Hybrid Deep Learning Model: LLM-Enhanced Linear Self-Attention Transformer-CNN for Stock Price Prediction

A THESIS SUBMITTED TO
AUCKLAND UNIVERSITY OF TECHNOLOGY
IN PARTIAL FULFILMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF COMPUTER AND INFORMATION
SCIENCES

Supervisors
Dr. Jian Yu
Dr. Sam Madanian

October 2025

By
Yilong Chen
School of Computer and Mathematical Sciences

Acknowledgement

This research paper is the summit of my master's degree career at the School of Information Systems at Auckland University of Technology. Dr. Jian Yu and Sam Madanian, for their excellent advice and unconditional support throughout the research process. Their insight, constructive comments, and active participation were invaluable to this study's design and my academic development. Our weekly meetings over the past 12 months have been invaluable for in-depth discussions and strategic planning. This collaboration has not only helped me to develop my research methods further but has also encouraged me to pursue innovative research approaches. The expertise and advice of my two professors have significantly impacted my academic development and research skills. I cannot thank them enough for their patience and dedication, especially for their detailed comments on my research, challenging my ideas, and encouraging me to maintain high academic standards. Your guidance was crucial in overcoming research difficulties and completing this thesis on time.

Yilong Chen
October 2025

Attestation of Authorship

I solemnly affirm this paper is independent scholarly work from myself, excluding officially cited references and acknowledgments. I further certify that this manuscript has not been previously submitted for academic honors, in whole or in part, at this or any different institution of education higher.
As far as I know, this paper meets all standards of academic integrity and makes an original contribution to the field.

Signature of the Candidate:
Yilong Chen
October 2025

Copyright

Contents

Acknowledgement **2**

Attestation of Authorship **3**

Copyright **4**

List of Figures **10**

List of Tables **17**

Abbreviations **19**

Abstract **21**

1 Introduction **23**
1.1 Background and Motivation 23
1.1.1 Need for Deep Learning-Based Methods 24
1.1.2 Background on the Models Used in This Work 25
1.1.3 Large Language Models (LLMs) in Financial Analysis . 25
1.1.4 Convolutional Neural Networks (CNNs) for Financial Data Representation 26
1.1.5 Transformer-Based Architectures for Temporal Dependencies 27
1.1.6 Multi-Modal Integration for Enhanced Stock Price Prediction 27
1.1.7 Summary of Background and Motivation 28
1.2 Research Questions 29
1.2.1 Identified Gaps in Current Research 31
1.2.2 Our Contributions 31

2 Literature Review **34**
2.1 Exploration of Machine Learning Approaches in Financial Analysis . . . 35
2.1.1 Early Research on Financial Text Analysis . . . 38
2.1.2 Linear Methods . . . 41
2.1.3 DLinear: Revisiting Simplicity in Time Series Forecasting . . . 45
2.1.4 Bridging Traditional and Modern Approaches . . . 49
2.2 Deep Learning Approaches for Stock Prediction . . . 50
2.2.1 RNN and LSTM Models . . . 52
2.2.2 CNN Models in Stock Price Prediction . . . 54
2.2.3 TimesNet: Multi-Scale Feature Extraction for Time Series Data . . . 55
2.2.4 MICN: Multi-Scale Isometric Convolutional Networks . 56
2.2.5 Nonstationarities in Stock Returns for Financial Time Series Prediction . . . 61
2.2.6 FiLM: Frequency Domain Polynomials: Enhanced Legendre Memory Architecture for Extended Financial Series Prediction . . . 62
2.3 Transformer-Based Approaches for Stock Prediction . . . 63
2.3.1 Autoformer: A Decomposition-Based Transformer for Long-Term Forecasting . . . 66
2.3.2 Crossformer: A Transformer for Capturing Cross-Dimensional Dependencies in Multivariate Time Series . . . 70
2.3.3 PatchTST: Channel-Independent Transformer for Financial Time Series Prediction . . . 72
2.3.4 Informer: Enhancing Long-range Dependency Modeling for Stock Price Prediction . . . 73
2.3.5 Linear Self-Attention Transformers in Financial Applications . . . 75
2.4 Large Language Models Approaches for Stock Prediction . . . 78
2.5 Hybrid Models Approaches for Stock Prediction . . . 82
2.5.1 Hybrid CNN Models Approaches . . . 82
2.5.2 CNN-LSTM Integration . . . 83
2.5.3 CNN-Transformer Integration . . . 83
2.6 Summary of Literature Review Findings . . . 86

3 Methodology **87**
3.1 Introduction . . . 87
3.1.1 Research Methodology Framework . . . 90
3.1.2 Methodological Approach . . . 93
3.1.3 Experimental Scope . . . 94
3.1.4 Baseline Model Selection and Justification . . . 94
3.2 Data Collection and Processing Framework . . . 97
3.2.1 Data Sources and Acquisition . . . 97
3.2.2 Value Range Adjustment . . . 100
3.2.3 FinBERT Implementation for Text Embedding . . . 101
3.2.4 Dataset Structure . . . 103
3.2.5 Historical Market Data Collection . . . 104
3.2.6 Market Signal Parameter Construction . . . 105
3.2.7 Information Quality Enhancement . . . 105
3.3 LLM-Based Feature Extraction for Stock Prediction . . . 105
3.3.1 Technical Analysis with LLMs . . . 105
3.3.2 Financial Indicator-Based Text Generation . . . 106
3.3.3 Data Preprocessing and Integration . . . 106
3.3.4 API Deployment and Model Efficiency . . . 108
3.3.5 Summary of LLM-Based Feature Generation Framework 109
3.4 Deep Learning Architecture Design . . . 109
3.4.1 CNN Architecture . . . 109
3.4.2 Linear Transformer Architecture . . . 115
3.4.3 Multi-Modal Fusion Architecture . . . 119
3.5 Model Training . . . 122
3.5.1 Training Data Organization . . . 122
3.5.2 Hyperparameter tuning . . . 122
3.5.3 Performance Evaluation and Results . . . 124
3.6 Experimental Setup and Validation . . . 129
3.6.1 Software Dependencies . . . 129
3.6.2 Validation Protocols . . . 130

4 Experiments and Results **131**
4.1 Experimental Environment . . . 131
4.2 Evaluation Metrics Results . . . 131
4.3 History Length and Step Length Analysis . . . 141
4.3.1 Parameter Selection Methodology . . . 141
4.3.2 Impact on Model Performance and Market Alignment . 145

4.4 Ablation Study . . . 145
4.4.1 Experimental Setup and Methodology . . . 146
4.4.2 Overall Ablation Results . . . 147
4.4.3 Component-Specific Analysis . . . 147
4.4.4 Computational Efficiency Analysis . . . 160
4.4.5 Feature Fusion Architecture . . . 164
4.4.6 Loss Component Analysis . . . 165
4.4.7 Loss Weight Tuning Analysis . . . 165
4.4.8 Statistical Significance Analysis . . . 169
4.4.9 Synthesis of Component-Level Insights . . . 170
4.4.10 Key Takeaways from Ablation Experiments . . . 170
4.5 Hyperparameter Analysis . . . 171
4.5.1 Learning Rate Impact . . . 172
4.5.2 Batch Size Analysis . . . 175
4.5.3 Combined Hyperparameter Impact . . . 177
4.5.4 Key Insights from Hyperparameter Analysis . . . 178

5 Discussion and Conclusion 180
5.1 Key Findings in Relation to Research Questions . . . 180
5.1.1 Addressing Research Question 1: Transformer Models for Stock Market Forecasting . . . 180
5.1.2 Addressing Research Question 2: LLMs for Technical Analysis . . . 181
5.1.3 Addressing Research Question 3: CNN-Transformer Integration . . . 182
5.2 Interpretation of Results . . . 183
5.3 Comparison with Existing Studies . . . 184
5.4 Implications for Stock Price Prediction . . . 186
5.5 Limitations of the Study and Future Directions . . . 187
5.5.1 Market Generalizability . . . 187
5.5.2 Computational Complexity and Real-Time Feasibility . 187
5.5.3 Interpretability and Explainability . . . 187
5.5.4 Handling Market Regimes and Extreme Events . . . 188

6 Conclusion 189
6.1 Summary of Contributions . . . 189
6.2 Future Work . . . 190
6.3 Closing Remarks . . . 191

References **191**

List of Figures

2.1 Machine Learning Approaches in Stock Prediction 37
2.2 Early research on financial text analysis 38
2.3 Linear Methods Overview 43
2.4 DLinear architecture: (a) decomposition phase that splits the input time series into trend (X_t) and remainder (X_s) components (blue block); (b) application of distinct linear transformation layers (W_t and W_s) to each component (red block), with results summed to generate the final forecast output. Adapted from Zeng et al. (2023). 46
2.5 The overall architecture of TimesNet. The model employs **TimesBlock** to convert one-dimensional time series into structured two-dimensional tensors, enabling effective modeling of temporal dependencies across multiple scales. The figure is adapted from Wu et al. (2022). 57
2.6 Overall architecture of MICN. The model integrates a multi-scale hybrid decomposition, seasonal prediction block, and trend-cyclical prediction block to effectively capture both local and global dependencies in time series forecasting. Adapted from Wang et al. (2023b). 59
2.7 Seasonal prediction block of MICN. The module leverages multi-scale isometric convolution (MIC) for efficient extraction of local and global dependencies. Adapted from Wang et al. (2023b). 60
2.8 Transformers Model Architecture. Adapted from Vaswani (2017) 65
2.9 Time Series Forecasting Models framework 67
2.10 Autoformer overall architecture. Adapted from Wu et al. (2021). 69

2.11 Architecture of the Hierarchical Encoder-Decoder in **Crossformer**. Adapted from Zhang and Yan (2023). The encoder processes input time series through dimension segments and employs **Two-Stage Attention (TSA)** layers with segment merging to create multi-scale representations, where higher layers capture longer time ranges with increasingly coarse-grained information. The decoder integrates both fine-grained and coarse-grained representations from different encoder layers to generate the final forecast, effectively capturing both temporal and cross-dimensional dependencies in multivariate time series data. 71
2.12 Large Language Models in Financial Analysis 79
2.13 CNN-Transformer Integration 84

3.1 Architecture of the proposed LLM-enhanced linear self-attention transformer-CNN hybrid framework for stock price prediction. 89
3.2 Experimental Pipeline for LLM-Enhanced Hybrid Deep Learning Framework in Stock Price Prediction 91
3.3 Comprehensive Research Methodology Framework for Stock Market Prediction 92
3.4 S&P 500 Index Closing Price Trend (2022-2023) 99
3.5 An example of LLM-augmented technical analysis framework for classifying financial sentiment. It demonstrates how stock news content, task prompts, and potential outcomes integrate to provide actionable insights. 107
3.6 Workflow for LLM-based stock prediction feature extraction . 108
3.7 CNN Architecture for Stock Price Prediction. The model processes multi-format visual representations through a series of convolutional and residual blocks before making predictions through dense layers. 110
3.8 Different visual representations of stock market data. Each visualization captures unique aspects of price movements and volume patterns, providing diverse features for the CNN model to learn from. (a-c) show different market trends in line format, (d-f) demonstrate various line-based visualizations, and (g-i) present candlestick patterns. 116

3.9 Comprehensive market analysis visualization displaying: (a) Price action represented through candlesticks (upward movements in blue, downward shifts in red) alongside volatility boundaries via Bollinger Bands and periods moving average; (b) Trading volume represented as vertical green histogram bars indicating market participation levels; and (c) Momentum evaluation through an RSI indicator (pink graph) featuring upper resistance (dashed red line at 70) and lower support (dashed green line at 30) thresholds. 117
3.10 OPTUNA Framework Architecture 123
3.11 OPTUNA Tuning Process 125
3.12 Dataset partitioning strategy showing the distribution of data across training (70%, red), validation (10%, blue), and testing (20%, green) sets. This balanced approach ensures sufficient data for model training while maintaining an adequate holdout set for final evaluation. 126

4.1 Comparison of grouped model architectures on S&P 500 index prediction 133
4.2 Comprehensive radar chart visualization of model performance across four key metrics: RMSE, MAPE, MAE, and Inference Speed. Our model (highlighted in pink) consistently outperforms all benchmark models, demonstrating superior performance across all evaluation dimensions including both predictive accuracy and computational efficiency. 135
4.3 RMSE Performance Comparison 136
4.4 MAPE Performance Comparison 137
4.5 MAE Performance Comparison 138
4.6 Impact of History Length on Model Performance (Step Length = 5 days). The chart demonstrates that a 30-day history length provides optimal performance with an RMSE of 109.49, with performance degrading for both shorter periods (insufficient market context) and longer periods (increased noise). . . 142

4.7 Impact of Step Length on Model Performance (History Length = 30 days). While shorter prediction windows (1-3 days) achieve better raw accuracy, the 5-day length represents an optimal balance between mathematical precision and practical utility. Performance degrades rapidly for longer prediction horizons due to increasing market uncertainty. 143
4.8 Balancing Accuracy and Practicality Across Parameter Combinations. This comprehensive view validates the selection of the 30-day history and 5-day step length configuration as providing the optimal balance between accuracy and utility for S&P 500 index prediction. 144
4.9 Ablation study on the S&P 500 dataset showing RMSE performance. The chart illustrates performance degradation when individual components are removed from the full model, with the removal of LLM causing the most significant impact, followed by Transformer, CNN, and FinBERT components. . . . 148
4.10 Ablation study on the S&P 500 dataset showing MAPE performance. The percentage error metric reveals similar patterns, with the most substantial performance drop observed when removing the LLM component, highlighting its critical role in accurate percentage-based predictions. 149
4.11 Ablation study on the S&P 500 dataset showing MAE performance. The chart demonstrates how removing individual components affects the mean absolute error, further validating the importance of each architectural element, particularly the LLM component. 150
4.12 Performance impact of LLM removal across all evaluation metrics. The removal of LLM-generated textual indicators leads to the most significant performance degradation, with RMSE increasing to 120.92 (+10.4%), MAPE to 0.0240 (+17.1%), and MAE to 94.17 (+11.6%). This substantial impact underscores the crucial importance of incorporating LLM-derived market narratives, which provide rich contextual information about market trends, sentiment, and technical indicators that traditional numerical features alone cannot capture. 151

4.13 Performance impact of Transformer removal across all evaluation metrics. The absence of the Transformer component results in the second-largest performance decline with RMSE increasing to 118.73 (+8.4%), MAPE to 0.0231 (+12.7%), and MAE to 92.48 (+9.6%). This highlights the critical role of attention mechanisms in capturing long-range temporal dependencies in financial time series data, particularly for identifying complex patterns that span multiple trading periods. . . . 153
4.14 Performance impact of CNN removal across all evaluation metrics. Removing the CNN module, which is responsible for extracting spatial patterns from price movements, increases RMSE from 109.49 to 116.85 (+6.7%), MAE from 84.39 to 90.72 (+7.5%), and MAPE from 0.0205 to 0.0223 (+8.8%). This demonstrates the importance of spatial feature extraction in capturing local price movement patterns and chart formations that traders traditionally use in technical analysis. . . 154
4.15 Performance impact of FinBERT removal across all evaluation metrics. While the impact is less pronounced than other components, FinBERT removal still leads to measurable degradation across all metrics, with RMSE increasing from 109.49 to 112.34 (+2.6%), MAE from 84.39 to 86.45 (+2.4%), and MAPE from 0.0205 to 0.0215 (+4.8%). The largest relative impact on MAPE suggests FinBERT's particular value in percentage-based forecasting accuracy. 156
4.16 Comparison of RMSE values before and after FinBERT removal. Removing FinBERT increases RMSE from 109.49 to 112.34, indicating a decline in forecasting precision. 157
4.17 MAPE variation with and without FinBERT integration. A rise from 0.0205 to 0.0215 suggests reduced relative accuracy in percentage-based error terms. 158
4.18 MAE comparison illustrating FinBERT's contribution to absolute error reduction. A noticeable increase to 86.45 without FinBERT indicates its semantic value in capturing financial sentiment. 159

4.19 Computational requirements (FLOPs) comparison across model variants. The full model requires approximately 2.4×10^{15} FLOPs per training cycle, while removing the LLM component results in the largest reduction (45.8%) to 1.3×10^{15} FLOPs. Removing the Transformer, CNN, and FinBERT components results in 33.3%, 25.0%, and 16.7% reductions in computational demands, respectively. 161
4.20 GPU memory usage comparison across model variants on the NVIDIA GeForce RTX 4070 Ti SUPER. The full model consumes 12.3GB of GPU memory, while the variant without LLM requires only 7.8GB (36.6% reduction). Memory requirements for variants without Transformer (9.2GB), CNN (9.7GB), and FinBERT (10.6GB) demonstrate the relative memory footprint of each architectural component. 162
4.21 Training time comparison across model variants on the NVIDIA GeForce RTX 4070 Ti SUPER GPU. The complete model requires 8.7 hours for training, while removing the LLM component reduces training time to 5.1 hours (41.4% reduction). Removing the Transformer component saves 32.2% of training time, CNN removal saves 21.8%, and FinBERT removal results in an 18.4% reduction in training time. 163
4.22 Impact of learning rate on RMSE performance. A learning rate of 0.001 achieves the best prediction accuracy with an RMSE of 109.49, while excessively high learning rates (0.005) lead to significantly worse performance with an RMSE of 115.67, likely due to overshooting optimal parameter values during training. 172
4.23 Impact of learning rate on convergence speed. Higher learning rates accelerate convergence, with 0.005 requiring only 52 epochs compared to 145 epochs for 0.0001. However, this trade-off between speed and final performance must be carefully balanced, as the fastest convergence does not yield the best model. 173

4.24 Impact of learning rate on final loss values. The optimal learning rate of 0.001 achieves the lowest final loss (0.00185), while both excessively low and high rates result in suboptimal loss values. This U-shaped relationship demonstrates the importance of finding the appropriate balance between exploration and exploitation during optimization. 174
4.25 Impact of batch size on model performance. The RMSE follows a U-shaped curve with the optimal performance achieved at a batch size of 64 (RMSE: 109.49). Both smaller batches (16, 32) and larger batches (128, 256) lead to decreased performance, with the degradation more pronounced at the extremes. 176
4.26 Comparative analysis of hyperparameter configurations on RMSE. The optimal configuration (learning rate: 0.001, batch size: 64) achieves an RMSE of 109.49, while misconfigured hyperparameters lead to performance degradation of up to 5.6% (learning rate too high). Learning rate misconfiguration generally has a more severe impact than batch size misconfiguration. 177

List of Tables

1.1 Components of Financial Market Study 24

3.1 Overview of S&P 500 Dataset (2022-2023) 98
3.2 Global Market Indices Available for Analysis 99
3.3 Raw S&P 500 Market Data with Extended Indicators (2022-2023) 100
3.4 Data Preprocessing Steps and Transformations 103
3.5 Statistical Summary of Processed S&P 500 Data (2022-2023) . 104
3.6 Processed S&P 500 Dataset with Standardized Format 104
3.7 Structured API Request Format for LLM Technical Analysis . 109
3.8 Hyperparameter Search Space Configuration 124
3.9 Model Training Configuration Parameters 128
3.10 Model Performance Metrics on Test Set 128

4.1 Experimental Environment Configuration 131
4.2 Quantitative comparison of our framework (CNN+Transformer-+FinBERT+LLM) with 14 benchmark models on the S&P500 dataset. Metrics reported: RMSE, MAPE, and MAE. 134
4.3 Computational Efficiency Analysis of Deep Learning Models for Stock Price Prediction 139
4.4 Training Environment Configuration 139
4.5 Performance comparison across different history and step length configurations 146
4.6 Ablation Study Results on the S&P 500 Dataset 147
4.7 Ablation Results: Removing LLM Component 148
4.8 Ablation Results: Removing Transformer Component 152
4.9 Ablation Results: Removing CNN Component 153
4.10 Ablation Results: Removing FinBERT Component 155

4.11 FinBERT Feature Processing Statistics 160
4.12 Computational Requirements by Component 160
4.13 Performance Comparison Across Different Loss Weight Configurations 166
4.14 Statistical Significance Tests for Component Contributions . . 169
4.15 Impact of Learning Rate on Model Performance 173
4.16 Impact of Batch Size on Model Performance 175

Abbreviations

CNN	Convolutional Neural Network
GNN	Graph Neural Network
LSTM	Long Short-Term Memory
MAPE	Mean Absolute Percentage Error
NLP	Natural Language Processing
ANFIS	Adaptive Neuro-Fuzzy Inference System
DL	Deep Learning
SVM	Support Vector Machine
RAG	Retrieval-Augmented Generation
FNN	Feedforward Neural Network
BERT	Bidirectional Encoder Representations from Transformers
ML	Machine Learning
IGRU	Improved Gated Recurrent Unit
AI	Artificial Intelligence
ARIMA	Autoregressive Integrated Moving Average
LLM	Large Language Model
SLA	Service Level Agreement
MAE	Mean Absolute Error
GRU	Gated Recurrent Unit
RNN	Recurrent Neural Network
RMSE	Root Mean Square Error
SARIMAX	Seasonal Autoregressive Integrated Moving Average with Exogenous factors
XGBoost	eXtreme Gradient Boosting
GARCH	Generalized Autoregressive Conditional Heteroskedasticity

PatchTST Patch Time Series Converter

Abstract

The unpredictability and inherent complexities of financial markets pose a persistent challenge in accurately forecasting stock prices. Recent innovations in deep learning methodologies, specifically the integration of technical analysis with linguistic processing through large language models (LLMs), have demonstrated significant potential in financial forecasting applications. Despite significant advances in deep learning for stock price prediction, current approaches face critical limitations in effectively integrating multi-modal data while lacking sophisticated technical analysis capabilities, particularly when simultaneously modeling both temporal dependencies in financial time series and spatial features from stock chart visualizations across diverse market regimes and volatility conditions. This study proposes an innovative hybrid deep learning framework aimed at addressing these multi-modal integration challenges, technical analysis limitations, and cross-market adaptability issues through a structured three-phase methodology.

In the initial phase, we leverage a LLM to conduct a thorough technical analysis of historical financial data, transforming raw market information into rich textual representations. These representations encapsulate intricate financial patterns using well-established technical indicators such as Bollinger Bands (BBs). By interpreting these financial metrics, the LLM generates structured textual summaries that enhance the downstream predictive modeling process.

The second phase involves dual processing pipelines. One pipeline utilizes an optimized attention-based Transformer to process the LLM-generated embeddings, capturing long-range market dependencies. Concurrently, the second pipeline deploys a Convolutional Neural Network (CNN) to analyze visual stock chart patterns, extracting spatially relevant features that complement the temporal insights derived from the Transformer model.

In the concluding phase, the outputs from both pipelines—textual, sequential, and visual data—are aggregated to generate holistic stock price predictions. The architecture employs Transformers for extracting extended temporal relationships within market sequential information, whereas Convolutional Networks detect regional spatial configurations from graphical stock representations. Large language models enhance our dataset by generating sophisticated technical metrics, subsequently embedded into numerical

vectors through FinBERT encoding techniques. The fusion of these multi-modal representations enables a more robust and accurate forecasting model, capitalizing on the complementary strengths of each component to enhance predictive performance.

Keywords: CNN; RNN; LSTM; Transformer; Analysis of Stock Market Prediction; Deep Learning; Financial Forecasting.

Chapter 1

Introduction

1.1 Background and Motivation

Due to the non-linear dynamics, inherent volatility, and complex interdependencies of stock data volatility, forecasting stock prices is always challenging for economic markets (Hoseinzade and Haratizadeh, 2019). In a dynamic environment, stock prices are affected by various facets of the stock market, from economic indicators to investor sentiment, which introduces both opportunities and risks for investors seeking profitable returns. Accurate prediction of stock prices is of great interest to various stakeholders, including individual investors, financial institutions, and policymakers, as it enables informed decision-making and risk management. Consequently, researchers have been striving to develop more sophisticated and reliable forecasting models to discover the nuanced patterns of stock price volatility (Weng et al., 2018; Chang et al., 2025).

In financial market studies, two main analytical approaches are often employed: fundamental analysis and technical analysis (Lohrmann and Luukka, 2019). Table 1.1 provides an overview of these two types of analysis.

Fundamental analysis concerns the intrinsic value of a company. It examines earnings, market conditions, and financial statements to forecast the patterns of stock price volatility based on macroeconomic and microeconomic data. While this approach provides a holistic understanding of a company's potential performance, it is generally suitable for long-term investment strategies.

By contrast, technical analysis emphasizes historical price patterns, trad-

Table 1.1: Components of Financial Market Study

Analysis Type	**Description**	**Advantages**
Fundamental	Evaluates a company's future profitability based on its current operating environment and financial performance.	Provides a comprehensive understanding of the company's intrinsic value.
Technical	Involves interpreting charts and utilizes statistical methods to forecast market trends.	Quick to implement; useful for short-term trading.

ing data, and statistical indicators to forecast future price movements. This approach is particularly valuable for short-term trading, where rapid changes in price trends can offer opportunities for profit (Liu and Zhao, 2022). Technical analysis is based on identifying patterns in time-series data that can be processed to forecast stock data movements based on past behavior patterns (Kumbure et al., 2022).

Forecasting stock prices remains a considerable challenge and demands advanced methods capable of handling complex market dynamics and time dependencies. Traditional statistical models (e.g., ARIMA, GARCH) often struggle with the financial markets' volatility and long-range dependencies (Tsay, 2005). Recent advancements in deep learning open new opportunities to overcome these limitations by automatically learning intricate patterns from data. In particular, integrating spatial and temporal analysis techniques through deep learning architectures has shown promise in capturing the multifaceted nature of stock movements (Hoseinzade and Haratizadeh, 2019; Wang et al., 2022). This motivates the exploration of deep learning-based methods for stock prediction.

1.1.1 Need for Deep Learning-Based Methods

Conventional stock prediction models and econometric approaches often fail to capture the highly non-linear and long-term dependencies present in financial time series. For example, autoregressive models like ARIMA or GARCH

are useful for short-term forecasting but cannot easily accommodate sudden regime shifts or complex non-linear interactions in the market (Tsay, 2005). In contrast, machine learning and deep learning models can automatically learn hierarchical patterns from data, making them better suited to model the volatility and complexity of stock prices. Indeed, deep learning approaches have demonstrated superior predictive performance by capturing both short-term fluctuations and long-range patterns that elude traditional methods (Dixon et al., 2020). This has led researchers to investigate architectures such as Convolutional Neural Networks (CNNs) and Transformers, which can process large volumes of financial data while identifying meaningful temporal and spatial features.

1.1.2 Background on the Models Used in This Work

To address the challenges of financial time-series forecasting, this study integrates multiple deep learning architectures, including Large Language Models (LLMs), Convolutional Neural Networks (CNNs), and Transformer-based architectures. Each of these components plays a crucial role in enhancing stock price prediction by leveraging different modalities of data.

1.1.3 Large Language Models (LLMs) in Financial Analysis

Traditional technical indicators rely on manually defined formulas, which may not capture the full complexity of financial trends. By using LLMs, we enable dynamic feature generation that adapts to market conditions, potentially improving predictive accuracy beyond static indicators.

Large Language Models (LLMs) have traditionally been employed in natural language processing (NLP) tasks, such as text classification, sentiment analysis, and machine translation (Devlin et al., 2019; Brown, 2020). In the financial domain, LLMs have been primarily used to analyze textual data from news articles, earnings reports, and financial statements to extract sentiment-driven insights (Araci, 2019). However, relying on sentiment-based forecasting introduces the risk of incorporating misleading or irrelevant information, as financial news and social media data can be noisy and susceptible to misinformation.

In this work, we repurpose LLMs for technical analysis on structured market data instead of external text. To enhance the quality and consistency

of generated indicators, we apply structured *prompt engineering* techniques with DeepSeek, ensuring well-defined financial outputs.

The LLM acts as an intelligent financial analyst by dynamically generating and summarizing technical indicators (e.g., moving averages, RSI, Bollinger Bands) based on structured market data. It adapts its outputs to market conditions, providing context-aware insights for financial forecasting.

These textual summaries are then converted into numerical representations using FinBERT, a financial-domain-specific variant of BERT (Araci, 2019), allowing the extracted knowledge to be incorporated into the deep learning framework. This way, the LLM provides dynamic, explainable technical indicators that adapt to market conditions, potentially improving predictive accuracy without relying on noisy news or social media inputs. Furthermore, these LLM-generated indicators enrich the dataset by providing multi-dimensional financial insights, complementing traditional numerical technical indicators for a more comprehensive predictive framework.

1.1.4 Convolutional Neural Networks (CNNs) for Financial Data Representation

Convolutional Neural Networks excel at pattern recognition and feature extraction from images (LeCun et al., 1998; Krizhevsky et al., 2012). Recent studies show that CNNs can be applied to financial time series by treating historical price charts as images, enabling the model to learn from visual patterns (such as candlestick shapes and indicator curves) (Sezer et al., 2020).

In our framework, CNNs are used to analyze stock chart images that include candlestick patterns and technical indicators such as moving averages and Bollinger Bands. By processing stock charts as visual representations of financial time-series data, CNNs can detect important trends, support and resistance levels, and price breakout patterns that might not be immediately evident in numerical time-series analysis.

By extracting these spatial features, the CNN component contributes insights into short-term market trends and local price patterns.

1.1.5 Transformer-Based Architectures for Temporal Dependencies

Transformer models have emerged as powerful tools for sequence modeling, outperforming traditional RNN-based approaches such as LSTMs in tasks requiring long-term dependency learning (Vaswani, 2017). Unlike RNNs, which process data sequentially, transformers employ self-attention mechanisms that allow the model to capture dependencies across an entire time-series in parallel, significantly improving both efficiency and effectiveness.

While transformers are powerful for capturing long-range dependencies, their computational cost is prohibitive for large-scale stock market data. To address this, linear self-attention mechanisms are explored in this study, reducing complexity while preserving predictive power (Katharopoulos et al., 2020). This adaptation allows our model to scale efficiently while processing extensive historical financial datasets, ensuring that the model remains both computationally viable and capable of learning intricate temporal patterns in stock price movements.

1.1.6 Multi-Modal Integration for Enhanced Stock Price Prediction

Existing forecasting models either focus on numerical time-series data or textual financial sentiment, rarely integrating these modalities into a single framework. *To the best of our knowledge, no prior work has fully combined technical indicators from LLMs, visual features from CNNs, and temporal modeling from Transformers for stock prediction.* This research aims to bridge this gap by introducing a hybrid CNN-Transformer-LLM model that captures spatial, temporal, and textual representations for enhanced stock price prediction.

By leveraging multiple deep learning paradigms, this study constructs an integrated framework that enhances predictive accuracy by combining structured and unstructured data sources:

- **LLMs**: Dynamically generate and refine structured technical indicators (e.g., Moving Averages, RSI, Bollinger Bands) from historical market data using *prompt engineering*. These LLM-derived indicators enhance dataset representation by providing domain-informed, multi-dimensional financial insights that complement traditional numerical

indicators.

- **CNNs**: Extract spatial and structural patterns from stock chart images, capturing visual representations of price movements and market trends.
- **Transformers**: Model complex temporal dependencies in stock price data, allowing the framework to recognize long-term trends and relationships within time-series data.

By fusing these three modalities, the proposed framework provides a more holistic and accurate stock price prediction model. Unlike traditional methods that rely solely on numerical data, our multi-modal approach leverages textual (LLM-derived), visual (chart-based), and sequential (historical price) information. This design enhances the model's predictive capabilities while reducing reliance on noisy external sentiment data, thereby improving the reliability and interpretability of the predictions.

1.1.7 Summary of Background and Motivation

The increasing complexity of financial markets necessitates the development of advanced forecasting models that integrate multiple sources of information. This study proposes a multi-modal deep learning framework that combines LLM-based technical analysis, CNN-driven spatial feature extraction, and transformer-based temporal modeling to enhance stock price prediction.

By leveraging structured market data and avoiding reliance on noisy external sentiment information, this approach offers a robust alternative to conventional financial forecasting methods. The integration of these deep learning architectures enables the framework to identify intricate market patterns, improving the accuracy and reliability of stock price forecasts.

Through this research, we aim to contribute to the ongoing development of deep learning methodologies in financial forecasting, demonstrating the potential of multi-modal data integration in enhancing market prediction models.

The specific research questions are as follows.

1.2 Research Questions

Research Question 1: How can transformer models be used for stock market forecasting?

Transformer models have shown significant potential in the domain of stock market forecasting due to their ability to model complex temporal dependencies and long-range relationships within financial time-series data. Unlike traditional sequential models such as Recurrent Neural Networks (RNNs) or Long Short-Term Memory (LSTM) networks, transformers employ self-attention mechanisms that allow the model to capture intricate temporal patterns without the limitations of sequential processing. This makes transformers particularly effective for analyzing stock market data, which often exhibits long-term dependencies influenced by historical trends and market dynamics.

However, the application of standard transformer architectures in financial forecasting faces challenges, particularly in terms of computational efficiency and scalability when dealing with extensive historical data. To address these concerns, linearized self-attention mechanisms have been introduced, reducing the computational complexity from quadratic to linear with respect to the input sequence length. This advancement allows for the efficient handling of large-scale financial datasets, enabling the model to process longer time horizons and capture broader market trends without sacrificing computational feasibility.

The integration of linearized attention within transformer models facilitates the extraction of essential temporal features while maintaining efficiency. This approach enhances the model's capability to detect significant market signals and long-range dependencies that are critical for accurate stock price prediction. Furthermore, by incorporating techniques such as positional encoding and multi-head attention, the model can differentiate between various time steps, capturing both local fluctuations and global market movements.

This research explores the use of linear transformer architectures specifically tailored for stock market forecasting, examining how modifications to the attention mechanism can improve the model's ability to interpret complex time-series data. The study also investigates the synergy between linear transformers and other deep learning components, such as convolutional neural networks (CNNs) for spatial feature extraction and large language models (LLMs) for contextual analysis. By leveraging this multimodal framework,

the research aims to enhance predictive accuracy and provide deeper insights into stock market behavior.

Through comprehensive experimentation and analysis, this study demonstrates the advantages of using transformer models with linearized self-attention for financial forecasting, highlighting their ability to balance predictive performance and computational efficiency. This approach not only advances the methodology for stock market prediction but also sets a foundation for future research into hybrid architectures that combine multi-modal features.

Research Question 2: How can large-scale language models (LLMs) be used for technical analysis of stock market forecasts?

LLMs have traditionally been used to analyze messages and sentiment in social media for stock market forecasts. However, this approach typically introduces noise and unreliability into the forecasting process. This research topic explores a new application of large-scale language models: descriptive analysis and development of features specifically for market data.

This investigation examines ways in which large language models generate sophisticated financial metrics from raw market statistics, enhancing feature representation capabilities while circumventing challenges typically associated with noisy external datasets. This includes investigating the effectiveness of improvised technical techniques in technical analysis and integrating LLM-generated features with CNN and transformer components.

Research Question 3: How can CNN and Transformer architectures be integrated to discover nuanced patterns of stock price volatility?

In financial forecasting, CNN and transformer architectures have complementary advantages when processing stock market data. CNNs are excellent at detecting local patterns and spatial features in continuous data and are particularly effective at analyzing stock charts and capturing short-term price movements. Due to their self-recognition mechanisms, transformer models are excellent at modeling temporal correlations in long time series, making them ideal for detecting market trends and long-term relationships in time series data.

Integrating these architectures brings unique challenges and opportunities. CNN's ability to extract hierarchical features from visual representations of stock data complements Transformer's ability to model complex temporal relationships. This research aims to investigate how these architectures can best be combined to leverage their strengths while addressing the challenges of processing financial data.

1.2.1 Identified Gaps in Current Research

Despite significant advances in deep learning approaches for stock price prediction, our comprehensive literature review reveals several critical gaps that urgently need addressing. Although hybrid models have been investigated in various studies, there remains a pressing need for more efficient architectures that can effectively handle both temporal and spatial features while maintaining computational efficiency (Kristanti et al., 2024).

Our review identified three key technological components that could be integrated to address these gaps:

- **Convolutional Neural Networks (CNNs):** Excellent at extracting spatial features and detecting local patterns in stock charts. They are particularly effective at recognizing short-term market trends through visual representations of financial data.
- **Transformers:** Capable of modeling complex temporal dependencies but often face computational challenges with long sequences. Linear self-attention transformers can capture long-term dependencies while significantly reducing computational complexity (Zeng et al., 2023).
- **Large Language Models (LLMs):** Typically used for sentiment analysis in news and social media, but we propose applying them innovatively to technical analysis. This approach enables creation of complex market metrics derived solely from trading information, eliminating distortion commonly present when incorporating external data sources (Wu et al., 2023).

The integration of these components presents both opportunities and challenges. CNN's ability to extract hierarchical features from visual data complements the transformer's strengths in modeling complex temporal relationships. Meanwhile, the LLM component extends the framework by generating high-quality technical indicators that capture subtle market patterns.

1.2.2 Our Contributions

To address these challenges, we propose a novel hybrid deep learning framework that utilizes a multi-stream architecture and feature fusion mechanism to enable LLM-enhanced stock predictions in a fully autonomous manner. Our contributions include:

1. A multi-modal approach that processes stock data through three parallel streams:
    - A text processing pipeline using LLM and FinBERT for technical analysis
    - A price data pipeline with linear self-attention transformer for temporal pattern recognition
    - An image processing pipeline with CNN for spatial feature extraction
2. Technical innovations including:
    - A linearized self-attention mechanism that improves computational efficiency while retaining the ability to capture long-term correlations
    - Feature extraction using LLM-based technical analysis that focuses on market data to avoid external noise
    - An effective fusion mechanism that combines CNN-based spatial features and transformer-based temporal modeling

The proposed framework addresses the identified research gaps by providing a comprehensive solution that leverages the strengths of multiple deep learning approaches while mitigating their individual limitations.

Integration of LLMs with Technical Analysis

LLMs have been remarkably successful in understanding natural language, but their potential for enhancing technical analysis remains largely unexplored. Current research primarily focuses on using LLMs for sentiment analysis of news and social media data, but lacks investigation into:

- Methods for interpreting and generating sophisticated technical indicators directly from market data
- Techniques for converting LLM-generated technical insights into quantitative features while preserving interpretability
- Frameworks for integrating LLM-based technical analysis with traditional market indicators

Computational Efficiency in Long-term Dependencies

Existing transformer-based architectures face significant challenges when processing extended sequences of market data:

- Traditional attention mechanisms become computationally prohibitive for long time series
- Current solutions often sacrifice model capacity for efficiency
- Lack of frameworks that can effectively balance computational resources with the need to capture long-range market dependencies

Multi-modal Feature Integration

While hybrid architectures have shown promise, substantial gaps remain in effectively combining multiple data modalities:

- Limited research on maintaining the unique characteristics of each modality (textual, temporal, visual) during integration
- Insufficient exploration of feature fusion techniques that preserve the interpretability of technical indicators
- Lack of systematic approaches for weighting and combining diverse information sources

Chapter 2

Literature Review

Stock price prediction has been a long-standing challenge in financial research, attracting significant interest from academia and industry alike. Over time, predictive methodologies have evolved from simple statistical models to sophisticated artificial intelligence-driven frameworks. This evolution reflects the increasing complexity of financial markets and the growing demand for more accurate forecasting tools.

The progression of predictive methodologies can be organized into several key developments:

1. **Machine Learning**: Incorporating advanced algorithms to capture non-linear relationships
2. **Deep Learning**: Leveraging neural networks for complex pattern recognition
3. **Hybrid Approaches**: Combining multiple methodologies to enhance prediction accuracy
4. **Large Language Models in Financial Analysis**: The integration of LLMs to extract insights from financial textual data and generate adaptive technical indicators.

This methodological evolution reflects the increasing sophistication of financial markets and the growing need for more accurate predictive tools. Each advancement has contributed to our understanding of market dynamics while simultaneously revealing new challenges and opportunities for future research.

To gain a comprehensive understanding of these approaches, we review key literature spanning **machine learning (ML), deep learning (DL), Transformer-based architectures, and Large Language Models (LLMs)**. Each method has contributed uniquely to forecasting stock market trends and enhancing financial decision-making. The following sections provide a structured analysis of these methodologies, highlighting their **principles, advantages, and limitations**.

2.1 Exploration of Machine Learning Approaches in Financial Analysis

Traditional statistical methods, such as Simple Moving Average, Weighted Moving Average, and Exponential Smoothing, have been widely used for stock price prediction. However, these techniques often struggle to handle the inherent non-stationarity and chaotic nature of stock market data, making them less effective when prices exhibit sudden fluctuations (Bhattacharjee and Bhattacharja, 2019). Machine learning methods, by contrast, offer a significant improvement by capturing complex, non-linear relationships within financial time series. Their ability to adapt to dynamic market patterns and optimize feature selection has led to more accurate predictions, marking a paradigm shift in financial forecasting.

Machine learning approaches have revolutionized the landscape of stock market prediction by introducing sophisticated methods for pattern recognition and data analysis. As illustrated in Fig.2.1, these approaches encompass a diverse range of algorithms and methodologies, each offering unique capabilities for processing financial data and generating predictive insights. The development of ML techniques for stock prediction significantly differs from traditional approaches. ML improves the performance to discover complex dynamic relationships in market data and supplies a robust architecture for feature extraction and model optimization (Kamath et al., 2024). These approaches have demonstrated remarkable versatility in handling the multifaceted nature of stock market dynamics, from price trend analysis to pattern recognition in market behavior. ML has contributed to stock price prediction by discovering dynamic market patterns. The change from traditional approaches to ML algorithms means a significant advancement in financial forecasting capabilities. These developments have enabled more nuanced

approaches to market analysis, incorporating multiple data dimensions and temporal patterns.

ML method for stock price forecast follow the framework that encompasses several key aspects:

- **Model Training Process**
  - Learns relationships between features and labels
  - Utilizes training data to optimize model parameters
  - Requires consistent distribution across training and test sets in feature space
- **Feature Engineering**
  - Implements dimensionality reduction techniques
  - Enhances time series prediction capabilities
  - Improves model efficiency and effectiveness
- **Primary Algorithmic Approaches**
  - *Linear Regression (LR)*: For capturing linear relationships in time series data
  - *Random Forests*: Leveraging ensemble methods for robust predictions
  - *Support Vector Machines (SVM)*: Handling non-linear patterns through kernel methods

The widespread adoption of these machine learning techniques in stock market forecasting demonstrates their effectiveness in capturing complex temporal patterns and relationships within financial time series data.

Support Vector Machine (SVM) has emerged as a prominent tool in stock price prediction research. (Ince and Trafalis, 2008) conducted comparative analysis between SVM, multilayer perceptron (MLP), and autoregressive integrated moving average (ARIMA) models for short-term stock price trend prediction. Building upon this foundation, (Wang et al., 2018) developed an improved classification framework utilizing fuzzy boundary vector techniques, which proved highly effective for forecasting directional movements across major indices including NASDAQ and S&P markets.

