MSPRACFormer

A Multi-Scale Transformer Architecture for Cardano Price Forecasting with Retrieval-Augmented Context

Huali Zhao, BSc, MSc

Supervised by Prof Martin Crane and Dr Marija Bezbradica



A thesis presented for the degree of Doctor of Philosophy

School of Computing

Dublin City University

Dec 2025

Declaration

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the award of Doctor of Philosophy is entirely my own work, and that I have exercised reasonable care to ensure that the work is original and have conformed to the regulations on the use and declaration of Generative AI, and does not to the best of my knowledge breach any law of copyright, and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work.

Signed (Huali Zhao):

ID No.: 18214581

Date: 01/12/2025

Reporting of Gen AI Tool Use in Research Theses

Tools: Clearly indicate the tool name and model (or version) you used and the date(s) of use. If you used a tool multiple times, provide the range of dates over which the tool was used and the model/versions used.	• Google Gemini, 2.0 Flash, used between 15/05/2025 and 17/06/2025 • Google Gemini, 2.5 Flash, used between 17/06/2025 and 30/11/2025 • NotebookLM, version not specified, used between 15/05/2025 and 30/11/2025
Outputs: Specify what tool outputs were generated and used by you and for what purpose.	• I used Gemini to help me find most relevant and high-quality literatures for my literature review. • I used Gemini for grammar checking and suggest improvements. • I used NotebookLM to create summaries of articles. It helped me to decide whether to add them into my reading list.
Prompts: Provide a record of the prompts you used here or link to an appendix where the prompts used are listed.	• I asked Gemini to "Find me the recent peer-reviewed, published journal papers that focus on cryptocurrency forecasting with attention mechanisms". The resulting papers are manually retrieved via Google and then reviewed. • I asked Gemini to "Identify the grammar errors, improper word usages, and suggest appropriate academic phrasing for the lines…". The returned suggestions are manually reviewed and assessed. The final written lines in this thesis are revised manually with or without incorporating the tool's suggestions. • I asked NotebookLM to "Find me papers that is relative to a topic or a dataset". The returned results are manually reviewed to determine its relevance for a citation.
Iterations: Provide some commentary on the number and type of iterations. This can be linked to your report on prompts and outputs above.	For finding a paper, I use two iterations. Firstly, I ask Gemini to list out recent high quality papers that related to a specific topic. The follow up prompt ask for more similar papers with a time range, e.g. publish time is after 2023.
Critical Reflection: If Gen AI tools have been used minimally, a very brief critical reflection on its role in the research submitted for examination can be included here. Otherwise, specify the parts of the thesis where critical reflection on the use and role of Gen AI tools is included.	Gemini is very efficient on searching out relevant papers, and helpful on written improvement. However, I find Gemini occasionally hallucinate information and misinterpret published materials. Therefore, all the returns from Gemini are manually reviewed and verified. Similarly, all the answers from NotebookLM are assessed to avoid misunderstanding of the resources.

Acknowledgements

I still remember the first few meetings with my supervisors, Prof. Martin Crane and Dr. Marija Bezbradica, they repetitively reminded me that Phd would be a long and challenging journey that may bring mixed feelings. I listened and thought I understood what lay ahead. However, research is something you truly learn by doing. To me, it is a lot like Hyrox (I was going to say a marathon, but I have never actually run one.) It began with excitement and energy, followed by fatigue, exhaustion, watching others passing by, and trying to remind myself not to give up. There were times I questioned why I even choose to enter the game in the first place. Then I paused, took small breaks, and keep moving forward. At times I felt a spark of excitements again and the cycle repeated until I hit the finish line, received the medal, and felt proud. A heartfelt thank you to my supervisors, Marin and Marija, who have been like trainers throughout this race. They constantly reminding me that it is not a competition and everyone runs their own course. At the same time, they offered gentle nudges to keep me on track with timeline. Most importantly, they dedicated countless late nights and weekends to support my work. Without their unwavering guidance and encouragement, I might have stepped out of the race long ago.

Throughout this journey, I have received support from many people. I would like to thank:

Dr. An Pham Ngoc Nguyen for kindly sharing a key dataset essential to my research, providing examples of his paper and thesis, and offering valuable feedback on my work.

Prof. Cathal Currin for encouraging words during our annual review meetings which motivated me.

Dr. Michael Scriney and Dr. Liting Zhou for their valuable suggestions during my transfer talk.

Although, as a part time Phd student, I was not able to be on campus regularly, I felt part of a warm and friendly community where we could share both our ideas and our lives. I am deeply grateful to the researchers at the School of Computing for making me feel included and supported through this journey.

