	Multiple	RMSE	Outperforms 7 baselines

Table B.3: Comparison of graph neural network methods for stock market prediction. Graph construction strategies and the type of relational information captured vary substantially across approaches.

Method	**Graph Type**	**Edge Definition**	**Market**	**Key Result**
GC-CNN	Static	Sector, correlation	Chinese (6)	Superior to non-graph CNN
GNN + Attn	Multi-source	Edge weight + GRU/LSTM	Multiple	+16.64% vs dim. reduction
MG-Conv	Static + Dynamic	Sector + rolling corr.	Chinese (42)	−5.11% error vs LSTM
TRR	Dynamic	Learned from returns	Multiple	Dynamic ranking
DGDNN	Dynamic	Entropy-driven	Multiple	Over static baselines
SH-Conv	Hypergraph	Higher-order groups	US equities	Over pairwise GNN
NodeFormer	Adaptive	Learned attention	General	Linear-complexity

Table B.4: Sentiment analysis approaches used in financial forecasting. The evolution from lexicon-based to transformer-based methods reflects gains in contextual understanding, while fusion strategies determine how effectively sentiment information is integrated with quantitative features.

Approach	**Study**	**Text Source**	**Fusion**	**Key Result**
Lexicon (LM)	Loughran & McDonald (2011)	10-K filings	N/A	Finance-specific lists outperform generic
Twitter mood	Bollen et al. (2011)	Twitter	SOFNN	87.6% DJIA direction accuracy
Event tuples	Ding et al. (2015)	News	Embedding	∼6% over baselines on S&P 500
Topic sentiment	Si et al. (2013)	Twitter	Topic model	Topic-level > aggregate sent.
AZFin	Schumaker & Chen (2009)	News	Hybrid	Text + price > either alone
StockNet	Xu & Cohen (2018)	Twitter + price	VAE	Joint model on 88 stocks
FinBERT	Araci (2019)	Financial text	Fine-tuned	SOTA on Financial PhraseBank