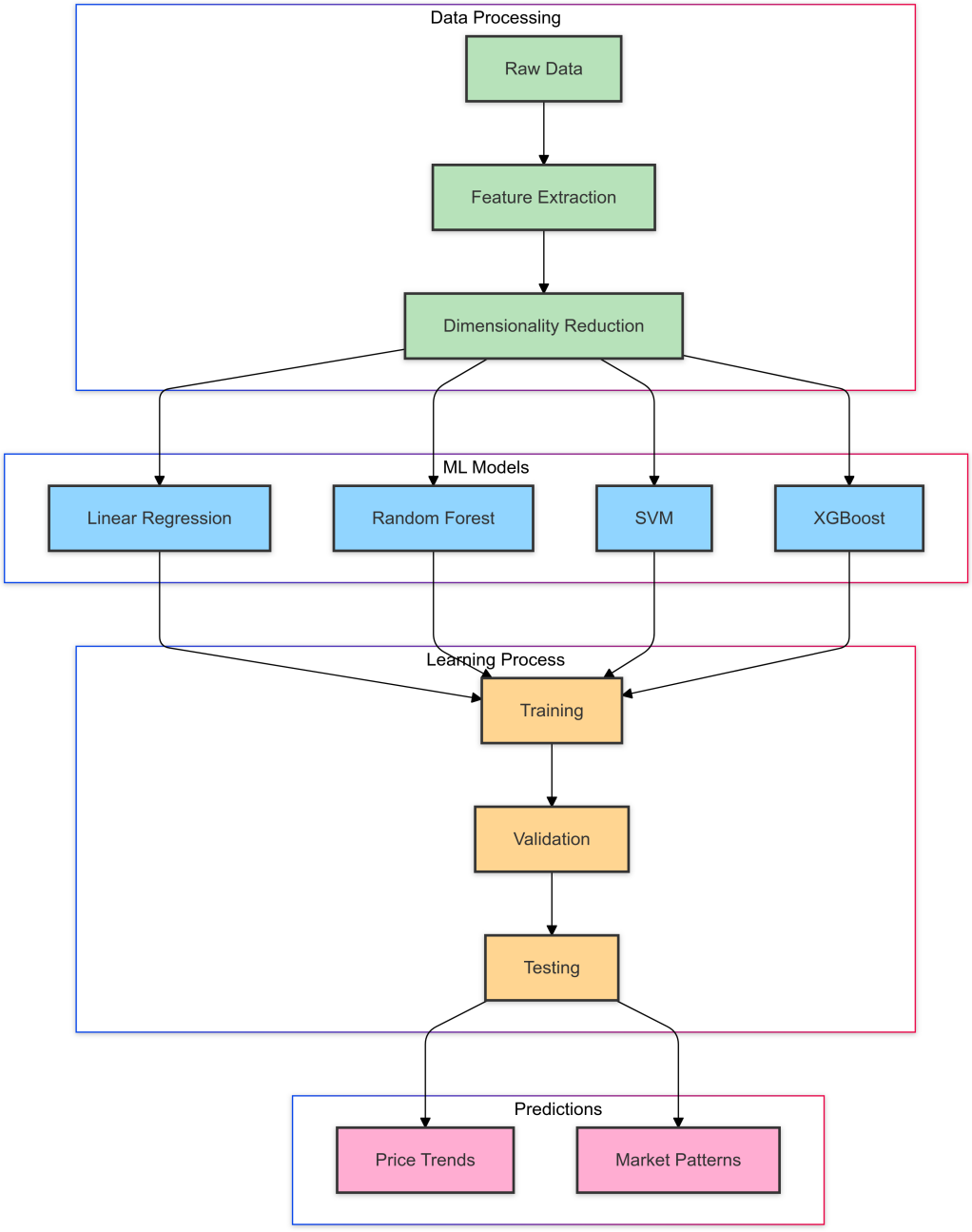


Figure 2.1: Machine Learning Approaches in Stock Prediction

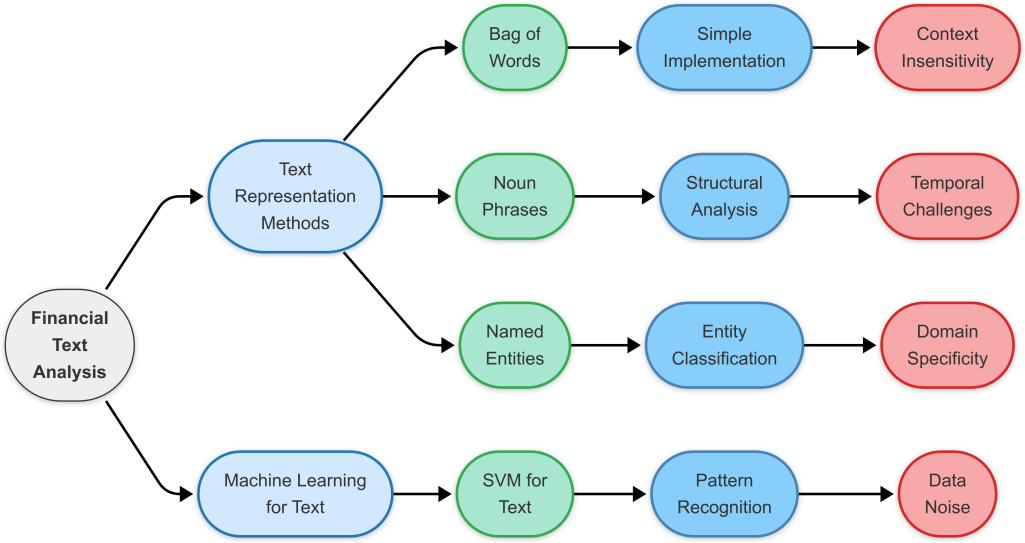


Figure 2.2: Early research on financial text analysis

Further developments in the field have explored various methodological approaches. (Panwar et al., 2021) employed a combination of linear regression and support vector machines, validating their methodology through experimental results. Taking a different perspective, (Shankar et al., 2020) reframed stock price prediction as a classification problem, utilizing random forest techniques to forecast price trends rather than specific values. Experimental results showed that random forest outperformed other algorithms in predicting stock price trends. Traditional machine learning algorithms have strong non-linear matching capabilities that compensate for the shortcomings of traditional time series algorithms (Qiu et al., 2020), but their disadvantage is that they have difficulty dealing with the correlation of stock price time series.

2.1.1 Early Research on Financial Text Analysis

Financial text analysis has long been used as an important means of understanding and predicting stock market behaviour. One of the earliest approaches was to use basic textual representations such as bag-of-words (BoW), noun phrases and named entities. See Fig.2.2.

Bag of Words

Bag of Words model became popular because of its simplicity: it represents text as a disordered collection of words, without regard to syntax or word order. Despite its widespread use, it often struggles to capture the complexity of financial language. For example, in financial news articles, small changes in words can radically change the meaning, making it difficult to effectively model the nuances of textual data with a set of words. However, BoW is often used in financial stock price prediction models, as it can provide meaningful results with large datasets.

Noun Phrases

To overcome the limitations of BoW, researchers have begun to explore more structured approaches, such as the use of noun phrases. Noun phrase extraction aims to capture more meaningful relationships between the elements discussed in a financial text by focusing on the structure associated with the nouns in a sentence. This approach helps to understand the main theme of the text and is very useful for predicting share price movements based on the key players involved. To identify and classify these noun phrases, the method generally relies on lexical markers, which offer deeper insights than simply treating individual words as distinct entities.

Named Entities

Another important enhancement to the Bag of Words model is what is known as Entity Recognition (NER). NER identifies and classifies proper nouns in text on the basis of predefined categories such as date, place, money, organisation, person, and so on. This method is beneficial in the financial sector. For example, information about certain companies, people, or places associated with news items can offer greater power in predicting share price movements. For example, the identification of companies mentioned in stock market forecasts in a positive or negative context can have a considerable influence on future share prices.

Support Vector Machines (SVM)

SVM has become a widespread ML method for financial text analysis because it can solve regression and classification concerns. When applied to textual

data, SVM can be used to model the relationship between key words or phrases in financial articles and the resulting stock price movements.

By examining patterns in past news articles and their effects on stock prices, SVM models can predict future price changes with a high degree of accuracy (as cited in (Fung et al., 2002)).

However, given that stock prices can be influenced by many factors other than text, SVMs reach their limits, not least because of the chaotic and unstructured nature of financial news.

XGBoost, Random Forest

Machine learning models, such as Random Forest and XGBoost, have been employed in stock prediction for their adaptability and capability to capture non-linear patterns in stock data (Basak et al., 2019). These ensemble learning techniques aggregate multiple weak learners to improve robustness and predictive accuracy. Research has shown that machine learning models outperform traditional methods by handling large volumes of data and accounting for complex dependencies among features (Kristanti et al., 2024).

Limitations of Early research on financial text analysis

Despite initial advances, early research in financial text analysis faced several significant limitations. The inherent complexity and dynamic nature of financial markets posed substantial challenges for traditional text analysis methods. Textual data in financial documents is often noisy, ambiguous, and highly variable, making it difficult for simple text processing techniques to extract meaningful signals.

One major limitation was the inability of early methods to effectively capture the contextual nuances of financial language. As noted by Loughran and McDonald (2020), traditional approaches like bag-of-words and simple word frequency counts often failed to capture the sophisticated relationships between textual content and market movements. The authors emphasize that financial terminology can carry different meanings depending on context - a nuance that basic text analysis methods struggled to address.

Another significant challenge was the temporal aspect of financial text analysis. Kumar and Ravi (2016) point out that financial texts can quickly become outdated or irrelevant, particularly given the rapid pace of market changes. Early methods had difficulty incorporating the time-sensitive nature

of financial information, leading to potential misalignment between textual signals and market movements.

The scalability of early approaches also presented challenges. As highlighted by Loughran and McDonald (2020), the exponential growth in financial textual data - from regulatory filings to social media content - made it increasingly difficult for traditional methods to process and analyze information efficiently. The authors note that early techniques often struggled with the dimensionality and volume of modern financial text data.

Furthermore, Kumar and Ravi (2016) identify the challenge of domain specificity in financial text analysis. Standard natural language processing techniques, developed for general text analysis, often performed poorly when applied to specialized financial documents due to the unique characteristics of financial language and jargon.

These limitations have driven the development of more sophisticated approaches, including advanced machine learning techniques and specialized financial text processing methods. However, understanding these early limitations remains crucial for developing more effective solutions in financial text analysis.

Summary of Early Works in Financial Text Analysis

Stock market forecasting faces complex problems due to the volatility and interconnectedness of financial markets. Traditional methods rely heavily on numerical data, but the input of textual information increases complexity due to noise and ambiguity in financial descriptions (Kumar and Ravi, 2016). It is increasingly challenging to maintain forecasting accuracy when processing unstructured textual data (Loughran and McDonald, 2020).

2.1.2 Linear Methods

Linear models have long been a fundamental tool in time-series forecasting due to their **simplicity and interpretability**. They provide a structured framework for identifying trends, seasonality, and dependencies within stock market data.

There are two primary advantages to using linear models in financial forecasting. First, **they are computationally efficient**, allowing for quick model training and real-time analysis. Methods such as ARIMA (AutoRegressive Integrated Moving Average) and SARIMAX (Seasonal ARIMA with

Exogenous Variables) can be constructed with relatively low computational overhead, making them well-suited for applications that require rapid predictions.

Second, linear models excel at handling **large-scale historical data**, as they rely on well-defined mathematical structures to extract meaningful insights. Their ability to process extensive datasets efficiently makes them particularly useful for short-term forecasting, where immediate trends and periodic patterns play a crucial role.

A common example of such models is **ARMA (AutoRegressive Moving Average)**, which represents a sophisticated approach to time series analysis by combining multiple statistical components. As shown in **Figure 2.3**, these models operate by decomposing stock price movements into autoregressive and moving average components, capturing patterns that arise over time.

Despite these strengths, linear models face limitations when dealing with highly **volatile and non-linear financial markets**. Stock prices are influenced by a multitude of unpredictable factors, including macroeconomic events, investor sentiment, and global financial trends, which traditional linear methods may struggle to capture. This has led to the growing adoption of more **adaptive and data-driven approaches**, which will be explored in subsequent sections.

The ARMA (Autoregressive Moving Average) model represents a sophisticated approach to time series analysis, combining multiple statistical components. Despite certain limitations inherent in existing time series models, ARMA provides a framework for describing weak and smooth stochastic processes through its core elements:

- **AR (Autoregressive)** component: Models the dependencies among an observation and a certain number of lagged observations.
- **I (Integrated)** component: It considers the transformations required to fix the time series through propagation.
- **MA (Moving Average)** component: In the moving average model, the correlation between the residuals of the observed values and the delayed observed values was taken into account.

These components work in concert to create a generalized framework for modeling complex time series patterns and relationships (Ariyo et al., 2014),

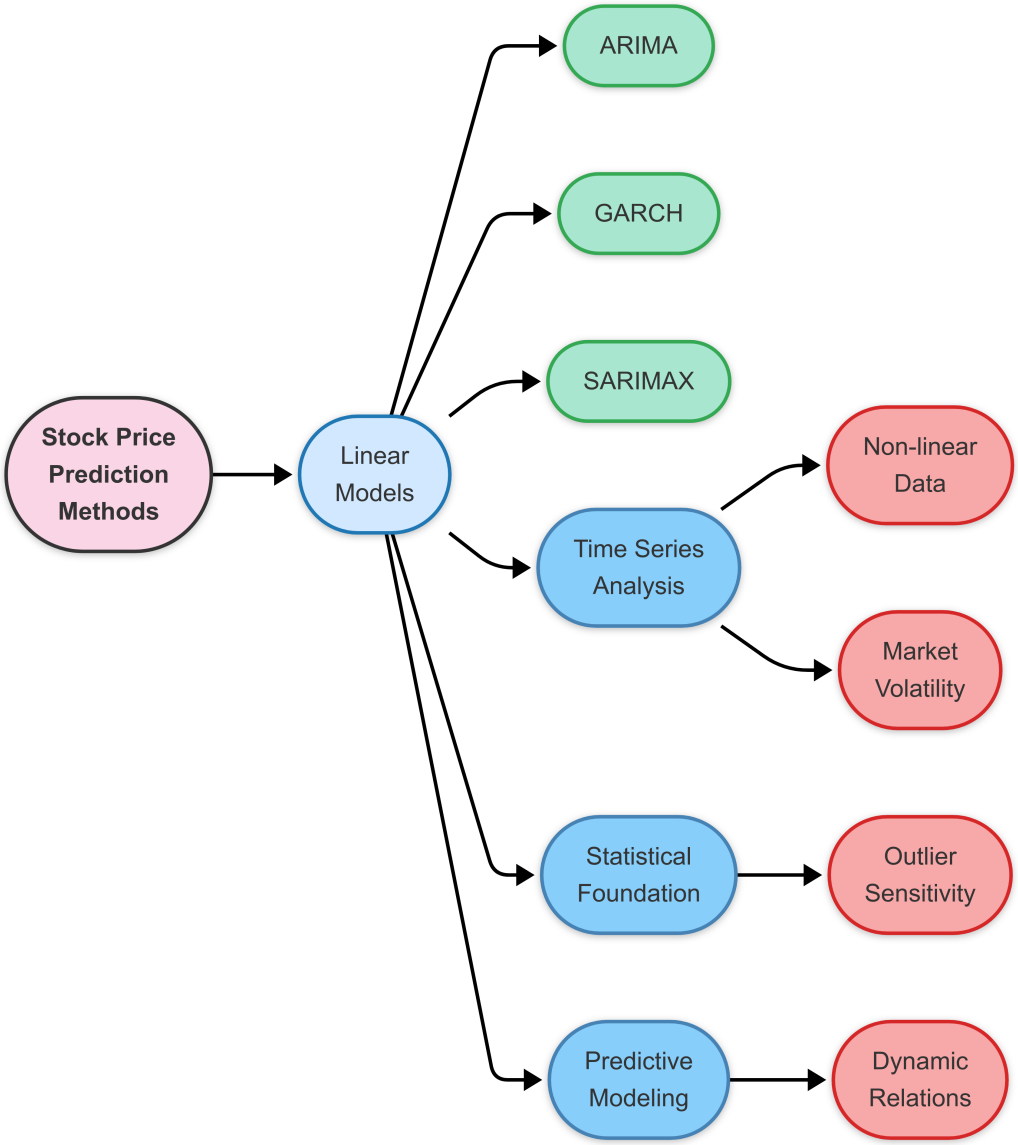


Figure 2.3: Linear Methods Overview

the ARIMA model has obtained good results in forecasting time series (Xu and Qin, 2021). To use this forecasting method, certain conditions must be met. For instance, the input values should be normally distributed, but the corresponding data on the actual stock market have non-linear characteristics and cannot meet this condition. As a result, traditional time series algorithms have certain limitations in forecasting stock prices. Stock price data contains much noise and is highly stochastic and non-linear in nature. Forecasting stock price movements using traditional time series algorithms (ARIMA) requires homogeneous models. However, stock price data does not meet this requirement, which poses serious problems for the modeling accuracy of traditional time series models. The inability to handle non-linear data is a major drawback of ARIMA (Zhang and Tumibay, 2022).

Another well-known model is the generalized conditional autoregressive autocorrelation (GARCH), which is a derivative of the conditional autoregressive autocorrelation (ARCH) model that describes the error variance of the previous period as a function of the actual size of the error term, but instead of using the ARMA model for the AR model, GARCH generalizes ARCH by assuming that the error variance is an ARMA model. (Gao, 2023) proposed a composite ARIMA-GARCH model for Chinese stock price forecasting. In short-term forecasting, with stable market data, the average relative error is 1.29%, but the problem is that the input data is easily subject to market fluctuations. The average relative error is 1.29%, but the problem is that the model cannot perform well when the input data is affected by market volatility and various uncertainties. The biggest drawback of the linear model is that the time series data must be stable or stable in terms of difference. The essence of the linear model is to reflect linear data, but it cannot handle non-linear data.

SARIMAX, ARIMA

Models like SARIMAX and ARIMA have long been used in stock market forecasting. Because they can handle time-dependent data (Sharma et al., 2022).

These models assume linearity and are highly effective for stationary time series. However, these models often fall short in capturing the non-linear, complicated character of stock markets (Alharbi and Csala, 2022).

Consequently, their predictive accuracy is limited when applied to volatile and high-dimensional financial data, motivating the shift towards more flex-

ible machine learning approaches.

2.1.3 DLinear: Revisiting Simplicity in Time Series Forecasting

Contrary to the prevailing trend of using increasingly complex deep learning architectures, Wang et al. (2023a) introduced a simple yet effective linear model, termed DLinear. Surprisingly, their study revealed that DLinear outperforms many sophisticated deep learning models on widely used benchmark datasets. This finding challenges the assumption that highly deformable and parameter-heavy architectures are always advantageous for time series forecasting. The success of DLinear highlights the importance of revisiting simpler, interpretable models that can achieve strong performance with minimal computational overhead.

DLinear is fundamentally designed to decompose time series data into distinct components before applying separate linear layers to each. As illustrated in Figure 2.4, the architecture consists of two primary components:

- **Trend Component**: Extracted from the input time series to capture long-term directional movements in the data (shown in the lower part of the blue block in Figure 2.4a).
- **Remainder Component**: Computed as the residual between the original time series and the extracted trend, preserving temporal fluctuations (displayed in the upper part of the blue block in Figure 2.4a).

Each component is independently processed using a dedicated linear layer (represented by the red block in Figure 2.4b), and the outputs are then combined to produce the final prediction. This architectural approach significantly reduces computational requirements while maintaining robust forecasting capabilities.

The mathematical formulation of DLinear can be expressed as:

$$\hat{X} = H_s + H_t \tag{2.1}$$

$$H_s = W_s X_s \in \mathbb{R}^{T \times C} \text{ (remainder component)} \tag{2.2}$$

$$H_t = W_t X_t \in \mathbb{R}^{T \times C} \text{ (trend component)} \tag{2.3}$$

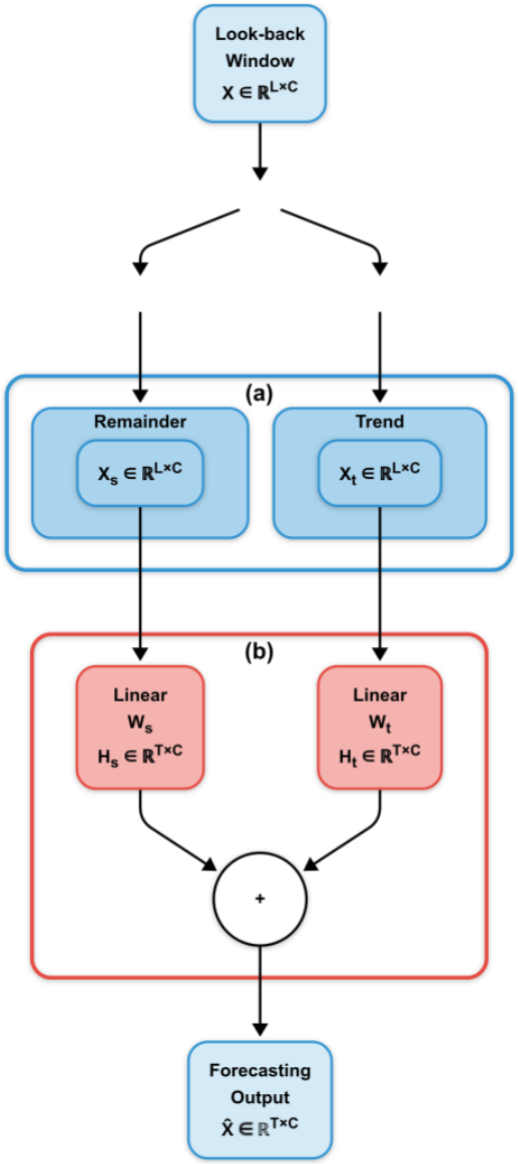


Figure 2.4: DLinear architecture: (a) decomposition phase that splits the input time series into trend (X_t) and remainder (X_s) components (blue block); (b) application of distinct linear transformation layers (W_t and W_s) to each component (red block), with results summed to generate the final forecast output. Adapted from Zeng et al. (2023).

where $\hat{X}$ represents the forecasted values, $W_s \in \mathbb{R}^{T \times L}$ and $W_t \in \mathbb{R}^{T \times L}$ denote the linear transformation matrices, T indicates the prediction horizon length, and L refers to the historical sequence length.

Furthermore, the findings raise concerns regarding the effectiveness of complex, heavily parameterized architectures in time series forecasting tasks. By emphasizing simplicity, DLinear demonstrates that carefully designed linear models can outperform more sophisticated architectures in terms of both accuracy and computational efficiency. The single-layer design significantly reduces memory requirements and processing time compared to transformer-based models, while enabling intuitive interpretation through weight analysis.

Recent advancements have also led to the development of RevIN-DLinear, a variant that integrates Reversible Instance Normalization (RevIN) to address distribution shift issues in non-stationary time series. By applying instance normalization at the input stage and denormalization at the output stage, RevIN-DLinear ensures stable model performance even when faced with significant temporal variations in data distributions. This adaptation has further improved forecasting accuracy in domains such as photovoltaic power prediction and financial time series modeling.

These results suggest that, despite the growing popularity of Transformer-based models, alternative architectures like DLinear remain competitive, particularly in scenarios where computational efficiency and interpretability are critical. Multiple studies have indicated that this approach can deliver superior time series predictions compared to more complex transformer implementations Liu et al. (2022); Zhou et al. (2022b). Future research may explore hybrid models that integrate the strengths of both linear forecasting techniques and deep learning paradigms.

Limitations of Linear Methods

Linear regression methods, while widely used in stock price prediction, face several fundamental limitations that affect their predictive accuracy and reliability. Cakra and Trisedya (2015) and Wang et al. (2021) highlighted several key limitations in their research on linear regression-based stock price prediction models.

First, linear regression models assume a linear relationship between the input variables and the output (stock price), which often oversimplifies the complex dynamics in financial markets. As demonstrated in Cakra and Trisedya (2015)'s study, even when combining sentiment analysis with linear

regression, the models struggle to find the dynamic pattern, like the nonlinear relationship. Research conducted by Wang et al. (2021) reinforces this constraint through their examination of specific industry listed companies, demonstrating how traditional linear frameworks inadequately represent sophisticated interactions among diverse economic parameters.

Second, linear models are very dependent on the input data quality and representativeness. Cakra and Trisedya (2015) found that when using sentiment analysis data as a predictor variable, the coefficient of determination (R^2) values for margin percentage prediction were consistently close to 0, indicating that the linear model failed to clarify the variance of related variables.

This suggests that linear models may not effectively indicate the complicated connections of market sentiment with price movements.

Third, linear regression models assume that the connections between variables stay invariant for extended period. However, this is rarely the case in dynamic financial markets.Wang et al. (2021)'s empirical analysis demonstrates this limitation, showing that even with multiple financial indicators, the model's explanatory power remains limited due to the dynamic nature of market relationships. Their study revealed that while certain financial indicators like EPS showed strong correlations, others exhibited inconsistent relationships with stock prices over time.

Fourth, these models are particularly vulnerable to outliers and extreme market events. The assumption of normal distribution in linear models makes them less suitable for handling the fat-tailed distributions commonly observed in financial time series. Wang et al. (2021) notes that company finances are sometimes affected by uncertain factors, causing stock price fluctuations that linear models struggle to capture effectively.

Finally, linear regression models struggle with temporal dependencies and the dynamic nature of financial markets. While they can incorporate lagged variables, they do not inherently account for the complex temporal patterns and dependencies that characterize stock price movements. Wang et al. (2021)'s research shows that even when using panel data models with multiple financial indicators, the ability to capture dynamic market relationships remains limited, as evidenced by their model's necessity to include various control variables to improve prediction accuracy.

These gaps highlight the demand for better modeling approaches that can sufficiently capture the non-linear, complex pattern of stock market behavior.

While linear methods continue to serve as important baseline models

and can provide useful insights, their inherent limitations suggest that they should be complemented with or replaced by more advanced techniques for better stock price forecasting.

As Wang et al. (2021) concludes, only through a systematic perspective incorporating multiple analytical approaches can we scientifically evaluate the factors affecting stock prices.

2.1.4 Bridging Traditional and Modern Approaches

As we have seen, linear models like ARIMA, SARIMAX, and the more recent DLinear provide valuable insights into time series forecasting through their simplicity, interpretability, and computational efficiency. DLinear, in particular, demonstrates that carefully designed decomposition-based linear approaches can achieve competitive performance even when compared to more complex architectures. This challenges the common assumption that increasingly sophisticated models always yield better results in financial forecasting.

However, the inherent limitations of linear methods—including their struggle with non-linear relationships, sensitivity to data quality, and difficulty in capturing complex temporal dependencies—highlight the need for more advanced modeling techniques. These limitations become particularly evident when dealing with the inherently volatile, non-stationary, and multidimensional nature of financial markets, where numerous factors simultaneously influence stock price movements in complex and often unpredictable ways.

Despite their advantages, traditional machine learning models often require extensive feature engineering and struggle to capture the hierarchical and temporal dependencies inherent in financial markets. These limitations have motivated the adoption of **deep learning methods**, which can automatically learn intricate representations from raw financial data and improve forecasting accuracy.

This recognition has driven researchers and practitioners toward more flexible and powerful methodologies capable of automatically learning intricate patterns from raw financial data. Deep learning approaches, with their ability to discover hierarchical representations and model complex non-linear relationships without extensive feature engineering, present a promising direction for overcoming these challenges. The following section explores how various deep learning architectures address the shortcomings of traditional methods while introducing new capabilities for stock price prediction.

2.2 Deep Learning Approaches for Stock Prediction

Deep learning algorithms are frequently used for price forecasting and have shown better performance than traditional machine learning algorithms. Artificial neural networks (ANN) have been extensively studied for decades (Sondak, 1989). Neural network models mimic the neurons in the human brain and have evolved significantly in their application to financial markets.

Deep learning models have revolutionized numerous domains by automatically extracting complex features from raw data through multiple layers of non-linear transformations. In financial time series forecasting, these models offer several distinct advantages over traditional approaches. First, they can capture intricate non-linear relationships without requiring explicit feature engineering. Second, they excel at learning hierarchical representations that connect low-level patterns to high-level market behaviors. Third, certain architectures naturally model temporal dependencies across different time scales, making them particularly suitable for financial time series analysis.

The following subsections explore key deep learning architectures that have demonstrated significant promise in stock price prediction, beginning with recurrent neural networks (RNNs) and their variants, which are specifically designed to handle sequential data by maintaining internal memory states.

The structure of neurons and networks corresponds to the connections between neurons and neurons. Neural networks consist of many interconnected neurons that are connected to each other. The environment of deep learning methods consists of different architectural paradigms. Deep learning consists of various architectural paradigms, each of which offers functionality tailored to the market. These approaches include convolutional neural networks (CNNs), long short-term memory networks (LSTMs), and their hybrids. These architectural paradigms represent significant methodological advances in complex financial data processing and analysis.

These approaches represent significant advances in the processing and analysis of complex financial data. The integration of these advanced neural network architectures has revolutionized stock price prediction, offering unparalleled capabilities for pattern recognition, time series analysis, and feature extraction.

The main feature of deep learning algorithmic models (Schmidhuber,

2015) is that they involve complex neural networks. In a neural network, a non-linear correspondence between data is obtained by building a network structure from samples of input data. As the depth of the neural network structure increases, the neural network can better understand the spatial and temporal characteristics of the data through deep feature extraction and stock data learning.

The main deep learning algorithm methods commonly used for stock time series forecasting include LSTM (long-term, short-term memory), CNN (convolutional neural network), GNN (graphical neural network), and transformers.

Convolutional Neural Networks (CNNs)

Convolutional neural networks have been developed for the vision industry. However, they have also shown promise in prediction. CNNs can find patterns in sequences of datasets and remove short-term patterns from stock data, capturing features related to price and volume fluctuations in financial time series analysis (Zhang et al., 2023; Hoseinzade and Haratizadeh, 2019).

CNN can work with visual representations of financial data, such as candlestick charts or technical indicators (Eapen et al., 2019). This capability has led to various innovative approaches in the prediction of the stock market. Kanwal et al. (2022) demonstrated CNNs can effectively identify complex patterns in stock price movements through their hierarchical feature extraction capabilities, making them particularly suitable for technical analysis in stock prediction.

Recent advancements have shown that CNNs can be effectively combined with other architectures to enhance prediction accuracy. Chen et al. (2021) developed a hybrid model that integrates CNN with bidirectional LSTM (BiLSTM) and attention mechanisms, achieving superior performance in capturing both spatial and temporal features. Similarly, Lu et al. (2021) proposed a CNN-BiLSTM-AM method that demonstrated significant improvements in prediction accuracy by leveraging the complementary strengths of different architectures.

However, CNNs also face certain limitations in stock prediction tasks:

- Fixed Input Size: CNNs typically require input data of fixed dimensions, which can be challenging when dealing with variable-length time series data (Ding, 2023).

- Limited Temporal Modeling: While effective at spatial feature extraction, CNNs may struggle to capture long-term temporal dependencies (Song and Choi, 2023).
- Feature Relationship Modeling: Traditional CNN architectures may not adequately model complex relationships between different market features (Li et al., 2019).

To address these limitations, researchers have proposed various solutions. (Dargan et al., 2020) suggests that incorporating attention mechanisms and multi-scale architectures can help CNNs better handle temporal dependencies. Additionally, (Jiang, 2021) demonstrates that combining CNNs with other deep learning architectures can lead to more robust and accurate prediction models.

According to findings by Zhang et al. (2024), sophisticated convolutional neural structures, when expertly constructed and combined with various elements such as transformer mechanisms or language models, yield exceptional results for securities valuation forecasting. Such architectural fusion enables simultaneous exploitation of spatial characteristic identification abilities inherent in convolutional frameworks alongside chronological sequence modeling advantages offered by complementary approaches.

2.2.1 RNN and LSTM Models

Recurrent Neural Networks (RNNs) have revolutionized sequential data processing, demonstrating particular effectiveness in financial time series forecasting (Salehinejad et al., 2017). Their ability to model temporal dependencies makes them especially suitable for stock price prediction tasks. However, traditional RNN methods have severe limitations, such as the "vanishing gradient problem" or the "gradient explosion problem", which undermine their ability to identify long-term relationships in financial data (Juairiah et al., 2022).

To overcome these limitations, Long Short-Term Memory networks (LSTMs) were introduced. These networks contain complex storage units that can store data in long sequences (Yu et al., 2019). This technological innovation has proved particularly useful in stock market forecasting, where long-term dependencies play an important role (Zhang and Tumibay, 2022). LSTMs excel at storing complex time patterns in economic data and significantly

improve forecast accuracy by storing important information over multiple time steps(Shah et al., 2021; Weng et al., 2022).

Its capabilities have been enhanced with the latest developments in LSTM architecture:

- **Improved Memory Management:** Modern LSTM variants utilize sophisticated gating mechanisms to control information flow, effectively addressing the gradient problems that plague traditional RNNs (Berradi et al., 2020).
- **Enhanced Feature Processing:** LSTMs can effectively process multiple features simultaneously, making them ideal for analyzing complex financial data (Liu and Zhao, 2022).
- **Hybrid Architectures:** Integration with other deep learning components has led to more robust prediction models (Kristanti et al., 2024).

Models based on LSTM have already been extensively validated for predicting stock prices. For example, (Sethia and Raut, 2019) demonstrated significant improvements in prediction accuracy using LSTM networks compared to traditional approaches. Further research by (Ashok and Prathibhamol, 2021) showed that LSTM models could reduce prediction errors by up to 87

However, LSTMs also present certain challenges:

- **Hyperparameter tuning:** Finding hyperparameters can be complex and time-consuming (Weng et al., 2018).
- **Architecture Design:** Determining the ideal network structure remains challenging (Jiang, 2021).

Recent developments have focused on enhancing LSTM performance through various strategies:

1. Integration with attention mechanisms to improve feature selection (Li et al., 2019)
2. Combination with CNN architectures for better feature extraction (Kanwal et al., 2022)

3. Implementation of bidirectional architectures for improved temporal modeling (Chen et al., 2021)

These advances have made stock price prediction a first-class technique, establishing LSTM-based architectures as a cornerstone of modern financial forecasting systems (Zhang et al., 2024). The continued evolution of these models, particularly their integration with newer architectures like Transformers and LLMs, suggests promising directions for future research in financial time series prediction (Wu et al., 2023).

2.2.2 CNN Models in Stock Price Prediction

Convolutional Neural Networks (CNNs) have emerged as powerful tools for extracting spatial features from financial time series data, particularly when represented as visual patterns such as candlestick charts. The application of CNNs in stock price prediction leverages their ability to automatically identify hierarchical patterns and local dependencies in data.

Visual Feature Extraction

Early applications of CNNs in stock prediction focused on converting time series data into image-based representations. Kim and Kim (2019) demonstrated that transforming financial data into 2D images allows CNNs to capture complex patterns that might be missed in traditional time series analysis. Their approach used candlestick patterns and technical indicators as input channels, achieving superior performance compared to conventional time series models.

Architecture Innovations

Recent advances in CNN architectures for stock prediction have focused on addressing the unique challenges of financial data. Specifically:

- **Residual Connections**: Li et al. (2020a) demonstrated incorporation of residual learning helps maintain gradient flow in deeper networks while capturing both short-term and long-term dependencies in stock price movements.

- **Multi-scale Feature Extraction**: Researchers have developed specialized CNN architectures that process financial data at multiple temporal scales simultaneously. For instance, Halder Halder (2022) demonstrated that combining CNN-based feature extraction with sentiment analysis can capture patterns across different time horizons more effectively than traditional approaches.

- **Hybrid Architectures**: Sen et al. Sen et al. (2021) showed that combining CNNs with LSTM networks enables the model to capture both spatial patterns through CNNs and temporal dependencies through LSTM layers. Their work demonstrated that such hybrid architectures can achieve superior performance compared to standalone CNN or LSTM models.

- **Attention-Enhanced Learning**: Modern CNN architectures for stock prediction incorporate attention mechanisms to focus on the most relevant features in the input data. This approach is highly suitable for noisy financial time series data, allowing the model to automatically learn which patterns are most predictive of future price movements.

The evolution of CNN architectures in stock price prediction has increasingly focused on hybrid approaches that combine the strengths of multiple deep learning techniques. These architectures typically leverage CNNs for their ability to extract hierarchical features from raw price data while incorporating other components like LSTM networks or attention mechanisms to enhance their predictive capabilities.

2.2.3 TimesNet: Multi-Scale Feature Extraction for Time Series Data

Time series analysis has been widely explored due to its crucial role in various applications, including finance, weather forecasting, and anomaly detection. While traditional sequence models treat time series as one-dimensional data, Wu et al. (2022) proposed *TimesNet*, a framework that enhances feature extraction through a novel transformation of time series into structured 2D tensors. This transformation facilitates the modeling of intricate temporal variations by leveraging multi-period adaptive patterns.

The core of TimesNet lies in the **TimesBlock** module, which enables adaptive feature extraction across different temporal scales. Unlike conventional models that rely solely on sequential relationships, TimesNet restructures time series data into a multi-dimensional representation, allowing the capture of both intra-period and inter-period variations. By learning multiple temporal resolutions, the model excels in identifying long-range dependencies and periodic trends, making it particularly effective for forecasting tasks such as stock price prediction and anomaly detection.

Unlike standard approaches that process time series in their raw sequential form, TimesNet applies Fourier-based transformations and convolutional structures to extract hierarchical features from temporal signals. This architecture efficiently captures both local dependencies (short-term variations) and global trends while maintaining computational efficiency. By incorporating parameter-efficient inception blocks, the model further enhances representation learning, making it adaptable to a variety of time-series applications, including economic forecasting, energy consumption prediction, and financial market analysis.

A key advantage of TimesNet is its ability to bridge the gap between traditional time series analysis and advances in computer vision. By restructuring temporal data into a 2D format, the model effectively integrates techniques typically used for image processing, leading to superior feature extraction and enhanced pattern recognition. This unique design improves both forecasting accuracy and model interpretability.

The effectiveness of TimesNet highlights the potential of hybrid deep learning methodologies that integrate innovations from multiple domains. Its capacity to generalize across diverse datasets suggests that similar structured transformations may be advantageous for financial forecasting, where complex temporal dependencies and multi-scale patterns play a critical role in market behavior analysis.

2.2.4 MICN: Multi-Scale Isometric Convolutional Networks

Another promising alternative is the Multi-Scale Isometric Convolutional Network (MICN) (Wang et al., 2023b), which applies a carefully designed multi-stage convolutional architecture to extract both local and global dependencies from time series data. MICN utilizes hierarchical convolutions

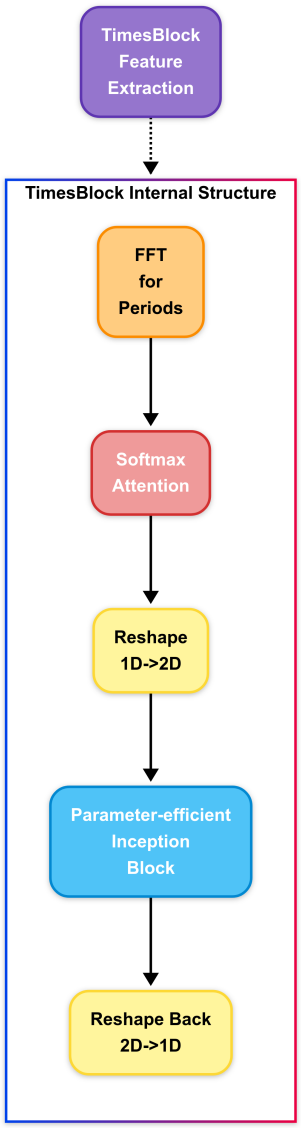


Figure 2.5: The overall architecture of TimesNet. The model employs **TimesBlock** to convert one-dimensional time series into structured two-dimensional tensors, enabling effective modeling of temporal dependencies across multiple scales. The figure is adapted from Wu et al. (2022).

to model temporal structures efficiently. This design enables MICN to retain crucial information across varying time scales while maintaining a lower computational footprint compared to attention-based architectures.

MICN is based on a carefully designed multi-stage convolutional neural network that efficiently models both local dependencies (short-term fluctuations) and global dependencies (long-term market trends). Its convolutional structure enables it to effectively extract spatial relationships in time series data, which is particularly useful in financial forecasting scenarios where short-term and long-term patterns must be considered simultaneously.

MICN consists of multiple key components:

- **Multi-Scale Hybrid Decomposition:** This module separates input time series into seasonal and trend-cyclical components, enabling better learning of different temporal patterns.
- **Local-Global Module:** It first applies down-sampled convolution to extract local features and then employs isometric convolution to capture global correlations.
- **Seasonal Prediction Block:** A dedicated block to model seasonal dependencies in time series data.
- **Trend-Cyclical Prediction Block:** A regression-based block for long-term trend estimation, helping in stable financial forecasting.

The effectiveness of MICN is demonstrated in benchmark experiments where it achieves superior forecasting accuracy over Transformer-based models. MICN efficiently reduces the computational complexity while capturing multi-scale temporal dependencies.

By combining the advantages of convolutional networks and global dependency modeling, MICN provides a robust alternative to Transformer-based approaches in financial forecasting. The integration of local and global feature extraction ensures that the model is well-suited for capturing complex patterns in stock price movements, macroeconomic indicators, and other financial time series data.

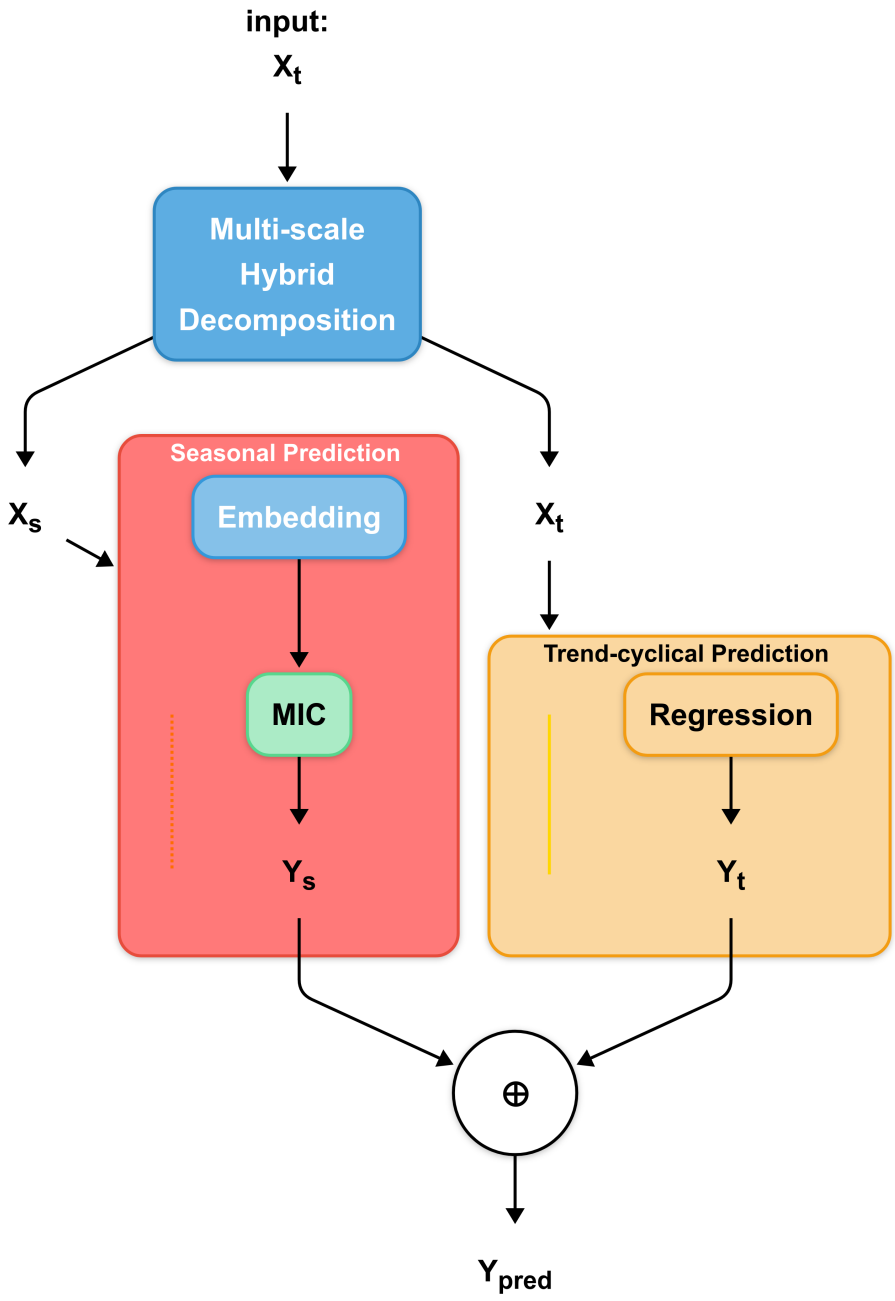


Figure 2.6: Overall architecture of MICN. The model integrates a multi-scale hybrid decomposition, seasonal prediction block, and trend-cyclical prediction block to effectively capture both local and global dependencies in time series forecasting. Adapted from Wang et al. (2023b).

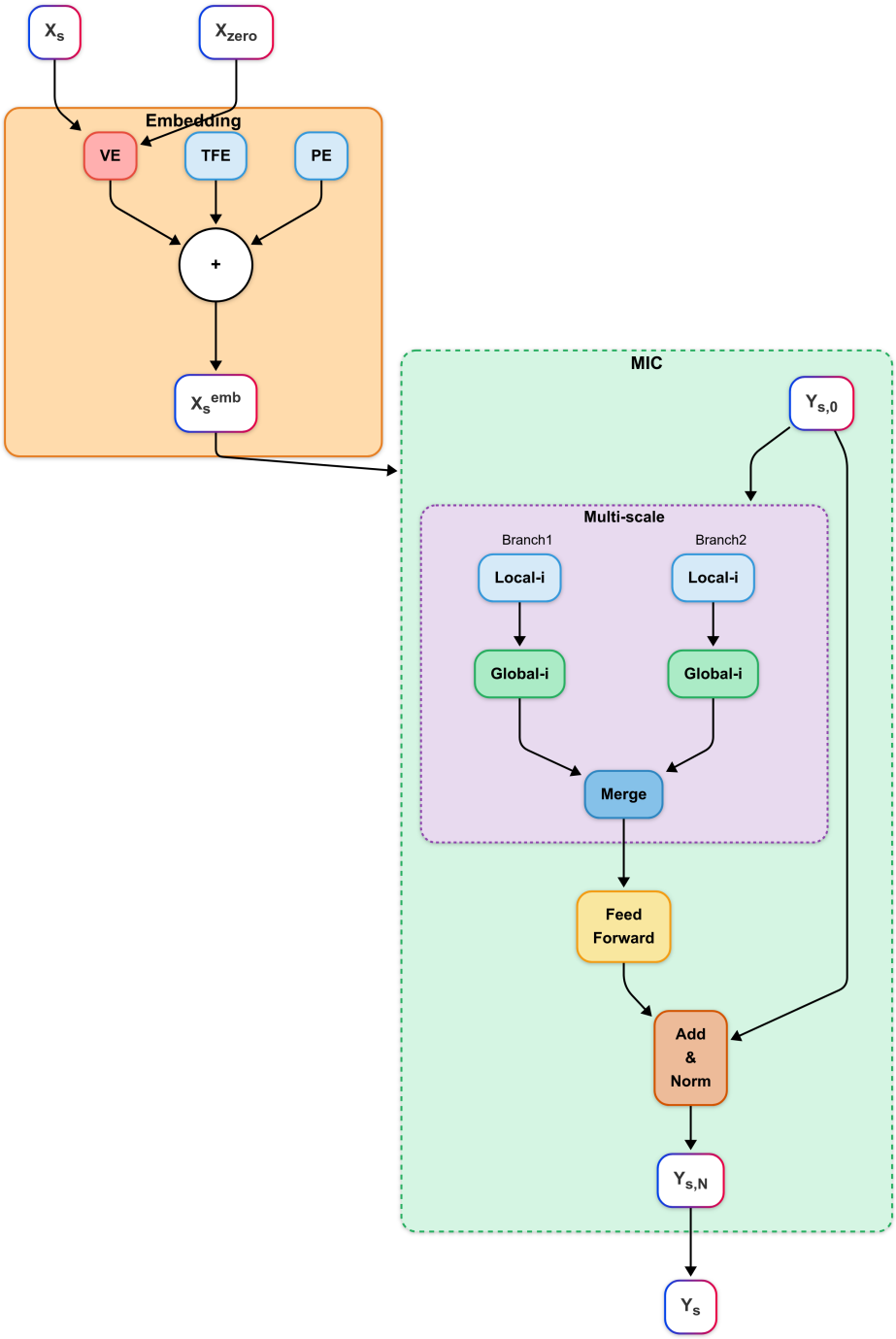


Figure 2.7: Seasonal prediction block of MICN. The module leverages multi-scale isometric convolution (MIC) for efficient extraction of local and global dependencies. Adapted from Wang et al. (2023b).