Contents

Declaration i

Acknowledgements iii

List of Figures viii

List of Tables xiii

List of Abbreviations xvi

List of Publications xix

Abstract 1

1 Introduction 3

2 Literature Review 12

2.1 Introduction 12

2.2 Predicting Foreign Exchange 13

2.2.1 Summary 16

2.3 Cryptocurrency Price Forecasting Methods 16

2.3.1 Statistical Models 17

2.3.2 Conventional Machine Learning Models 30

2.3.3 Deep Neural Networks 41

2.3.4 Synthesis of Cryptocurrency Forecasting Methods: Data, Strengths and Limitations, Challenges and Research Gaps 52

2.4 Recent Transformer-based architectures for TSF 57
2.4.1 Limitations of canonical Transformers in TSF 57
2.4.2 Computational complexity reduction 58
2.4.3 Patching - Enhancing temporal information modeling 59
2.4.4 Channel-independence (CI) vs Channel-dependence (CD) . . . 61
2.5 Retrieval Augmented(RA) Techniques for TSF 64
2.6 Summary 74

3 Methodology 77
3.1 Introduction 77
3.2 Dataset 78
3.2.1 Data collection 78
3.2.2 Extreme Values of Volatility 80
3.2.3 Cross-Asset Correlation of Cryptocurrency Markets 81
3.2.4 Revealing Recurring Cycles: The Fast Fourier Transform (FFT) 83
3.2.5 Dataset for experiments 85
3.3 Experimental Setup 86
3.3.1 Forecasting problem definition 86
3.3.2 Experimental pipeline 87
3.3.3 Evaluation Metrics 89
3.4 MSPRACFormer: Multi-Scale Patching with Retrieval-Augmented Context Transformer 91

4 Empirical Evaluation of Architectural Options 96
4.1 Introduction 96
4.2 LSTM vs Transformer 97
4.3 Patching 104
4.3.1 Single-scale patching vs Multi-scale patching 104
4.3.2 FFT-based patching scale selection 108
4.4 Channel-Dependent (CD) vs Channel-Independent (CI) 111

4.4.1 Experimental results and findings . . . 113
4.5 Retrieval-Augmented Context . . . 116
4.5.1 Experimental results and findings . . . 120
4.6 Conclusion . . . 126

5 Results and discussion **131**
5.1 Introduction . . . 131
5.2 Experimental results and analysis . . . 132

6 Conclusion **141**
6.1 Contribution . . . 142
6.2 Limitations and Future Work . . . 146

A Data analysis **A1**

B A history of Transformer **B1**
B.1 Transformer . . . B1
B.1.1 A history of attention: from fixed vector to Transformer's scale-dot attention . . . B2
B.1.2 Transformer architecture . . . B6

C Predicted vs. True Value Plots for ADA, BTC and ETH forecasting results **C1**

D Artefects **D1**

List of Figures

1.1 Overview of the thesis structure 11

2.1 Models used in cryptocurrency forecasting. 18

2.2 The Process of Patching 60

2.3 Comparison of CI and CD processing for training multivariate TSF(reproduced from Han et al.[54]). In CI processing, each channel is learned separately using a shared set of model parameters across all channels. In contrast, CD processing learns all channels jointly, explicitly modeling inter-channel dependencies. 63

3.1 Historical price patterns of the top 10 market capitalization cryptocurrencies from 1st January 2020 to 1st January 2025. The panel displays the log-price (in USD) series to highlight relative growth dynamics. 80

3.2 Boxplots of daily log-returns for the top 10 cryptocurrencies 82

3.3 Power spectral density (PSD) of ADA, BTC, and ETH log-returns, restricted to periods between one trading week (5-day) and one trading year (252-day). 85