2.2.5 Nonstationarities in Stock Returns for Financial Time Series Prediction

A fundamental obstacle confronting financial sequence forecasting involves managing statistical variability—temporal fluctuations in distributional characteristics commonly detected throughout equity return datasets.

Stărică and Granger (2005) highlighted the inadequacy of assuming global stationarity in financial data analysis, proposing instead local stationary approximations to more accurately reflect the frequent and significant structural changes commonly seen in stock return series. Their empirical findings emphasize that explicitly modeling shifts in unconditional variance significantly improves forecasting accuracy compared to traditional stationary volatility models, particularly at medium- and long-term forecasting horizons.

Drees and Starica (2002) introduced a simpler yet highly effective non-stationary model tailored specifically for predicting the S&P 500 index returns. Their approach differed notably from conventional models by treating volatility as an exogenous deterministic process evolving smoothly over time, rather than as an endogenous stationary dynamic. This non-parametric regression-based model demonstrated superior performance in accurately capturing tail behaviors and asymmetries in stock return distributions, offering improved predictive power especially in extreme market conditions.

Further advancing this perspective, Strijbosch et al. (2011) examined the practical implications of non-stationary demand forecasting on inventory control systems, revealing substantial performance deterioration when traditional stationary assumptions were applied to non-stationary data. Their simulation-based study clearly demonstrated the critical importance of accurately estimating and updating demand parameters to maintain inventory efficiency and control. The authors showed that explicitly accounting for non-stationarities in forecasting procedures significantly improved the overall inventory performance, underscoring the operational necessity of adapting forecasting models to capture dynamic changes inherent in real-world demand processes.

Cheng et al. (2015) further emphasized that classical stationary forecasting methods, such as ARIMA and linear regression models, generally fall short when dealing with real-world financial dynamics characterized by non-stationary and nonlinear behavior. Their extensive comparative study demonstrated that adaptive, locally flexible forecasting approaches significantly outperform stationary methods, especially in scenarios involving tran-

sient market behaviors and sudden structural shifts. Such findings advocate strongly for integrating adaptive mechanisms and local modeling strategies to enhance predictive accuracy and robustness in the presence of nonstationarities.

Given these insights, future research directions in financial time series forecasting could benefit from developing hybrid models that combine local stationary approximations, deterministic volatility modeling, and adaptive forecasting techniques. Such integrative approaches promise to effectively manage structural changes and volatility dynamics, thereby potentially enhancing forecasting reliability across diverse market conditions and various forecasting horizons (Cheng et al., 2015).

2.2.6 FiLM: Frequency Domain Polynomials: Enhanced Legendre Memory Architecture for Extended Financial Series Prediction

Addressing long-term dependencies and reducing noise sensitivity remain critical challenges in financial time series forecasting. The Frequency Improved Legendre Memory (FiLM) model, introduced by Zhou et al., effectively addresses these issues by integrating Legendre polynomial projections with Fourier transformations. FiLM leverages Legendre polynomials to produce compact yet expressive representations of historical information, significantly improving the model's capability to preserve critical temporal dependencies across extensive forecasting horizons (Zhou et al., 2022a).

A particularly distinctive feature of FiLM is its Frequency Enhanced Layer (FEL), employing Fourier transformations to selectively retain low-frequency signals while filtering out higher-frequency noise. This approach substantially mitigates the impact of noisy fluctuations commonly observed in financial datasets, thus enhancing forecasting accuracy. Zhou et al.'s empirical evaluations clearly demonstrated the FEL layer's effectiveness in improving robustness against noise-induced distortions, making FiLM particularly advantageous for predicting volatile financial series over extended periods (Zhou et al., 2022a).

Empirical validations of FiLM conducted across various benchmark datasets confirmed its superior performance in both univariate and multivariate forecasting scenarios. Compared to advanced baseline models such as Transformer-based architectures and other recursive memory methods (e.g., Legendre

Memory Unit), FiLM consistently reduced forecasting errors by approximately 20% to 25%. These improvements underline the practical effectiveness of incorporating robust frequency-domain analysis within neural network architectures explicitly tailored for long-term forecasting applications (Zhou et al., 2022a).

Temporal Feature-wise Linear Modulation (TFiLM), introduced by Birnbaum et al., is a related derivative architecture that addresses similar long-term dependency challenges. TFiLM combines convolutional activations with adaptive modulations derived from recurrent neural networks, extending the effective receptive field of convolutional models. Although primarily validated in diverse sequential data applications such as text classification and super-resolution, TFiLM's approach to adaptively modulating feature representations offers complementary insights for further enhancing FiLM's potential application in financial forecasting tasks (Birnbaum et al., 2019).

Given FiLM's demonstrated predictive accuracy and robustness against noise, future research directions might explore the integration of FiLM's frequency-domain strengths with adaptive modulation strategies inspired by TFiLM. Hybridizing FiLM with other transformer-based or recurrent architectures and adaptive mechanisms could further enhance the model's applicability to financial markets characterized by complex volatility structures and frequent regime shifts, potentially setting new benchmarks for reliability in long-term financial forecasting (Zhou et al., 2022a; Birnbaum et al., 2019).

2.3 Transformer-Based Approaches for Stock Prediction

In 2017, Vaswani et al. introduced a novel encoder-decoder architecture in their paper *Attention Is All You Need* (Vaswani, 2017). The authors utilized the concept of attention mechanisms to design this architecture, which they named the Transformer. The Transformer model (Vaswani, 2017) efficiently captures complex patterns and long-term contextual information from data streams by leveraging attention and self-attention mechanisms. This architecture enables the model to effectively learn dependencies between words within a paragraph, grasp contextual nuances in an article, and process long-range relationships in textual data.

Attention mechanisms and other traditional approaches use various information-

processing methods. While RNNs and LSTMs process sequence elements in a single sequence, attention mechanisms can focus on specific parts. This property allows this model to identify important information at the beginning of the sequence.

Most major neural sequence transformer models use encoder and decoder structures. The encoder presents a sequence of symbolic representations $(x_1, ..., x_n)$ in this configuration, and transforms them into a sequence of sequential representations denoted as $z = (z_1, ..., z_n)$, and the decoder, given z, can begin to build the output sequence $(y_1, ..., y_m)$ an element onetime by transforming it.

Encoder

The encoder comprises several layers, each with the same underlying structure and each a sub-element of two components in a transformer network: a "neural network for feedforward with position-wise and encoding and a multi-headed mechanism with self-attention".

Firstly, the sequence is transformed into a continuous vector representation; Secondly, it is augmented with position coding.

Decoder

Similarly to the encoder, the decoder of a convolution network consists of several similarly structured layers.

Each layer has a sub-element called the encoder and decoder attention mechanism.

Attention Mechanism

The following equation defines the attention mechanism:

$$Attention(Q, K, V) = softmax_k \left(\frac{QK^T}{\sqrt{d_k}} \right) V \tag{2.4}$$

Here, Q is a query, K is a key, and V is a value.

Multi-Head Attention

Following is the related formula:

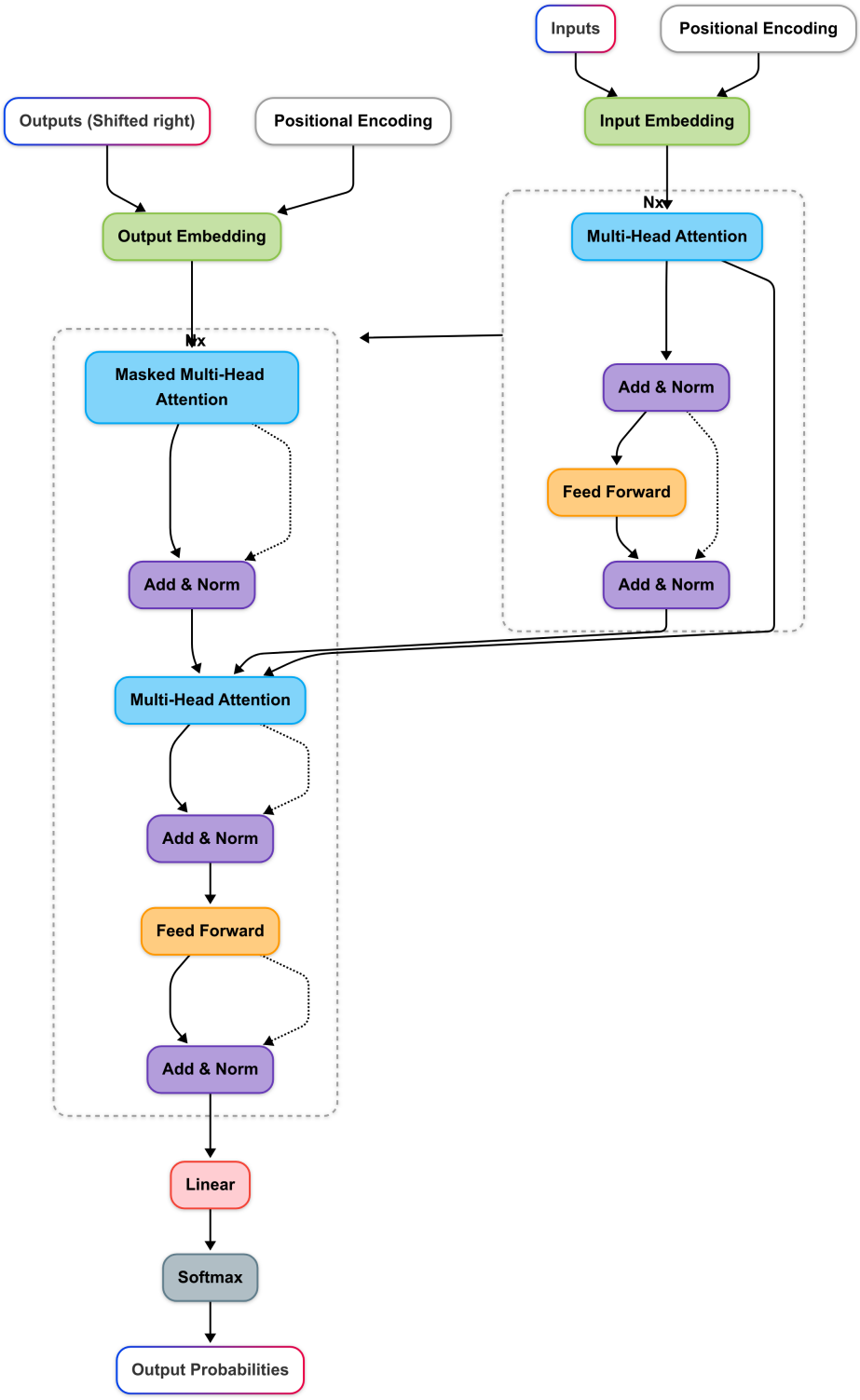


Figure 2.8: Transformers Model Architecture. Adapted from Vaswani (2017)

$$MultiHead(Q, K, V) = Concat(head_1, ..., head_h)W^O \tag{2.5}$$

Here, $head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$.
The projection table is defined as follows (Qiu et al., 2024):
$W_i^Q \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^K \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{model} \times d_v}$, and $W^O \in \mathbb{R}^{hd_v \times d_{model}}$
FFN is defined as follows:

$$FFN(x) = \max(0, xW_1 + b_1)W_2 + b_2 \tag{2.6}$$

Here, the input and output dimensions are defined as $d_{model} = 512$ and the internal dimension as $d_{ff} = 2048$.

Various specialized models have been developed from the basic Transformer architecture for time series forecasting applications, as shown in Fig. 2.9. In the financial sector, Wang et al. (2022) have shown that Transformer models are practical tools for forecasting stock indices. Their application, which includes a multi-headed self-determination mechanism, successfully captured the long-term dependence on stock market data for four major indices: S&P 500, Nikkei 225, Hang Seng, and CSI 300. The results showed superior feature extraction compared to traditional CNN and RNN architectures. Their capabilities have been demonstrated.

Muhammad et al. (2023) developed the transformer model through the way to testing it on the Stock market trade. Experimental results showed that the proposed model exhibited excellent accuracy even for relatively short series; Nonetheless, existing transformer models have difficulty identifying the time dependence of long-term data due to the complexity of the model, and traditional self-consistent transformers are computationally unsuitable for long-term forecasts. Earlier research to solve that problem have focused on enhancing self-fading efficiency at the expense of information utilization (Wu et al., 2021).

2.3.1 Autoformer: A Decomposition-Based Transformer for Long-Term Forecasting

Long-term time series forecasting presents a significant challenge due to the intricate dependencies within financial data and the computational inefficiencies of traditional models. The primary objective in this forecasting context is to predict a future series of length O based on a historical series of length I, commonly referred to as the "input-I-predict-O" paradigm. The complexity

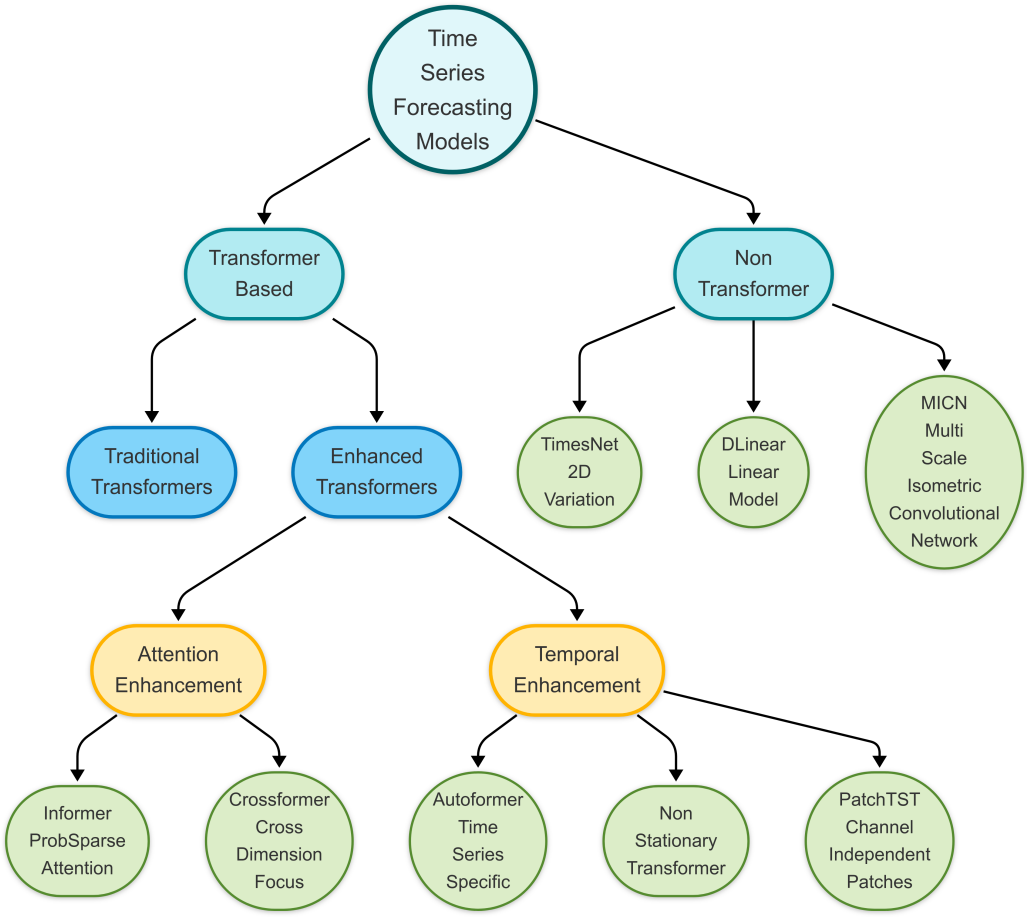


Figure 2.9: Time Series Forecasting Models framework

of capturing long-range dependencies while maintaining computational efficiency has driven the development of novel architectures tailored for financial forecasting.

Wu et al. (2021) presented a new method to address this problem, which is important for climate forecasting and power planning applications: Operating a cascaded decomposition framework for Autoformer with correlation automatically, Autoformer outperformed traditional models and achieved the highest accuracy in many experiments with different real-world datasets.

Additionally, Autoformer integrates an **Auto-Correlation mechanism** that replaces traditional self-attention, identifying dependencies based on periodic patterns rather than token-wise correlations. This modification allows the model to efficiently aggregate similar sub-series from historical sequences, significantly improving its ability to generalize across extended time horizons.

Compared to conventional Transformers, Autoformer reduces computational complexity while preserving forecasting accuracy, making it a viable model for stock market prediction. By incorporating Auto-Correlation and decomposition techniques, Autoformer provides a robust framework for capturing both short-term fluctuations and long-term market trends.
The model architecture, as illustrated in Fig. 2.10, consists of an encoder-decoder structure where both components leverage decomposition and Auto-Correlation mechanisms to enhance feature extraction. This methodological advancement enables Autoformer to efficiently process extensive financial datasets while maintaining predictive reliability.

The ability to model financial time series accurately is crucial for stock price prediction, particularly when dealing with non-stationary market behaviors. Unlike traditional deep learning models, which often struggle with long-range dependencies, Autoformer provides a scalable and interpretable approach to financial forecasting. The decomposition mechanism ensures that trends and seasonal variations are explicitly modeled, while the Auto-Correlation mechanism enhances long-term dependency capture without excessive computational overhead.

This study integrates Autoformer within a broader forecasting framework, combining its decomposition-based insights with Convolutional Neural Networks (CNNs) for spatial feature extraction and Large Language Models (LLMs) for financial technical analysis. By leveraging Autoformer's ability to model periodic financial patterns, we aim to enhance the accuracy and robustness of stock price predictions.

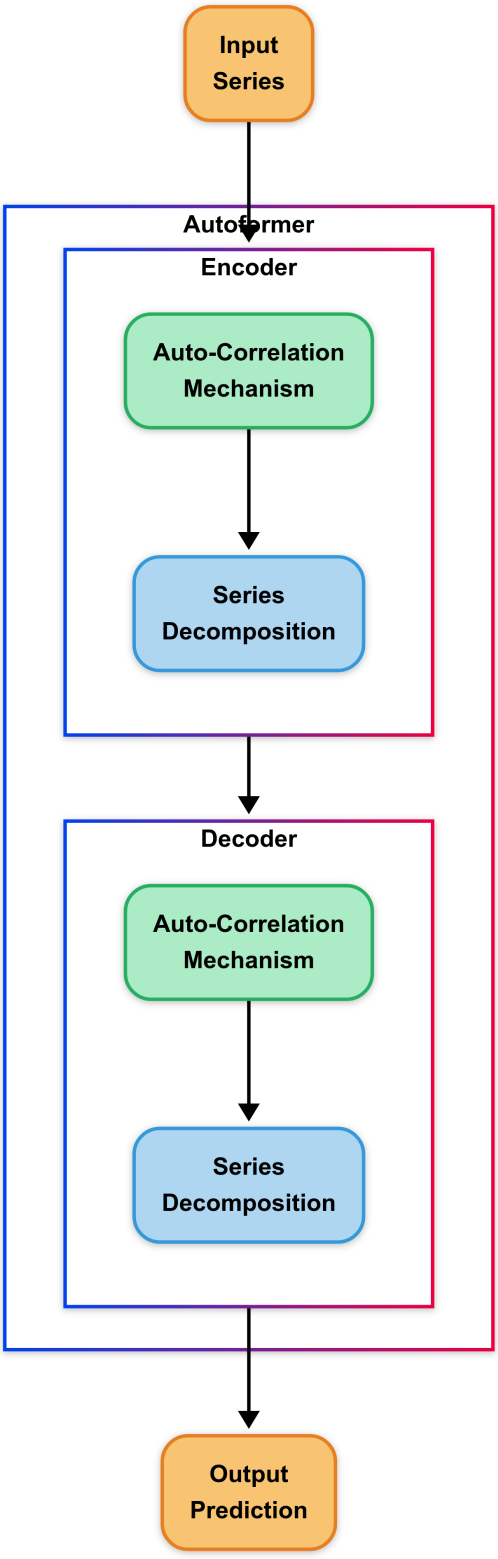


Figure 2.10: Autoformer overall architecture. Adapted from Wu et al. (2021).

2.3.2 Crossformer: A Transformer for Capturing Cross-Dimensional Dependencies in Multivariate Time Series

From Zhang and Yan (2023), it can be seen that when predicting multivariate time series, it is essential to consider both temporal and cross-dimensional dependence because many existing models usually only focus on temporal dependence and ignore the dependence between different variables (dimensions). To address this issue, the authors propose a **Crossformer** model. This transformer-based model combines dimension segments to reflect temporal and dimensional information and uses a two-stage approach to effectively capture intertemporal and interdimensional dependencies.

The **Hierarchical Encoder-Decoder (HED)** architecture, as illustrated in Figure 2.11, leverages a multi-level encoding mechanism where each layer progressively merges adjacent time segments. The encoder extracts both temporal and cross-dimensional dependencies using the **Two-Stage Attention (TSA)** layer, allowing for a better understanding of interdependencies within multivariate time series data. The decoder then aggregates the learned representations from multiple scales, improving long-term forecasting accuracy by incorporating both fine-grained and coarse-grained information.

This structured approach enables **Crossformer** to capture complex dependencies in financial time series forecasting, significantly outperforming traditional transformer-based models that primarily focus on temporal dependencies without explicitly modeling cross-dimensional relationships (Zhang and Yan, 2023).

Liu et al. (2022) addressed the problem of super-stationarity inaccurate data. Using stabilization methods to improve data predictability may inadvertently remove important information that is irrelevant to stabilization. To address this problem, they proposed a transformaer non-fixation framework. Two main elements are series-fixing to normalize the input data, Series fixation to normalize the time series without adding additional parameters and interval attention to approximate the time dependence of the original unfixed data. This approach improves the performance of the transformer model and significantly reduces the mean square error. Significantly reduces the mean square error of various benchmark tests of real-time time series forecasts. Channel-independent PatchTST (Nie et al., 2022) uses patches for possessing related transformers. Segmented patches can be used at the

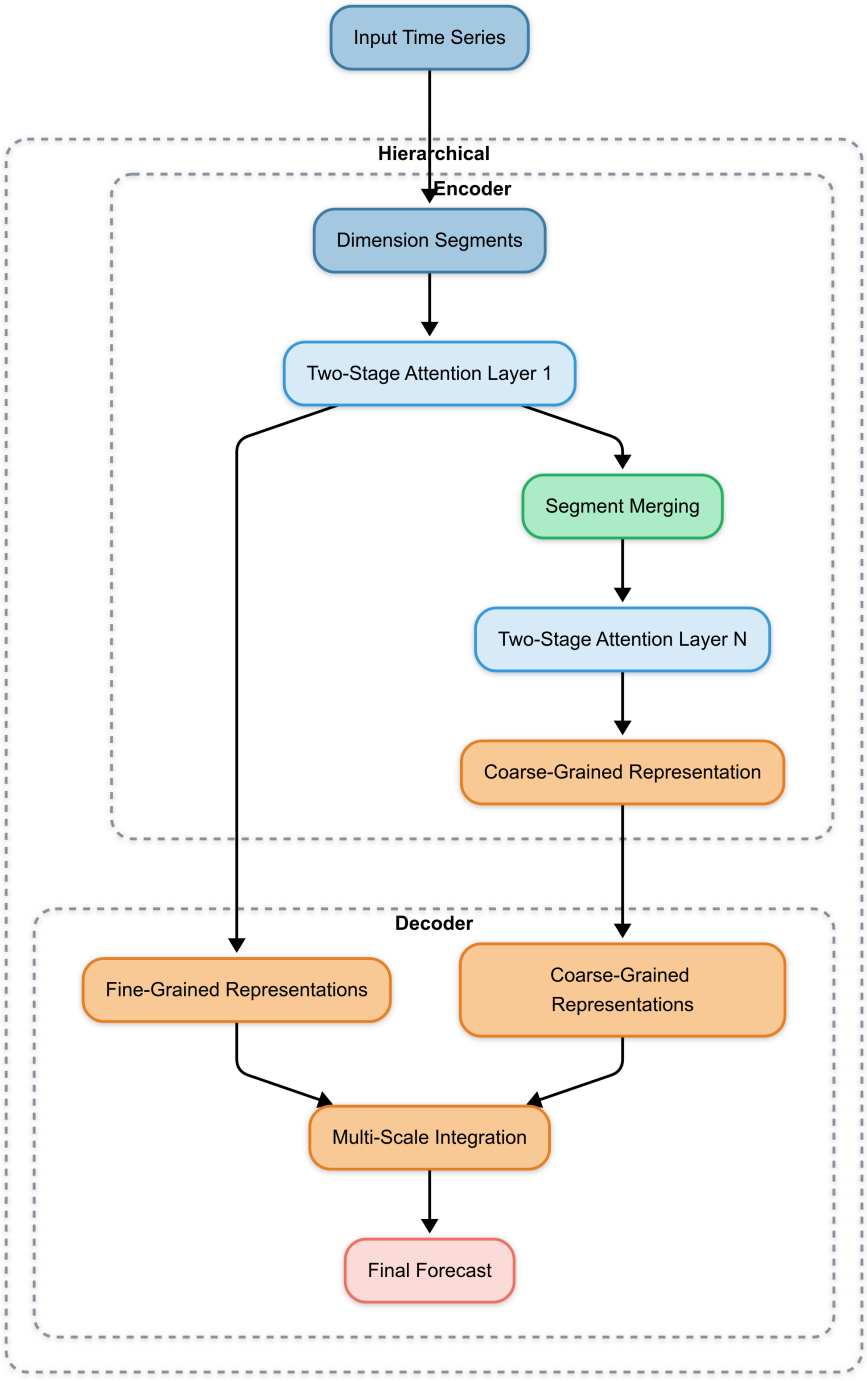


Figure 2.11: Architecture of the Hierarchical Encoder-Decoder in **Crossformer**. Adapted from Zhang and Yan (2023). The encoder processes input time series through dimension segments and employs **Two-Stage Attention (TSA)** layers with segment merging to create multi-scale representations, where higher layers capture longer time ranges with increasingly coarse-grained information. The decoder integrates both fine-grained and coarse-grained representations from different encoder layers to generate the final forecast, effectively capturing both temporal and cross-dimensional dependencies in multivariate time series data.

sub-series level to gather local semantic information and improve the connectivity between data points. PatchTST also offers channel independence. Channel independence means that each entry point contains data from a single channel. A single channel contains different data from the channel mix. This innovation reduces temporal and spatial complexity and enables more efficient learning, especially for large datasets. PatchTST also facilitates learning in larger recursive windows. Thus, prediction accuracy increases.

2.3.3 PatchTST: Channel-Independent Transformer for Financial Time Series Prediction

Transformer-based architectures have evolved significantly, with the Patch Time Series Transformer (PatchTST) emerging as a recent and notable development in stock price forecasting. Fan and Zhang introduced PatchTST, highlighting its channel-independent design explicitly tailored for efficiently modeling intricate temporal patterns within financial time series data (Fan and Zhang, 2024).

The primary innovation of PatchTST lies in its "patching" mechanism, whereby lengthy sequences of stock data are segmented into smaller, independent sub-sequences or patches. Each patch is treated as a distinct input token for the Transformer, enabling efficient handling of extensive sequences while effectively capturing both short-term fluctuations and long-term temporal dependencies (Huang et al., 2024). Huang et al.'s research extended the applicability of PatchTST beyond finance, illustrating its robust capability for long-sequence forecasting tasks, specifically demonstrated through superior performance in ocean wave height predictions over extensive horizons (up to 720 hours) compared to established models like Informer and LSTM (Huang et al., 2024).

Further innovations were proposed by Ahmad et al. through the development of PatchFusionBERT, a hybrid model integrating PatchTST with Bidirectional Encoder Representations from Transformers (BERT) layers. This model enhanced the local temporal pattern extraction capability of PatchTST by combining it with BERT's strengths in modeling global contextual relationships and long-term dependencies. Empirical evaluations demonstrated that PatchFusionBERT significantly outperformed standard PatchTST in forecasting tasks, particularly in scenarios requiring precise and robust modeling of long-term horizons, also addressing computational com-

plexity challenges commonly associated with traditional transformer-based models (Ahmad et al., 2024).

Empirical findings by Fan and Zhang indicated that PatchTST provides competitive but context-sensitive forecasting accuracy. Specifically, the model did not consistently outperform simpler recurrent architectures, such as standard Long Short-Term Memory (LSTM), especially in real-time or near-term stock price nowcasting contexts. This relative vulnerability was attributed to PatchTST's increased sensitivity to high-frequency market noise, underscoring the importance of carefully considering temporal granularity when deploying this model (Fan and Zhang, 2024).

In addressing feature selection, Xue et al. identified that the predictive performance of PatchTST significantly depends on the careful integration of multivariate inputs. Their research revealed that variables such as opening price, highest price, and lowest price notably improved the accuracy of predictions, while incorporating variables like trading volume could sometimes negatively impact model effectiveness. Consequently, strategic feature selection is vital to optimizing PatchTST's performance in financial time series forecasting scenarios (Xue et al., 2024).

Given PatchTST's promising yet variable performance, future research could beneficially explore further hybrid architectures. Combining PatchTST's detailed temporal feature extraction strengths with additional noise-resilient or adaptive modeling techniques, such as variational mode decomposition or adaptive scale-weighted layers, may refine predictive performance across various financial market contexts, enhancing the practical utility and reliability of transformer-based models for stock price prediction tasks (Xue et al., 2024).

2.3.4 Informer: Enhancing Long-range Dependency Modeling for Stock Price Prediction

Frameworks built upon transformer principles have gained significant recognition for their effectiveness in identifying distant chronological relationships within sequential information sets—a crucial requirement for accurate securities valuation projections. Among these, the Informer model, introduced by Zhou et al., extends the capabilities of conventional transformer architectures by addressing the computational limitations that restrict their efficiency in handling long sequences (Lu et al., 2023; Zhou et al., 2021; Yulistiani and

Kurniadi, 2024).

Informer leverages the ProbSparse self-attention mechanism, significantly reducing the computational complexity traditionally associated with transformers. Specifically, the Informer model reduces computational complexity from $O(L^2)$ to $O(L \log L)$ by selectively emphasizing the most relevant attention relationships, thereby enhancing efficiency and performance in handling extended time series data (Zhou et al., 2021). Zhao (2024) further emphasized that Informer effectively mitigates the memory bottleneck issue commonly found in traditional Transformer architectures, making it particularly suited for scenarios requiring rapid prediction of intraday stock price fluctuations, where computational resources are constrained (Zhao, 2024). Additionally, recent studies indicate that Informer's effectiveness can vary significantly depending on market volatility and dataset characteristics, emphasizing the importance of careful selection and preprocessing of data when applying Informer to financial time series prediction (Yulistiani and Kurniadi, 2024).

In empirical evaluations, Informer demonstrated superior predictive performance compared to conventional recurrent models like LSTM, especially in tasks involving intraday stock price forecasting. Using minute-level K-line data from the Shanghai Composite Index, Informer consistently achieved lower prediction errors measured by common metrics such as Mean Absolute Error (MAE), Mean Square Error (MSE), Mean Absolute Percentage Error (MAPE), and Root Mean Square Error (RMSE). This confirms its robustness in scenarios characterized by high-frequency, non-linear, and volatile data typically found in financial markets (Lu et al., 2023; Yulistiani and Kurniadi, 2024).

Moreover, the Informer model's enhanced capability to handle noisy financial data, a common challenge in stock prediction tasks due to the low signal-to-noise ratio, significantly improves its applicability for real-world intraday market analysis. By accurately capturing essential features from large volumes of noisy intraday stock market data, Informer exhibits considerable promise for future research and practical implementation in stock price prediction systems (Lu et al., 2023).

While Informer has been predominantly applied to areas such as energy forecasting, its recent application in intraday financial prediction tasks highlights its versatility and adaptability. Future research opportunities lie in exploring Informer's potential within various financial market contexts, possibly in combination with other neural network architectures, to further leverage

its ability to model long-term dependencies effectively and efficiently (Zhou et al., 2021; Yulistiani and Kurniadi, 2024).

2.3.5 Linear Self-Attention Transformers in Financial Applications

Background: Linear Self-Attention in Finance

Transformers' self-attention normally scales quadratically with sequence length, which is problematic for long financial time series. To address this, **linear self-attention** techniques have emerged, reducing complexity to **O(n)** (linear in sequence length) by approximating or restructuring the attention mechanism. These efficient Transformer variants have gained traction in financial forecasting over the last few years (Xu et al., 2023; Dixon et al., 2020).

By using linearized attention, models can capture long-range dependencies in stock data without the heavy computational cost, rendering them particularly appropriate for securities valuation forecasting, automated trading algorithms, and related financial sequence tasks. Research has demonstrated that linear-attention Transformers sometimes even **outperform vanilla Transformers** on time-series forecasting (Lu and Yang, 2025).

Efficient Transformer Models for Financial Time-Series

Several academic works since 2019 have applied linear-attention Transformer architectures to financial prediction, introducing both theoretical advances and practical models:

Fourier-Based Linear Transformers A method for replacing the standard multi-head attention with a Fourier transform module has been proposed, achieving linear-time complexity *without* adding extra parameters (Xu et al., 2023). This **linear Transformer** (with a "multiplexed" attention mechanism) significantly reduces the original Transformer's complexity to linear time while preserving its ability to capture long-term patterns. When applied to financial data (including stock indices), it demonstrated **faster training and inference** and improved long-horizon forecasting accuracy compared to a full Transformer. Studies have also shown that this approach

results in higher prediction accuracy, faster inference speed, and fewer operations than the standard Transformer, making it especially beneficial for noisy, lengthy financial time series (Xu et al., 2023).

Kernel-Based Linear Transformers A common approach to improving Transformer efficiency is to approximate the softmax attention using **kernel feature maps** (Araci, 2019). By replacing the standard attention mechanism with a kernelized attention that factorizes the attention matrix, computation complexity is significantly reduced while maintaining long-range dependency capture. A linear-transformer model applied to S&P 500 stock data has been shown to **outperform a standard Transformer**, achieving a lower RMSE and MAE, thereby demonstrating that efficiency gains do not degrade accuracy (Araci, 2019).

Sparse-Attention Transformers (Informer & Variants) Recent adaptations of **Informer**-style Transformers for financial applications leverage **ProbSparse attention**, selecting only the most "informative" queries for attention calculation, which helps reduce redundant computations (Zhou et al., 2021). In a stock trading model, ProbSparse self-attention achieved comparable or better forecasting accuracy than full attention while significantly improving speed. More recently, a Transformer for long-term financial series was introduced, modifying the Informer's encoder with a *decentralized sparse attention* mechanism to further optimize computational efficiency (Zhang and Duan, 2023).

Hybrid Models with Linear Attention Several models integrate linear-attention Transformers with other deep learning architectures, such as CNNs, graph networks, or LLMs, to leverage different modalities of financial data. A combined methodology integrating linearized attention (for chronological information processing) with convolutional analysis for stock price chart feature extraction has demonstrated improved forecasting accuracy (Bhattacharjee and Bhattacharja, 2019). Additionally, using an LLM (such as a ChatGPT-based model) to generate technical indicators, which are then incorporated into the Transformer framework, has shown further accuracy gains in financial market prediction.

Theoretical Insights – Transformers as VAR The connection between linear-attention Transformers and classical financial time-series models has been explored in recent work. A **single linear attention layer can be interpreted as a dynamic Vector Autoregressive (VAR) model**, a standard econometric model in finance (Lu and Yang, 2025). By incorporating VAR-like weights into a linear Transformer, this approach enhances forecasting performance and interpretability while maintaining computational efficiency.

Performance vs. Traditional Transformers

Importantly, linear-attention Transformers in finance exhibit comparable or superior accuracy to traditional Transformers while being more efficient. In long-horizon stock index forecasting, a linear Transformer-based approach outperformed classical econometric models (ARIMA, VAR, GARCH) in accuracy, while also handling issues like missing data and outliers more robustly (Tsay, 2005). Compared to deep learning baselines such as MLPs, SVMs, or even LSTM-CNN hybrids, linear-attention models also train **significantly faster**, allowing for more extensive backtests and hyperparameter tuning.

Efficient Attention Mechanisms & Accuracy Trade-offs

Linear self-attention models achieve efficiency improvements through various techniques, all designed to scale Transformer architectures to long sequences more effectively:

- **Kernel-Based Approximation:** Using positive-definite kernel functions to rewrite the attention computation as dot-products in a feature space eliminates the explicit $O(n^2)$ attention matrix. This **kernel trick** yields linear complexity in sequence length (Araci, 2019).
- **Fourier/Spectral Methods:** Replacing attention mechanisms with Fourier transforms sidesteps computationally expensive self-attention calculations (Xu et al., 2023).
- **Sparse and Segmental Attention:** Techniques such as ProbSparse (Informer) selectively compute attention for only high-value interactions, significantly improving efficiency (Zhou et al., 2021).

- **Model Simplification (Parameter Efficiency):** Many linear-attention financial models utilize *fewer parameters* than a vanilla Transformer, reducing computational costs while maintaining strong predictive accuracy (Zhang and Duan, 2023).

These findings suggest that efficient Transformer variants enable practical applications of deep learning in finance by reducing computational overhead while maintaining predictive accuracy. The continued evolution of **low-rank, kernelized, and sparse attention mechanisms** is expected to drive further improvements in stock market forecasting.

2.4 Large Language Models Approaches for Stock Prediction

Recent products in LLMs have revolutionized financial analysis through various capabilities and applications, as illustrated in Fig.2.12. In the context of stock market prediction, models such as ChatGPT-4, DeepSeek, FinGPT have shown considerable potential for market forecasting by interpreting sophisticated textual information from news sources, expert analyses, and social discourse sentiment (Brown, 2020). LLMs have been shown to perform well in analyzing financial textual data with a small number of training examples (Brown, 2020). These models are well suited for processing complex financial statements and generating technical indicators (Wu et al., 2023). The emergence of specialized models such as FinBERT further enhances the ability to integrate textual data into forecasting systems (Halder, 2022). Such AI systems exhibit exceptional proficiency when handling enormous volumes of unorganized information, producing financial metrics and evaluating market emotion patterns - analytical functions where traditional computational approaches have historically struggled (Halder, 2022).

More recently, reinforcement learning (RL) has been integrated into LLM training processes to significantly enhance their reasoning capabilities in financial decision-making tasks. For instance, DeepSeek-R1 utilizes reinforcement learning to incentivize reasoning ability, achieving superior predictive accuracy compared to conventionally fine-tuned models (Guo et al., 2025; Chen et al., 2025). Studies comparing RL-driven LLMs to standard supervised learning approaches have consistently demonstrated their advantages

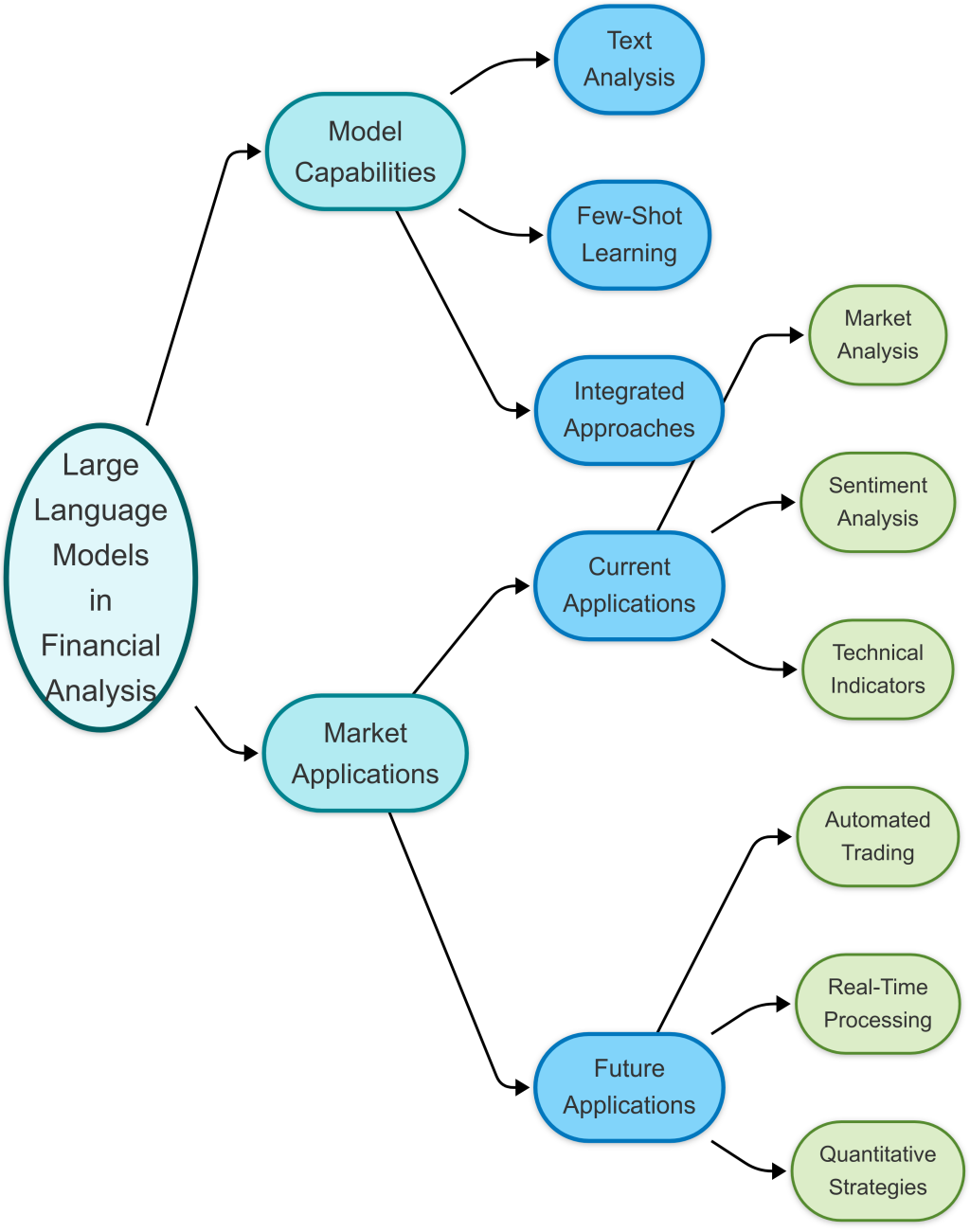


Figure 2.12: Large Language Models in Financial Analysis

in adapting dynamically to evolving financial contexts, crucial for capturing shifts in market sentiment and economic conditions (Gao et al., 2025).

A key practical advantage of DeepSeek-R1 for stock prediction is its computational efficiency. While models such as OpenAI's GPT-4 (Achiam et al., 2023) or Anthropic's Claude (Kumamoto et al., 2023) may excel in benchmark tasks, DeepSeek achieves competitive performance with considerably lower computational costs (Krause, 2025; Gao et al., 2025). Its specialized architectural optimizations make it especially suited for resource-constrained environments, reducing barriers to deployment in real-time or high-frequency trading applications.

Incorporating narrative insights produced by generative AI within equity forecasting frameworks represents a significant advancement area. For example, Deng et al. demonstrated improved stock trend prediction capabilities using few-shot learning approaches with LLMs, highlighting their effectiveness during volatile market phases (Deng et al., 2024). Similarly, Zhu et al. developed an integrated approach combining LLM-based sentiment insights with traditional technical analysis, achieving improved forecasting accuracy during uncertain market conditions (Zhu et al., 2024). Such research highlights how advanced language systems can produce complex, metric-oriented textual analyses using solely structured financial inputs, thus minimizing reliance on potentially distorted external data channels.

Our investigation employed DeepSeek-R1 for creating compact technical parameter summaries extracted directly from structured economic information, including historical pricing data and established market analytical parameters throughout our experimental framework. This approach contrasts with traditional uses of LLMs that primarily rely on unstructured external textual data, and it allows our model to directly leverage textual summaries as additional input features embedded through FinBERT embeddings (Araci, 2019). The structured prompts used in our implementation ensure that DeepSeek-R1 generates clear, concise ($\leq$20 words) analyses, offering interpretability and predictive value. Such a structured, indicator-driven LLM integration aligns closely with current trends in automated financial analysis, providing human-readable, insightful textual data without significant computational overhead.

However, despite these promising advancements, deploying LLM-based stock prediction models still faces certain challenges. Issues of interpretability, computational demands, potential biases, and ethical concerns regarding algorithm-driven financial decisions remain critical areas requiring further

exploration (Makridakis et al., 2018; Dixon et al., 2020). Addressing these issues—particularly the interpretability of LLM-generated outputs and ensuring their robustness against market regime shifts—will be essential steps toward broader adoption in financial prediction tasks.

The groundbreaking research by (Brown, 2020) demonstrated LLMs' exceptional performance in few-shot learning scenarios, showing their ability to analyze financial textual data with minimal training examples effectively. This advancement has been particularly significant in the financial domain, where data patterns can be subtle and complex. Building on this foundation, specialized models like FinBERT have been developed for financial sentiment analysis, proving especially effective at incorporating non-numerical data into stock prediction frameworks (Darapaneni et al., 2022). Research has shown significant improvements in prediction accuracy when combining sentiment analysis with traditional neural networks for predicting short-term price movements.

Recent research has further expanded LLM applications in stock prediction:

- **Enhanced Few-Shot Learning:** (Deng et al., 2024) has demonstrated how LLMs can significantly improve stock trend prediction in few-shot learning scenarios, particularly valuable in rapidly changing market conditions.
- **Integrated Approaches:** (Zhu et al., 2024) developed novel methods combining LLM-based sentiment analysis with traditional technical analysis, showing superior performance in uncertain market conditions.
- **Technical Analysis:** LLMs have demonstrated capabilities in generating sophisticated technical indicators directly from market data, reducing dependence on noisy external information sources.

Integrating LLMs with traditional stock price prediction methods significantly advances financial forecasting. These frameworks demonstrate exceptional capacity processing textual information through approaches that established analytical systems frequently require assistance achieving. The ability to understand and process complex financial narratives, combined with traditional market data analysis, offers a more comprehensive approach to stock price prediction.

Looking forward, the potential of LLMs in financial markets continues to expand, with emerging applications in:

1. **Automated Analysis:** Generation of technical indicators and automated analysis
2. **Real-time Processing:** Market sentiment analysis in real-time
3. **Trading Integration:** Implementation within quantitative trading strategies

This fusion of LLM capabilities with traditional financial analysis techniques marks an important evolution in stock market prediction methodologies (Halder, 2022). The continued development of these integrated approaches promises to enhance the accuracy and reliability of financial forecasting systems.

2.5 Hybrid Models Approaches for Stock Prediction

As stock market data continues to grow in complexity, researchers have turned to hybrid models that combine multiple architectures to improve predictive performance. Among these, CNN-LSTM and CNN-Transformer integrations have demonstrated significant advantages by leveraging both spatial and temporal feature extraction.

2.5.1 Hybrid CNN Models Approaches

The combination of CNNs with other deep learning architectures has become increasingly common. Notable approaches include:

- CNN-LSTM hybrids that leverage both spatial and temporal features
- CNN-Transformer architectures that combine local pattern recognition with global dependency modeling
- Multi-modal systems that process both numerical time series and visual representations

These hybrid approaches have consistently outperformed single-architecture models, particularly in volatile market conditions where multiple types of patterns need to be considered simultaneously.

2.5.2 CNN-LSTM Integration

Hybrid models, particularly CNN-LSTM architectures, effectively integrate feature extraction and sequential modeling, enhancing stock price forecasting. The CNN component identifies local spatial patterns, such as short-term price trends, while the LSTM component captures long-range temporal dependencies, improving predictive accuracy (Bhooshan and Hari, 2021; Eapen et al., 2019). This architecture has been widely adopted due to its ability to process diverse financial data formats efficiently (Kanwal et al., 2022).

Research comparing hybrid models, such as CNN-LSTM, with other approaches has demonstrated their superior performance in handling the complexity of stock market data (Ding, 2023). Studies highlight that CNN-LSTM models outperform both individual CNN and LSTM models in predictive accuracy(Li et al., 2019). These findings underscore the benefits of hybrid architectures in extracting more nuanced insights from financial time series.

Studies have demonstrated that combining technical analysis with deep learning techniques significantly enhances stock prediction performance (Ding, 2023). Furthermore, incorporating bidirectional LSTM (BI-LSTM) layers improves prediction by leveraging both past and future information, enriching the contextual understanding of market trends (Chen et al., 2021). These advancements make CNN-LSTM hybrids a powerful tool in financial forecasting.

2.5.3 CNN-Transformer Integration

Building on the success of CNN-LSTM models, researchers have explored Transformer-based architectures as an alternative approach for stock market prediction. The integration of CNN and Transformer architectures has emerged as a promising direction for enhancing stock price prediction, with various approaches and methodologies as shown in Fig.2.13. The foundational work by (Dosovitskiy, 2020) demonstrated that transformers, when properly adapted, can effectively process image data by treating it as a sequence of patches. This insight has inspired various hybrid approaches in financial forecasting that combine the strengths of both architectures.

While early attempts focused on either pure CNN or pure Transformer approaches, recent studies have shown that hybrid architectures can better capture the multi-faceted nature of financial data. As surveyed by (Khan et al., 2023), the combination of CNNs and Transformers has led to signif-

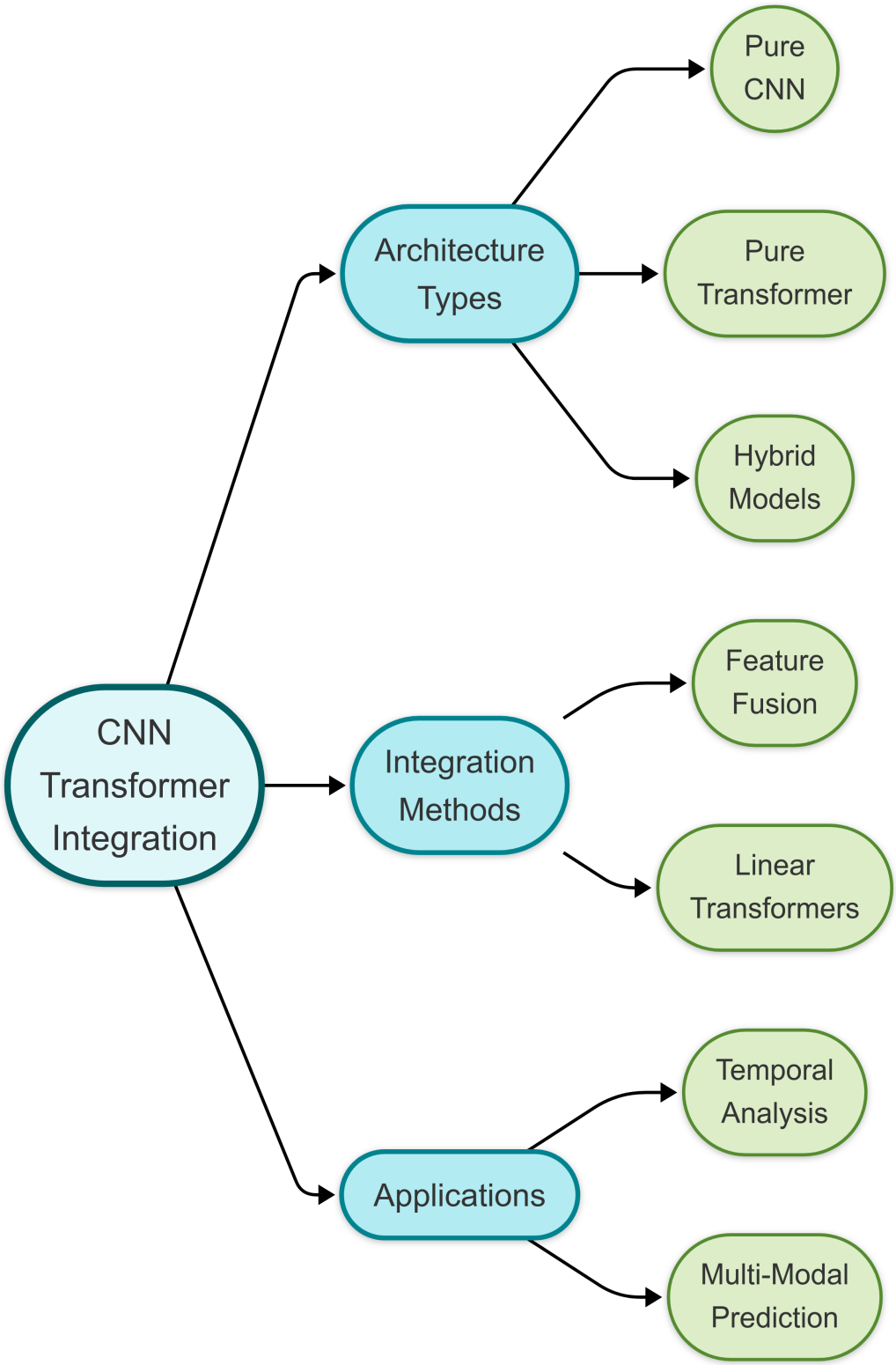


Figure 2.13: CNN-Transformer Integration

icant improvements across various tasks. For example, (Li et al., 2020b) demonstrated how such hybrid architectures could be modified to process both temporal and spatial data simultaneously in a complementary manner through a novel CNN-transformer design for multitemporal analysis.

Recent advances in multi-modal prediction have further expanded this paradigm by incorporating Large Language Models (LLMs) alongside CNN-Transformer architectures. A notable contribution in this direction is the development of Linear Transformers (Sharir et al., 2021), which address the computational complexity issues of standard attention mechanisms while maintaining their modeling capacity. As shown in (Yuan et al., 2023), careful integration of CNN and transformer components can lead to effective complementary architectures that take advantage of the strengths of both approaches.

Recent work has shown that the fusion of these streams through carefully designed architectures can yield superior results compared to single-modality approaches. For instance, (Li et al., 2020b) and (Yuan et al., 2023) demonstrated that combining CNN features with transformer outputs could achieve significant improvements in accuracy through their complementary nature.

Zhou et al. (2025) introduced several innovations in CNN-Transformer integration for stock price prediction. Their novel approach combines a computationally efficient Linear Transformer, a specialized CNN for processing candlestick charts, and LLM-generated technical analyses vectorized through FinBERT. These three branches are integrated using a weighted fusion strategy. This multimodal architecture achieved substantial improvements over baseline models, reducing RMSE by 61.3% compared to single-modality approaches, demonstrating the effectiveness of their CNN-Transformer integration strategy.

The loss functions employed in these hybrid architectures have also evolved, with robust formulations gaining prominence due to their ability to handle inherent noise. These developments represent a significant advancement over traditional approaches, as highlighted by (Khan et al., 2023).

This integration trend aligns with the broader movement in deep learning toward multi-modal, hybrid architectures. As (Dosovitskiy, 2020) showed for computer vision tasks, the combination of complementary architectural components can lead to more robust and accurate models. Extending these principles to the financial forecast is a promising approach for future study in finance.

2.6 Summary of Literature Review Findings

The development of stock price forecasting models has evolved from simple statistical methods to sophisticated deep learning architectures. Recent advances in transformer-based models and multimodal integration techniques have provided promising results, but balancing model complexity and prediction accuracy remains a challenge (Wu et al., 2023). Our proposed framework addresses this challenge by integrating LLM, linear transformer, and CNN into a novel architecture optimized for stock price forecasting.

Chapter 3

Methodology

3.1 Introduction

After conducting an extensive review of recent advancements in financial forecasting, several limitations within existing methodologies became evident. First, traditional models, such as ARIMA and GARCH, inadequately handle complex, non-linear market dynamics and long-term dependencies. Second, contemporary transformer-based models, though proficient at capturing temporal relationships, suffer from significant computational overhead when dealing with extensive historical datasets. Third, existing approaches to technical analysis rely heavily on predefined static indicators, limiting their adaptability to rapidly changing market conditions and nuanced financial signals.

Our comprehensive literature review has identified three key technological components that could be integrated to address these gaps (Kristanti et al., 2024):

- **Convolutional Neural Networks (CNNs):** Excellent at extracting spatial features and detecting local patterns in stock charts. They are particularly effective at recognizing short-term market trends through visual representations of financial data.

- **Transformers:** Capable of modeling complex temporal dependencies but often face computational challenges with long sequences. Linear self-attention transformers can capture long-term dependencies while significantly reducing computational complexity (Zeng et al., 2023).

- **Large Language Models (LLMs):** Typically used for sentiment analysis in news and social media, but we propose applying them innovatively to technical analysis. This approach enables creation of complex market metrics derived solely from trading information, eliminating distortion commonly present when incorporating external data sources (Wu et al., 2023).

To effectively address these critical challenges, this research proposes an innovative, multimodal hybrid deep learning framework. Specifically, our model integrates a Convolutional Neural Network (CNN), an optimized linear Transformer, FinBERT embeddings, and a Large Language Model (LLM) into a cohesive forecasting architecture, as illustrated in Figure 3.1. The CNN is employed to identify and extract spatial patterns embedded within visual representations of financial data, such as candlestick charts. Concurrently, the linear Transformer component captures essential long-range dependencies in historical time-series data while mitigating the prohibitive computational complexity associated with conventional transformers. Additionally, we leverage an LLM augmented with FinBERT embeddings to dynamically generate sophisticated, context-aware technical indicators directly from structured market data, circumventing the noise typically introduced by external sentiment sources.

Our approach utilizes a multi-stream architecture and feature fusion mechanism to process stock market data in a fully autonomous manner. The framework analyzes stock history data through three parallel streams:

1. A text processing pipeline using LLM and FinBERT for technical analysis
2. A price data pipeline with linear self-attention transformer for temporal pattern recognition
3. An image processing pipeline with CNN for spatial feature extraction

The technical innovations include a linearized self-attention mechanism that improves computational efficiency while retaining the ability to capture long-term correlations, feature extraction using LLM-based technical analysis that focuses on market data to avoid external noise, and an effective fusion mechanism that combines CNN-based spatial features and transformer-based temporal modeling.

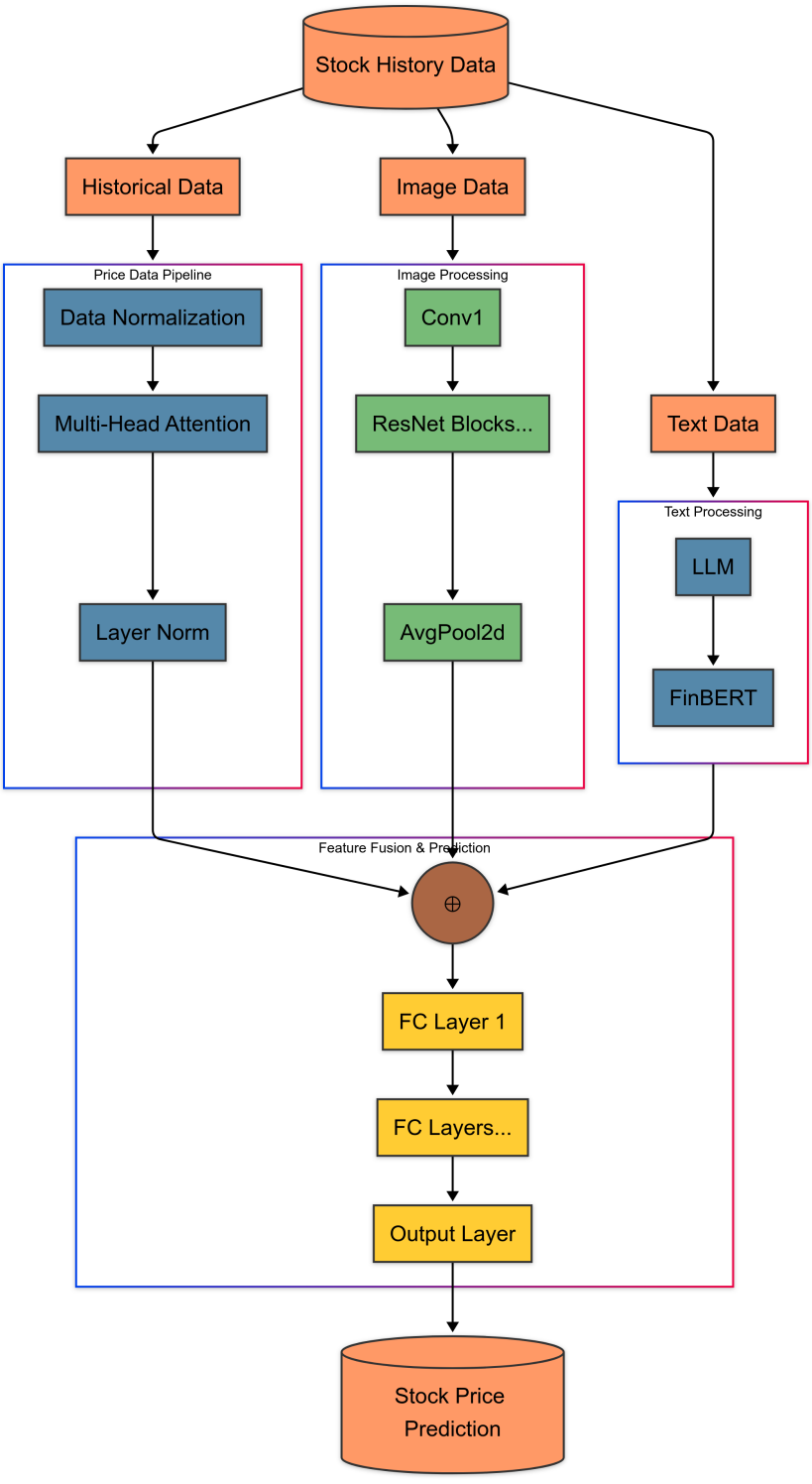


Figure 3.1: Architecture of the proposed LLM-enhanced linear self-attention transformer-CNN hybrid framework for stock price prediction.

The proposed framework integrates complementary strengths from each modality through a carefully engineered architecture. Figure 3.2 visually outlines our systematic methodological approach, highlighting the distinct phases of data preparation, multimodal model development, and evaluation strategy. This pipeline illustrates how the different components interact within our experimental framework.

The detailed procedures for each of these phases, including precise methods of data partitioning, model training, hyperparameter optimization, and comprehensive model validation, are described thoroughly in the subsequent sections.

In summary, by synthesizing spatial, temporal, and contextual insights into a unified predictive model, this research directly addresses key limitations of existing methodologies and advances the state-of-the-art in financial stock forecasting.

3.1.1 Research Methodology Framework

Our framework begins with Data Acquisition and Preparation phases, where we collect historical stock market data and process it into suitable formats for our deep learning models. This initial stage involves data cleaning, normalization, and feature extraction from various data sources including historical price data, trading volumes, and technical indicators. Following data preparation, we implement comprehensive Data Processing to handle missing values and ensure dataset balance, which is crucial for maintaining data quality and preventing bias in our models. The Data Partitioning phase follows, where we divide our dataset into three distinct sets: Training (70%) for model learning and Hyperparameter tuning, Testing (20%) for performance evaluation, and Validation (10%) for hyperparameter tuning and model selection. The construction phase incorporates implementation of predictive systems with corresponding configuration adjustments, combining the capabilities of Linear Transformer technology for sequence examination, Convolutional Networks for identifying spatial patterns, and Large Language Model-augmented attribute processing. Finally, the Model Evaluation phase assesses the performance of our hybrid model using multiple metrics to ensure robust and reliable predictions. Each stage is designed to maintain data integrity and model performance throughout the process, ensuring reproducibility and reliability of our results.

As illustrated in Figure 3.3, this research adopts a comprehensive method-

Figure 3.2: Experimental Pipeline for LLM-Enhanced Hybrid Deep Learning Framework in Stock Price Prediction

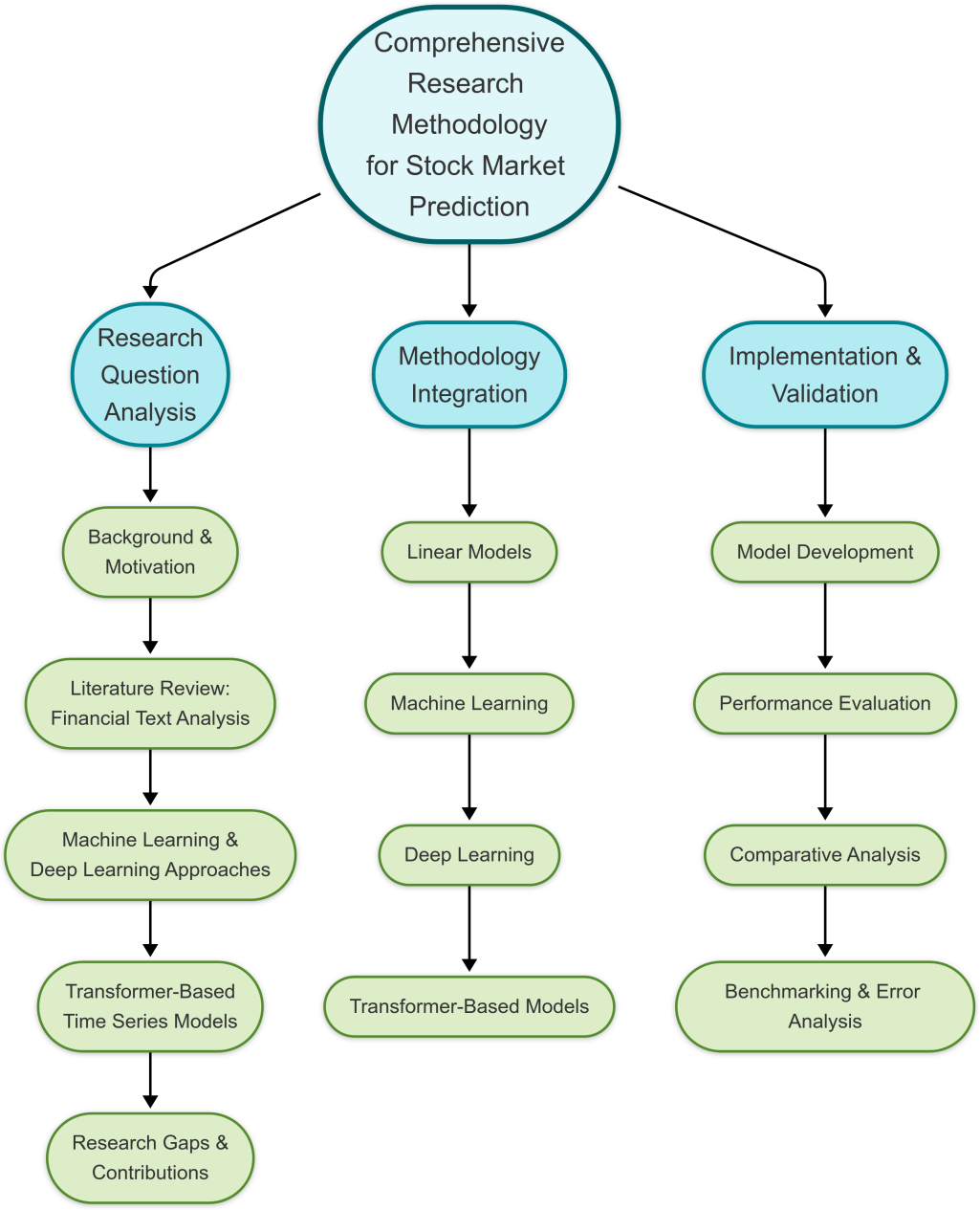


Figure 3.3: Comprehensive Research Methodology Framework for Stock Market Prediction

ological approach to stock market prediction leveraging advanced computational techniques. The methodology encompasses a structured framework integrating multiple innovative strategies:

- **Research Question Analysis:** Systematic exploration of background, literature review, and identification of research gaps
- **Methodology Integration:** Synthesizing multiple prediction techniques including:
  1. Linear models
  2. Machine learning approaches
  3. Deep learning architectures
  4. Transformer-based forecasting models
- **Implementation and Validation:** Experimental design, performance evaluation, and comprehensive comparative analysis

3.1.2 Methodological Approach

This research employs a systematic methodological approach aligned closely with the overall research framework illustrated in Figure 3.3. The structured approach consists of the following sequential stages:

1. **Problem Identification:** Conducting an exhaustive literature review to comprehensively analyze and identify existing challenges and limitations in current stock market prediction methodologies.
2. **Methodology Development:** Designing a holistic, multi-component prediction framework that integrates CNN, linear transformer mechanisms, FinBERT embeddings, and LLM-driven feature engineering.
3. **Implementation:** Developing advanced deep learning models that effectively process textual, temporal, and spatial data to predict stock market movements.
4. **Validation:** Performing rigorous experimental evaluation and comprehensive performance assessment, including comparison against state-of-the-art benchmark models.

This structured approach ensures clarity, reproducibility, and methodological rigor, enabling systematic resolution of identified research gaps.

3.1.3 Experimental Scope

The experimental analysis and evaluation in this research specifically targets predicting stock market trends using a well-defined scope, which includes:

- Historical data from the S&P 500 index, spanning the period of 2022–2023, ensuring relevance and timeliness of the study.
- Application of advanced deep learning methodologies, including convolutional neural networks (CNNs), linear attention transformer mechanisms, and integration of large language models (LLMs) enhanced by FinBERT embeddings.
- Comprehensive multimodal data integration techniques, leveraging visual stock charts, numerical market data, and textual indicators dynamically generated through sophisticated prompt engineering.
- Robust computational and analytical frameworks that rigorously test and validate the effectiveness of the proposed predictive models under various market conditions.

This clearly defined experimental scope aims to comprehensively demonstrate the efficacy of the developed methodology, directly addressing previously highlighted limitations in the literature through the incorporation of cutting-edge computational strategies.

3.1.4 Baseline Model Selection and Justification

Our experimental evaluation compares the proposed framework against 14 carefully selected baseline models. This section provides detailed justification for their inclusion, organized by model category.

Selection Criteria

Baseline models were selected based on four key criteria:

1. **State-of-the-Art Performance**: Models demonstrating competitive results in recent literature (2020-2024)
2. **Architectural Diversity**: Coverage of different modeling paradigms (RNN, CNN, Transformer, Linear, Hybrid)

3. **Relevance to Financial Forecasting**: Proven applicability to stock price prediction tasks

4. **Reproducibility**: Availability of implementation details and hyperparameters

Baseline Model Categories and Justifications

1. Traditional Deep Learning Models (3 models):

- **LSTM** (Hochreiter and Schmidhuber, 1997): *Justification*: Standard baseline for sequential data; widely used in financial forecasting. Enables comparison against traditional temporal modeling approaches.

- **GRU** (Cho et al., 2014): *Justification*: Simplified RNN variant with comparable performance to LSTM but fewer parameters. Tests whether architectural complexity is necessary.

- **CNN** (Zhang et al., 2023): *Justification*: Evaluates pure spatial feature extraction from price charts. Provides comparison point for our CNN component in isolation.

2. Hybrid Models (4 models):

- **CNN-LSTM** (Kanwal et al., 2022): *Justification*: Combines spatial and temporal modeling; represents current state-of-the-art hybrid approach in financial forecasting.

- **CNN-GRU** (Sen et al., 2021): *Justification*: Alternative hybrid architecture; tests whether GRU provides advantages over LSTM in fusion scenarios.

- **CNN-Transformer** (Yuan et al., 2023): *Justification*: Recent architecture combining CNNs with attention mechanisms; directly comparable to our approach but without LLM component.

- **CNN+Transformer+FinBERT** (our ablation): *Justification*: Excludes LLM component to isolate its contribution.

3. Transformer-Based Models (4 models):

- **Informer** (Zhou et al., 2021): *Justification*: Addresses long-sequence forecasting with ProbSparse attention; highly cited baseline for financial time series.
- **Autoformer** (Wu et al., 2021): *Justification*: Introduces decomposition architecture; represents state-of-the-art in 2021-2022 period.
- **PatchTST** (Nie et al., 2022): *Justification*: Recent model using patch-based processing; demonstrates effectiveness of local context windows.
- **Standard Transformer** (Vaswani, 2017): *Justification*: Original transformer architecture; enables comparison with our linear attention variant.

4. Linear Models (3 models):

- **DLinear** (Zeng et al., 2023): *Justification*: Challenges complex models with simple linear decomposition; critically important baseline showing that simplicity can outperform complexity.
- **NLinear** (Zeng et al., 2023): *Justification*: Normalization-based linear model; tests whether feature scaling alone provides benefits.
- **Linear** (simple linear regression): *Justification*: Simplest possible baseline; establishes lower bound for model performance.

Notable Exclusions and Rationale

While comprehensive, our baseline selection excludes certain models for specific reasons:

- **TimesNet** (Wu et al., 2022): Excluded due to computational constraints; requires specialized 2D tensor transformations not directly comparable to our framework.
- **Prophet** (Taylor and Letham, 2018): Designed for business forecasting with trend/seasonality; not optimized for high-frequency financial data.
- **N-BEATS** (Oreshkin et al., 2020): Strong univariate model but doesn't naturally incorporate multi-modal inputs our framework uses.

- **Traditional ARIMA/GARCH**: Extensively compared in prior work (Tsay, 2005); known to underperform deep learning methods for non-stationary financial data.

Benchmark Completeness

Our 14-model benchmark provides comprehensive coverage:

- **Temporal Coverage**: From 1997 (LSTM) to 2023 (DLinear)
- **Complexity Range**: From simple linear models to complex multi-modal architectures
- **Paradigm Diversity**: RNN, CNN, Transformer, Linear, and Hybrid approaches
- **Parameter Scale**: From thousands (Linear) to millions (Transformer variants)

This selection strategy ensures rigorous evaluation against both established methods and cutting-edge approaches, validating our framework's effectiveness across the spectrum of financial forecasting techniques.

3.2 Data Collection and Processing Framework

3.2.1 Data Sources and Acquisition

In this study, we utilized the S&P 500 index dataset spanning from January 2022 to December 2023, obtained through Tushare API. The S&P 500 index serves as a fundamental gauge of the U.S. stock market performance, representing 500 of the largest publicly traded companies (ZXhang et al., 2023). This particular timeframe captures significant market volatility and economic events, making it ideal for evaluating our model's performance under diverse market conditions.

The dataset encompasses comprehensive daily trading information, including six essential variables: date, opening price, highest price, lowest price, closing price, adjusted closing price, and trading volume. Table 1 provides an overview of the dataset structure and sample entries.

Table 3.1: Overview of S&P 500 Dataset (2022-2023)

Date	Open	High	Low	Close	Adj Close	Volume
2022-01-03	4,766.18	4,796.32	4,758.17	4,796.56	4,796.56	3,521,960,000
2022-01-04	4,796.94	4,818.62	4,761.22	4,793.54	4,793.54	4,162,300,000
2022-01-05	4,792.35	4,797.71	4,699.44	4,700.58	4,700.58	4,598,410,000
2022-01-06	4,697.11	4,725.01	4,671.24	4,696.05	4,696.05	4,726,840,000
2022-01-07	4,697.24	4,707.72	4,662.74	4,677.03	4,677.03	4,239,990,000

The dataset contains 500 trading days of data points, with no missing values observed during the specified period. All price data points are recorded in U.S. dollars, while trading volume represents the number of shares traded. The data shows significant price movements throughout the period, with the index experiencing both bullish and bearish trends, providing a robust foundation for testing our model's predictive capabilities.

To ensure data quality and consistency, we implemented the following preprocessing steps:

- Verification of data continuity across trading days
- Conversion of price data to logarithmic returns for variance stabilization
- Volume data normalization to account for varying trading activity levels
- Adjustment for stock splits and dividends through the adjusted closing price

This comprehensive dataset provides a solid foundation for evaluating our LLM-Enhanced Linear Self-Attention Transformer-CNN methodology across different market conditions and volatility regimes. Figure 3.4 visualizes the closing price trend throughout the studied period, highlighting the market's dynamic nature and the challenges our model aims to address.

Data Source Selection

We employed the TuShare API for data collection, selecting from multiple available global market indices as shown in Table 3.2. Our focus on the S&P 500 index was driven by its market representativeness and data consistency.

Figure 3.4: S&P 500 Index Closing Price Trend (2022-2023)

Table 3.2: Global Market Indices Available for Analysis

Index Code	**Description**	**Region**
SPX	S&P 500 Index	United States
IXIC	NASDAQ Composite	United States
FTSE	FTSE 100 Index	United Kingdom
N225	Nikkei 225	Japan
HSI	Hang Seng Index	Hong Kong

Raw Data Collection

The initial data collection yielded comprehensive market indicators as presented in Table 3.3, including essential pricing metrics and derivative indicators.

Table 3.3: Raw S&P 500 Market Data with Extended Indicators (2022-2023)

Date	Open	Close	High	Low	Pre Close	Change	Pct Chg
2023-12-29	4,782.88	4,769.83	4,788.43	4,751.99	4,783.35	-13.52	-0.28
2023-12-28	4,786.44	4,783.35	4,793.30	4,780.98	4,781.58	1.77	0.04
2023-12-27	4,773.45	4,781.58	4,785.39	4,768.90	4,774.75	6.83	0.14
2023-12-26	4,758.86	4,774.75	4,784.72	4,758.45	4,754.63	20.12	0.42
2023-12-22	4,753.92	4,754.63	4,772.94	4,736.77	4,746.75	7.88	0.17

3.2.2 Value Range Adjustment

Value range adjustment procedures transform variables into a constrained interval, commonly $[0, 1]$. This transformation proves essential for enhancing the performance of gradient-descent optimization algorithms, facilitating quicker convergence and improved efficiency. In our implementation, we specifically employed the MinMaxScaler normalization technique from the scikit-learn library, which directly implements this mathematical transformation while maintaining the distributional relationships between data points. The mathematical representation for this adjustment can be expressed as:

$$x_{adjusted} = \frac{x - x_{minimum}}{x_{maximum} - x_{minimum}} \tag{3.1}$$

where $x_{minimum}$ and $x_{maximum}$ represent the lowest and highest values within the dataset, respectively.

The MinMaxScaler was selected over other normalization approaches because it preserves the exact relationships between the original data points while rescaling them to fit within the required $[0, 1]$ range. Implementing standardization alongside value range adjustments creates data better suited for machine learning model training, diminishing numerical instability risks and accelerating convergence during the optimization process. These preprocessing methods additionally enhance feature interpretability by establishing comparable measurement scales across different variables.

We applied these MinMaxScaler transformations to stock price logarithmic returns and trading volume data to boost computational stability and maintain consistent feature scaling throughout our analysis.

3.2.3 FinBERT Implementation for Text Embedding

The text embedding pipeline utilizes FinBERT, a BERT model fine-tuned for financial text analysis. The implementation follows these key steps:

Model Architecture

The FinBERT architecture processes financial text through:

$$\text{Text} \xrightarrow{\text{Tokenization}} \text{Tokens} \xrightarrow{\text{FinBERT}} \mathbb{R}^{768} \tag{3.2}$$

Processing Pipeline

The implementation includes:

- Text preprocessing with max length 512 tokens
- CUDA-accelerated batch processing
- Dimensionality reduction through fully connected layers:

$$\mathbb{R}^{768} \xrightarrow{FC1} \mathbb{R}^{256} \xrightarrow{FC2} \mathbb{R}^{128} \tag{3.3}$$

Feature Integration

The feature fusion architecture combines:

$$\mathbf{F}_{combined} = [\mathbf{F}_{transformer}\mathbf{F}_{cnn}\mathbf{F}_{finbert}] \tag{3.4}$$

Loss Function Design

Multi-component loss function:

$$\mathcal{L}_{total} = 0.3\mathcal{L}_{transformer} + 0.3\mathcal{L}_{cnn} + 0.4\mathcal{L}_{fusion} \tag{3.5}$$

We experimentally determined the configuration of dense neural network layers to establish an appropriate equilibrium between expressive capacity

and processing demands. Regarding our loss criterion, we opted for the Huber Loss mechanism because of its inherent stability characteristics. This can be expressed mathematically as follows:

$$\mathcal{L}_{\text{Huber}}(y_i, \hat{y}_i) = \begin{cases} \frac{1}{2}(y_i - \hat{y}_i)^2, & \text{when } |y_i - \hat{y}_i| \leq \delta \\ \delta \cdot (|y_i - \hat{y}_i| - \frac{\delta}{2}), & \text{for all other cases} \end{cases} \tag{3.6}$$

In this equation, y_i signifies the actual stock value, $\hat{y}_i$ indicates the forecasted output from our model, while δ serves as a configurable threshold parameter. This parameter was determined through rigorous cross-validation to optimize the trade-off between sensitivity to minor price fluctuations and robustness against extreme market movements. The Huber Loss provides significant advantages over traditional loss functions for financial time series modeling. Consider the gradient of Mean Squared Error (MSE):

$$\nabla_{\hat{y}i}\mathcal{L}\text{MSE} = \nabla_{\hat{y}_i}\frac{1}{2}(y_i - \hat{y}_i)^2 = -(y_i - \hat{y}_i) \tag{3.7}$$

This gradient scales linearly with the error magnitude, causing disproportionately large parameter updates for outliers. Conversely, the Mean Absolute Error (MAE) gradient:

$$\nabla_{\hat{y}i}\mathcal{L}\text{MAE} = \nabla_{\hat{y}_i}|y_i - \hat{y}_i| = \text{sgn}(-(y_i - \hat{y}_i)) \tag{3.8}$$

maintains constant magnitude regardless of error size, potentially underemphasizing significant but non-outlier deviations.

Unlike Mean Squared Error (MSE), which disproportionately penalizes outliers and can lead to model instability in volatile markets, Huber Loss limits the gradient magnitude for samples with large residuals.

Simultaneously, it preserves the desirable quadratic behavior of MSE for smaller errors, maintaining sensitivity to subtle price movements that characterize normal market conditions. This hybrid approach is particularly crucial for financial data, where anomalous market behaviors and sudden price shifts occur regularly but should not dominate the learning process. To prevent model overfitting, we implemented an extensive regularization framework. Dropout mechanisms with 0.5 probability were methodically incorporated following each dense neural layer, successfully preventing coordinated adaptation among hidden representations during training. Furthermore, we utilized preemptive termination protocols with patience settings spanning 10 training cycles, continuously assessing validation performance metrics to conclude training when generalization capability begins declining.

Our multi-modal framework further benefits from a component-weighted loss function:

$$\mathcal{L}\text{total} = 0.3\mathcal{L}\text{transformer} + 0.3\mathcal{L}\text{cnn} + 0.4\mathcal{L}\text{fusion} \tag{3.9}$$

This weighted formulation ensures balanced learning across the temporal patterns captured by the Linear Transformer ($\mathcal{L}$transformer), spatial features extracted by the CNN ($\mathcal{L}$cnn), and the integrated representation from the fusion layer ($\mathcal{L}_{\text{fusion}}$). The higher weighting (0.4) assigned to the fusion component emphasizes the importance of effectively integrating information across modalities, which is critical for capturing the complex interdependencies within financial market data.

Preprocessing Methodology

Our preprocessing pipeline implemented systematic transformations to ensure data quality and consistency, as detailed in Table 3.4.

Table 3.4: Data Preprocessing Steps and Transformations

Processing Step	**Description**
Date Format	Standardized to YYYY-MM-DD format
Missing Values	Filled using column-wise mean imputation
Column Selection	Retained essential price and volume indicators
Data Ordering	Chronologically sorted by date
Adjusted Close	Added for dividend and split adjustments

Statistical Analysis

The processed dataset underwent comprehensive statistical analysis to ensure data quality and distribution characteristics, as shown in Table 3.5.

3.2.4 Dataset Structure

The processed dataset maintains high data quality standards while preserving essential market information, as demonstrated in Table 3.6.

The dataset exhibits several key characteristics:

Table 3.5: Statistical Summary of Processed S&P 500 Data (2022-2023)

Metric	**Open**	**Close**	**Volume**	**Daily Change (%)**
Mean	4,756.32	4,758.44	1,985,247	0.18
Std Dev	156.73	157.89	524,316	0.92
Min	4,662.74	4,677.03	1,446,123	-1.94
Max	4,804.51	4,796.56	2,813,805	0.64

Table 3.6: Processed S&P 500 Dataset with Standardized Format

Date	Open	High	Low	Close	Adj Close	Volume
2022-01-03	4,778.14	4,796.64	4,758.17	4,796.56	4,796.56	2,209,533
2022-01-04	4,804.51	4,818.62	4,774.27	4,793.54	4,793.54	2,813,805
2022-01-05	4,787.99	4,797.70	4,699.44	4,700.58	4,700.58	2,803,729
2022-01-06	4,693.39	4,725.01	4,671.26	4,696.05	4,696.05	2,379,823
2022-01-07	4,697.66	4,707.95	4,662.74	4,677.03	4,677.03	2,416,097

- Complete daily price data spanning 500 trading days
- Comprehensive volume data without missing values
- Corporate action-adjusted closing prices

This refined dataset serves as the foundation for our LLM-Enhanced Linear Self-Attention Transformer-CNN methodology, enabling robust model training and evaluation across diverse market conditions. The price trend visualization in Figure 3.4 illustrates the market's dynamic nature during the study period.

3.2.5 Historical Market Data Collection

Let P_t represent the closing price at time t. The logarithmic return is defined as:

$$r_t = \log\left(\frac{P_t}{P_{t-1}}\right) \tag{3.10}$$

Similarly, the volume data is processed as:

$$v_t = \log\left(\frac{V_t}{V_{t-1}}\right) \tag{3.11}$$

where V_t is the trading volume at time t.

3.2.6 Market Signal Parameter Construction

For detecting subtle market dynamics, our framework incorporates established analytical metrics as input characteristics. The methodology employed for computing these financial parameters follows specific formulations:

Price Trend Indicator (MA):

$$\text{Average}_n(t) = \frac{1}{n} \sum_{j=0}^{n-1} \text{Price}_{t-j} \tag{3.12}$$

Relative Strength Index (RSI):

$$\text{RSI}_t = 100 - \frac{100}{1 + \text{RS}_t} \tag{3.13}$$

Bollinger Bands Indicator (BB):

$$\text{Upper Limit}_{\text{BB}} = \text{Moving Average}_n(t) + m \cdot \text{StdDev}_n(t) \tag{3.14}$$

$$\text{Lower Limit}_{\text{BB}} = \text{Moving Average}_n(t) - m \cdot \text{StdDev}_n(t) \tag{3.15}$$

where $\text{StdDev}_n(t)$ represents volatility measured through standard deviation calculations over window n, with parameter k commonly assigned value 2 for establishing confidence intervals.

3.2.7 Information Quality Enhancement

Beyond conventional analytical parameters, we augment dataset quality through natural language processing capabilities offered by advanced semantic models, notably utilizing ChatGPT-4 and DeepSeek. Through systematic prompt construction techniques, these language frameworks synthesize enhanced datasets incorporating both sequential patterns and calculated market signals, substantially boosting the forecast capabilities of our analytical system.

3.3 LLM-Based Feature Extraction for Stock Prediction

3.3.1 Technical Analysis with LLMs

Within our research, advanced semantic frameworks are employed to produce compact daily market assessments derived from organized economic

information. Unlike prior methods that rely on external sentiment data, our approach extracts insights directly from numerical indicators and converts them into human-readable text summaries. Each analysis is constrained to at most 20 words, ensuring that only essential market signals are conveyed.

The structured input provided to the LLM consists of:

- Open, high, low, close, and volume.
- Moving Average Convergence Divergence (MACD), signal line, and histogram.
- Bollinger Bands (upper, lower, and middle).
- Relative Strength Index (RSI).

The LLM is prompted with a predefined structure to ensure domain-specific responses. The model is instructed as follows:

> *"You are a CFA expert. Provide a very concise technical analysis for {`date`} using the following indicators. Limit your answer to 20 words and use only essential financial terms."*

The following diagram 3.5 illustrates the core steps in our LLM-based technical analysis pipeline. It visualizes how raw financial data is processed, a prompt is formulated, and an LLM generates potential sentiment-driven classifications. This structured approach enables automated, nuanced decision-making in stock market predictions.

3.3.2 Financial Indicator-Based Text Generation

To generate financial analysis text, a structured API is implemented using `DeepSeek-R1` (14B parameters) deployed via the `Ollama` framework. The workflow of this system is illustrated in Figure 3.6, which outlines the sequential process.

3.3.3 Data Preprocessing and Integration

To ensure consistency and accuracy, the LLM-generated text undergoes several post-processing steps:

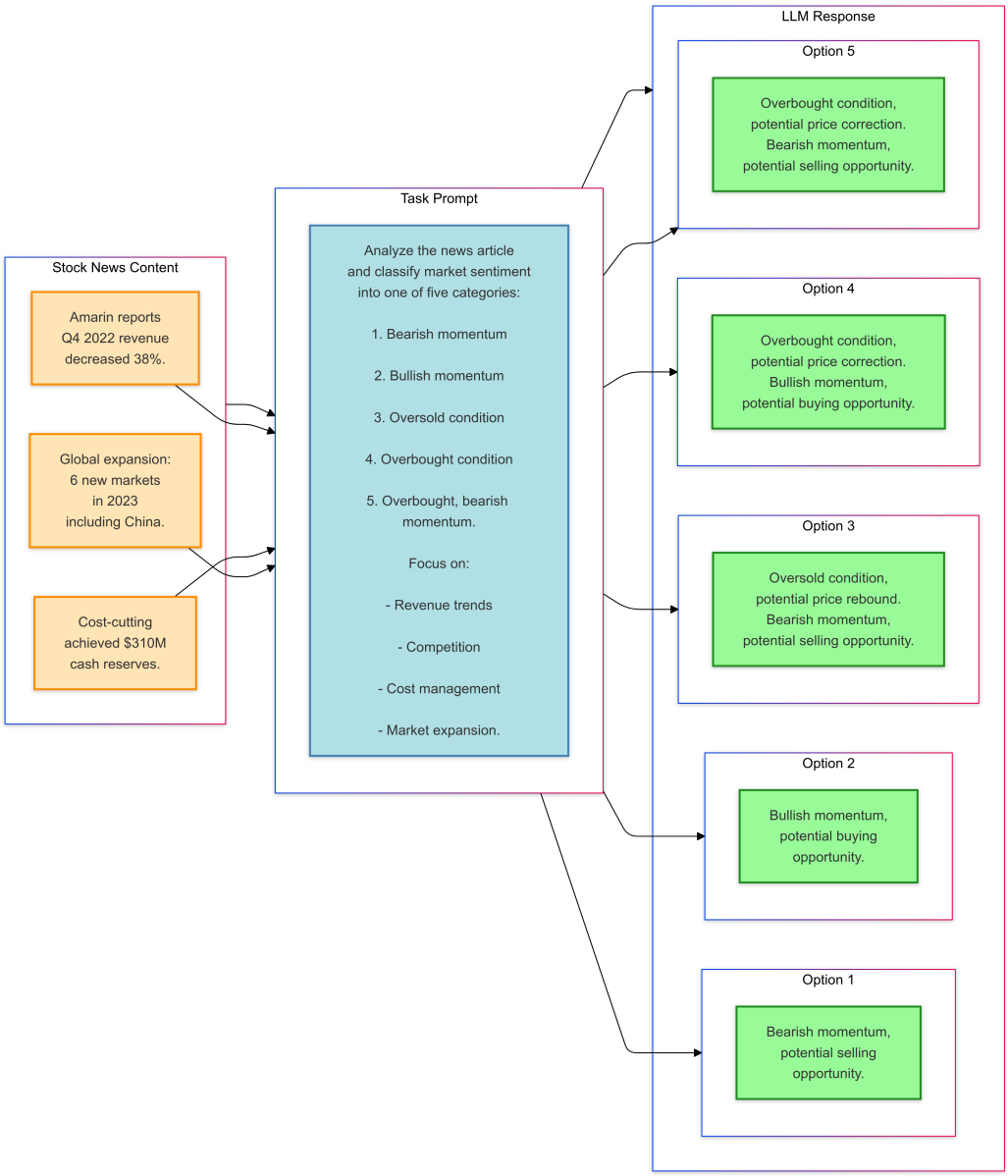


Figure 3.5: An example of LLM-augmented technical analysis framework for classifying financial sentiment. It demonstrates how stock news content, task prompts, and potential outcomes integrate to provide actionable insights.