3.4 Workflow of the proposed experimental pipeline, from raw data preprocessing to model testing. 87

3.5 MSRACFormer architecture 93

4.1 Methods chart: This chart links each sub-section of Section Empirical Evaluation of Architectural Choices to its corresponding hypothesis, highlighting the key findings that shape the proposed MSPRACFormer architecture. 98
4.2 LSTM and Transformer prediction results comparison with input sequence length of 21 days for different prediction horizons 101
4.3 LSTM and Transformer prediction results comparison with input sequence length of 60 days for different prediction horizons 102
4.4 Proposed multi-scale patching architecture (CD). A historical time series window of length L ($L = 8$ in the example) and N ($N = 4$ in the example) features is transformed into multiple patch representations at different scales, processed in a CD way through the Transformer, and aggregated into the final forecast. 106
4.5 Proposed multi-scale patching architecture (CI). A historical time series window of length L ($L = 8$ in the example) and N features ($N = 4$ in the example) is transformed into multiple patch representations at different scales, processed in a CI way through the Transformer, and aggregated into the final forecast. 112
4.6 Comparison of time series forecasting before and after RAC. It can be observed that an informative historical sequence is retrieved and concatenated with the input sequence prior to its processing by the forecasting model. The augmented input sequence is hypothesized to enable the forecasting model to yield more accurate predictions. . . . 118
4.7 Percentage change ($\Delta\%$) in MAPE for each model after applying RAC across different forecasting horizons (input sequence length of 21). Negative values indicate improved performance (lower error), while positive values indicate performance degradation relative to the no-RAC baseline. 121

4.8 Percentage change (Δ%) in RMSE for each model after applying RAC across different forecasting horizons (input sequence length of 21). Negative values indicate performance improvement, while positive values indicate degradation relative to the no-RAC baseline. . . . 122

4.9 Percentage change (Δ%) in MAPE for each model after applying RAC across different forecasting horizons (input sequence length of 7). Negative values indicate performance improvement, while positive values indicate degradation relative to the no-RAC baseline. 123

4.10 Percentage change (Δ%) in RMSE for each model after applying RAC across different forecasting horizons (input sequence length of 7). Negative values indicate performance improvement, while positive values indicate degradation relative to the no-RAC baseline. 124

5.1 RMSE and MAE comparison across different horizons for ADA price forecasting using MSPRACFormer, PatchTST, iTransformer, MSGNet, TimeXer, LSTM and DLinear. Lower values indicate better performance. The best performing model for each horizon is highlighted with a star, and the second best performer is denoted with a diamond. 138

5.2 RMSE and MAE comparison across different horizons for BTC price forecasting using MSPRACFormer, PatchTST, iTransformer, MSGNet, TimeXer, LSTM and DLinear. Lower values indicate better performance. The best performing model for each horizon is highlighted with a star, and the second best performer is denoted with a diamond. 139

5.3 RMSE and MAE comparison across different horizons for ETH price forecasting using MSPRACFormer, PatchTST, iTransformer, MSGNet, TimeXer, LSTM and DLinear. Lower values indicate better performance. The best performing model for each horizon is highlighted with a star, and the second best performer is denoted with a diamond. 140

A.1 Rolling Pearson correlation between ADA and the other top nine cryptocurrencies using a 30 days window. A1
A.2 Rolling Pearson correlation between ADA and the other top nine cryptocurrencies using a 60 days window. A2
A.3 Rolling Pearson correlation between ADA and the other top nine cryptocurrencies using a 90 days window. A2
A.4 Rolling Pearson correlation between ADA and the other top nine cryptocurrencies using a 180 days window. A3
A.5 Average rolling Pearson correlation heatmaps with window sizes of 30, 60, 90, and 180 days. A3
A.6 Rolling Spearman correlation between ADA and the other top nine cryptocurrencies using a 30 days window. A4
A.7 Rolling Spearman correlation between ADA and the other top nine cryptocurrencies using a 60 days window. A4
A.8 Rolling Spearman correlation between ADA and the other top nine cryptocurrencies using a 90 days window. A5
A.9 Rolling Spearman correlation between ADA and the other top nine cryptocurrencies using a 180 days window. A5
A.10 Average rolling Spearman correlation heatmaps with window sizes of 30, 60, 90, and 180 days. A6

B.1 Attention architecture proposed by Bahdanau et al.[245] B3
B.2 Global and local attention mechanisms[246] B4

B.3 Demonstration of Self-Attention in action[247] B5
B.4 The canonical Transformer architecture[20] B7

C.1 ADA prediction with MSPRACFormer C1
C.2 ADA prediction with iTransformer C2
C.3 ADA prediction with PatchTST C2
C.4 ADA prediction with MSGNet C3
C.5 ADA prediction with TimeXer C3
C.6 ADA prediction with DLinear C4
C.7 ADA prediction with LSTM C4
C.8 BTC prediction with MSPRACFormer C5
C.9 BTC prediction with iTransformer C5
C.10 BTC prediction with PatchTST C6
C.11 BTC prediction with MSGNet C6
C.12 BTC prediction with TimeXer C7
C.13 BTC prediction with DLinear C7
C.14 BTC prediction with LSTM C8
C.15 ETH prediction with MSPRACFormer C8
C.16 ETH prediction with iTransformer C9
C.17 ETH prediction with PatchTST C9
C.18 ETH prediction with MSGNet C10
C.19 ETH prediction with TimeXer C10
C.20 ETH prediction with DLinear C11
C.21 ETH prediction with LSTM C11