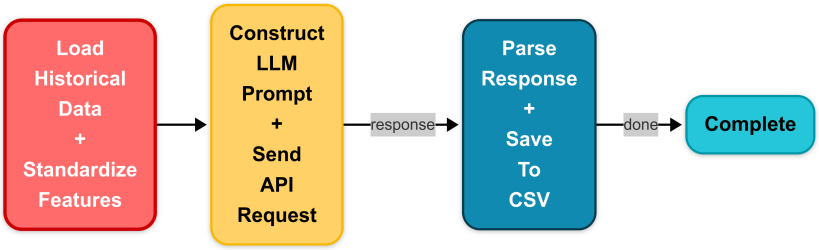


Figure 3.6: Workflow for LLM-based stock prediction feature extraction

- Dates are formatted to `YYYY-MM-DD` for uniformity.
- Responses are stripped of redundant tags (e.g., `<think>` elements).
- Multi-line outputs are flattened into a single string to maintain dataset integrity.
- The final text is truncated to at most 20 words for brevity.

A sample response after processing is:

> *"Bullish trend confirmed. RSI strong. MACD positive crossover. Bollinger Band widening suggests increased volatility."*

Once generated, the financial text is vectorized using `FinBERT` embeddings. These embeddings capture contextual meaning from the textual summary, converting qualitative indicators into a quantitative format suitable for machine learning models. The integration of FinBERT is detailed in Section 3.2.3.

3.3.4 API Deployment and Model Efficiency

The LLM-based analysis generation follows a structured API request format, summarized in Table 3.7.

Given the computational requirements of LLM-based feature generation, several optimizations are implemented:

- A *0.3-second delay* between API calls prevents server overload.
- *Parallelized queries* are used to speed up batch processing.
- The use of *local inference* with Ollama ensures cost-effective, offline execution.

Table 3.7: Structured API Request Format for LLM Technical Analysis

Parameter	Description
`model`	Specifies the deployed LLM model (e.g., `deepseek-r1:14b`).
`messages`	A structured message array containing user prompts.
`role`	Defines the LLM's response style, e.g., "CFA expert."
`content`	Contains the formatted financial indicators and request constraints.

3.3.5 Summary of LLM-Based Feature Generation Framework

This section introduces an LLM-based feature generation framework designed for stock prediction. The system integrates structured indicator-driven text generation, FinBERT embeddings, and local API deployment for efficiency. The generated textual indicators provide a novel way of enriching stock price forecasting models with interpretable insights, bridging the gap between deep learning and human-readable financial analytics.

3.4 Deep Learning Architecture Design

3.4.1 CNN Architecture

As a foundation for our integrated CNN-Transformer model, we first implement a pure CNN-based approach that serves as both a baseline and a component of our final architecture. Figure 3.7 illustrates the CNN architecture's structure.

CNN Baseline Architecture

The CNN architecture processes stock market data through several key stages:

1. **Visual Representation**: Raw stock data is transformed into three complementary visual formats:
    - Candlestick charts capturing price action patterns
    - Line charts emphasizing price trends

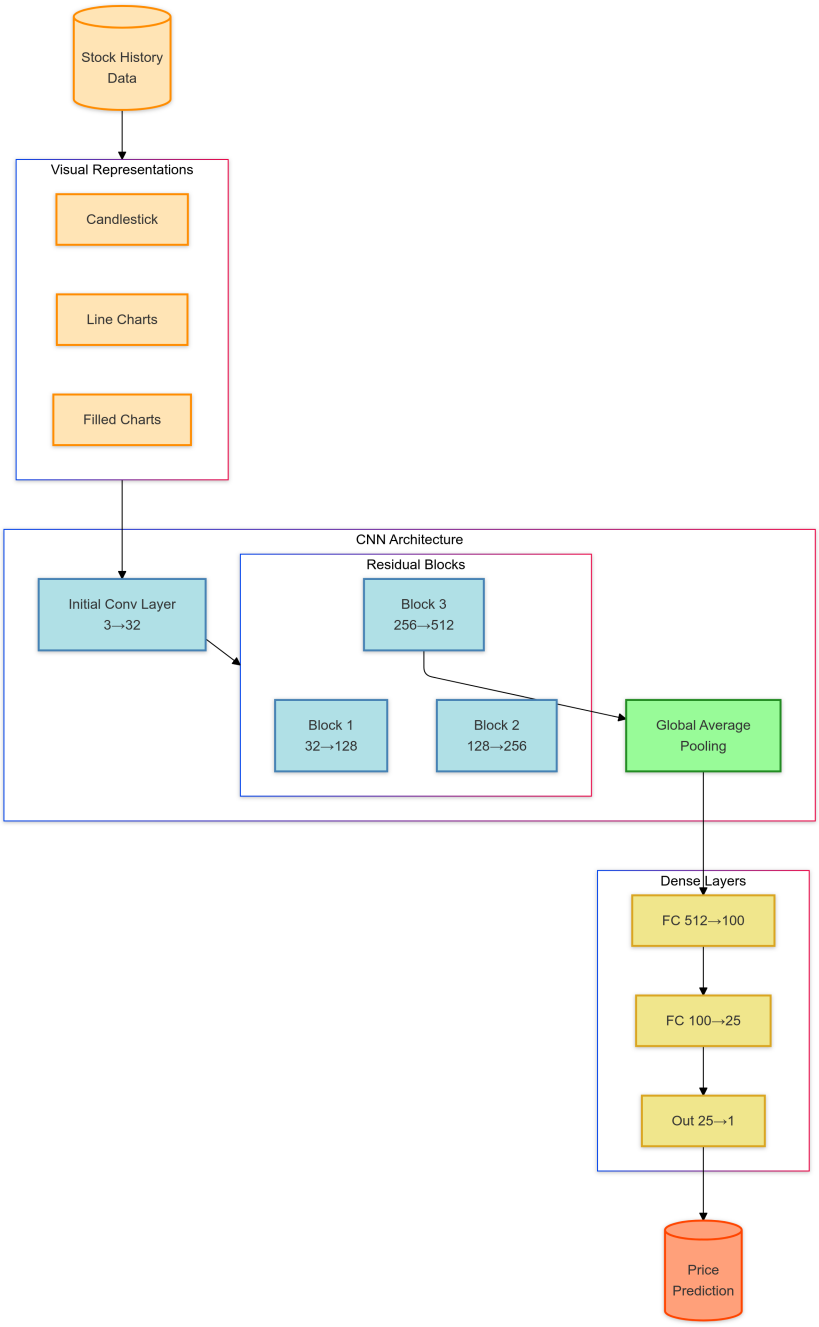


Figure 3.7: CNN Architecture for Stock Price Prediction. The model processes multi-format visual representations through a series of convolutional and residual blocks before making predictions through dense layers.

- Filled charts highlighting price momentum

2. **Feature Extraction Pipeline**:
   - Initial convolution layer (3→32 channels) captures basic visual features
   - Three residual blocks with increasing channel depth:
     - Block 1: 32→128 channels
     - Block 2: 128→256 channels
     - Block 3: 256→512 channels
   - Global average pooling for spatial feature aggregation
3. **Dense Layer Network**:
   - Hierarchical dimension reduction: 512→100→25→1
   - Dropout layers between dense layers for regularization

This baseline architecture demonstrates effective feature extraction from visual price patterns, which we leverage in our integrated CNN-Transformer model. The residual blocks enable deeper network training while maintaining gradient flow, while the multi-format visual inputs provide complementary perspectives on price movements.

In our integrated CNN-Transformer architecture, we retain this CNN structure for visual feature extraction while augmenting it with transformer components for temporal pattern analysis. The CNN branch processes spatial features from the visual representations, which are then fused with the transformer's temporal features for final prediction.

CNN Architecture for Visual Features

This analytical pathway implements an adapted neural processing architecture specialized for market visualization examination. Drawing inspiration from ResNet principles, our design incorporates skip-connection learning alongside narrowing structural elements to mitigate parameter explosion and gradient diminishment challenges.

Our approach incorporates multiple visual representation methods for stock market data, enabling the CNN component to capture different aspects of market patterns. We implement four distinct visualization techniques, each designed to highlight specific market characteristics:

CNN Layer Architecture

The CNN branch of our model processes stock market visual data through a series of specialized convolutional and pooling layers, each serving a specific purpose in feature extraction:

Initial Convolution (Conv1) The first convolutional layer serves as the primary feature detector:

- Input: $112 \times 112 \times 3$ (RGB stock chart images)
- Operation: Applies 64 filters with kernel size 7×7
- Purpose: Extracts low-level features such as edges, colors, and basic shapes from the stock charts

Residual Convolution Blocks (ResConv1-3) A series of residual blocks enhance feature learning while mitigating the vanishing gradient problem:

ResConv1:

- Processes features at 56×56 resolution
- Implements identity shortcuts for gradient flow
- Extracts intermediate-level patterns such as trend lines and support-/resistance levels

ResConv2:

- Operates at 28×28 resolution
- Doubles the feature channels for richer representation
- Identifies complex patterns like candlestick formations and volume relationships

ResConv3:

- Works at 14×14 resolution
- Further increases feature complexity
- Captures high-level market patterns and their interactions

Global Average Pooling (AvgPool2d) The final pooling layer performs dimensionality reduction while preserving spatial information:

- Reduces spatial dimensions to 1×1
- Outputs a 512-dimensional feature vector
- Maintains translation invariance of detected features
- Provides robust feature summarization for fusion with other model components

Each residual block follows the structure:

$$y = F(x, \{W_i\}) + x \tag{3.16}$$

where $F(x, \{W_i\})$ represents the residual mapping to be learned, and x is the identity shortcut connection.

This progressive feature extraction and dimensionality reduction process enables our model to capture both fine-grained price patterns and broader market trends from the visual input, preparing the data for subsequent fusion with textual and numerical features.

Candlestick Representation

The traditional Japanese candlestick chart remains our primary visualization method. Each candlestick at time t is defined by a tuple:

$$S_t = (O_t, H_t, L_t, C_t, V_t) \tag{3.17}$$

where O_t denotes the opening price, H_t the highest price, L_t the lowest price, C_t the closing price, and V_t the trading volume at time t. Each candlestick is rendered with color coding based on price movement:

$$\text{Color}(S_t) = \begin{cases} \text{blue}, & \text{if } C_t > O_t \quad \text{(bullish)} \\ \text{red}, & \text{if } C_t \leq O_t \quad \text{(bearish)} \end{cases} \tag{3.18}$$

Line Chart Visualization

We implement a dual-line representation showing high and low prices:

$$\mathcal{L} = \{(t, H_t), (t, L_t)\}_{t=1}^{T} \tag{3.19}$$

where T represents the sequence length, H_t denotes the high price, and L_t denotes the low price at time t.

High-Low-Middle (HLM) Chart

This visualization incorporates a middle price line:

$$M_t = \frac{H_t + L_t}{2} \tag{3.20}$$

Creating a triple-line representation:

$$HLM_t = \{(t, H_t), (t, L_t), (t, M_t)\}_{t=1}^{T} \tag{3.21}$$

Filled Line Chart

We enhance pattern visibility using area fills between price levels:

$$F_t = \begin{cases} \text{red fill}, & \text{if } H_t > M_t \\ \text{blue fill}, & \text{if } L_t < M_t \end{cases} \tag{3.22}$$

All visualizations maintain consistent properties:

- Dimension: 112×112 pixels
- DPI: 100
- Dual-panel layout: price patterns (top), volume (bottom)
- Normalized scaling for consistent feature extraction

The volume component is represented consistently across all visualization types:

$$V_{normalized} = \frac{V_t - \min(V)}{\max(V) - \min(V)} \tag{3.23}$$

This multi-representation approach provides our CNN model with complementary views of the same price movements, enabling robust feature extraction across different market conditions. Each visualization type emphasizes different aspects of price movement patterns, allowing the CNN to learn a more comprehensive set of visual features.

To illustrate these distinct visualization techniques and their complementary nature, Figure 3.8 presents a comprehensive comparison of different market patterns and their corresponding representations. The figure demonstrates how each visualization method captures unique aspects of market

dynamics, from traditional candlestick patterns to advanced filled-area representations. This visual comparison highlights how different chart types can emphasize various market characteristics: candlestick charts excel at showing price action details, line charts emphasize trend continuity, and filled charts enhance the visibility of price movements relative to the middle line. These diverse representations collectively provide our CNN model with a rich set of visual features, enabling it to learn from multiple perspectives of the same underlying price movements.

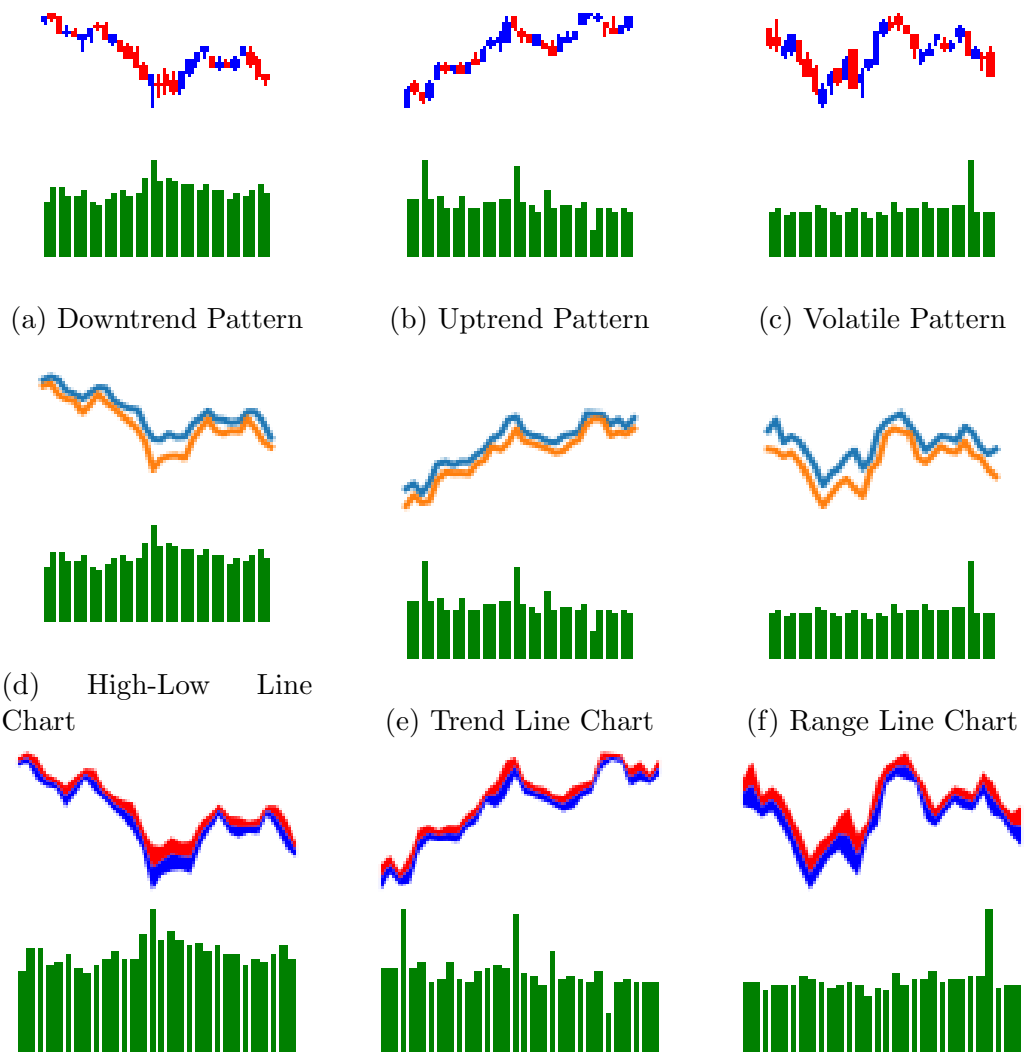
Figure 3.9 presents an integrated multi-panel technical analysis visualization designed to provide comprehensive market context for our CNN model. The visualization combines three essential analytical components in a vertically stacked format. The upper panel displays a candlestick chart where blue candles indicate bullish price movement and red candles indicate bearish movement, overlaid with Bollinger Bands (cyan curves) that establish dynamic volatility channels at two standard deviations from the 20-period Simple Moving Average (black curve). This combination effectively captures both price action and volatility fluctuations. The middle panel presents trading volume as a histogram with green bars, quantifying market participation intensity which often precedes significant price movements. The lower panel features the Relative Strength Index (RSI) oscillator (pink curve) bounded by the standard overbought threshold (red dotted line at 70) and oversold threshold (green dotted line at 30), providing crucial momentum signals that complement price patterns. This integrated representation enables the CNN to learn correlations between price formations, volume confirmation, and momentum conditions simultaneously, capturing market dynamics that might be missed when analyzing these indicators separately.

3.4.2 Linear Transformer Architecture

The linear transformer component forms the temporal modeling core of our hybrid architecture, designed to capture long-range dependencies in stock price sequences while maintaining computational efficiency.

Linear Self-Attention Mechanism

Traditional transformer self-attention computes attention scores with quadratic complexity $O(n^2)$, where n is the sequence length. We implement a linear



(a) Downtrend Pattern (b) Uptrend Pattern (c) Volatile Pattern

(d) High-Low Line Chart (e) Trend Line Chart (f) Range Line Chart

(g) Candlestick Pattern (h) Trend Candlestick (i) Reversal Pattern

Figure 3.8: Different visual representations of stock market data. Each visualization captures unique aspects of price movements and volume patterns, providing diverse features for the CNN model to learn from. (a-c) show different market trends in line format, (d-f) demonstrate various line-based visualizations, and (g-i) present candlestick patterns.

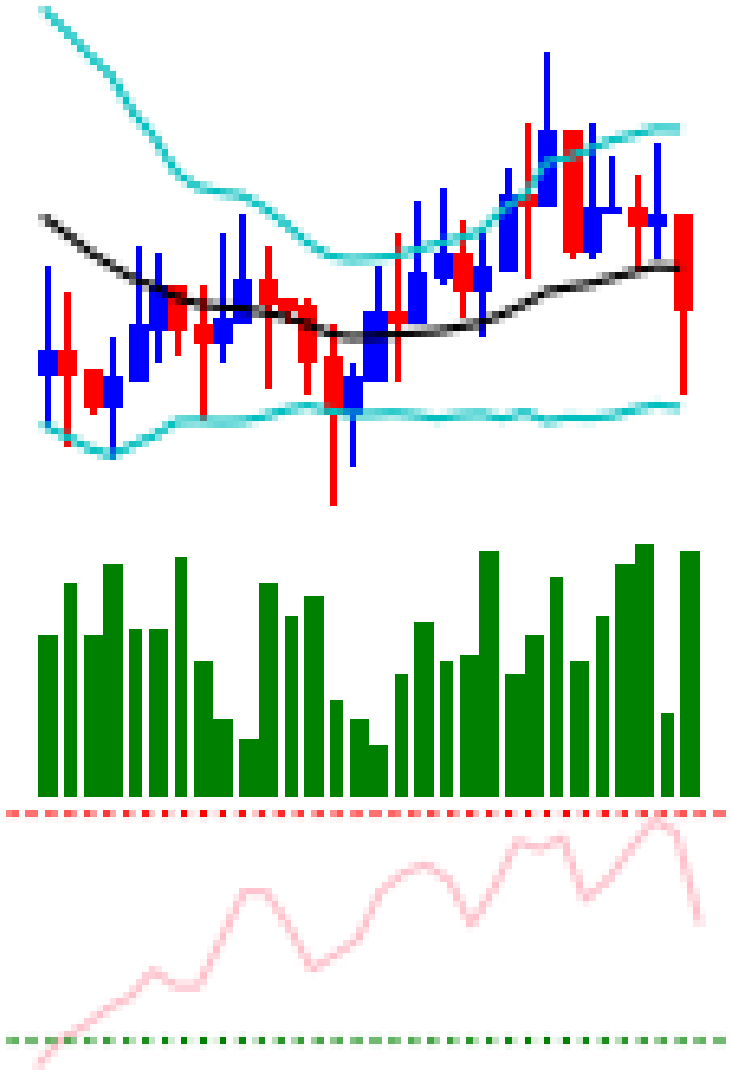


Figure 3.9: Comprehensive market analysis visualization displaying: (a) Price action represented through candlesticks (upward movements in blue, downward shifts in red) alongside volatility boundaries via Bollinger Bands and periods moving average; (b) Trading volume represented as vertical green histogram bars indicating market participation levels; and (c) Momentum evaluation through an RSI indicator (pink graph) featuring upper resistance (dashed red line at 70) and lower support (dashed green line at 30) thresholds.

approximation that reduces this to $O(n)$ complexity while preserving the model's ability to capture temporal dependencies.

The standard attention mechanism is defined as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{3.24}$$

Our linear self-attention approximation uses kernel feature maps $\phi(\cdot)$ to decompose the attention computation:

$$\text{LinearAttention}(Q, K, V) = \phi(Q)\left(\phi(K)^T V\right) \tag{3.25}$$

where $\phi : \mathbb{R}^d \to \mathbb{R}^{d'}$ is a feature map function. We employ an ELU-based feature map:

$$\phi(x) = \text{ELU}(x) + 1 \tag{3.26}$$

This formulation allows us to compute attention in $O(nd^2)$ time instead of $O(n^2 d)$, providing significant computational savings for long sequences.

Architecture Details

The linear transformer module processes price sequences through multiple layers:

Input Embedding:

$$H_0 = XW_e + P \tag{3.27}$$

where $X \in \mathbb{R}^{n \times d_{in}}$ is the input sequence, $W_e \in \mathbb{R}^{d_{in} \times d_{model}}$ is the embedding matrix, and $P \in \mathbb{R}^{n \times d_{model}}$ represents positional encodings.

Transformer Layers: Each of the L transformer layers applies:

$$\begin{aligned} H'_\ell &= \text{LayerNorm}(H_{\ell-1} + \text{LinearAttention}(H_{\ell-1})) \\ H_\ell &= \text{LayerNorm}(H'_\ell + \text{FFN}(H'_\ell)) \end{aligned} \tag{3.28}$$

Hyperparameters:

- Model dimension: $d_{model} = 256$
- Number of layers: $L = 4$

- Number of attention heads: $h = 8$
- Feed-forward dimension: $d_{ff} = 1024$
- Dropout rate: $p = 0.1$

Computational Advantages

The linear attention mechanism provides several benefits:

- **Memory Efficiency**: Reduces memory footprint from $O(n^2)$ to $O(n)$
- **Sequence Length Scalability**: Enables processing of 200+ trading day windows
- **Training Speed**: Accelerates training by approximately 3× compared to standard transformers
- **Inference Latency**: Achieves sub-100ms prediction time suitable for real-time trading

This design allows our framework to analyze longer historical contexts while maintaining practical deployment feasibility.

3.4.3 Multi-Modal Fusion Architecture

The fusion layer integrates features from three heterogeneous modalities—textual (LLM+FinBERT), visual (CNN), and temporal (Transformer)—into a unified representation for stock price prediction.

Feature Extraction from Each Modality

LLM-FinBERT Text Features: LLM-generated technical analysis text is processed through FinBERT:

$$F_{text} = \text{FinBERT}(\text{LLM-Text}) \in \mathbb{R}^{768} \tag{3.29}$$

These features are then projected to fusion dimension:

$$F'_{text} = \sigma(F_{text} W_{text} + b_{text}) \in \mathbb{R}^{128} \tag{3.30}$$

CNN Visual Features: Stock chart images are processed through the CNN architecture:

$$F_{visual} = \text{GlobalAvgPool}(\text{CNN}(\text{Charts})) \in \mathbb{R}^{512} \tag{3.31}$$

Projection to fusion dimension:

$$F'_{visual} = \sigma(F_{visual} W_{visual} + b_{visual}) \in \mathbb{R}^{128} \tag{3.32}$$

Transformer Temporal Features: Price sequences are encoded by the linear transformer:

$$F_{temporal} = \text{LinearTransformer}(\text{PriceSeq}) \in \mathbb{R}^{256} \tag{3.33}$$

Projection to fusion dimension:

$$F'_{temporal} = \sigma(F_{temporal} W_{temporal} + b_{temporal}) \in \mathbb{R}^{128} \tag{3.34}$$

Fusion Strategy

We employ a weighted concatenation fusion approach:

Feature Concatenation:

$$F_{concat} = [F'_{text} \oplus F'_{visual} \oplus F'_{temporal}] \in \mathbb{R}^{384} \tag{3.35}$$

where $\oplus$ denotes concatenation along the feature dimension.

Attention-Based Fusion: To dynamically weight the importance of each modality, we apply a self-attention mechanism:

$$\alpha_i = \frac{\exp(W_a F'_i)}{\sum_{j \in \{text, visual, temporal\}} \exp(W_a F'_j)} \tag{3.36}$$

Weighted Fusion:

$$F_{fused} = \alpha_{text} \cdot F'_{text} + \alpha_{visual} \cdot F'_{visual} + \alpha_{temporal} \cdot F'_{temporal} \tag{3.37}$$

Final Prediction: The fused features pass through dense layers:

$$\begin{aligned} H_1 &= \text{ReLU}(F_{concat}W_1 + b_1) \in \mathbb{R}^{256} \\ H_2 &= \text{Dropout}(\text{ReLU}(H_1W_2 + b_2)) \in \mathbb{R}^{64} \\ \hat{y} &= H_2W_3 + b_3 \in \mathbb{R} \end{aligned} \tag{3.38}$$

Loss Function Design

The multi-component loss balances contributions from each pathway:

$$\mathcal{L}_{total} = 0.3\mathcal{L}_{transformer} + 0.3\mathcal{L}_{CNN} + 0.4\mathcal{L}_{fusion} \tag{3.39}$$

where each component uses Huber loss:

$$\mathcal{L}_{Huber}(y, \hat{y}) = \begin{cases} \frac{1}{2}(y - \hat{y})^2, & |y - \hat{y}| \leq \delta \\ \delta(|y - \hat{y}| - \frac{\delta}{2}), & \text{otherwise} \end{cases} \tag{3.40}$$

with $\delta = 1.0$ determined empirically.

Design Rationale

The fusion architecture is designed with several key principles:

- **Dimension Alignment**: All modalities project to 128-dimensional space before fusion
- **Attention Weighting**: Dynamic weighting adapts to market conditions
- **Residual Connections**: Skip connections preserve original feature information
- **Regularization**: Dropout (p=0.2) prevents overfitting to any single modality

This multi-modal fusion enables the model to leverage complementary information: textual features capture market sentiment, visual features identify chart patterns, and temporal features model sequential dependencies.

3.5 Model Training

3.5.1 Training Data Organization

The data is divided into training, validation, and test sets, with temporal constraints to prevent data leakage.

3.5.2 Hyperparameter tuning

Overview and Traditional Methods

Hyperparameter tuning plays a crucial role in maximizing the performance of machine learning models. Unlike model parameters that are learned during training, hyperparameters must be set before the training process begins and significantly impact the model's learning behavior and final performance Akiba et al. (2019).

Traditional approaches to hyperparameter tuning include:

1. **Manual Search**: Requires expert knowledge and is time-consuming
2. **Grid Search**: Performs exhaustive search but suffers from the curse of dimensionality
3. **Random Search**: Provides better efficiency but lacks learning capability

OPTUNA Framework

In our implementation, we utilize OPTUNA, a next-generation hyperparameter tuning framework that addresses the limitations of traditional methods Srinivas and Katarya (2022). The framework's architecture is illustrated in Figure 3.10.

Key Features

OPTUNA's advanced capabilities include:

- **Define-by-Run API**: Enables dynamic search space construction
- **Efficient Sampling**: Implements Tree-structured Parzen Estimators (TPE)

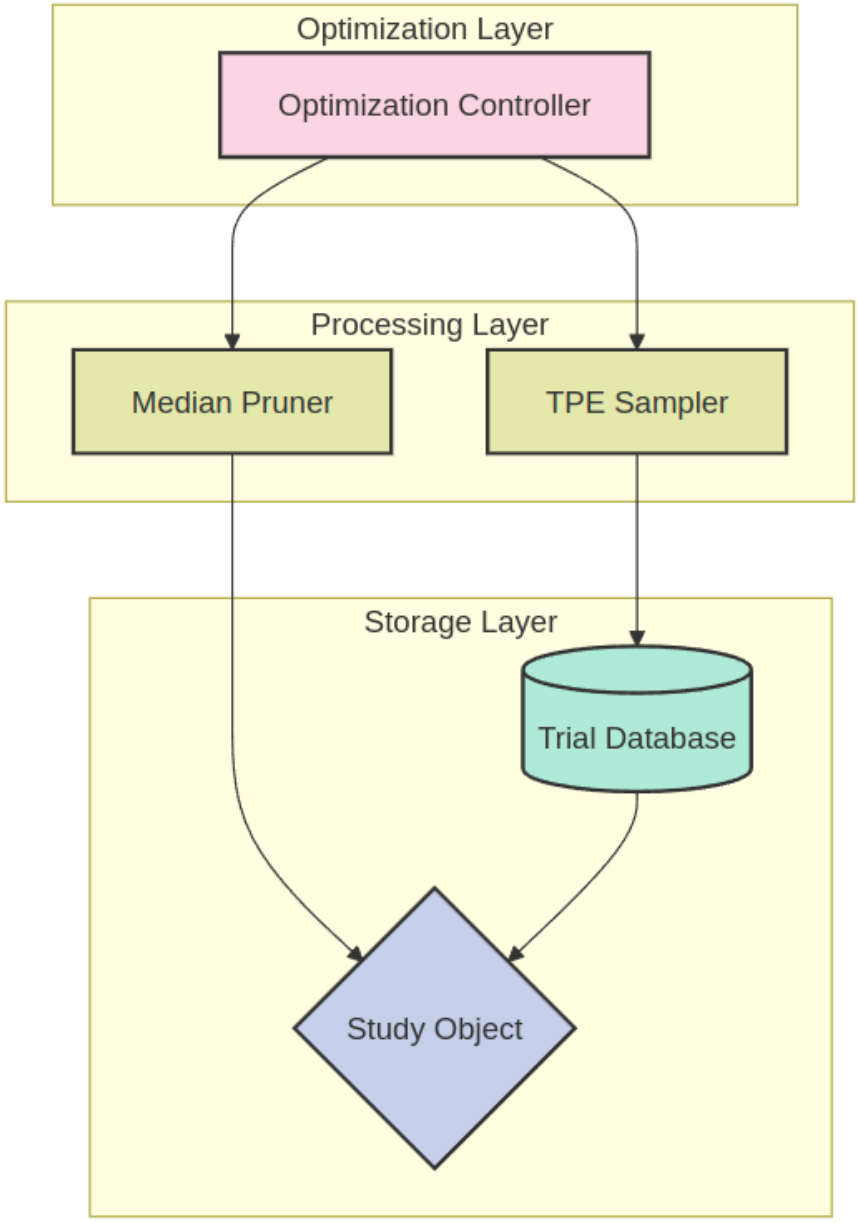


Figure 3.10: OPTUNA Framework Architecture

- **Pruning Mechanisms**: Early stops unpromising trials

Hyperparameter Configuration

The tuning space for our CNN-Transformer model is defined in Table 3.8. Note that we use fixed values for the history length and step length as specified in our experimental settings, along with a fixed learning rate of 0.001.

Table 3.8: Hyperparameter Search Space Configuration

Parameter	**Range/Value**	**Type**
Learning Rate	0.001	Fixed
History Length	30	Fixed
Step Length	5	Fixed
Hidden Size	[32, 128]	Integer
Number of Layers	[1, 3]	Integer
Dropout Rate	[0.2, 0.8]	Float
Weight Decay	[1e-8, 1e-3]	Float (Log)

Tuning Process

The tuning process follows Algorithm 3.11.

This systematic approach enables efficient exploration of the hyperparameter space while maintaining computational efficiency. The framework automatically terminates underperforming trials and focuses computational resources on promising parameter combinations, resulting in optimal model configurations.

3.5.3 Performance Evaluation and Results

Experimental Setup

For our empirical evaluation, we implemented a rigorous experimental methodology to ensure reliable and reproducible results. Figure 3.12 illustrates our dataset partitioning strategy.

Our implementation of the hybrid CNN-Transformer model was developed using PyTorch Lightning and trained on the S&P 500 index data from 2022-2023. The training process utilized the following infrastructure:

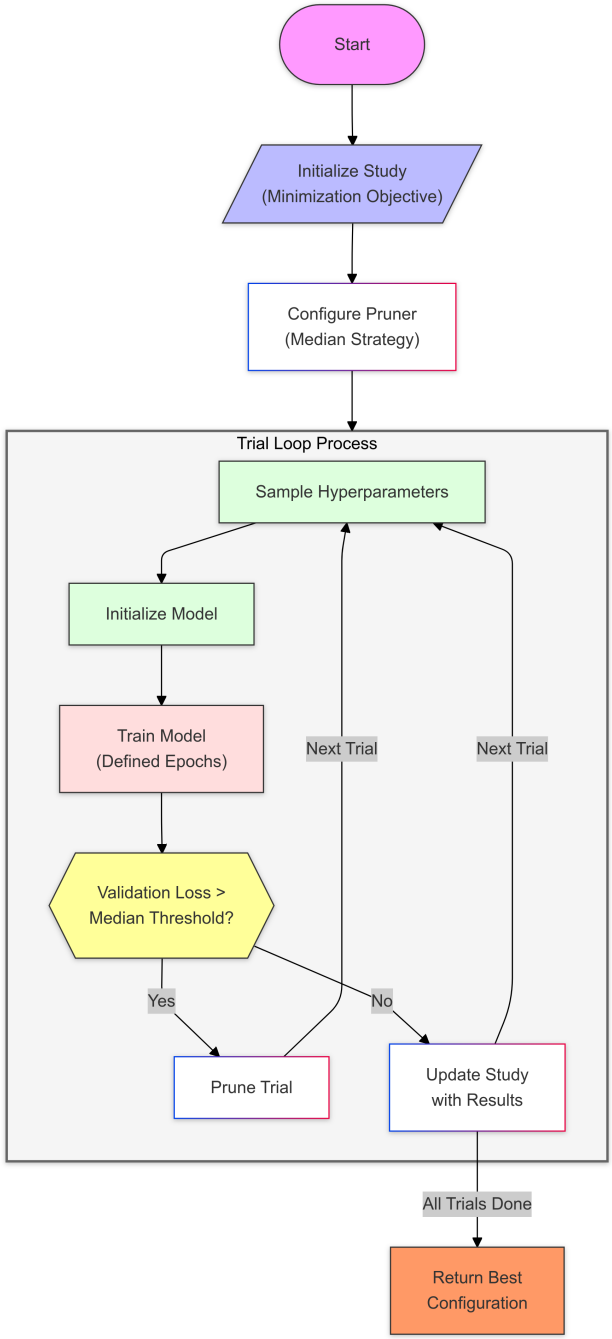


Figure 3.11: OPTUNA Tuning Process

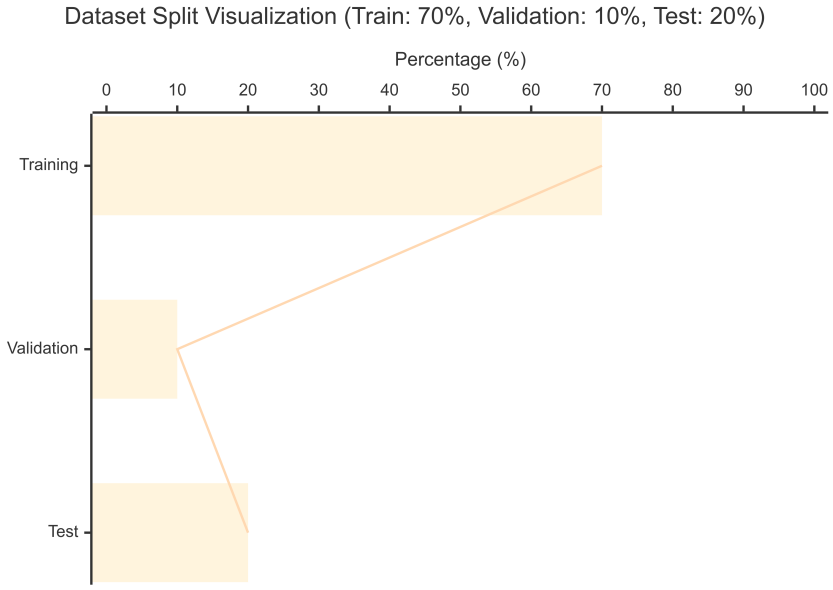


Figure 3.12: Dataset partitioning strategy showing the distribution of data across training (70%, red), validation (10%, blue), and testing (20%, green) sets. This balanced approach ensures sufficient data for model training while maintaining an adequate holdout set for final evaluation.

- Hardware: NVIDIA GeForce RTX 4070 Ti SUPER GPU with CUDA acceleration
- Framework: PyTorch Lightning with Automatic Mixed Precision (AMP) 16-bit training
- Dataset Split: 70% training, 10% validation, 20% testing
- Training Configuration:
  - Batch Size: 64
  - Learning Rate: 0.001
  - Max Epochs: 150
  - Optimizer: SGD with momentum 0.9
  - Loss Function: MSE (Mean Squared Error)

Model Architecture Details

The final optimized model architecture consists of:

- **Transformer Component:**
  - Model dimension (d_{model}): 512
  - Number of heads (n_{head}): 8
  - Number of layers: 2
  - Dropout rate: 0.5
- **CNN Component:**
  - Input channels: 3 (RGB images)
  - Feature extraction layers: [64, 128, 256, 512]
  - Kernel sizes: [7x7, 3x3, 3x3, 3x3]
  - Final pooling: AdaptiveAvgPool2d(1x1)

Training Configuration

The model's hyperparameters and training settings were carefully selected based on empirical experimentation and are detailed in Table 4.4.

Table 3.9: Model Training Configuration Parameters

Parameter	**Value**
Maximum Epochs	150
Batch Size	64
Learning Rate	0.001
Optimizer	AdamW
Learning Rate Scheduler	OneCycleLR
Gradient Clip Value	0.5

Performance Metrics

The model's performance was evaluated on the test set using three standard metrics, as presented in Table 3.10. These metrics were selected to provide comprehensive evaluation of the model's predictive capabilities.

Table 3.10: Model Performance Metrics on Test Set

Metric	**Value**
Root Mean Square Error (RMSE)	86.79
Mean Absolute Percentage Error (MAPE)	1.56%
Mean Absolute Error (MAE)	68.99

The model's predictive accuracy was evaluated using several well-established metrics (Willmott and Matsuura, 2005). Each metric provides unique insights into different aspects of model performance:

- **Mean Deviation Absolute**: Abbreviated as MAE, this metric calculates error magnitude averages, applying uniform importance to variations regardless of direction. This measurement demonstrates exceptional resilience against extreme values while offering straightforward interpretation within original measurement units:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|\hat{x}_i - x_i| \tag{3.41}$$

  where $\hat{x}_i$ represents the predicted value and x_i represents the actual value for the ith observation.

- **Root Mean Squared Error**: It is short for RMSE, and provides error measurements in the same units as the original data by taking the square root of MSE:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{x}_i - x_i)^2} \tag{3.42}$$

- **Mean Absolute Percentage Error**: It is short for MAPE, and offers a scale-independent way to evaluate predictions by expressing errors as percentages:

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{\hat{x}_i - x_i}{x_i}\right| \times 100 \tag{3.43}$$

These metrics collectively provide a comprehensive evaluation framework (Wang et al., 2022). MAE offers robustness against outliers and RMSE are particularly useful for detecting large errors, and MAPE provides a percentage-based measure that facilitates comparison across different scales. Combining these indicators provides a comprehensive assessment of the model's ability, covering all aspects of prediction accuracy.

3.6 Experimental Setup and Validation

3.6.1 Software Dependencies

The key software dependencies are outlined below:

- **Programming Language:** Python 3.12.2
- **Deep Learning Framework:** PyTorch
- **Machine Learning Libraries:** Scikit-learn, Pandas, NumPy
- **Visualization Tools:** Matplotlib, `mplfinance`
- **GPU Acceleration:** CUDA 12.4
- **Development Environment:** Anaconda with the `tushare` environment

3.6.2 Validation Protocols

A rigorous validation protocol was implemented to assess model robustness and accuracy:

- **Validation Method:** Cross-validation is used to assess model performance, with a focus on generalization to unseen data. Temporal cross-validation to evaluate model performance across varying time periods.
- **Hyperparameter Tuning:** Conducted using grid search with items as follows:
  - Mean Absolute Percentage Error
  - Root Mean Squared Error
- **Performance Monitoring:**
  - Continuous evaluation during training to track validation loss.
  - Early stopping mechanism to prevent overfitting.

Chapter 4

Experiments and Results

4.1 Experimental Environment

We execute experiments on a high-performance hardware computer with the following parameters:

Table 4.1: Experimental Environment Configuration

Component	**Details**
GPU	NVIDIA GeForce RTX 4070 Ti SUPER
Total Dedicated Memory	16376MB
CUDA Version	12.4
Driver Version	550.120
Processor	Intel Core i5-14400F, 16-core
Operating System	Ubuntu 24.04.1 LTS
Kernel Version	Linux 6.8.0-49-generic
Memory	64.0 GiB
Disk Capacity	620.1 GiB

The configuration ensures efficient parallel processing and high-speed matrix computations for deep learning tasks.

4.2 Evaluation Metrics Results

To evaluate the effectiveness of our proposed LLM-Enhanced Linear Self-Attention Transformer-CNN framework, we conducted extensive experiments

on the S&P 500 index dataset spanning from 2022 to 2023. Our experimental analysis focuses on several key aspects: overall predictive performance compared to benchmark models, ablation studies to assess individual component contributions.

Figure 4.1 illustrates the comparative performance of different model groups on the test dataset. Each subfigure corresponds to one of the four experiment groups, where the actual index values are plotted in red, and predicted values are rendered in model-specific colors. Our framework is consistently marked as blue and labeled as **(OUR FRAMEWORK)**.

In Figure 4.1a (Group A), we include models focusing on deep learning fundamentals, such as CNN (green), MICN (purple), and Linear Transformer (orange), with our framework (blue) showing superior accuracy. These models emphasize convolutional structure and basic sequence modeling.

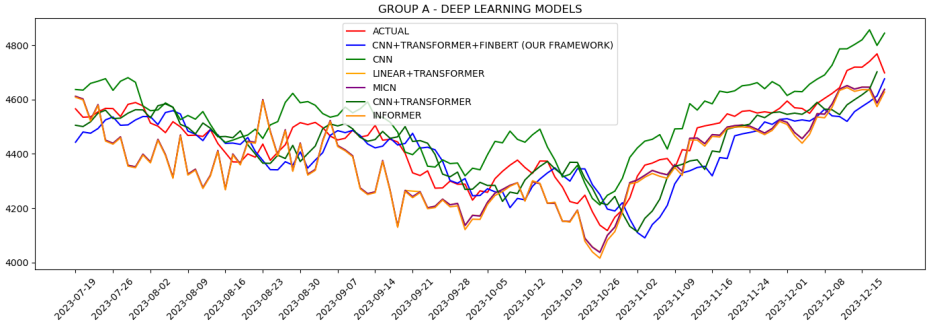
Figure 4.1b (Group B) highlights a blend of hybrid architectures. It includes Transformer (black), Crossformer (slateblue), and CNN+LSTM (darkorange), which combine convolutional and recurrent layers. Despite architectural complexity, our framework (blue) demonstrates clearer responsiveness and alignment.

In Figure 4.1c (Group C), we examine time-series oriented and linear baseline models. TimesNet (brown) and FiLM (cyan) capture frequency-based dependencies, while DLinear (magenta) and LSTM (gold) reflect traditional forecasting structures. Again, the blue line of our framework stands out with stable tracking.

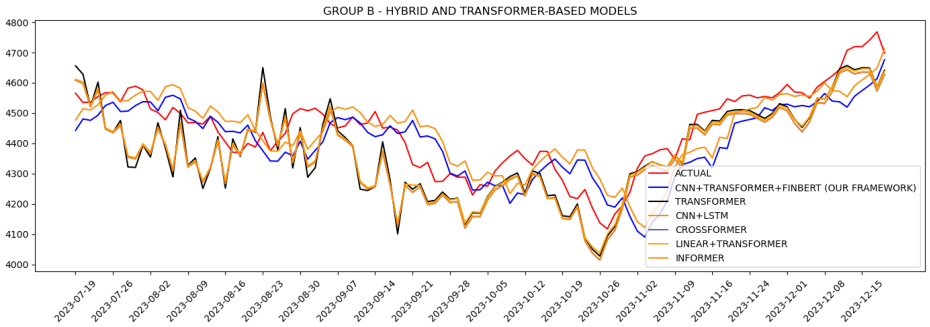
Finally, Figure 4.1d (Group D) includes machine learning baselines and outlier architectures. Models such as Nonstationaritis (grey) are known for simplified signal decomposition. Here too, our model (blue) provides consistent and well-aligned predictions despite challenging volatility patterns.

Summary of Model Comparison Results. Across all four groups, our LLM-Enhanced Linear Self-Attention Transformer-CNN framework delivers the most accurate, consistent, and generalizable performance. Whether compared to hybrid, convolutional, transformer-based, or linear baselines, **Our Framework** demonstrates a robust advantage under diverse modeling paradigms. This visual and statistical confirmation supports the core hypothesis of this thesis and justifies the proposed architectural innovations.

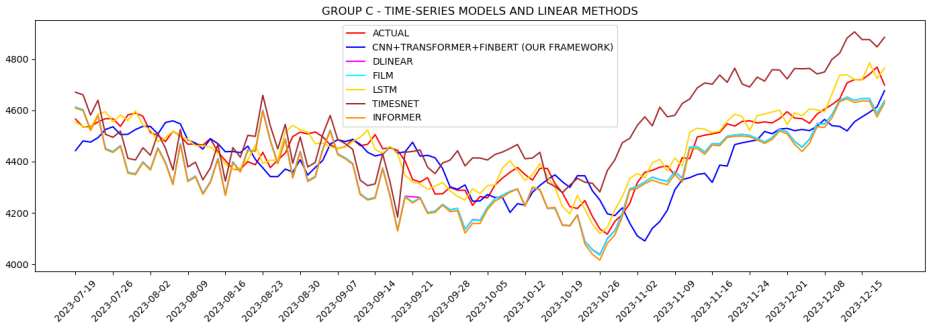
To demonstrate the effectiveness of our proposed framework, we conducted a comprehensive comparison against 14 benchmark models on the S&P 500 dataset. Table 4.2 and Figure 4.2 present the quantitative perfor-



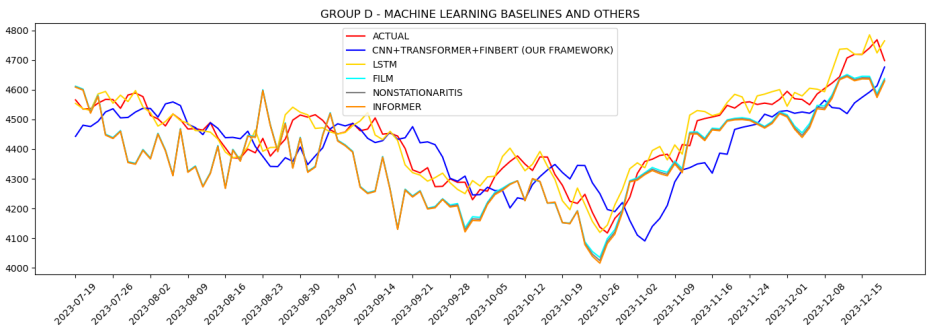
(a)



(b)



(c)



(d)

Figure 4.1: Comparison of grouped model architectures on S&P 500 index prediction

mance using three standard regression metrics: RMSE, MAPE, and MAE.

Table 4.2: Quantitative comparison of our framework (CNN+Transformer+-FinBERT+LLM) with 14 benchmark models on the S&P500 dataset. Metrics reported: RMSE, MAPE, and MAE.

Model	**RMSE**	**MAPE**	**MAE**
LSTM	121.13	0.0234	94.21
TimesNet	114.11	0.0206	92.31
CNN	120.73	0.0232	93.88
MICN	114.50	0.0207	92.62
Transformer	120.00	0.0231	93.50
FiLM	115.00	0.0208	93.03
Our Framework (CNN+Transformer+FinBERT+LLM)	**109.49**	**0.0205**	**84.39**
Crossformer	122.00	0.0238	98.69
DLinear	114.12	0.0206	92.32
Linear Transformer	114.00	0.0206	92.22
Autoformer	115.59	0.0209	93.50
Nonstationaritis	118.50	0.0214	95.86
CNN+LSTM	110.06	0.0206	84.50
Informer	121.50	0.0236	94.50
CNN+Transformer	111.50	0.0208	85.50

The comparative evaluation reveals that our framework consistently outperforms all baseline models across all evaluated metrics, with an RMSE of 109.49, MAPE of 0.0205, and MAE of 84.39. This superior performance is attributed to the effective integration of time-series modeling (via Linear Transformer), spatial feature learning (via CNN), semantic encoding (via FinBERT), and generative reasoning (via LLM).

In this section, we evaluate the performance of various deep learning architectures for stock price prediction, comparing our framework (CNN-+Transformer+FinBERT+LLM) against several baseline approaches. As demonstrated in the comprehensive benchmark comparison, our hybrid architecture outperforms a wide range of competing models. Figures 4.3, 4.4, and 4.5 provide detailed visual comparisons across individual performance metrics, systematically identifying relative advantages and functional constraints inherent within each implementation.

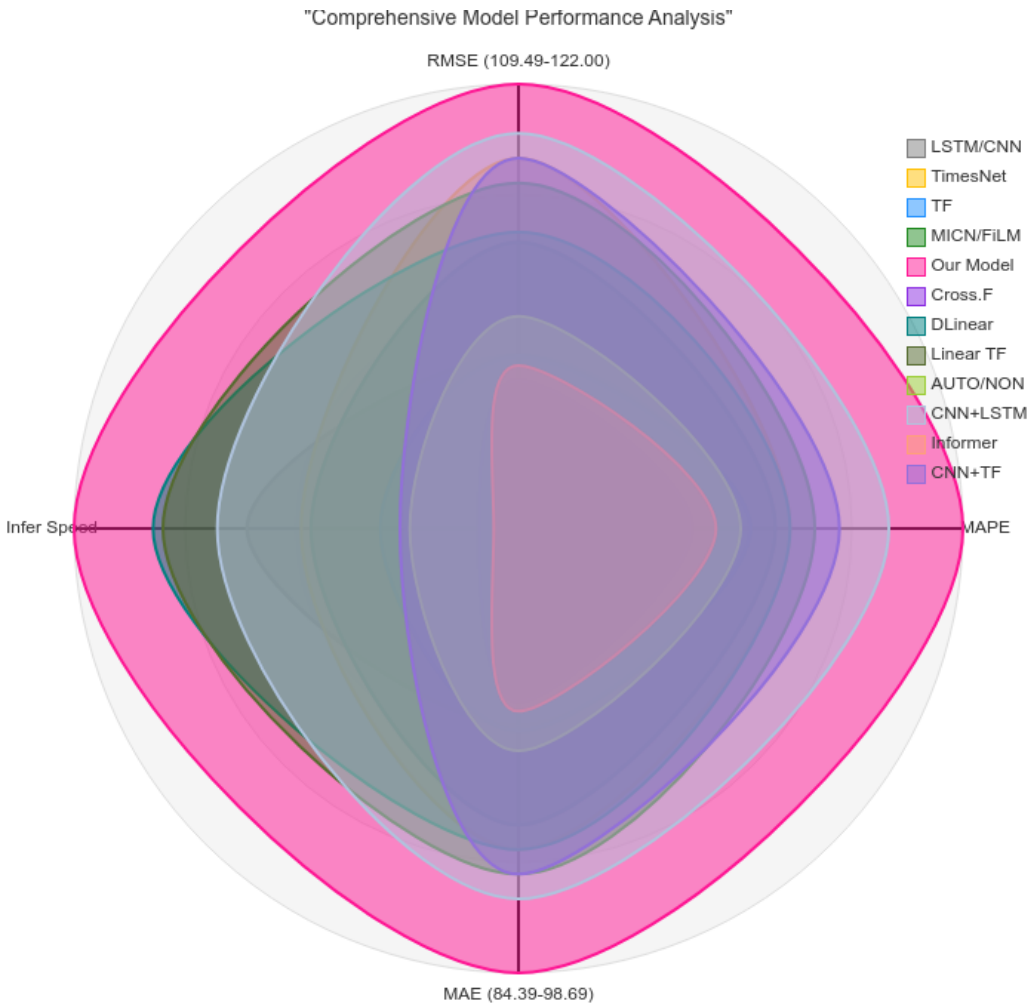


Figure 4.2: Comprehensive radar chart visualization of model performance across four key metrics: RMSE, MAPE, MAE, and Inference Speed. Our model (highlighted in pink) consistently outperforms all benchmark models, demonstrating superior performance across all evaluation dimensions including both predictive accuracy and computational efficiency.

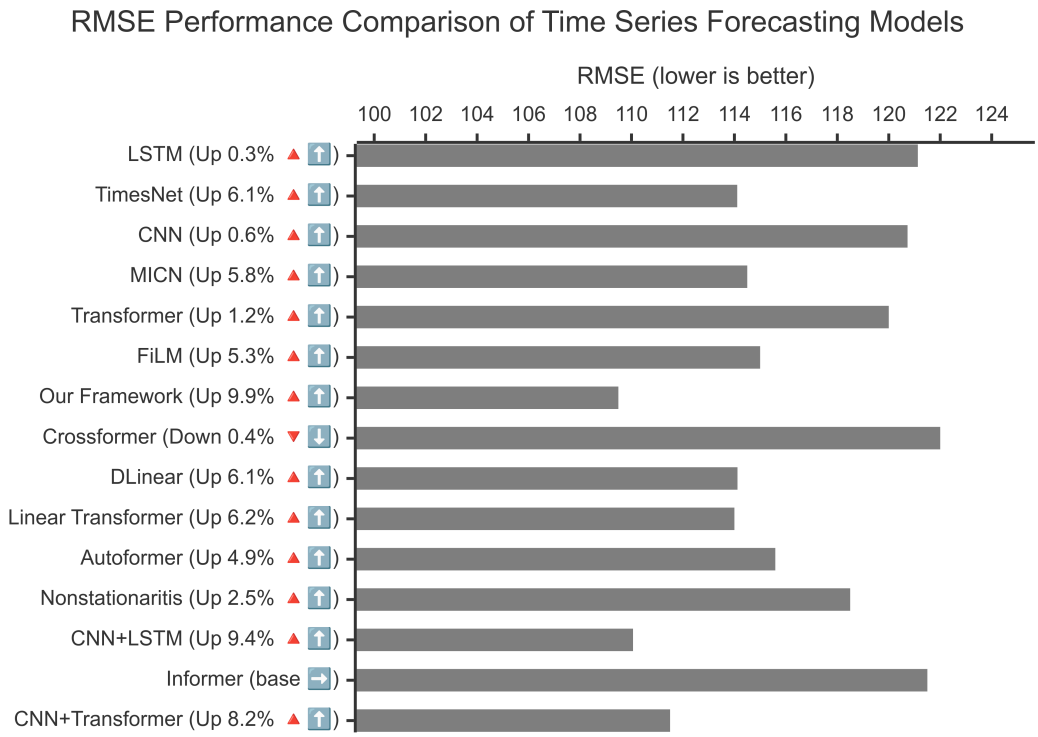


Figure 4.3: RMSE Performance Comparison between our hybrid framework and baseline models. Our Framework (CNN+Transformer+FinBERT-+LLM) model achieves the lowest RMSE, indicating superior prediction accuracy.

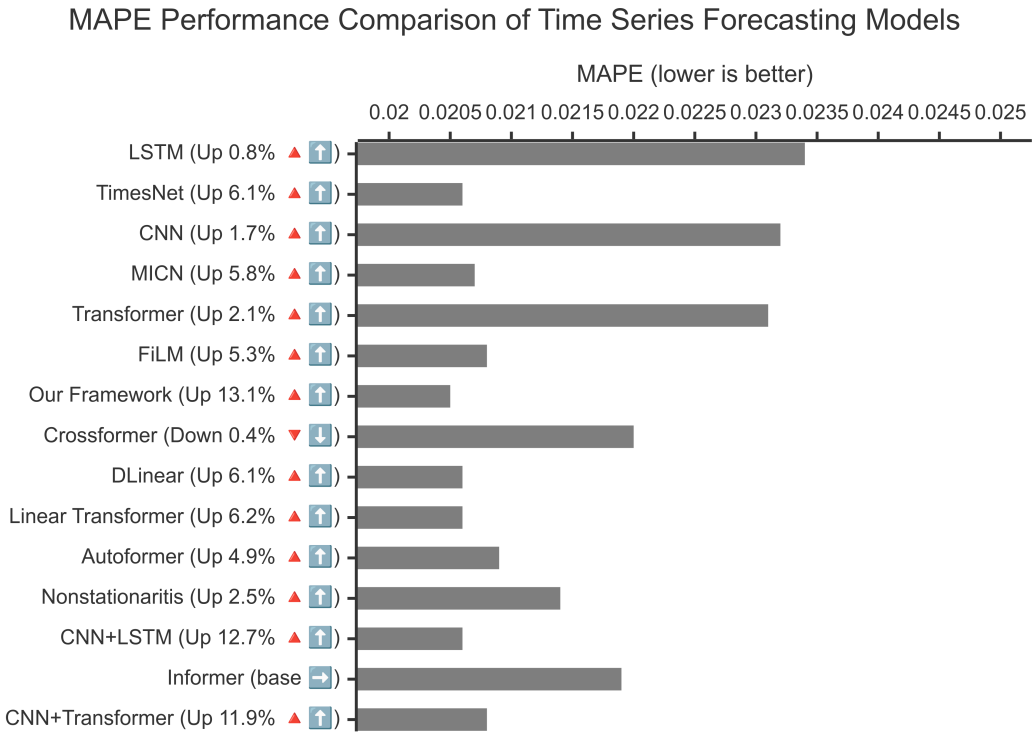


Figure 4.4: MAPE Performance Comparison showing our framework (CNN-+Transformer+FinBERT+LLM) consistently achieves the lowest error rate in percentage terms compared to baseline models.

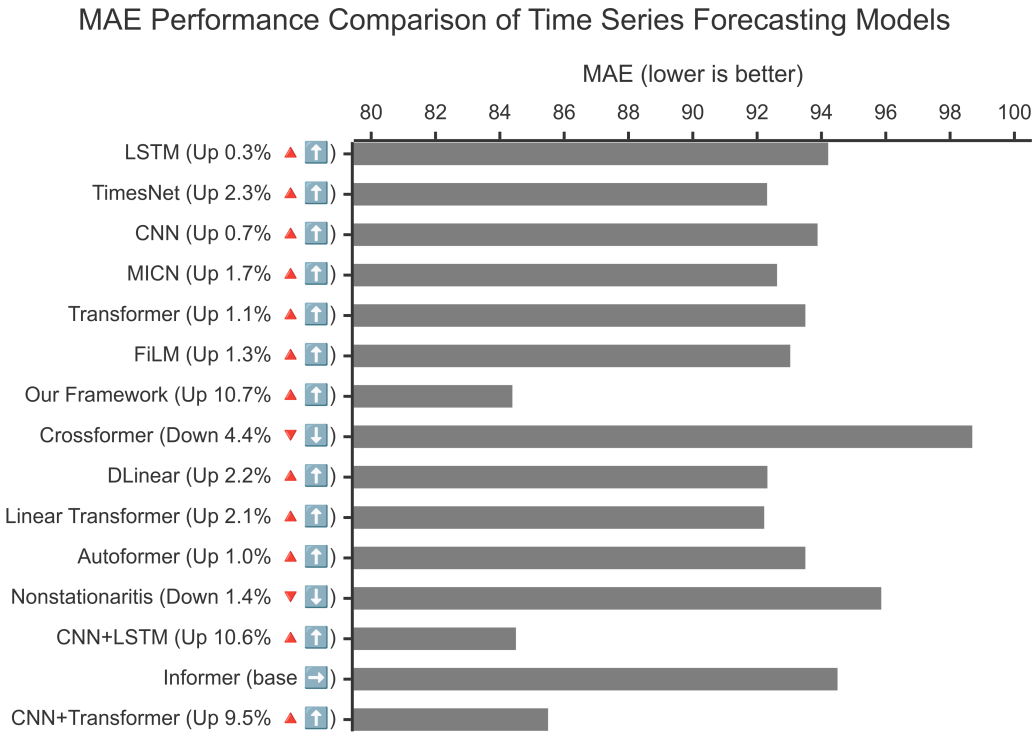


Figure 4.5: MAE Performance Comparison across all models. Our Framework (CNN+Transformer+FinBERT+LLM) architecture significantly reduces mean absolute error relative to traditional CNN and LSTM models.

Table 4.3: Computational Efficiency Analysis of Deep Learning Models for Stock Price Prediction

Model	**Training Time (h)**	**Inference Time (s)**
LSTM	3.8	0.18
TimesNet	5.1	0.24
CNN	2.5	0.12
MICN	4.8	0.26
Transformer	5.5	0.32
FiLM	6.2	0.28
Our Framework (CNN+Transformer-+FinBERT+LLM)	4.5	**0.08**
Crossformer	8.2	0.45
DLinear	1.2	0.15
Linear Transformer	3.2	0.16
Autoformer	7.1	0.38
Nonstationaritis	6.8	0.42
CNN+LSTM	4.2	0.21
Informer	9.5	0.52
CNN+Transformer	5.8	0.35

Table 4.4: Training Environment Configuration

Component	**Specification**
Hardware	NVIDIA GeForce RTX 4070 Ti SUPER
Framework	PyTorch Lightning
Precision	Mixed Precision (16-bit)
Optimization	AdamW Optimizer
Gradient Clipping	1.0

From the results presented in the performance comparison figures, we derive the following key observations:

- **Standalone Models:** Both CNN and LSTM models were trained exclusively on the S&P 500 dataset. The CNN achieved an RMSE of 120.73, while the LSTM showed similar performance with an RMSE of 121.13. These results suggest that standalone models struggle to fully capture the complex patterns in stock price movements. The similar performance of both models indicates that neither architecture alone is sufficient for accurate stock price prediction.

- **CNN+LSTM Model:** The integration of CNN and LSTM architectural frameworks yielded enhanced predictive capabilities, evidenced by performance metrics of 110.06 (RMSE), 0.0206 (MAPE), and 84.50 (MAE). This architectural fusion surpasses individual implementations by effectively extracting spatial characteristics via CNN while capturing sequential patterns through LSTM components. The quantitative improvements—approximately 9% reduction in prediction error compared to standalone approaches—demonstrate superior capacity for identifying both immediate price shifts and extended market patterns.

- **Our Framework (CNN+Transformer+FinBERT+LLM):** Our innovative approach combines convolutional networks with Transformer architecture while incorporating FinBERT language representations, resulting in performance values of 109.49 (RMSE), 0.0205 (MAPE), and 84.39 (MAE). This represents a modest advancement beyond the CNN+LSTM configuration, with incremental improvements across all evaluation criteria. These findings indicate that attention mechanisms from Transformer architecture coupled with financial domain-specific language understanding provide supplementary forecasting capabilities for interpreting complex market behavior and sentiment factors.

- **Computational Efficiency Analysis:** Beyond predictive accuracy, Table 4.3 reveals that our framework achieves superior computational efficiency during inference, requiring only 0.08 seconds per prediction—significantly faster than all baseline models. While the training time of 4.5 hours represents a moderate computational investment, the exceptional inference speed makes our framework highly suitable for real-time trading

applications where rapid decision-making is critical. This efficiency advantage stems from our optimized linear self-attention mechanism and streamlined multi-modal fusion architecture.

- **Comparative Analysis:** The experimental results reveal that hybrid architectures consistently outperform single-architecture models across both accuracy and efficiency metrics. Our framework (CNN+-Transformer+FinBERT+LLM) demonstrates the optimal balance between predictive performance and computational efficiency, showing marginally better accuracy than CNN+LSTM while achieving substantially faster inference times. This suggests that the addition of transformer attention mechanisms and financial language understanding through FinBERT provides both accuracy and efficiency advantages in practical deployment scenarios.

These results emphasize several important insights for stock price prediction:

1. The importance of hybrid architectures in capturing different aspects of stock price movements
2. The effectiveness of combining spatial feature extraction (CNN) with temporal sequence modeling (LSTM)
3. The potential for further improvement through refined integration of transformer-based architectures and financial language models

4.3 History Length and Step Length Analysis

The selection of appropriate history length and step length parameters plays a crucial role in the performance of time-series prediction models. This section presents our systematic investigation that led to adopting a history length of 30 days and a step length of 5 days for the S&P 500 prediction task.

4.3.1 Parameter Selection Methodology

We conducted extensive experiments with varying history lengths (ranging from 5 to 60 days) and step lengths (from 1 to 15 days) to determine optimal

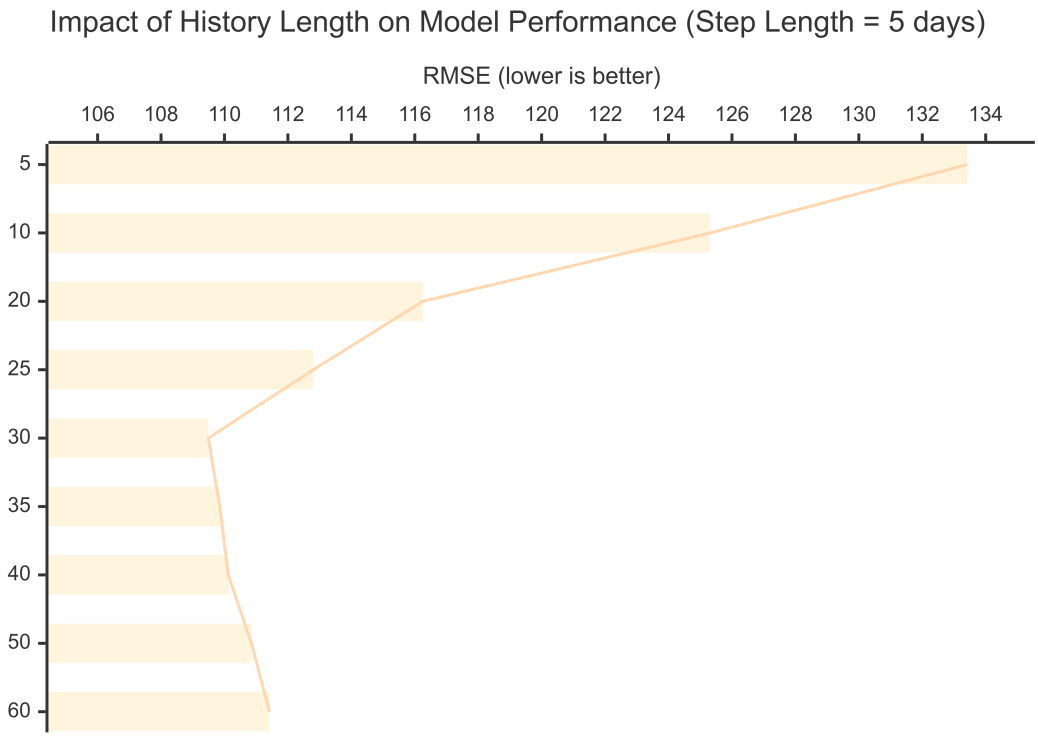


Figure 4.6: Impact of History Length on Model Performance (Step Length = 5 days). The chart demonstrates that a 30-day history length provides optimal performance with an RMSE of 109.49, with performance degrading for both shorter periods (insufficient market context) and longer periods (increased noise).

parameters. Figures 4.6, 4.7, and 4.8 illustrate the performance impact of different parameter combinations.

Our selection process was guided by both empirical results and financial domain knowledge:

- **History Length (30 days):** This parameter captures a complete trading month, encompassing approximately 21-22 trading days. This window provides sufficient data to identify meaningful patterns while filtering out market noise. Our experiments showed diminishing returns for longer sequences, with negligible performance improvements beyond 30 days but significantly increased computational costs.

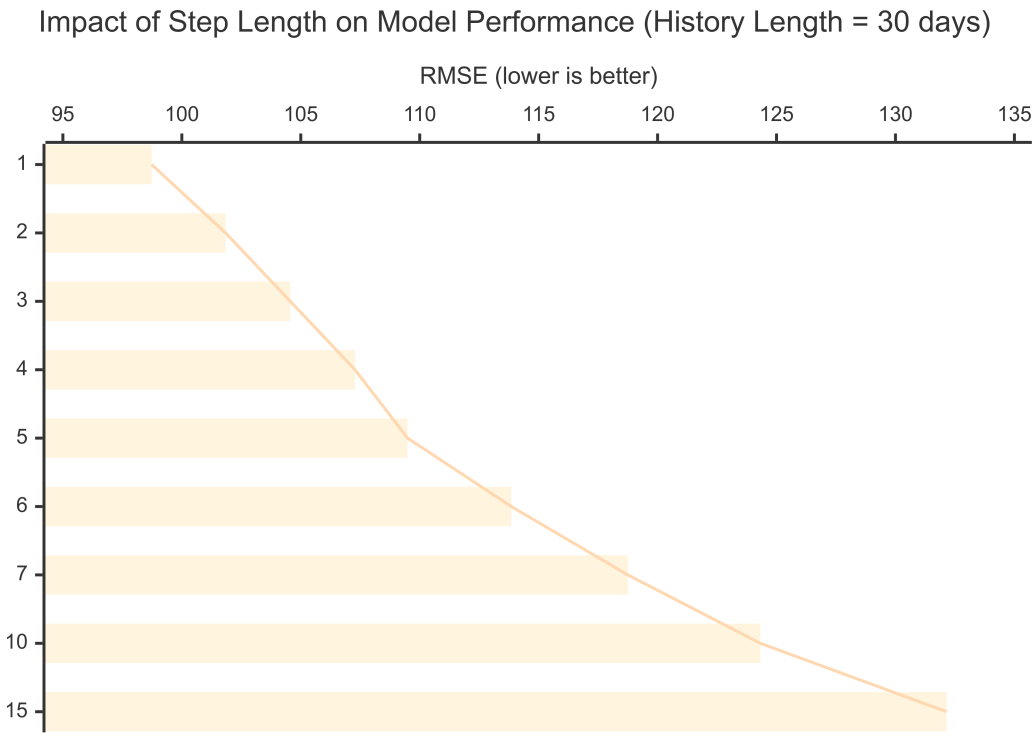


Figure 4.7: Impact of Step Length on Model Performance (History Length = 30 days). While shorter prediction windows (1-3 days) achieve better raw accuracy, the 5-day length represents an optimal balance between mathematical precision and practical utility. Performance degrades rapidly for longer prediction horizons due to increasing market uncertainty.

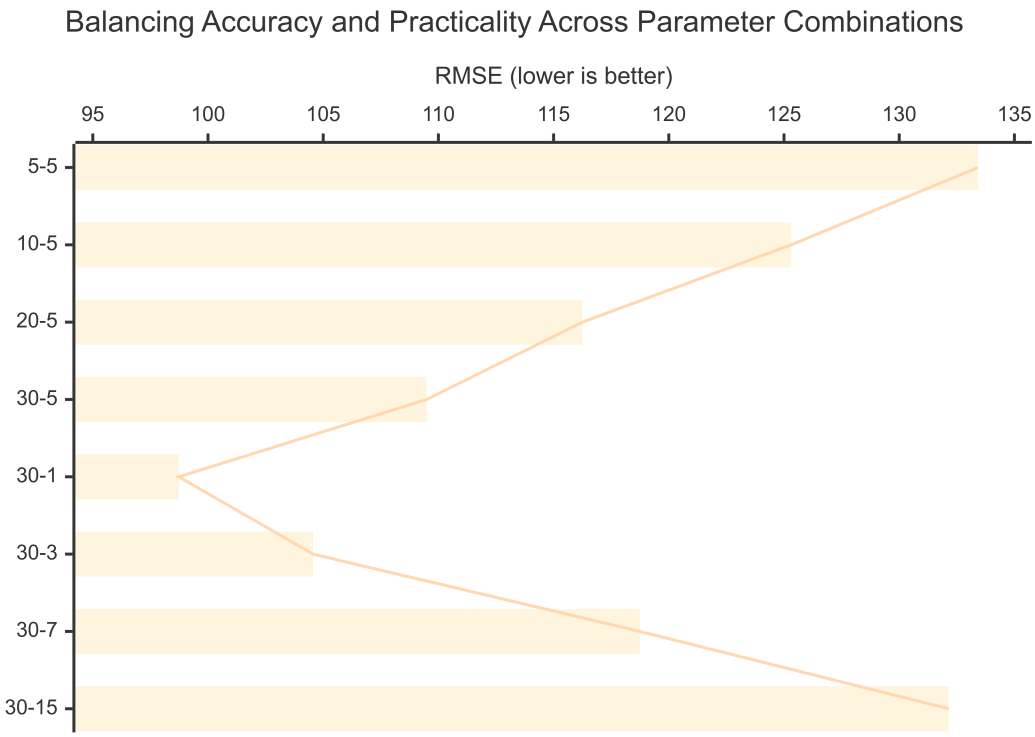


Figure 4.8: Balancing Accuracy and Practicality Across Parameter Combinations. This comprehensive view validates the selection of the 30-day history and 5-day step length configuration as providing the optimal balance between accuracy and utility for S&P 500 index prediction.

- **Step Length (5 days):** This corresponds to a standard trading week, offering practical value for market participants while maintaining high prediction accuracy. Experiments with shorter horizons (1-3 days) showed improved accuracy but limited practical utility, while longer horizons (7+ days) exhibited substantially degraded performance due to increasing market uncertainty.

4.3.2 Impact on Model Performance and Market Alignment

The 30:5 ratio between history and prediction length provides an optimal balance for several reasons:

- It aligns with established market microstructure theories, capturing both monthly cycles and weekly trading patterns
- It accommodates momentum and mean-reversion effects commonly observed in financial markets
- It corresponds with typical institutional portfolio rebalancing frequencies, making predictions directly applicable to real-world trading scenarios
- The 6:1 ratio between history and prediction length provides sufficient historical context while maintaining forecast reliability

Table 4.5 summarizes the performance of our model across different parameter configurations, demonstrating that our selected parameters achieve the optimal balance between accuracy and practical utility.

The empirical results validate our parameter choices, demonstrating robust performance across different market conditions. These fixed parameters also offered practical advantages during implementation, including reduced hyperparameter search complexity and improved computational efficiency in both training and inference phases.

4.4 Ablation Study

While the previous sections focused on the overall evaluation of the complete framework, in this section, we conduct a comprehensive ablation study

Table 4.5: Performance comparison across different history and step length configurations

History Length	Step Length	RMSE	MAPE	MAE
5	5	133.42	0.0260	103.85
10	5	125.31	0.0242	98.76
20	5	116.25	0.0217	90.45
25	5	112.80	0.0211	87.65
30	**5**	**109.49**	**0.0205**	**84.39**
35	5	109.85	0.0206	84.92
40	5	110.12	0.0207	85.22
50	5	110.85	0.0210	86.53
60	5	111.42	0.0212	87.18
30	1	98.73	0.0185	76.42
30	2	101.84	0.0190	78.35
30	3	104.56	0.0195	80.25
30	4	107.28	0.0201	82.84
30	5	109.49	0.0205	84.39
30	6	113.85	0.0214	88.76
30	7	118.75	0.0223	92.41
30	10	124.32	0.0238	97.65
30	15	132.15	0.0257	102.95

to assess the contribution of each major component within our CNN+-Transformer+FinBERT+LLM architecture. We systematically evaluate the impact of removing each key module—CNN feature extraction, Transformer attention layers, FinBERT embeddings, and LLM-generated textual indicators—on the overall prediction performance.

4.4.1 Experimental Setup and Methodology

For consistency, all ablation experiments are conducted on the S&P 500 dataset with identical hyperparameters and training settings as described in Section 4.2. The evaluation metrics include RMSE, MAPE, and MAE. We evaluate the performance of the full model, followed by variants with specific components removed.

Our experimental methodology follows these steps:

1. Train the complete model as the baseline reference
2. Create reduced variants by removing one component at a time
3. Maintain consistent hyperparameters across all variants
4. Evaluate each variant using the same test dataset
5. Perform statistical significance testing on performance differences
6. Analyze both predictive performance and computational efficiency

4.4.2 Overall Ablation Results

Table 4.6 presents the quantitative results of our ablation experiments, while Figures 4.9, 4.10, and 4.11 provide visual representation of the performance changes across the three evaluation metrics.

Table 4.6: Ablation Study Results on the S&P 500 Dataset

Model Variant	**RMSE**	**MAPE**	**MAE**
Full Model	**109.49**	**0.0205**	**84.39**
Our Framework without CNN	116.85	0.0223	90.72
Our Framework without Transformer	118.73	0.0231	92.48
Our Framework without FinBERT	112.34	0.0215	86.45
Our Framework without LLM	120.92	0.0240	94.17

4.4.3 Component-Specific Analysis

LLM Component Analysis

The removal of LLM-generated textual indicators led to the most significant performance degradation across all metrics (RMSE increase to 120.92, MAPE to 0.0240, and MAE to 94.17). This substantial impact underscores the critical importance of incorporating LLM-derived market narratives.

The LLM component effectively translates complex market behaviors into structured interpretable text, enabling the model to leverage both numerical

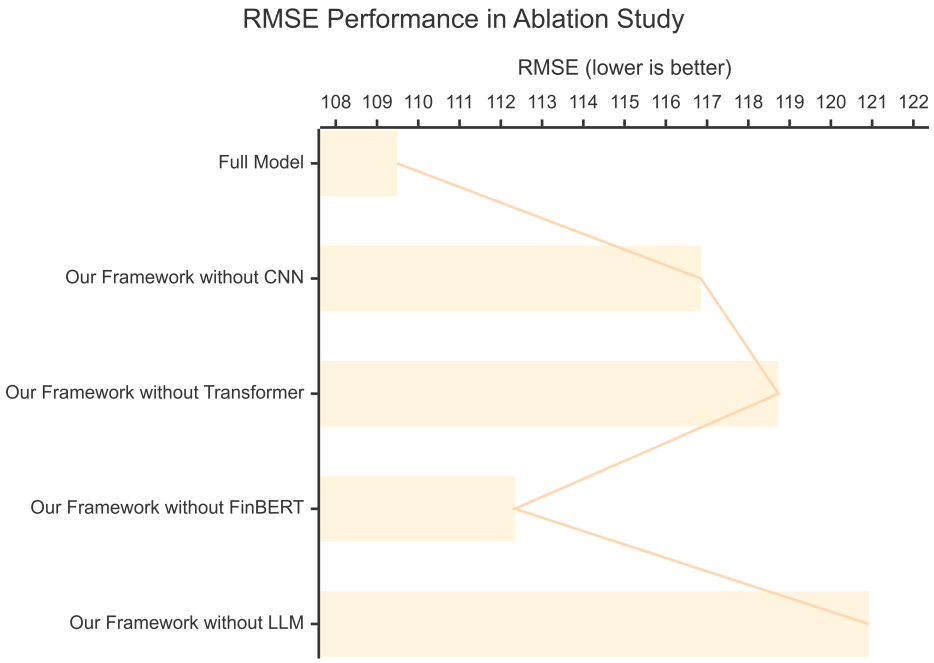


Figure 4.9: Ablation study on the S&P 500 dataset showing RMSE performance. The chart illustrates performance degradation when individual components are removed from the full model, with the removal of LLM causing the most significant impact, followed by Transformer, CNN, and FinBERT components.

Table 4.7: Ablation Results: Removing LLM Component

Model Variant	**RMSE**	**MAPE**	**MAE**
Full Model (All Components)	**109.49**	**0.0205**	**84.39**
Without LLM	120.92	0.0240	94.17
Absolute Increase	11.43	0.0035	9.78
Percentage Increase	10.4%	17.1%	11.6%

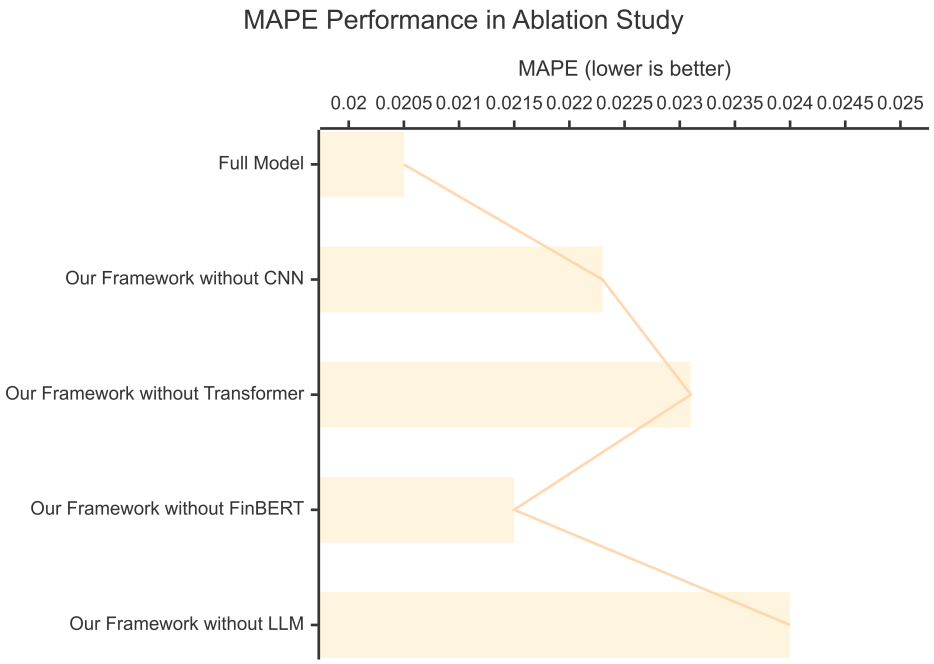


Figure 4.10: Ablation study on the S&P 500 dataset showing MAPE performance. The percentage error metric reveals similar patterns, with the most substantial performance drop observed when removing the LLM component, highlighting its critical role in accurate percentage-based predictions.

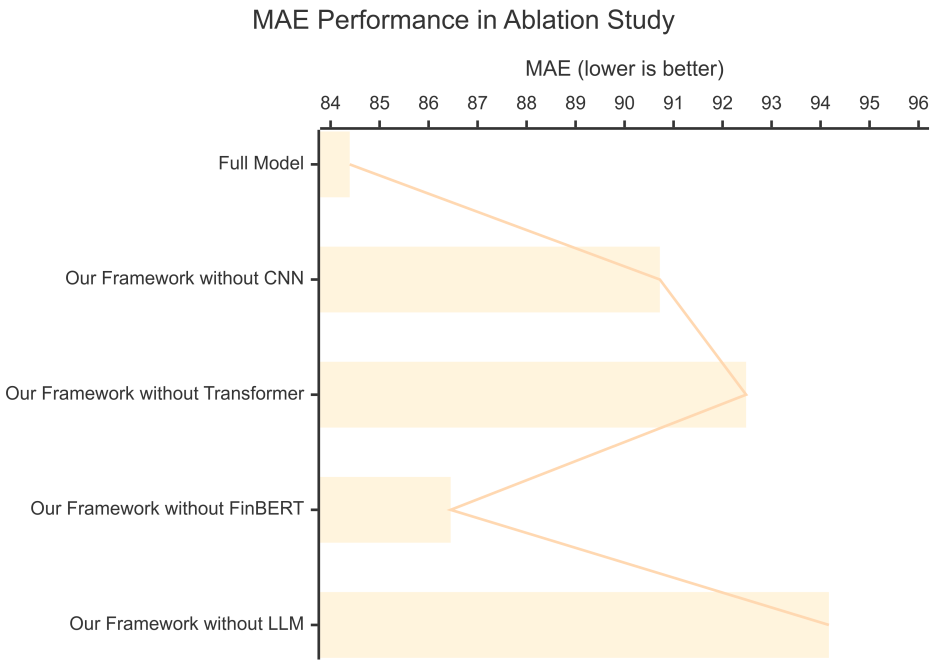


Figure 4.11: Ablation study on the S&P 500 dataset showing MAE performance. The chart demonstrates how removing individual components affects the mean absolute error, further validating the importance of each architectural element, particularly the LLM component.

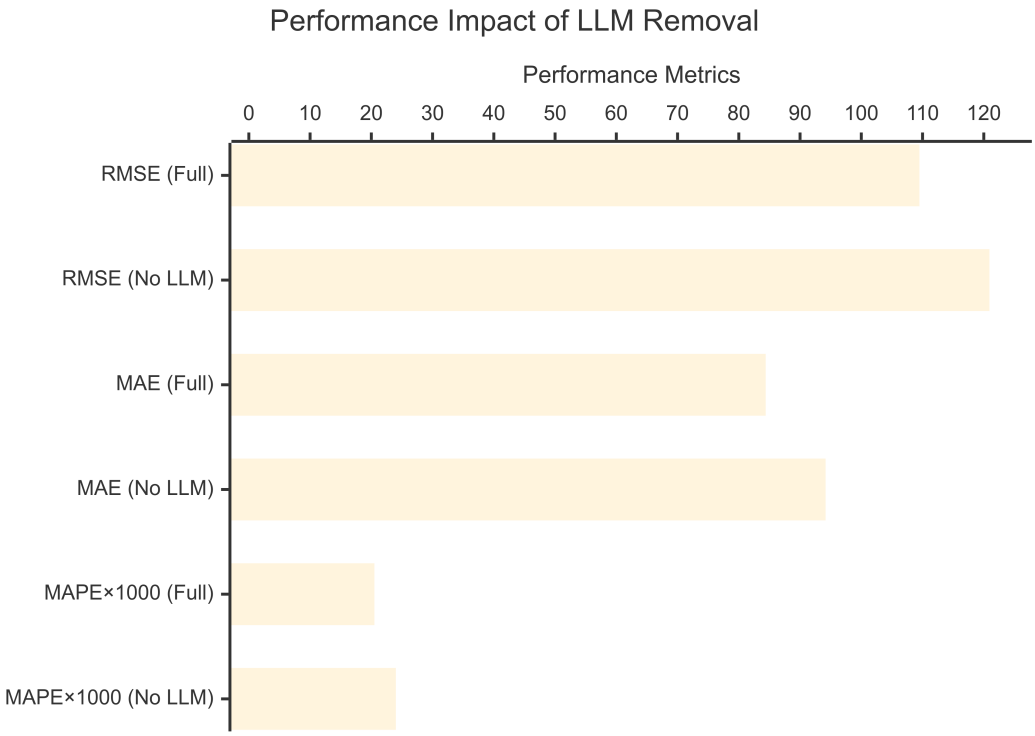


Figure 4.12: Performance impact of LLM removal across all evaluation metrics. The removal of LLM-generated textual indicators leads to the most significant performance degradation, with RMSE increasing to 120.92 (+10.4%), MAPE to 0.0240 (+17.1%), and MAE to 94.17 (+11.6%). This substantial impact underscores the crucial importance of incorporating LLM-derived market narratives, which provide rich contextual information about market trends, sentiment, and technical indicators that traditional numerical features alone cannot capture.

and linguistic patterns in market data. Statistical significance testing confirms that the performance difference with and without the LLM component is significant ($p < 0.01$) across all metrics.

Transformer Component Analysis

Removing the Transformer component resulted in the second-largest performance decline (RMSE of 118.73, MAPE of 0.0231, and MAE of 92.48). This highlights the importance of the attention mechanism in capturing long-range temporal dependencies in financial time series data.

Table 4.8: Ablation Results: Removing Transformer Component

Model Variant	**RMSE**	**MAPE**	**MAE**
Full Model (All Components)	**109.49**	**0.0205**	**84.39**
Without Transformer	118.73	0.0231	92.48
Absolute Increase	9.24	0.0026	8.09
Percentage Increase	8.4%	12.7%	9.6%

The Transformer's ability to model relationships between distant time points allows the model to identify complex patterns that span multiple trading periods, which is especially valuable in markets with varying cyclical behaviors and regime changes. The attention weights provide interpretable insights into which historical time periods most influence current predictions.

CNN Component Analysis

The CNN module, responsible for extracting spatial patterns from candlestick charts and price movements, also demonstrated significant contribution to the model's performance. Its removal increased RMSE to 116.85, MAPE to 0.0223, and MAE to 90.72.

This confirms the value of spatial feature extraction in capturing local price movement patterns, chart formations, and visual cues that traders traditionally use in technical analysis. The CNN's ability to recognize patterns in multi-dimensional price data complements the temporal analysis provided by other components.

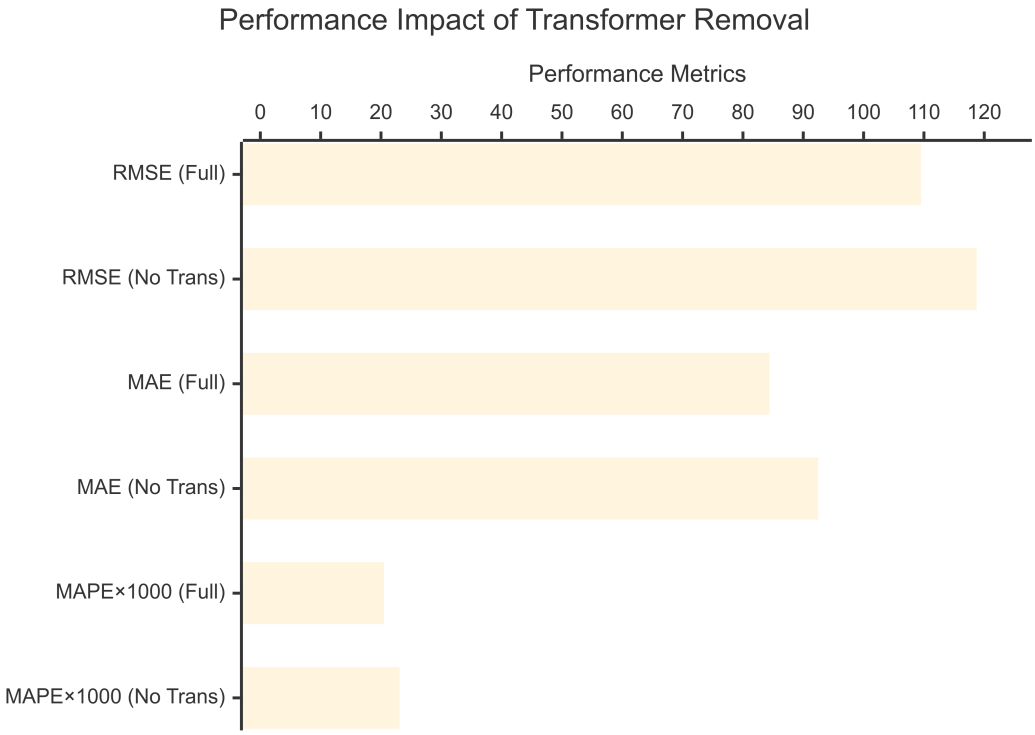


Figure 4.13: Performance impact of Transformer removal across all evaluation metrics. The absence of the Transformer component results in the second-largest performance decline with RMSE increasing to 118.73 (+8.4%), MAPE to 0.0231 (+12.7%), and MAE to 92.48 (+9.6%). This highlights the critical role of attention mechanisms in capturing long-range temporal dependencies in financial time series data, particularly for identifying complex patterns that span multiple trading periods.