D.1 Directory structure of the MSRACFormer framework D2

List of Tables

1.1 Research questions and associated hypotheses 9

2.1 Summary of literature on cryptocurrency TSF using statistical methods 29

2.2 Summary of literature on cryptocurrency TSF using machine learning methods 40

2.3 Summary of literature on cryptocurrency TSF using deep learning methods 51

2.4 Transformer-based models. This table is organized chronologically-based on the publication date of each model and ranked by the computational complexity of their attention mechanism. 61

2.5 Dataset description. This table presents detailed descriptions of the datasets utilized by the Transformer-based models. 62

2.6 Summary of the methods employed to measure similarity between the current input x_{input} and historical sequences x_h across RA-based time series studies. It also details the approaches used to integrate the retrieved historical sequences $x_{1...k}$ into downstream forecasting models. 73

3.1 Top 10 cryptocurrencies by market capitalization 78

3.2 Descriptive statistics of daily log-returns for the top 10 cryptocurrencies 81

3.3 Top 10 Fourier frequencies by amplitude for ADA, BTC and ETH. . . 84

3.4 Sample of multivariate daily closing prices for selected cryptocurrencies used in the forecasting experiments 86

3.5 Sample of the processed data before training 88

4.1 Architecture of the LSTM model. 97

4.2 Encoder-only Transformer architecture used for cryptocurrency forecasting, as recommended by Optuna. 99

4.3 Prediction results for LSTM and Transformer with input sequence length of 21. Best results for each prediction horizon are highlighted in bold. 100

4.4 Prediction results for LSTM and Transformer with input sequence length of 60. Best results per horizon are highlighted in bold. 100

4.5 Prediction performance comparison among single-scale patching Transformer, multi-scale patching Transformer, canonical Transformer, as well as the LSTM baseline, using an input sequence length of 21. The best results for each prediction horizon are highlighted in bold, the second best results are marked with the superscript $^{(*)}$ 107

4.6 Prediction performance comparison among Transformer-based models with single-scale, multi-scale patching, multi-scale patching with FFT selected patch scales and the baseline LSTM, using an input sequence length of 21. The best results for each prediction horizon are highlighted in bold, the second best results are marked with the superscript $^{(*)}$ 110

4.7 Prediction performance comparison among CI and CD version of Transformer-based models with single-scale, multi-scale patching, multi-scale patching with FFT selected patch sizes, as well as the LSTM baseline, using an input sequence length of 21 days. The best results for each prediction horizon are highlighted in bold, the second best result are marked with the superscript $^{(*)}$ 115

4.8 Forecasting performance across horizons with input sequence 21 (21 days of cryptocurrencies daily prices). For each model and horizon, RAC vs. no-RAC changes are indicated after the RAC values with arrows : ↓ means improvement (lower error), ↑ means degradation. The best result for each prediction horizon are highlighted in bold. . . . 127

4.9 Forecasting performance across forecasting horizons (1-, 3-, 5-, 7-day) with input sequence 7 (7 days of cryptocurrencies daily prices). For each model and horizon, RAC vs. no-RAC changes are indicated after the RAC values with arrows : ↓ means improvement (lower error), ↑ means degradation. The best result for each prediction horizon are highlighted in bold. 128

4.10 This table lists all the hypotheses and their corresponding sections where they are assessed. The last column demonstrates the key findings derived from each section. 130

5.1 ADA forecasting performance across horizons with input sequence 5 (5 days of cryptocurrencies daily prices). The best result for each prediction horizon are highlighted in bold, and the second best result is marked with (*). 133

5.2 BTC forecasting performance across horizons with input sequence 3 (3 days of cryptocurrencies daily prices). The best result for each prediction horizon are highlighted in bold, and the second best result is marked with (*). 134

5.3 ETH forecasting performance across horizons with input sequence 5 (5 days of cryptocurrencies daily prices). The best result for each prediction horizon are highlighted in bold, and the second best result is marked with (*). 135