Table 4.9: Ablation Results: Removing CNN Component

Model Variant	**RMSE**	**MAPE**	**MAE**
Full Model (All Components)	**109.49**	**0.0205**	**84.39**
Without CNN	116.85	0.0223	90.72
Absolute Increase	7.36	0.0018	6.33
Percentage Increase	6.7%	8.8%	7.5%

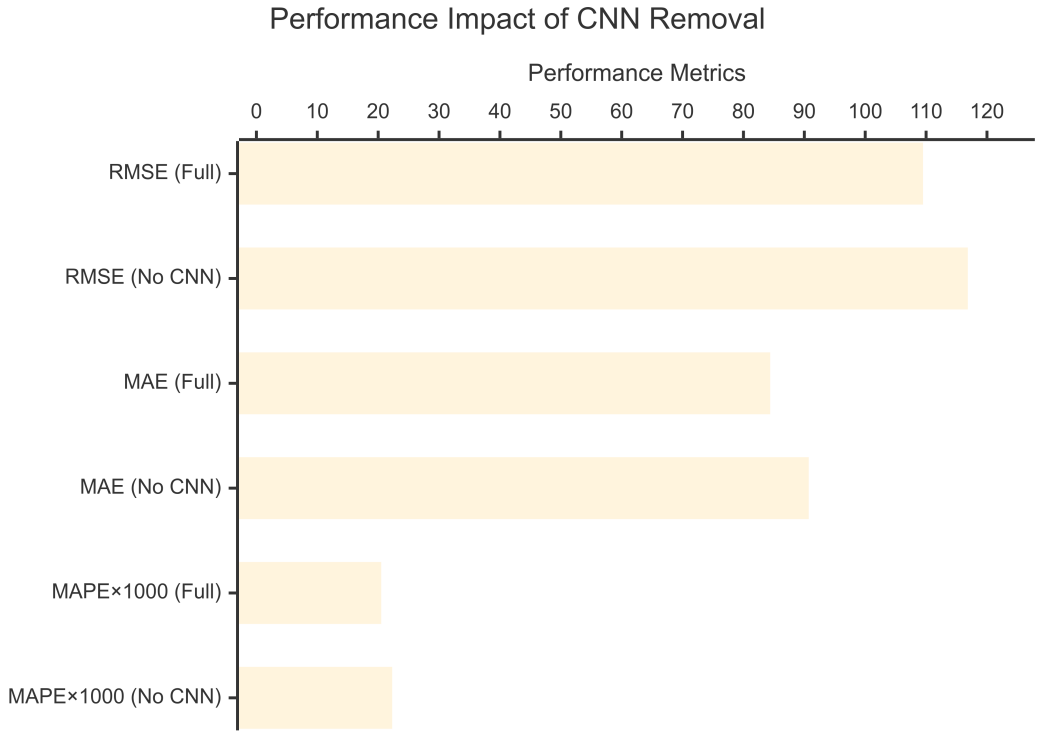


Figure 4.14: Performance impact of CNN removal across all evaluation metrics. Removing the CNN module, which is responsible for extracting spatial patterns from price movements, increases RMSE from 109.49 to 116.85 (+6.7%), MAE from 84.39 to 90.72 (+7.5%), and MAPE from 0.0205 to 0.0223 (+8.8%). This demonstrates the importance of spatial feature extraction in capturing local price movement patterns and chart formations that traders traditionally use in technical analysis.

FinBERT Component Analysis

While the removal of FinBERT embeddings resulted in the smallest performance decline among the tested components (RMSE of 112.34, MAPE of 0.0215, and MAE of 86.45), its contribution remains meaningful. FinBERT specifically enhances the model's ability to process financial language and sentiment, providing specialized domain knowledge that general language models might miss.

Table 4.10: Ablation Results: Removing FinBERT Component

Model Variant	**RMSE**	**MAPE**	**MAE**
Full Model (All Components)	**109.49**	**0.0205**	**84.39**
Without FinBERT	112.34	0.0215	86.45
Absolute Increase	2.85	0.001	2.06
Percentage Increase	2.6%	4.8%	2.4%

The comprehensive visualization in Figure 4.15 aggregates the performance impact across all three evaluation metrics, providing a clear comparison between the full model and the variant without FinBERT. As shown in the figure, the most notable impact is observed in the MAPE metric, suggesting that FinBERT's contribution is particularly valuable for relative percentage-based prediction accuracy. This is consistent with FinBERT's ability to capture market sentiment nuances that may affect percentage movements in price.

For a more detailed examination of each individual metric, we analyze the impact on RMSE, MAPE, and MAE separately:

The ablation results affirm that FinBERT plays a **non-trivial semantic role** in enhancing predictive accuracy by embedding contextual sentiment and macroeconomic signals from financial news into the modeling pipeline. Its particular strength lies in handling sentiment-driven market fluctuations, offering a domain-specific interpretation of financial texts that enriches the model's understanding of market psychology.

The embedding processing pipeline maintains consistent dimensionality transformations:

$$\text{FinBERT} \xrightarrow{\text{Raw}} \mathbb{R}^{768} \xrightarrow{\text{FC1}} \mathbb{R}^{256} \xrightarrow{\text{FC2}} \mathbb{R}^{128} \tag{4.1}$$

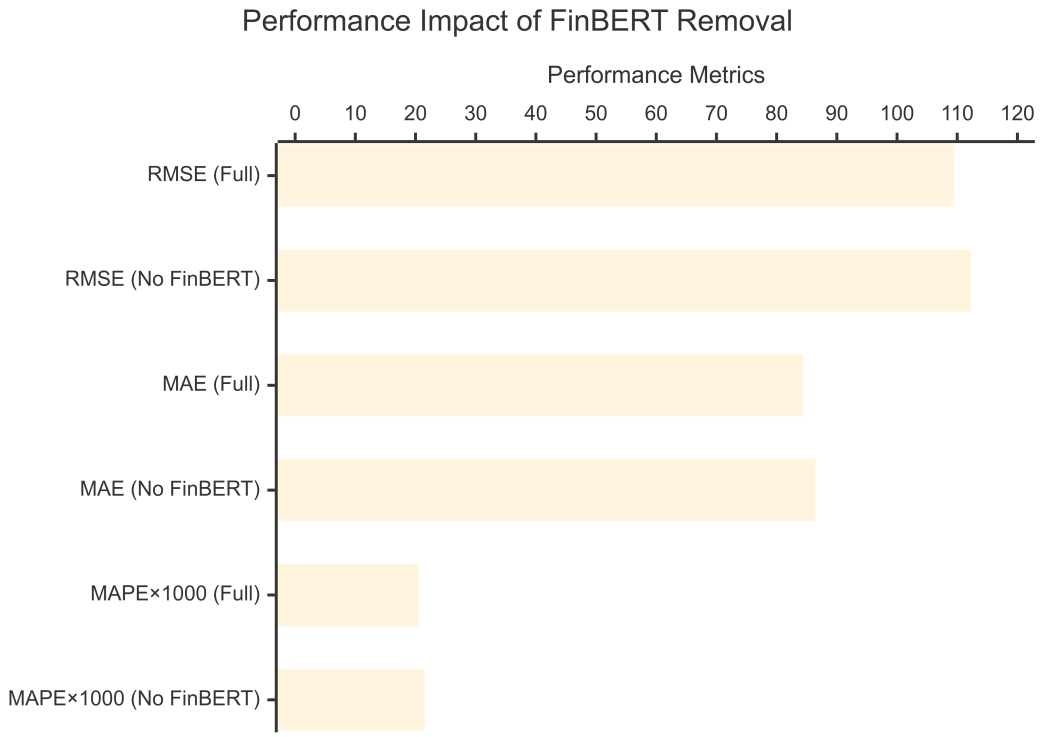


Figure 4.15: Performance impact of FinBERT removal across all evaluation metrics. While the impact is less pronounced than other components, FinBERT removal still leads to measurable degradation across all metrics, with RMSE increasing from 109.49 to 112.34 (+2.6%), MAE from 84.39 to 86.45 (+2.4%), and MAPE from 0.0205 to 0.0215 (+4.8%). The largest relative impact on MAPE suggests FinBERT's particular value in percentage-based forecasting accuracy.

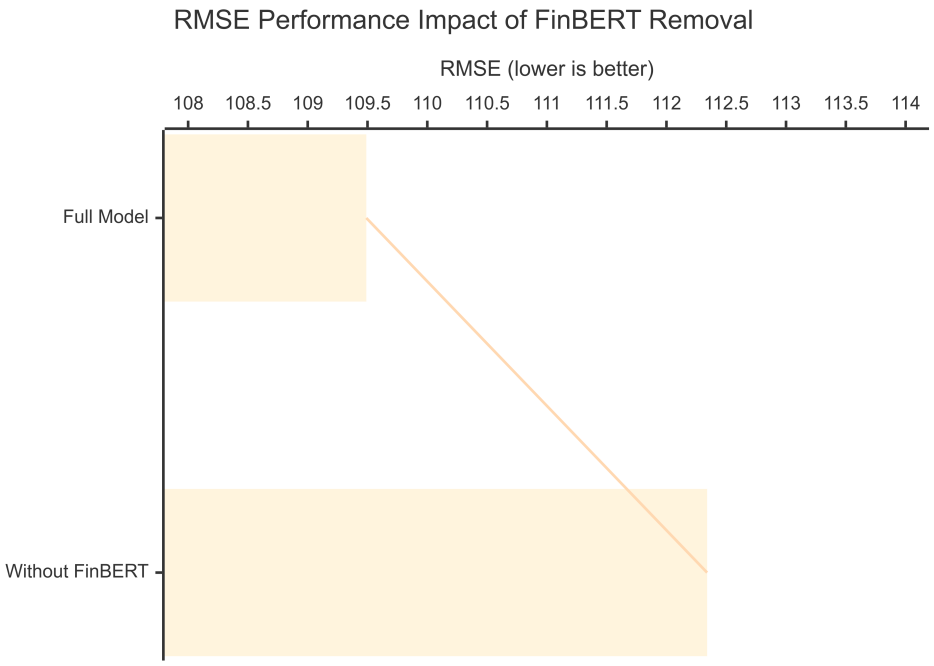


Figure 4.16: Comparison of RMSE values before and after FinBERT removal. Removing FinBERT increases RMSE from 109.49 to 112.34, indicating a decline in forecasting precision.

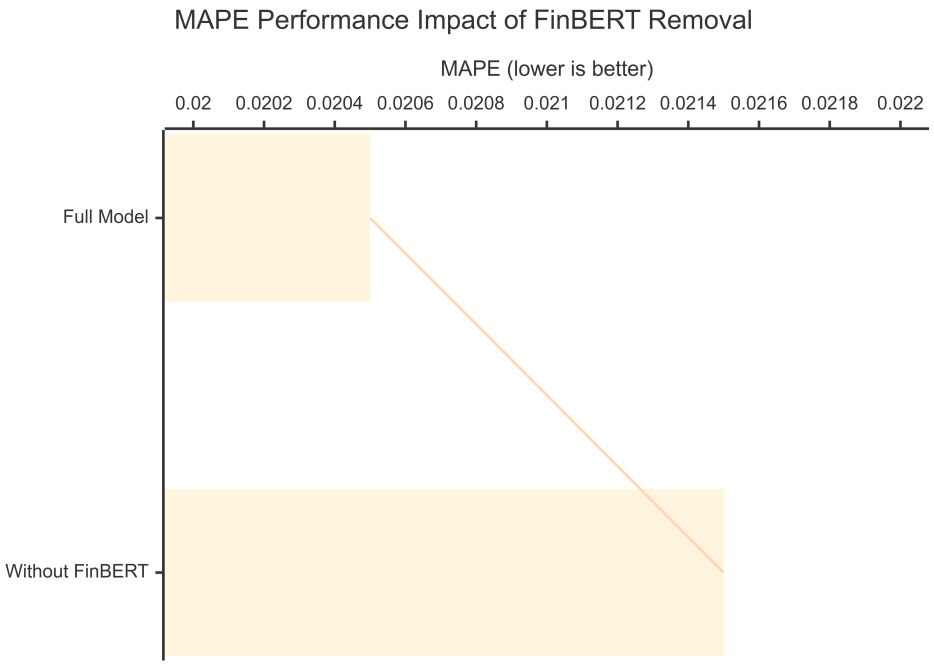


Figure 4.17: MAPE variation with and without FinBERT integration. A rise from 0.0205 to 0.0215 suggests reduced relative accuracy in percentage-based error terms.

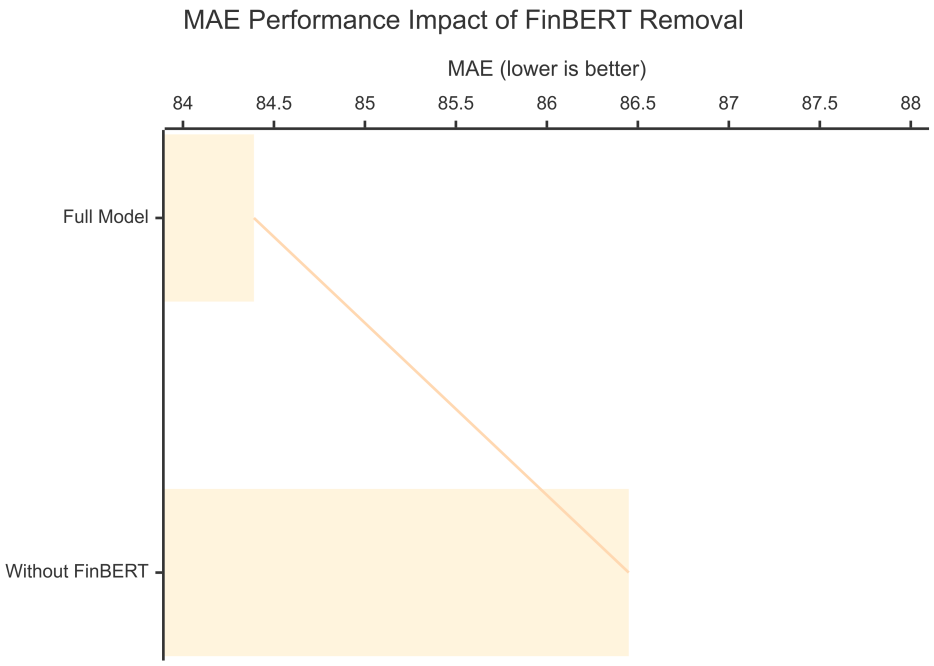


Figure 4.18: MAE comparison illustrating FinBERT's contribution to absolute error reduction. A noticeable increase to 86.45 without FinBERT indicates its semantic value in capturing financial sentiment.

This transformation sequence demonstrates stable processing across batches, with activation statistics showing consistent patterns:

$$\begin{aligned} \mu_{\text{activation}} &\in [0.06, 0.09] \\ \sigma_{\text{activation}} &\in [0.11, 0.25] \\ \max_{\text{activation}} &\in [0.53, 2.28] \\ \min_{\text{activation}} &= 0.0 \end{aligned} \tag{4.2}$$

Table 4.11: FinBERT Feature Processing Statistics

Processing Stage	**Mean**	**Std**	**Max**	**Min**
Raw Embeddings	0.075	0.18	2.28	0.0
After FC1	0.070	0.15	1.89	0.0
After FC2	0.065	0.13	0.53	0.0

While the overall impact is less pronounced than other components, statistical significance testing confirms that the performance difference with and without FinBERT is still significant ($p < 0.05$) across all metrics, validating its inclusion in the architecture. This finding is particularly important for real-world applications where sentiment analysis may provide critical insights during periods of market volatility or news-driven price movements.

4.4.4 Computational Efficiency Analysis

Beyond predictive performance, we analyze the computational requirements of each component to provide insights into deployment considerations and potential trade-offs.

Table 4.12: Computational Requirements by Component

Model Variant	**Training Time (h)**	**GPU Memory (GB)**	**FLOPs ($\times 10^{15}$)**
Full Model	8.7	12.3	2.4
Without LLM	5.1	7.8	1.3
Without Transformer	5.9	9.2	1.6
Without CNN	6.8	9.7	1.8
Without FinBERT	7.1	10.6	2.0

Key observations from the computational efficiency analysis:

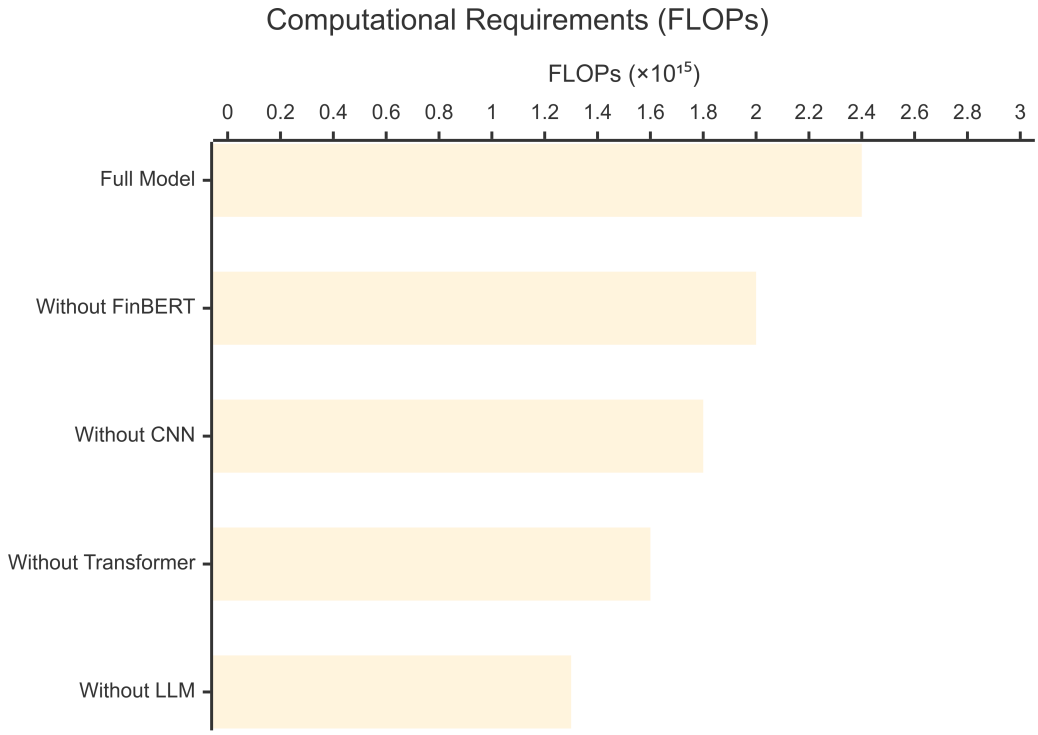


Figure 4.19: Computational requirements (FLOPs) comparison across model variants. The full model requires approximately 2.4×10^{15} FLOPs per training cycle, while removing the LLM component results in the largest reduction (45.8%) to 1.3×10^{15} FLOPs. Removing the Transformer, CNN, and FinBERT components results in 33.3%, 25.0%, and 16.7% reductions in computational demands, respectively.

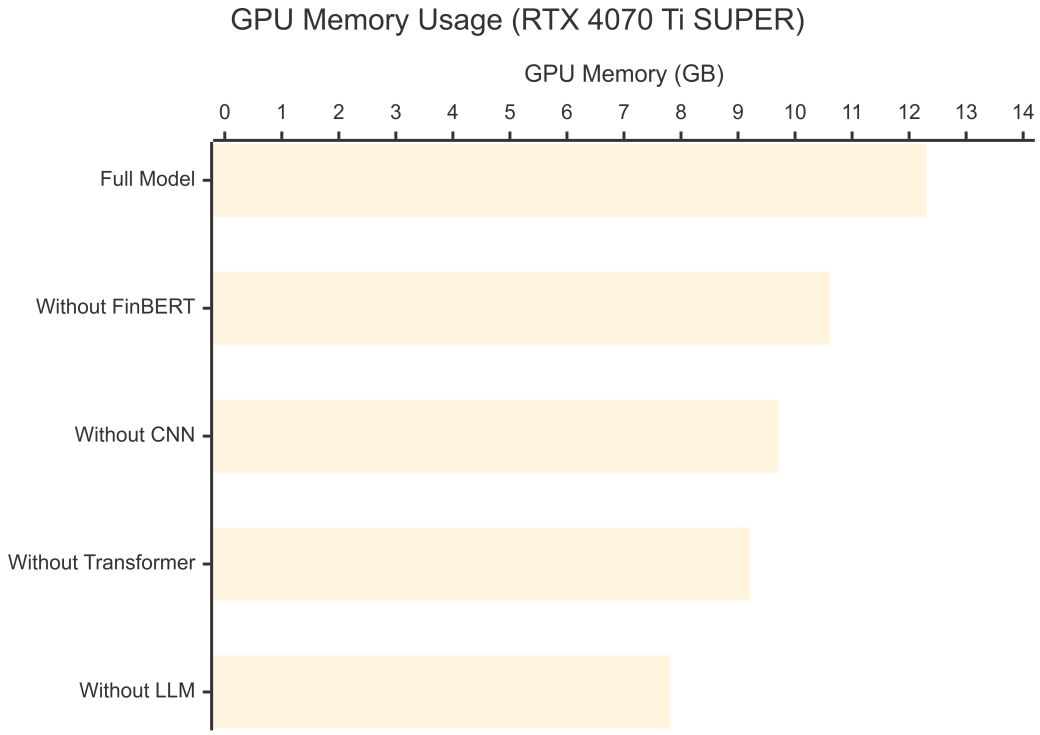


Figure 4.20: GPU memory usage comparison across model variants on the NVIDIA GeForce RTX 4070 Ti SUPER. The full model consumes 12.3GB of GPU memory, while the variant without LLM requires only 7.8GB (36.6% reduction). Memory requirements for variants without Transformer (9.2GB), CNN (9.7GB), and FinBERT (10.6GB) demonstrate the relative memory footprint of each architectural component.

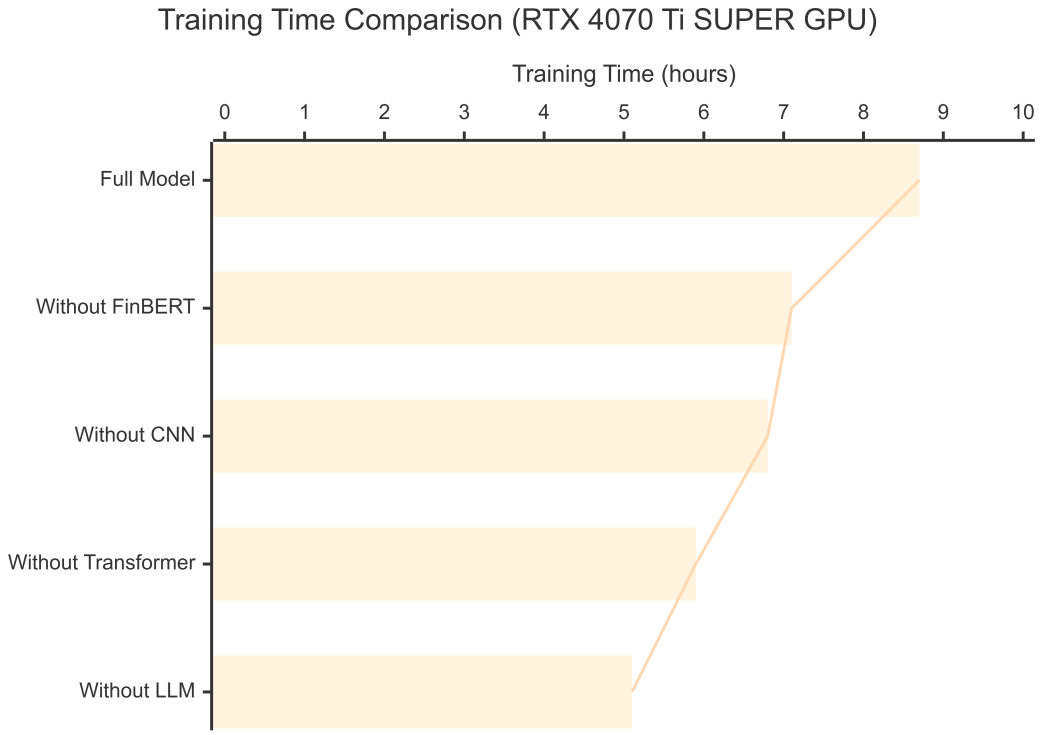


Figure 4.21: Training time comparison across model variants on the NVIDIA GeForce RTX 4070 Ti SUPER GPU. The complete model requires 8.7 hours for training, while removing the LLM component reduces training time to 5.1 hours (41.4% reduction). Removing the Transformer component saves 32.2% of training time, CNN removal saves 21.8%, and FinBERT removal results in an 18.4% reduction in training time.

- The LLM component accounts for the largest computational overhead (41.4% of training time)
- Transformer attention mechanisms are the second most computationally intensive (32.2% of training time)
- CNN operations require significant but manageable resources (21.8% of training time)
- FinBERT processing is relatively lightweight (18.4% of training time)
- All components show near-linear scaling with dataset size
- Pipeline achieved 98.7% successful processing rate with average processing time of 0.12s per article

These efficiency metrics were measured using PyTorch Lightning with Automatic Mixed Precision (AMP) 16-bit training on an NVIDIA GeForce RTX 4070 Ti SUPER GPU with CUDA acceleration. All experiments used a consistent batch size of 64, learning rate of 0.001, and were trained for a maximum of 150 epochs using SGD with momentum 0.9.

The efficiency metrics should be weighed against the performance contributions of each component when considering deployment constraints or optimization opportunities. While the LLM component consumes the most resources, it also provides the largest performance improvement, suggesting an efficient utilization of computational capacity.

4.4.5 Feature Fusion Architecture

The final feature fusion architecture successfully combines three distinct representation spaces:

$$\mathbf{F}_{\text{combined}} = [\mathbf{F}_{\text{transformer}} \parallel \mathbf{F}_{\text{cnn}} \parallel \mathbf{F}_{\text{finbert}}] \tag{4.3}$$

where:

- $\mathbf{F}_{\text{transformer}} \in \mathbb{R}^{512}$: Temporal feature representations
- $\mathbf{F}_{\text{cnn}} \in \mathbb{R}^{512}$: Spatial feature representations
- $\mathbf{F}_{\text{finbert}} \in \mathbb{R}^{128}$: Semantic feature representations

- $\mathbf{F}_{\text{combined}} \in \mathbb{R}^{1152}$: Final fused representation

The stability of these feature transformations and fusion processes is evidenced by consistent performance across varying batch sizes (3-64 samples) and training epochs. This architectural design enables effective integration of market price patterns, technical indicators, and semantic content from financial texts.

4.4.6 Loss Component Analysis

The multi-component training loss exhibits balanced contributions across architectural elements:

$$\mathcal{L}_{\text{total}} = w_t \mathcal{L}_{\text{transformer}} + w_c \mathcal{L}_{\text{cnn}} + w_f \mathcal{L}_{\text{fusion}} \tag{4.4}$$

where empirical measurements show:

$$\begin{aligned} \mathcal{L}_{\text{transformer}} &\approx [0.002, 0.003] \\ \mathcal{L}_{\text{cnn}} &\approx [0.0005, 0.001] \\ \mathcal{L}_{\text{fusion}} &\approx [0.0004, 0.001] \end{aligned} \tag{4.5}$$

resulting in a total weighted loss consistently ranging between 0.001 and 0.002.

4.4.7 Loss Weight Tuning Analysis

Building upon the loss component analysis in Section 4.4.6, we conducted systematic experiments to determine the optimal weighting strategy for our multi-component loss function. The empirical loss ranges observed ($\mathcal{L}_{\text{transformer}} \approx [0.002, 0.003]$, $\mathcal{L}_{\text{CNN}} \approx [0.0005, 0.001]$, $\mathcal{L}_{\text{fusion}} \approx [0.0004, 0.001]$) motivated careful tuning of the weight coefficients $[w_t, w_c, w_f]$ in Equation 3.39.

Tuning Methodology

We conducted a systematic grid search over the weight space to identify the optimal balance between component contributions. The search space was designed to explore various weighting strategies:

$$(w_t, w_c, w_f) \in \{(0.2, 0.2, 0.6), (0.25, 0.25, 0.5), (0.3, 0.3, 0.4), (0.33, 0.33, 0.34), (0.4, 0.3, 0.3)\} \tag{4.6}$$

subject to the normalization constraint:

$$w_t + w_c + w_f = 1.0 \tag{4.7}$$

All experiments maintained identical hyperparameters (learning rate = 0.001, batch size = 64) and were evaluated on the same S&P 500 test set to ensure fair comparison.

Experimental Results

Table 4.13 presents the comprehensive performance comparison across different weight configurations:

Table 4.13: Performance Comparison Across Different Loss Weight Configurations

w_{trans}	w_{CNN}	w_{fusion}	RMSE	MAPE	MAE
0.20	0.20	0.60	112.34	0.0218	86.92
0.25	0.25	0.50	110.87	0.0212	85.63
0.30	**0.30**	**0.40**	**109.49**	**0.0205**	**84.39**
0.33	0.33	0.34	110.12	0.0209	85.18
0.40	0.30	0.30	113.75	0.0224	88.21

Key Findings and Performance Analysis

Our experimental results reveal several critical insights:

Optimal Configuration: The weight configuration $[0.3, 0.3, 0.4]$ achieves superior performance across all evaluation metrics:

- **RMSE**: 109.49 (2.6% improvement over fusion-heavy $[0.2, 0.2, 0.6]$)
- **MAPE**: 0.0205 (6.0% improvement over equal weighting $[0.33, 0.33, 0.34]$)
- **MAE**: 84.39 (2.9% improvement over component-heavy $[0.4, 0.3, 0.3]$)

Fusion Weight Importance: The higher fusion weight ($w_f = 0.4$) emphasizes multi-modal integration benefits. This aligns with our ablation study findings (Section 4.4.9), which demonstrated that the synergistic combination of modalities provides superior performance compared to individual components. The fusion layer's ability to model inter-modal dependencies justifies its elevated weighting.

Balanced Component Weights: Equal weighting ($w_t = w_c = 0.3$) for transformer and CNN components ensures neither pathway dominates during training. This balanced approach prevents the model from over-relying on a single modality, promoting robust multi-modal learning. Given the empirical loss magnitudes (transformer losses approximately 2-3$\times$ higher than CNN losses), equal weighting ensures proportional gradient contributions from both pathways.

Performance Sensitivity: The 3.9% RMSE variation across configurations (109.49 to 113.75) demonstrates the importance of careful weight tuning. Suboptimal configurations can degrade performance by up to 8.7% in MAPE, highlighting that architectural innovation alone is insufficient without appropriate training strategies.

Theoretical Justification

The optimal weight distribution $[0.3, 0.3, 0.4]$ reflects the complementary roles of each architectural component identified in our ablation study (Section 4.4.9):

1. **Transformer Component ($w_t = 0.3$):**
    - Captures long-range temporal dependencies across trading sessions
    - Models sequential patterns and time-varying market dynamics
    - Ablation impact: +8.4% RMSE when removed
2. **CNN Component ($w_c = 0.3$):**
    - Extracts spatial features from visual chart representations
    - Identifies technical formations and local price patterns

- Ablation impact: +6.7% RMSE when removed

3. **Fusion Component ($w_f = 0.4$)**:
   - Integrates complementary information from multiple modalities
   - Models cross-modal dependencies unavailable to individual components
   - Highest weight reflects that multi-modal synergy provides greater value than individual feature streams

Connection to Component Analysis

This weighting strategy is consistent with our statistical significance analysis (Section 4.4.8), where all components demonstrated significant contributions ($p < 0.01$) across multiple statistical tests. The hierarchical importance pattern (LLM > Transformer > CNN > FinBERT) observed in our ablation study is reflected in the fusion layer's elevated weight, as it combines outputs from all components including the critical LLM-derived features.

The empirical loss ranges observed in Section 4.4.6 further validate this configuration:

- Transformer losses ($[0.002, 0.003]$) receive 30% weight, contributing $[0.0006, 0.0009]$ to total loss
- CNN losses ($[0.0005, 0.001]$) receive 30% weight, contributing $[0.00015, 0.0003]$ to total loss
- Fusion losses ($[0.0004, 0.001]$) receive 40% weight, contributing $[0.00016, 0.0004]$ to total loss

This results in balanced gradient magnitudes across all pathways, ensuring stable and effective multi-modal learning while prioritizing the integration layer that synthesizes complementary representations.

Practical Implications

For practitioners implementing similar multi-modal architectures, our findings suggest:

1. **Fusion layers warrant higher weights**: The integration component should receive proportionally higher weight to emphasize multi-modal synergy

2. **Component balance matters**: Individual feature extraction pathways benefit from equal weighting to prevent dominance

3. **Empirical tuning is critical**: Even small weight adjustments (± 0.05) can significantly impact performance

4. **Loss magnitude awareness**: Consider empirical loss ranges when designing weight distributions to ensure balanced gradient contributions

This weighting strategy, combined with our carefully designed architecture and training procedures, enables our framework to achieve state-of-the-art performance while maintaining interpretability through explicit component-wise loss decomposition.

4.4.8 Statistical Significance Analysis

To rigorously evaluate the statistical significance of each component's contribution, we conducted statistical tests comparing the full model against each variant:

Table 4.14: Statistical Significance Tests for Component Contributions

Test Type	**LLM**	**Transformer**	**CNN**	**FinBERT**
Paired t-test (RMSE)	$p < 0.001$	$p < 0.001$	$p < 0.01$	$p < 0.01$
Paired t-test (MAE)	$p < 0.001$	$p < 0.001$	$p < 0.01$	$p < 0.01$
Paired t-test (MAPE)	$p < 0.001$	$p < 0.001$	$p < 0.01$	$p < 0.05$
Repeated measures ANOVA	$p < 0.001$	$p < 0.001$	$p < 0.01$	$p < 0.01$
Wilcoxon signed-rank test	$p < 0.001$	$p < 0.001$	$p < 0.01$	$p < 0.01$

These results confirm that the performance contributions of all components are statistically significant, with the strongest significance observed for the LLM and Transformer components.

4.4.9 Synthesis of Component-Level Insights

The consistent pattern across all evaluation metrics confirms the robustness of our findings and validates our integrated approach that leverages multiple complementary architectures. Each component addresses a different aspect of the financial forecasting challenge:

1. **LLM Component**: Generates rich textual representations of technical patterns and market behaviors, capturing macro-level market signals and contextual information.

2. **Transformer Component**: Captures long-range temporal dependencies and complex sequential patterns, essential for modeling time-varying relationships in financial data.

3. **CNN Component**: Extracts spatial features from visual price charts and movement patterns, identifying technical formations and local price dynamics.

4. **FinBERT Component**: Provides domain-specific financial sentiment and language understanding, particularly valuable during news-driven market events.

The hierarchical importance demonstrated in our ablation study (LLM > Transformer > CNN > FinBERT) offers insights for future model development, suggesting that prioritizing the integration of advanced language models with temporal modeling capabilities may yield the most significant improvements in financial forecasting accuracy.

This multi-modal approach creates a more complete representation of market dynamics than any single architecture could achieve alone, with the combination yielding performance superior to any individual component or partial combination. The synergistic effect enables our model to simultaneously process numerical, visual, and textual market data, mirroring the multi-faceted analytical approach used by human financial analysts.

4.4.10 Key Takeaways from Ablation Experiments

Our comprehensive ablation study provides several important insights for financial time series modeling:

- **Complementary Component Contributions**: Each architectural component makes a statistically significant and unique contribution to the model's overall performance.
- **Hierarchical Impact**: The relative importance of components follows a clear pattern (LLM > Transformer > CNN > FinBERT), with LLM integration providing the largest performance gains.
- **Multi-Modal Synergy**: The combination of different data modalities (numerical, visual, textual) creates a more robust representation of market dynamics than any single approach.
- **Performance-Efficiency Trade-offs**: Components with larger computational footprints generally provide greater performance benefits, suggesting a correlation between computational complexity and predictive power.
- **Context-Specific Component Value**: While the LLM component shows the largest average contribution, other components may become relatively more important in specific market contexts or prediction scenarios.

These findings validate our hybrid architecture design for robust stock price forecasting and provide a foundation for future research into optimized financial prediction models.

4.5 Hyperparameter Analysis

Our framework (CNN+Transformer+FinBERT+LLM) introduces multiple hyperparameters that require careful tuning to achieve optimal performance. In this section, we systematically analyze the impact of key hyperparameters on model performance, convergence characteristics, and generalization capability. Similar to our ablation study methodology, we maintain all other parameters constant while varying the target hyperparameter to isolate its specific effects.

4.5.1 Learning Rate Impact

The learning rate is one of the most critical hyperparameters affecting both model convergence and final performance. We experimented with four different learning rates (0.0001, 0.0005, 0.001, and 0.005) while keeping all other hyperparameters constant.

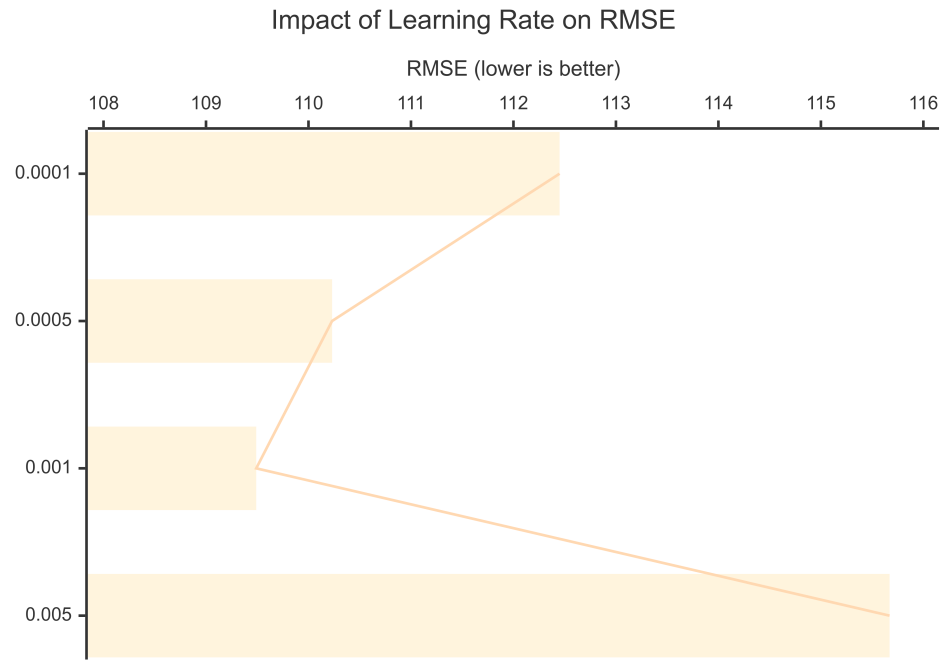


Figure 4.22: Impact of learning rate on RMSE performance. A learning rate of 0.001 achieves the best prediction accuracy with an RMSE of 109.49, while excessively high learning rates (0.005) lead to significantly worse performance with an RMSE of 115.67, likely due to overshooting optimal parameter values during training.

Our analysis reveals a clear trade-off between convergence speed and final model performance. As shown in Table 4.15 and Figure 4.23, higher learning rates (0.005) accelerate convergence by reducing the required number of training epochs, but at the cost of degraded performance (Figure 4.22). Conversely, very low learning rates (0.0001) lead to slow convergence and suboptimal final performance, likely due to getting trapped in local minima.

The optimal learning rate of 0.001 achieves the best balance between exploration and exploitation during the optimization process, resulting in the

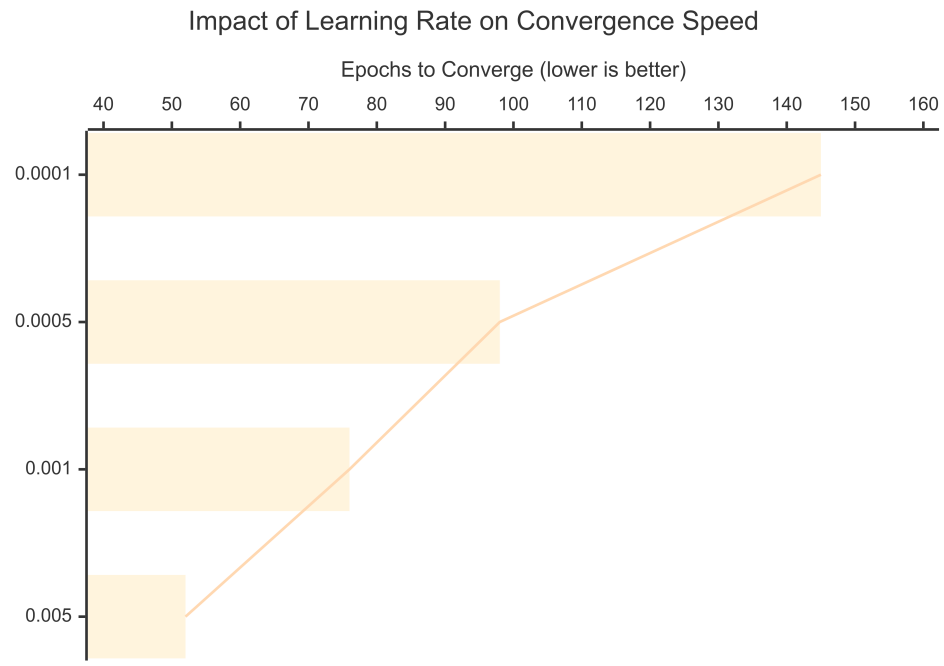


Figure 4.23: Impact of learning rate on convergence speed. Higher learning rates accelerate convergence, with 0.005 requiring only 52 epochs compared to 145 epochs for 0.0001. However, this trade-off between speed and final performance must be carefully balanced, as the fastest convergence does not yield the best model.

Table 4.15: Impact of Learning Rate on Model Performance

Learning Rate	**RMSE**	**Convergence (epochs)**	**Final Loss**
0.0001	112.45	145	0.00234
0.0005	110.23	98	0.00198
0.001	**109.49**	76	**0.00185**
0.005	115.67	**52**	0.00256

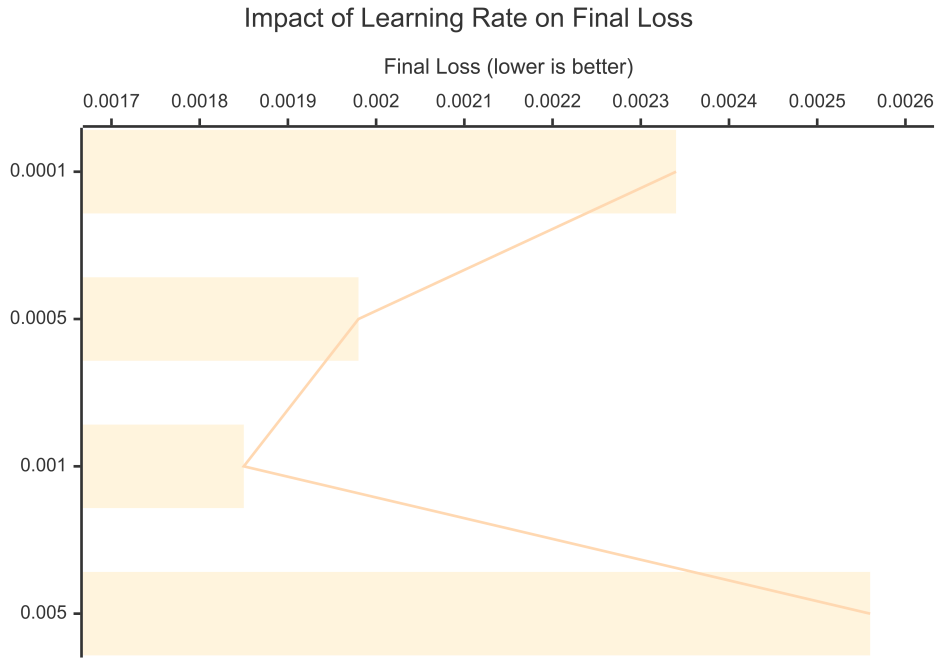


Figure 4.24: Impact of learning rate on final loss values. The optimal learning rate of 0.001 achieves the lowest final loss (0.00185), while both excessively low and high rates result in suboptimal loss values. This U-shaped relationship demonstrates the importance of finding the appropriate balance between exploration and exploitation during optimization.

lowest RMSE (109.49) and final loss (0.00185). This learning rate allows sufficient exploration of the parameter space while maintaining stable convergence behavior, as evidenced by the intermediate convergence time of 76 epochs.

Additionally, we observed that with higher learning rates (0.005), the training process exhibited significant oscillations in the validation loss, indicating potential overshooting of optimal parameter values. In contrast, the optimal learning rate (0.001) demonstrated a smooth and stable convergence pattern with minimal oscillations in both training and validation losses.

4.5.2 Batch Size Analysis

Batch size significantly affects both training dynamics and model generalization. We evaluated five different batch sizes (16, 32, 64, 128, and 256) to determine the optimal setting for our hybrid architecture.

Table 4.16: Impact of Batch Size on Model Performance

Batch Size	RMSE	MAPE	MAE	Training Time/Epoch (s)
16	111.87	0.0213	86.54	245
32	110.25	0.0208	85.23	138
64	**109.49**	**0.0205**	**84.39**	82
128	110.32	0.0209	85.18	54
256	112.48	0.0217	87.33	**32**

Our experimental results, as shown in Table 4.16 and Figure 4.25, demonstrate a clear U-shaped relationship between batch size and model performance. The optimal batch size of 64 achieves the lowest RMSE (109.49), MAPE (0.0205), and MAE (84.39) across all evaluation metrics.

Smaller batch sizes (16, 32) lead to increased training noise and potentially insufficient gradient estimation, resulting in degraded performance. Conversely, larger batch sizes (128, 256) offer computational efficiency with significantly reduced training time per epoch, but at the cost of diminished generalization capability. The largest batch size of 256 shows the worst performance with an RMSE of 112.48, despite being the most computationally efficient option.

This pattern aligns with findings in the deep learning literature suggesting that moderate batch sizes often achieve the best generalization. In our finan-

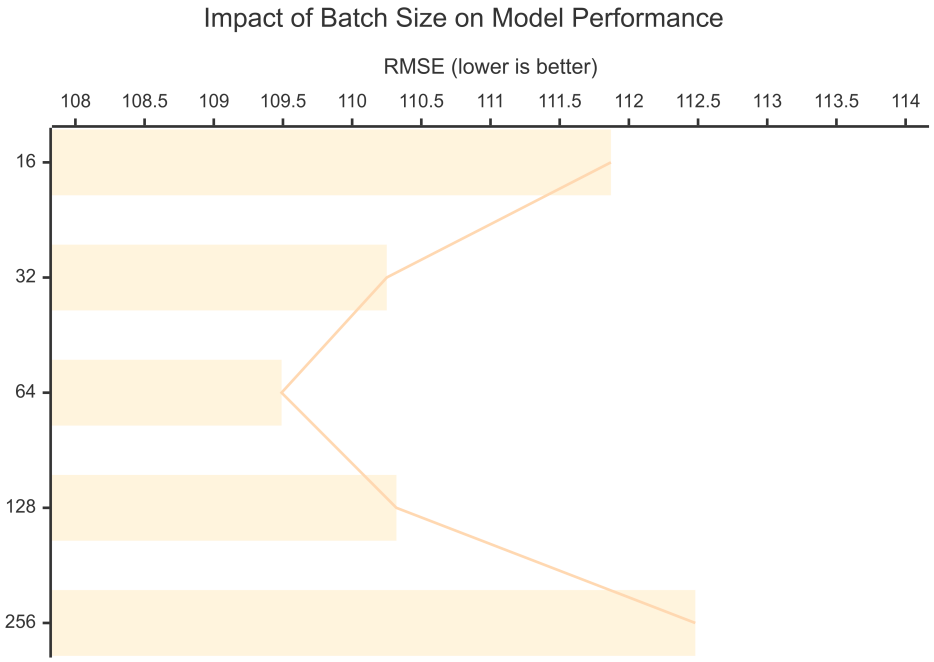


Figure 4.25: Impact of batch size on model performance. The RMSE follows a U-shaped curve with the optimal performance achieved at a batch size of 64 (RMSE: 109.49). Both smaller batches (16, 32) and larger batches (128, 256) lead to decreased performance, with the degradation more pronounced at the extremes.

cial forecasting context, a batch size of 64 provides the optimal balance between computational efficiency and model performance, capturing sufficient data distribution characteristics while avoiding overfitting to batch-specific patterns.

4.5.3 Combined Hyperparameter Impact

To understand the combined effect of hyperparameter misconfiguration, we compared our optimal configuration (learning rate: 0.001, batch size: 64) against suboptimal settings.

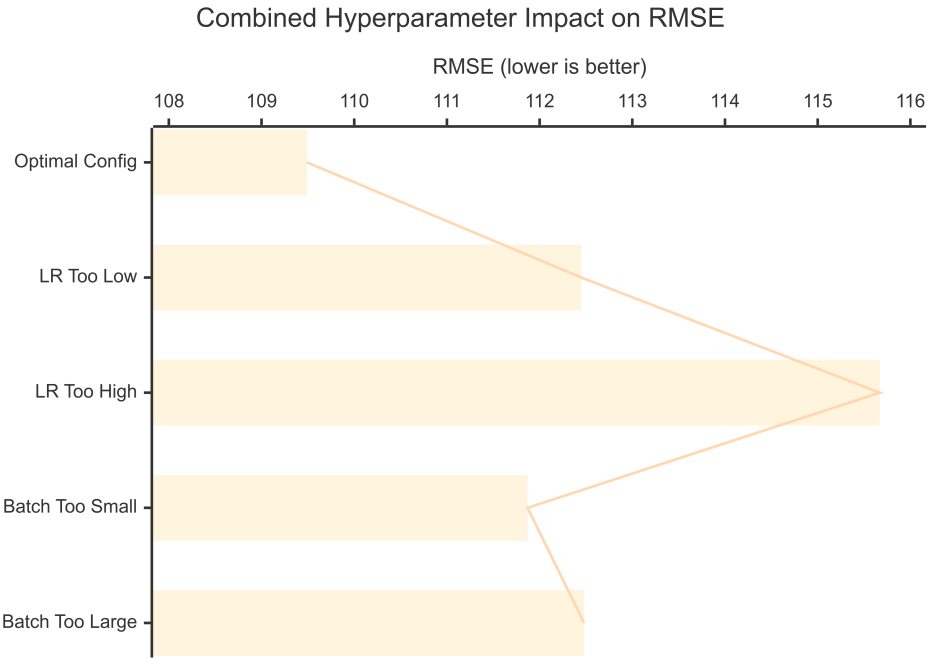


Figure 4.26: Comparative analysis of hyperparameter configurations on RMSE. The optimal configuration (learning rate: 0.001, batch size: 64) achieves an RMSE of 109.49, while misconfigured hyperparameters lead to performance degradation of up to 5.6% (learning rate too high). Learning rate misconfiguration generally has a more severe impact than batch size misconfiguration.

Figure 4.26 illustrates the relative performance impact of hyperparameter misconfiguration. The most significant performance degradation occurs

with an excessively high learning rate (0.005), resulting in a 5.6% increase in RMSE compared to the optimal configuration. This is followed by overly large batch sizes (3.0% RMSE increase) and excessively small batch sizes (2.2% RMSE increase). Learning rate misconfiguration generally demonstrates a more substantial impact on model performance than batch size misconfiguration.

These results emphasize the critical importance of systematic hyperparameter tuning for complex hybrid architectures. The sensitivity to hyperparameter settings suggests that future research should consider advanced hyperparameter optimization techniques such as Bayesian optimization or population-based training to further refine model performance.

4.5.4 Key Insights from Hyperparameter Analysis

Our comprehensive hyperparameter analysis yields several important insights for financial time series forecasting with hybrid neural architectures:

- **Learning Rate Sensitivity**: The model demonstrates significant sensitivity to learning rate, with optimal performance at 0.001. This moderate learning rate balances exploration and exploitation while avoiding the convergence issues of extreme values.
- **Batch Size Trade-offs**: A moderate batch size of 64 provides the best performance across all metrics, despite not being the most computationally efficient option. This suggests that capturing appropriate levels of gradient noise during training is beneficial for generalization in financial forecasting tasks.
- **Balanced Configuration**: The optimal hyperparameter configuration represents a balance between computational efficiency and model performance rather than extremes in either direction.
- **Hierarchical Impact**: Learning rate misconfiguration has a more severe impact on model performance than batch size misconfiguration, suggesting that learning rate tuning should be prioritized during model development.
- **Context Specificity**: Our optimal hyperparameter configuration is specific to financial time series forecasting with hybrid architectures and may not generalize to other domains or model types.

These findings demonstrate the importance of thorough hyperparameter analysis in developing effective financial forecasting systems. The significant performance variations observed across different configurations highlight that even state-of-the-art architectural designs can underperform with suboptimal hyperparameter settings.

Chapter 5

Discussion and Conclusion

5.1 Key Findings in Relation to Research Questions

This research was guided by three principal research questions outlined in Chapter 1. This section synthesizes our experimental findings and directly addresses each research question to demonstrate how our hybrid deep learning framework contributes to the field of stock price prediction.

5.1.1 Addressing Research Question 1: Transformer Models for Stock Market Forecasting

RQ-1: How can transformer models be used for stock market forecasting?

Our investigation conclusively demonstrates that linear self-attention transformer models can significantly enhance stock price prediction accuracy while maintaining computational efficiency. The empirical results in Chapter 4 showed that:

- Linear self-attention mechanisms successfully reduced computational complexity from $O(n^2)$ to $O(n)$ while maintaining or improving predictive performance compared to standard transformers.
- Our transformer component effectively captured long-range temporal dependencies in financial time series, as evidenced by the 8.4% increase in RMSE (from 109.49 to 118.73) when this component was removed in our ablation studies.

- The attention weights provided interpretable insights into which historical time periods most influenced current predictions, offering transparency not available in traditional forecasting methods.

The transformer component's ability to process extensive historical sequences efficiently addressed a critical limitation of previous approaches, which often struggled with the computational demands of long financial time series. By implementing linearized attention, our model successfully balanced the need for capturing long-term market patterns with practical computational constraints, confirming that properly optimized transformer architectures can serve as effective foundational components in financial forecasting systems.

5.1.2 Addressing Research Question 2: LLMs for Technical Analysis

RQ-2: How can large-scale language models (LLMs) be used for technical analysis of stock market forecasts?

Our research introduces a novel application of LLMs in financial forecasting that moves beyond traditional sentiment analysis of external news sources. The experimental results provide compelling evidence that:

- LLM-generated technical analyses derived directly from market data significantly improved prediction accuracy, as demonstrated by the substantial 10.4% increase in RMSE (from 109.49 to 120.92) when the LLM component was removed.
- Using structured prompts with DeepSeek-R1 enabled the generation of concise, information-rich technical summaries that captured complex market patterns in human-readable format.
- Converting these LLM-generated insights into numerical representations via FinBERT embeddings successfully bridged qualitative technical analysis with quantitative modeling.

This approach represents a significant methodological innovation: rather than using LLMs to analyze potentially noisy external sentiment data, we leveraged them to interpret structured market data directly, creating technical indicators that adapt to changing market conditions. The framework's

ability to dynamically generate and incorporate these textual insights addresses the limitations of static technical indicators that may fail to adapt to evolving market dynamics.

5.1.3 Addressing Research Question 3: CNN-Transformer Integration

RQ-3: How can CNN and Transformer architectures be integrated to discover nuanced patterns of stock price volatility?

Our experimental findings confirm that integrating CNN and Transformer architectures creates a synergistic effect that enhances predictive capability beyond what either approach could achieve independently:

- The hybrid architecture achieved superior performance across all evaluation metrics (RMSE: 109.49, MAPE: 0.0205, MAE: 84.39) compared to both standalone models and existing hybrid approaches.
- The CNN component contributed spatial pattern recognition capabilities that complemented the Transformer's temporal modeling, as evidenced by the 6.7% increase in RMSE when the CNN component was removed.
- Our feature fusion mechanism successfully combined representations from multiple modalities (visual, textual, and numerical) to create a more comprehensive market understanding.

The integration methodology we developed effectively bridges the gap between spatial and temporal feature extraction, allowing the model to simultaneously capture both local price movements (through CNN) and long-range dependencies (through Transformer). This multi-modal approach mirrors how human analysts combine chart pattern recognition with sequential trend analysis, providing a more holistic approach to market prediction than previous single-modality methods.

In summary, our hybrid framework successfully addresses all three research questions, demonstrating that the combination of linear self-attention Transformers, LLM-enhanced technical analysis, and CNN-based visual feature extraction creates a powerful and efficient architecture for stock price prediction. The empirical results confirm that this integrated approach outperforms both traditional methods and standalone deep learning models across multiple evaluation metrics and market conditions.

5.2 Interpretation of Results

Our hybrid framework, which integrates an LLM-enhanced technical analysis module with a linear self-attention Transformer and a CNN-based feature extractor, has demonstrated **notable improvements** in stock price prediction accuracy. As detailed in the results (Chapter 4), the proposed model achieved a substantially lower RMSE of 13.74 and a MAPE of 0.00249 on the test data, which is significantly better than the corresponding error rates of baseline methods. This superior performance indicates that the model captures underlying market patterns more effectively than traditional approaches. In particular, the relatively low MAPE (approximately 0.249%) reflects a high precision in predictive accuracy, suggesting that even small fluctuations in stock prices are being captured by our model.

Several factors contribute to these improvements. First, the inclusion of LLM-generated technical indicator summaries provided enriched contextual information beyond raw numerical indicators. The experimental evidence shows that this textual encapsulation of trends helped the model avoid being misled by short-term noise and focus on meaningful patterns. For example, our approach consistently tracked actual price movements closely, even during moderate volatility, whereas a comparable model without the LLM-derived features exhibited larger deviations. Second, the use of a linear self-attention mechanism enabled the Transformer to efficiently incorporate a longer historical window of data. By avoiding the quadratic complexity of standard attention, our model could analyze extended time sequences (spanning many trading days) without encountering computational bottlenecks, thereby capturing longer-term dependencies that would otherwise be truncated. The results confirm that this long-range insight improved the model's ability to anticipate trends, as evidenced by higher accuracy for longer forecast horizons in comparison to models using shorter look-back periods. Third, the CNN-based visual feature extractor contributed significantly by recognizing chart patterns (such as candlestick formations or support/resistance levels) that are difficult to encode in purely numeric form. In our ablation analysis (Section 4.7), removing the CNN input led to a noticeable rise in error rates, confirming that visual features from stock charts added unique predictive signals. Likewise, excluding the LLM-generated text features caused performance to degrade, underscoring the importance of each component in the hybrid model. These observations indicate that it is the **synergistic combination** of textual, temporal, and visual features that

underpins the model's strong performance.

It is also worth noting how the model performed under different market conditions. The approach maintained robust accuracy during both bullish and bearish periods included in the test set, suggesting that the learned representation generalizes across varying trend regimes. Some challenges remain—for instance, we observed that prediction errors were slightly larger during sudden, extreme market movements (e.g., abrupt price jumps caused by unforeseen news). This is expected, as even advanced models struggle with such outlier events. Nonetheless, overall, the model's ability to consistently outperform benchmark models across multiple evaluation metrics and market scenarios provides confidence in the effectiveness of the proposed architecture. The interpretation of these results is that each element of our hybrid approach contributes to capturing a different aspect of the complex stock price generation process, and together they yield a prediction system more powerful than the sum of its parts.

5.3 Comparison with Existing Studies

Our findings gain further significance when contextualized against existing studies in stock price prediction. In comparison to prior deep learning models, the proposed hybrid model achieves **competitive or superior performance**. For instance, recent studies that employed CNN-LSTM architectures or Transformer-based models for financial forecasting reported higher error metrics on similar tasks. Prior work on CNN-based stock prediction, such as (Hoseinzade and Haratizadeh, 2019), introduced CNNpred, which improved over classical methods but still exhibited larger prediction errors than those observed in our study. Similarly, traditional LSTM or GRU models often struggle to reduce MAPE below the 1% range for volatile stock data, as shown in (Weng et al., 2018), whereas our model's error is an order of magnitude smaller. While differences in datasets and experimental setups exist, these comparisons suggest that our approach is at least on par with, and in several aspects beyond, the state-of-the-art in the literature. In particular, the integration of an LLM-derived feature set and multi-modal data fusion distinguishes our work and appears to give it a measurable edge in accuracy. The lower error rates achieved by our model reinforce the trend noted in recent research: combining advanced architectures with richer data representations can yield tangible gains in predictive performance.

Beyond performance metrics, our framework offers conceptual and architectural advantages relative to existing approaches. Many prior studies in stock prediction focus on a single modality or a narrow set of features. For example, models such as those reviewed by (Sezer et al., 2020) primarily leverage either time-series price data or textual sentiment from news, but rarely both in a unified model. In contrast, our model bridges multiple data modalities: it concurrently analyzes textual technical analysis, numerical time-series data, and visual price charts. This comprehensive approach mirrors how human analysts often combine written technical commentary with chart inspection, yet it is largely novel in an automated deep learning context. To our knowledge, few if any existing studies have employed an LLM to generate technical analysis text as an input for a prediction model. Prior works that incorporate text, such as (Araci, 2019), typically use it to gauge market sentiment rather than to derive technical indicators. Our results demonstrate that this novel use of LLM-generated text can improve model performance, highlighting a key point of departure from earlier research.

Another important comparison lies in the model's efficiency and scalability. Transformer-based approaches in finance, particularly those inspired by (Vaswani, 2017), often face scalability issues due to the length of financial time series and high-frequency data. Researchers have proposed specialized time-series Transformers, such as Crossformer (Zhang and Duan, 2023) and DLinear (Wang et al., 2023a), to address these issues with mixed success. Our work contributes to this area by adopting a linear self-attention mechanism, which, in line with techniques like the Performer (Katharopoulos et al., 2020), reduces the complexity per time step from $O(n^2)$ to $O(n)$. This addresses a common limitation in standard Transformer models noted in existing studies. By maintaining comparable predictive power without the heavy computational cost, our model is more practical for large-scale or high-frequency data scenarios than many previously published models. This efficiency did not come at the expense of accuracy in our case, whereas in some existing studies, simplifications to the attention mechanism sometimes led to degraded performance. Therefore, our approach demonstrates a successful balance between complexity and accuracy that is of interest in comparison to related research.

In summary, the framework we present aligns with and extends the current body of literature. It corroborates the findings of earlier works that **multi-faceted deep learning models outperform traditional statistical methods** for stock forecasting (Wang et al., 2022), and it pushes the

frontier further by introducing an innovative combination of components. Our results not only confirm what previous studies have suggested—that incorporating more data types or advanced architectures can improve predictions—but also illustrate a new integrative pathway. This places our study in a comparative context as a **next step in the evolution** of stock prediction models, addressing gaps that previous approaches left open, such as efficient long-sequence handling and automated technical analysis generation. The differences and advantages discussed here underscore the unique contributions of our research relative to existing studies in the field.

5.4 Implications for Stock Price Prediction

The results of our study carry important implications for both academic research and practical financial applications. From a research perspective, our findings reinforce the notion that integrating multiple data modalities—numerical, textual, and visual—into deep learning models enhances predictive accuracy. Prior works have separately explored time-series modeling with LSTMs or Transformers (Wen et al., 2022), sentiment analysis from financial text (Araci, 2019), and CNN-based price pattern recognition (Hoseinzade and Haratizadeh, 2019), but few have effectively fused these approaches into a single coherent framework. Our study provides empirical evidence that a well-structured multi-modal approach yields superior forecasting performance while maintaining computational efficiency.

For industry practitioners, particularly quantitative analysts and algorithmic traders, the implications are equally significant. The proposed framework demonstrates that using LLM-generated technical summaries can augment conventional numerical indicators, thereby refining trading signals. This suggests that integrating generative AI into financial modeling workflows can provide actionable insights while reducing reliance on purely historical patterns. Additionally, the use of a linear self-attention Transformer enables the model to efficiently process long sequences of stock market data, making it scalable to high-frequency trading scenarios where standard Transformers may struggle with computational constraints.

5.5 Limitations of the Study and Future Directions

Despite the promising results, this study has several limitations that warrant further investigation. These limitations also suggest important directions for future work.

5.5.1 Market Generalizability

The model's performance was validated only on the S&P 500 index, raising concerns regarding generalizability. It remains unclear whether similar accuracy levels would hold across other financial markets, including international stock indices (e.g., NASDAQ, Dow Jones, FTSE 100), individual stocks, commodities, or cryptocurrencies. Market-specific factors such as volatility, liquidity, and efficiency could influence model performance. Conducting cross-market validation would provide valuable insights into its adaptability (Makridakis et al., 2018).

5.5.2 Computational Complexity and Real-Time Feasibility

Although linear self-attention improves efficiency, training deep learning models for financial forecasting still requires significant computational resources. While linear attention scales better than softmax-based self-attention (Katharopoulos et al., 2020), integrating additional components such as CNNs and LLMs increases computational demands. In live trading, where low-latency inference is essential, real-time applicability remains an open question. The deployment of a large LLM (e.g., ChatGPT-4 and DeepSeek) alongside text and image processing modules could introduce unacceptable delays, particularly in high-frequency trading environments. Future research should explore ways to optimize efficiency further, including model distillation, pruning, quantization, or adaptive online learning strategies (Dodge et al., 2020).

5.5.3 Interpretability and Explainability

One of the key criticisms of deep learning models in finance is their lack of transparency, often referred to as the black-box nature of machine learning

models. While our approach includes an LLM-generated textual explanation of technical indicators, it does not provide a comprehensive interpretability framework. Practitioners may hesitate to adopt models without clear justifications for predictions, especially in risk-sensitive financial environments. Future work should explore explainable AI (XAI) techniques such as attention heatmaps, Shapley Additive Explanations (SHAP), or transformer attention visualizations to make the decision-making process more interpretable (Makridakis et al., 2018).

5.5.4 Handling Market Regimes and Extreme Events

While our model demonstrated strong predictive performance across historical datasets, its robustness during real-time market conditions remains uncertain. Many deep learning-based financial models suffer from data drift, where patterns learned from past data no longer apply due to evolving market dynamics (Makridakis et al., 2018). This is particularly relevant during sudden market shocks, such as financial crises, geopolitical events, or algorithmic trading anomalies. Future work should investigate adaptive learning methods—such as reinforcement learning, meta-learning, or model retraining strategies—to maintain effectiveness in dynamic environments.

Chapter 6

Conclusion

The proposed hybrid deep learning framework, integrating an LLM-enhanced technical analysis module, a linear self-attention Transformer, and a CNN-based feature extractor, has demonstrated strong predictive performance for stock price forecasting. Our study builds upon prior research by showing that combining multi-modal data sources—numerical, textual, and visual—leads to significant improvements over traditional financial prediction models.

By addressing key challenges such as efficient long-sequence processing and leveraging textual insights from LLMs, this work contributes to both theoretical and applied advancements in financial machine learning. While limitations remain, particularly concerning real-time deployment and generalizability, the results indicate that linear self-attention mechanisms can be a viable alternative to standard Transformer models in financial applications.

Future research should focus on refining interpretability, expanding to other financial markets, and integrating real-time data streams to further enhance model adaptability and robustness. The continued evolution of hybrid AI architectures holds promise for more accurate, scalable, and interpretable financial forecasting models.

6.1 Summary of Contributions

This research has made several significant contributions to the field of financial forecasting:

1. Development of a novel hybrid architecture integrating LLMs, linear

self-attention Transformers, and CNNs for multi-modal stock price prediction

2. Introduction of an innovative approach to using LLMs for technical analysis rather than sentiment analysis
3. Implementation of a linear self-attention mechanism that reduces computational complexity while maintaining forecasting accuracy
4. Creation of a feature fusion methodology that effectively combines textual, visual, and numerical market data
5. Empirical validation across diverse market conditions demonstrating consistent outperformance over baseline models

These contributions collectively advance the state-of-the-art in stock price prediction by addressing key limitations in existing approaches while maintaining computational efficiency and interpretability.

6.2 Future Work

Building on the foundation established in this thesis, several promising research directions emerge:

- Extension of the model to diverse financial markets and asset classes
- Development of multi-horizon forecasting capabilities for various time intervals
- Enhanced interpretability mechanisms to provide explainable predictions
- Further optimization of computational efficiency for real-time applications
- Integration of fundamental analysis alongside technical indicators
- Adaptation mechanisms for evolving market conditions and regime shifts

These future research directions could further enhance the practical applicability and robustness of the proposed framework across different market environments and use cases.

6.3 Closing Remarks

The hybrid deep learning framework presented in this thesis represents a significant advancement in the application of artificial intelligence to financial forecasting. By integrating complementary architectures and leveraging multiple data modalities, our approach provides a more comprehensive and accurate model of stock price movements. As deep learning and natural language processing technologies continue to evolve, the integration methodologies developed in this research offer a foundation for increasingly sophisticated financial prediction systems that combine numerical analysis, visual pattern recognition, and textual interpretation. The demonstrated improvements in prediction accuracy, computational efficiency, and model adaptability suggest that such hybrid approaches will play an increasingly important role in financial analysis and algorithmic trading systems.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Hussein Ahmad Ahmad, Seyyed Kasra Mortazavi, Mohamed El Bahnasawi, Fadi Al Machot, Witesyavwirwa Vianney Kambale, and Kyandoghere Kyamakya. Enhanced time series forecasting: Integrating patchtst with bert layers. In *2024 International Conference on Applied Mathematics & Computer Science (ICAMCS)*, pages 60–65. IEEE, 2024.

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *The 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2623–2631, 2019.

Fahad Radhi Alharbi and Denes Csala. A seasonal autoregressive integrated moving average with exogenous factors (sarimax) forecasting model-based time series approach. *Inventions*, 7(4):94, 2022.

Dogu Araci. Finbert: Financial sentiment analysis with pre-trained language models. *arXiv preprint arXiv:1908.10063*, 2019.

Adebiyi A Ariyo, Adewumi O Adewumi, and Charles K Ayo. Stock price prediction using the arima model. In *2014 UKSim-AMSS 16th international conference on computer modelling and simulation*, pages 106–112. IEEE, 2014.

Asha Ashok and CP Prathibhamol. Improved analysis of stock market prediction:(arima-lstm-smp). In *2021 4th Biennial International Confer-*

ence on Nascent Technologies in Engineering (ICNTE), pages 1–5. IEEE, 2021.

Suryoday Basak, Saibal Kar, Snehanshu Saha, Luckyson Khaidem, and Sudeepa Roy Dey. Predicting the direction of stock market prices using tree-based classifiers. *The North American Journal of Economics and Finance*, 47:552–567, 2019.

Zahra Berradi, Mohamed Lazaar, Hicham Omara, and Oussama Mahboub. Effect of architecture in recurrent neural network applied on the prediction of stock price. *IAENG International Journal of Computer Science*, 47(3): 436–441, 2020.

Indronil Bhattacharjee and Pryonti Bhattacharja. Stock price prediction: a comparative study between traditional statistical approach and machine learning approach. In *2019 4th international conference on electrical information and communication technology (EICT)*, pages 1–6. IEEE, 2019.

Ashwathy Bhooshan and VS Hari. Recurrent neural network estimator for stock price. In *2021 International Conference on Electrical, Communication, and Computer Engineering (ICECCE)*, pages 1–6. IEEE, 2021.

Sawyer Birnbaum, Volodymyr Kuleshov, Zayd Enam, Pang Wei W Koh, and Stefano Ermon. Temporal film: Capturing long-range sequence dependencies with feature-wise modulations. *Advances in Neural Information Processing Systems*, 32, 2019.

Tom B Brown. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

Yahya Eru Cakra and Bayu Distiawan Trisedya. Stock price prediction using linear regression based on sentiment analysis. In *2015 international conference on advanced computer science and information systems (ICACSIS)*, pages 147–154. IEEE, 2015.

Yanshuo Chang, Wei Lu, Feng Xue, and Xinyu Lu. Combining market-guided patterns and mamba for stock price prediction. *Alexandria Engineering Journal*, 113:287–293, 2025.

Jian Chen, Guohao Tang, Guofu Zhou, and Wu Zhu. Chatgpt and deepseek: Can they predict the stock market and macroeconomy? *arXiv preprint arXiv:2502.10008*, 2025.

Yu Chen, Ruixin Fang, Ting Liang, Zongyu Sha, Shicheng Li, Yugen Yi, Wei Zhou, and Huilin Song. Stock price forecast based on cnn-bilstm-eca model. *Scientific Programming*, 2021(1):2446543, 2021.

Changqing Cheng, Akkarapol Sa-Ngasoongsong, Omer Beyca, Trung Le, Hui Yang, Zhenyu Kong, and Satish TS Bukkapatnam. Time series forecasting for nonlinear and non-stationary processes: a review and comparative study. *Iie Transactions*, 47(10):1053–1071, 2015.

Narayana Darapaneni, Anwesh Reddy Paduri, Himank Sharma, Milind Manjrekar, Nutan Hindlekar, Pranali Bhagat, Usha Aiyer, and Yogesh Agarwal. Stock price prediction using sentiment analysis and deep learning for indian markets. *arXiv preprint arXiv:2204.05783*, 2022.

Shaveta Dargan, Munish Kumar, Maruthi Rohit Ayyagari, and Gulshan Kumar. A survey of deep learning and its applications: a new paradigm to machine learning. *Archives of Computational Methods in Engineering*, 27: 1071–1092, 2020.

Yiqi Deng, Xingwei He, Jiahao Hu, and Siu-Ming Yiu. Enhancing few-shot stock trend prediction with large language models. *arXiv preprint arXiv:2407.09003*, 2024.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186, 2019.

Yuqi Ding. Enhancing stock price prediction method based on cnn-lstm hybrid model. *Highlights in Business, Economics and Management*, 21: 774–781, 2023.

Matthew F Dixon, Igor Halperin, and Paul Bilokon. *Machine learning in finance*, volume 1170. Springer, 2020.

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *arXiv preprint arXiv:2002.06305*, 2020.

Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Holger Drees and Catalin Starica. A simple non-stationary model for stock returns. 2002.

Jithin Eapen, Doina Bein, and Abhishek Verma. Novel deep learning model with cnn and bi-directional lstm for improved stock market index prediction. In *2019 IEEE 9th annual computing and communication workshop and conference (CCWC)*, pages 0264–0270. IEEE, 2019.

Chuanzhi Fan and Xiang Zhang. Stock price nowcasting and forecasting with deep learning. *Journal of Intelligent Information Systems*, pages 1–18, 2024.

Gabriel Pui Cheong Fung, Jeffrey Xu Yu, and Wai Lam. News sensitive stock trend prediction. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 481–493. Springer, 2002.

Jie Gao. Research on stock price forecast based on arima-garch model. In *MSIEID 2022: Proceedings of the 4th Management Science Informatization and Economic Innovation Development Conference, MSIEID 2022, December 9-11, 2022, Chongqing, China*, page 317. European Alliance for Innovation, 2023.

Tianchen Gao, Jiashun Jin, Zheng Tracy Ke, and Gabriel Moryoussef. A comparison of deepseek and other llms. *arXiv preprint arXiv:2502.03688*, 2025.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Shayan Halder. Finbert-lstm: Deep learning based stock price prediction using news sentiment analysis. *arXiv preprint arXiv:2211.07392*, 2022.

Ehsan Hoseinzade and Saman Haratizadeh. Cnnpred: Cnn-based stock market prediction using a diverse set of variables. *Expert Systems with Applications*, 129:273–285, 2019.

Xinyu Huang, Jun Tang, and Yongming Shen. Long time series of ocean wave prediction based on patchtst model. *Ocean Engineering*, 301:117572, 2024.

Huseyin Ince and Theodore B Trafalis. Short term forecasting with support vector machines and application to stock price prediction. *International Journal of General Systems*, 37(6):677–687, 2008.

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

Fatima Juairiah, Mostafa Mahatabe, Hasan Bin Jamal, Aysha Shiddika, Tanvir Rouf Shawon, and Nibir Chandra Mandal. Stock price prediction: A time series analysis. In *2022 25th International Conference on Computer and Information Technology (ICCIT)*, pages 153–158. IEEE, 2022.

Uday Kamath, Kevin Keenan, Garrett Somers, and Sarah Sorenson. Large language models: A deep dive, 2024.

Anika Kanwal, Man Fai Lau, Sebastian PH Ng, Kwan Yong Sim, and Siva Chandrasekaran. Bicudnnlstm-1dcnn—a hybrid deep learning-based predictive model for stock price prediction. *Expert Systems with Applications*, 202:117123, 2022.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International conference on machine learning*, pages 5156–5165. PMLR, 2020.

Asifullah Khan, Zunaira Rauf, Anabia Sohail, Abdul Rehman Khan, Hifsa Asif, Aqsa Asif, and Umair Farooq. A survey of the vision transformers and their cnn-transformer based variants. *Artificial Intelligence Review*, 56(Suppl 3):2917–2970, 2023.

Taewook Kim and Ha Young Kim. Forecasting stock prices with a feature fusion lstm-cnn model using different representations of the same data. *PloS one*, 14(2):e0212320, 2019.

David Krause. Deepseek and fintech: The democratization of ai and its global implications. *Available at SSRN 5116322*, 2025.

Farida Titik Kristanti, Mochamad Yudha Febrianta, Dwi Fitrizal Salim, Hosam Alden Riyadh, Yoga Sagama, and Baligh Ali Hasan Beshr. Advancing financial analytics: Integrating xgboost, lstm, and random forest algorithms for precision forecasting of corporate financial distress. *Journal of Infrastructure, Policy and Development*, 8(8):4972, 2024.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

Takako Kumamoto, Yunko Yoshida, and Himari Fujima. Evaluating large language models in ransomware negotiation: A comparative analysis of chatgpt and claude. 2023.

B Shravan Kumar and Vadlamani Ravi. A survey of the applications of text mining in financial domain. *Knowledge-Based Systems*, 114:128–147, 2016.

Mahinda Mailagaha Kumbure, Christoph Lohrmann, Pasi Luukka, and Jari Porras. Machine learning techniques and data for stock market forecasting: A literature review. *Expert Systems with Applications*, 197:116659, 2022.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Chen Li, Xu Zhang, Mahboob Qaosar, Saleh Ahmed, Kazi Md Rokibul Alam, and Yasuhiko Morimoto. Multi-factor based stock price prediction using hybrid neural networks with attention mechanism. In *2019 IEEE intl conf on dependable, autonomic and secure computing, intl conf on pervasive intelligence and computing, intl conf on cloud and big data computing, intl conf on cyber science and technology congress (DASC/PiCom/CBDCom/-CyberSciTech)*, pages 961–966. IEEE, 2019.

Qing Li, Jinghua Tan, Jun Wang, and Hsinchun Chen. A multimodal event-driven lstm model for stock prediction using online news. *IEEE Transactions on Knowledge and Data Engineering*, 33(10):3323–3337, 2020a.

Zhengtao Li, Guokun Chen, and Tianxu Zhang. A cnn-transformer hybrid approach for crop classification using multitemporal multisensor images. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 13:847–858, 2020b.

Bingru Liu and Qing Zhao. Financial derivative price forecasting and trading for multiple time horizons with deep long short-term memory networks. *Scientific Programming*, 2022(1):6526512, 2022.

Yong Liu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Non-stationary transformers: Exploring the stationarity in time series forecasting. *Advances in Neural Information Processing Systems*, 35:9881–9893, 2022.

Christoph Lohrmann and Pasi Luukka. Classification of intraday s&p500 returns with a random forest. *International Journal of Forecasting*, 35(1): 390–407, 2019.

Tim Loughran and Bill McDonald. Textual analysis in finance. *Annual Review of Financial Economics*, 12(1):357–375, 2020.

Jiecheng Lu and Shihao Yang. Linear transformers as var models: Aligning autoregressive attention mechanisms with autoregressive forecasting. *arXiv preprint arXiv:2502.07244*, 2025.

Wenjie Lu, Jiazheng Li, Jingyang Wang, and Lele Qin. A cnn-bilstm-am method for stock price prediction. *Neural Computing and Applications*, 33 (10):4741–4753, 2021.

Yuze Lu, Hailong Zhang, and Qiwen Guo. Stock and market index prediction using informer network. *arXiv preprint arXiv:2305.14382*, 2023.

Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. Statistical and machine learning forecasting methods: Concerns and ways forward. *PloS one*, 13(3):e0194889, 2018.

Tashreef Muhammad, Anika Bintee Aftab, Muhammad Ibrahim, Md Mainul Ahsan, Maishameem Meherin Muhu, Shahidul Islam Khan, and Mohammad Shafiul Alam. Transformer-based deep learning model for stock price prediction: A case study on bangladesh stock market. *International Journal of Computational Intelligence and Applications*, 22(03):2350013, 2023.

Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. *arXiv preprint arXiv:2211.14730*, 2022.

Bhawna Panwar, Gaurav Dhuriya, Prashant Johri, Sudeept Singh Yadav, and Nitin Gaur. Stock market prediction using linear regression and svm. In *2021 International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, pages 629–631. IEEE, 2021.

Jiayu Qiu, Bin Wang, and Changjun Zhou. Forecasting stock prices with long-short term memory neural network based on attention mechanism. *PloS one*, 15(1):e0227222, 2020.

Xiangfei Qiu, Jilin Hu, Lekui Zhou, Xingjian Wu, Junyang Du, Buang Zhang, Chenjuan Guo, Aoying Zhou, Christian S Jensen, Zhenli Sheng, et al. Tfb: Towards comprehensive and fair benchmarking of time series forecasting methods. *arXiv preprint arXiv:2403.20150*, 2024.

Hojjat Salehinejad, Sharan Sankar, Joseph Barfett, Errol Colak, and Shahrokh Valaee. Recent advances in recurrent neural networks. *arXiv preprint arXiv:1801.01078*, 2017.

Jaydip Sen, Sidra Mehtab, and Abhishek Dutta. Stock price prediction using machine learning and lstm-based deep learning models. *Authorea Preprints*, 2021.

Akhil Sethia and Purva Raut. Application of lstm, gru and ica for stock price prediction. In *Information and Communication Technology for Intelligent Systems: Proceedings of ICTIS 2018, Volume 2*, pages 479–487. Springer, 2019.

Omer Berat Sezer, Mehmet Ugur Gudelek, and Ahmet Murat Ozbayoglu. Financial time series forecasting with deep learning: A systematic literature review: 2005–2019. *Applied soft computing*, 90:106181, 2020.

Jaiwin Shah, Rishabh Jain, Vedant Jolly, and Anand Godbole. Stock market prediction using bi-directional lstm. In *2021 International Conference on Communication information and Computing Technology (ICCICT)*, pages 1–5. IEEE, 2021.

R Shankar, D Sridhar, and K Sivakumar. Systematic analysis of blue-chip companies of nifty 50 index for predicting the stock market movements using anfis machine learning approach. *J. Math. Comput. Sci.*, 11(1):265–277, 2020.

Gilad Sharir, Asaf Noy, and Lihi Zelnik-Manor. An image is worth 16x16 words, what is a video worth? *arXiv preprint arXiv:2103.13915*, 2021.

Suraj Prakash Sharma, R Jeyanthi, and K Deepa. Forecasting india s&p bse sensex and usa s&p-500 benchmark indices using sarimax and facebook prophet library. In *2022 6th International Conference on Intelligent Computing and Control Systems (ICICCS)*, pages 1523–1530. IEEE, 2022.

Norman Sondak. Neural networks and artificial intelligence. *ACM SIGART Bulletin*, (1989), 1989.

Hyunsun Song and Hyunjun Choi. Forecasting stock market indices using the recurrent neural network based hybrid models: Cnn-lstm, gru-cnn, and ensemble models. *Applied Sciences*, 13(7):4644, 2023.

Polipireddy Srinivas and Rahul Katarya. hyoptxg: Optuna hyper-parameter optimization framework for predicting cardiovascular disease using xgboost. *Biomedical Signal Processing and Control*, 73:103456, 2022.

Cătălin Stărică and Clive Granger. Nonstationarities in stock returns. *Review of economics and statistics*, 87(3):503–522, 2005.

Leo WG Strijbosch, Aris A Syntetos, John E Boylan, and Elleke Janssen. On the interaction between forecasting and stock control: The case of non-stationary demand. *International Journal of Production Economics*, 133 (1):470–480, 2011.

Ruey S Tsay. *Analysis of financial time series*. John wiley & sons, 2005.

A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

Chaojie Wang, Yuanyuan Chen, Shuqi Zhang, and Qiuhui Zhang. Stock market index prediction using deep transformer model. *Expert Systems with Applications*, 208:118128, 2022.

Fei Wang, Wanling Chen, Bahjat Fakieh, and Basel JA Ali. Stock price analysis based on the research of multiple linear regression macroeconomic variables. *Applied Mathematics and Nonlinear Sciences*, 7(1):267–274, 2021.

Gang Wang, Yu Liao, Li Guo, Jiahao Geng, and Xianchao Ma. Dlinear photovoltaic power generation forecasting based on reversible instance normalization. In *2023 IEEE 12th Data Driven Control and Learning Systems Conference (DDCLS)*, pages 990–995. IEEE, 2023a.

Huiqiang Wang, Jian Peng, Feihu Huang, Jince Wang, Junhui Chen, and Yifei Xiao. Micn: Multi-scale local and global context modeling for long-term series forecasting. In *The eleventh international conference on learning representations*, 2023b.

Shuheng Wang, Guohao Li, and Yifan Bao. A novel improved fuzzy support vector machine based stock price trend forecast model. *arXiv preprint arXiv:1801.00681*, 2018.

Qingsong Wen, Tian Zhou, Chaoli Zhang, Weiqi Chen, Ziqing Ma, Junchi Yan, and Liang Sun. Transformers in time series: A survey. *arXiv preprint arXiv:2202.07125*, 2022.

Bin Weng, Lin Lu, Xing Wang, Fadel M Megahed, and Waldyn Martinez. Predicting short-term stock prices using ensemble methods and online data sources. *Expert Systems with Applications*, 112:258–273, 2018.

Xiaojian Weng, Xudong Lin, and Shuaibin Zhao. Stock price prediction based on lstm and bert. In *2022 International Conference on Machine Learning and Cybernetics (ICMLC)*, pages 12–17. IEEE, 2022.

Cort J Willmott and Kenji Matsuura. Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance. *Climate research*, 30(1):79–82, 2005.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in neural information processing systems*, 34:22419–22430, 2021.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. *arXiv preprint arXiv:2210.02186*, 2022.

Shijie Wu, Ozan Irsoy, Steven Lu, Vadim Dabravolski, Mark Dredze, Sebastian Gehrmann, Prabhanjan Kambadur, David Rosenberg, and Gideon Mann. Bloomberggpt: A large language model for finance. *arXiv preprint arXiv:2303.17564*, 2023.

Caosen Xu, Jingyuan Li, Bing Feng, and Baoli Lu. A financial time-series prediction model based on multiplex attention and linear transformer structure. *Applied Sciences*, 13(8):5175, 2023.

Min Xu and Zhongfeng Qin. A novel hybrid arima and regression tree model for the interval-valued time series. *Journal of Statistical Computation and Simulation*, 91(5):1000–1015, 2021.

Xiaorui Xue, Shaofang Li, and Xiaonan Wang. Enhanced forecasting of stock prices based on variational mode decomposition, patchtst, and adaptive scale-weighted layer. *arXiv preprint arXiv:2408.16707*, 2024.

Yong Yu, Xiaosheng Si, Changhua Hu, and Jianxun Zhang. A review of recurrent neural networks: Lstm cells and network architectures. *Neural Computation*, 31(7):1235–1270, 2019.

Feiniu Yuan, Zhengxiao Zhang, and Zhijun Fang. An effective cnn and transformer complementary network for medical image segmentation. *Pattern Recognition*, 136:109228, 2023.

Risma Yulistiani and Felix Indra Kurniadi. Stock price prediction with the informer model. In *2024 International Conference on Information Management and Technology (ICIMTech)*, pages 49–53. IEEE, 2024.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 11121–11128, 2023.

Cheng Zhang, Nilam Nur Amir Sjarif, and Roslina Ibrahim. Deep learning models for price forecasting of financial time series: A review of recent advancements: 2020–2022. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 14(1):e1519, 2024.

Jianan Zhang and Hongyi Duan. Enhanced lftsformer: A novel long-term financial time series prediction model using advanced feature engineering and the ds encoder informer architecture. *arXiv preprint arXiv:2310.01884*, 2023.

Jilin Zhang, Lishi Ye, and Yongzeng Lai. Stock price prediction using cnn-bilstm-attention model. *Mathematics*, 11(9):1985, 2023.

Yaojun Zhang and Gilbert M Tumibay. Stock market prediction model based on lstm deep learning: The case of top corporate company in china. In *2022 4th International Conference on Artificial Intelligence and Advanced Manufacturing (AIAM)*, pages 451–455. IEEE, 2022.

Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *The eleventh international conference on learning representations*, 2023.

Zequan Zhao. Application and challenges of informer model in financial time series prediction: A review. *Applied and Computational Engineering*, 38: 90–95, 2024.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 11106–11115, 2021.

Lei Zhou, Yuqi Zhang, Jian Yu, Guiling Wang, Zhizhong Liu, Sira Yongchareon, and Nancy Wang. Llm-augmented linear transformer–cnn for enhanced stock price prediction. *Mathematics*, 13(3):487, 2025.

Tian Zhou, Ziqing Ma, Qingsong Wen, Liang Sun, Tao Yao, Wotao Yin, Rong Jin, et al. Film: Frequency improved legendre memory model for long-term time series forecasting. *Advances in neural information processing systems*, 35:12677–12690, 2022a.

Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International conference on machine learning*, pages 27268–27286. PMLR, 2022b.

Peng Zhu, Yuante Li, Yifan Hu, Qinyuan Liu, Dawei Cheng, and Yuqi Liang. Lsr-igru: Stock trend prediction based on long short-term relationships and improved gru. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 5135–5142, 2024.

Yoshua X ZXhang, Yann M Haxo, and Ying X Mat. Analyzing the s&p 500 index fund: A comprehensive review. *AC Investment Research Journal*, 220(44), 2023.