D.1 Key hyperparameters of MSPRACFormer D3

List of Abbreviations

Abbreviation	Full Term
ADA	Cardano (Cryptocurrency)
ANN	Artificial Neural Networks
APARCH	Asymmetric Power ARCH
ARCH	Autoregressive Conditional Heteroskedasticity
ARIMA	AutoRegressive Integrated Moving Average
Bi-GRU	Bidirectional GRU
Bi-LSTM	Bidirectional LSTM
BNB	BNB (Cryptocurrency)
BT	Bagged Tree
BTC	Bitcoin (Cryptocurrency)
CD	Channel-Dependent
CGARCH	Component GARCH
CI	Channel-Independent
CNN	Convolutional Neural Networks
DL	Deep Learning
DOGE	Dogecoin (Cryptocurrency)
DQN	Deep Q-Networks
DT	Decision Tree
DTW	Dynamic Time Warping
EGARCH	Exponential GARCH
ETH	Ethereum (Cryptocurrency)

FbProphet	Facebook Prophet
FFT	Fast Fourier Transform
FGI	Fear and Greed Index
GB	Gradient Boosting
GARCH	Generalized Auto-Regressive Conditional Heteroskedasticity
GJR-GARCH	Glosten–Jagannathan–Runkle GARCH
GMV	Global Minimum Variance
GRNN	General Regression Neural Network
GRU	Gated Recurrent Unit
IGARCH	Integrated GARCH
KNN	K-Nearest Neighbors
LGARCH	Log GARCH
LLM	Large Language Model
LR	Linear Regression
LSNet	Long- and Short-Term Time-Series Network
LSTM	Long Short-Term Memory Network
MAC	Multiply-and-Accumulate
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
MDA	Mean Directional Accuracy
ML	Machine Learning
MLP	Multilayer Perceptron
MSE	Mean Squared Error
MSP	channel-independent Multi-Scale Patching module
NLP	Natural Language Processing
NN	Neural Network
OBOT	Optuna-Based Optimized Transformer
OHLCV	Open, High, Low, Close prices and Volume
PoS	Proof-of-Stake

RBFN	Radial Basis Function Network
RF	Random Forest
RFR	Random Forest Regression
RAG	Retrieval Augmented Generation
RAG	Retrieval Augmented Context
RAF	Retrieval Augmented Forecasting
RMSE	Root Mean Squared Error
RNN	Recurrent Neural Network
RRSE	Root Relative Squared Error
RW	Random Walk
SARIMA	Seasonal AutoRegressive Integrated Moving Average
SOL	Solana (Cryptocurrency)
SOTA	State-Of-The-Art
SVR	Support Vector Regression
SVM	Support Vector Machine
TCN	Temporal Convolutional Network
TGARCH	Threshold GARCH
TRX	TRON (Cryptocurrency)
USDC	USD Coin (Cryptocurrency)
USDT	Tether (Cryptocurrency)
WN	White Noise
XGBoost	eXtreme Gradient Boosting
XRP	XRP (Cryptocurrency)

List of Publications

1. **Attention! Transformer with sentiment on cryptocurrencies price prerediction**

   Huali Zhao, Martin Crane, Marija Bezbradica

   7th International Conference on Complexity, Future Information Systems and Risk

MSPRACFormer

A Multiscale Patching Transformer Architecture for Cardano Forecasting with Retrieval-Augmented Context

Huali Zhao

Abstract

Forecasting cryptocurrency prices has been a challenge due to their high volatility, non-stationary and regime-dependent dynamics. Recent advances in deep learning, particular Transformer architectures, have shown their superiority in time series forecasting by leveraging self-attention to capture long range dependencies. However, studies have shown that the canonical Transformer model does not consistently outperform recurrent architectures such as LSTMs when applied to cryptocurrencies. This thesis investigates the limitations of the canonical Transformer architecture for cryptocurrency forecasting and systematically examines a series of architectural enhancements, including different patching strategies, channel-dependent versus channel-independent processing, and retrieval-augmented mechanisms for generating informative contextual inputs, aimed at improving the model's predictive performance on forecasting future cryptocurrency prices. Building upon the insights from the empirical experiment results, this thesis proposes a Transformer-based predictive framework, MSPRACFormer (Multi-Scale Patching with Retrieval-Augmented Context Transformer). The framework supports both single-scale and multi-scale patching, and incorporates a RAC module for contextual retrieval. Each module is designed as modular components that can be flexibly incorporated into existing forecasting architectures. The framework allows dynamic configuration depending on the dataset characteristics and model architecture, thereby improving flexibility and generalization in time series forecasting. Empirical evaluations on forecasting future cryptocurrency prices using historical data from 1 January 2020

to 1 January 2025 of the top-10 cryptocurrencies confirm that this design delivers superior performance in capturing complex temporal patterns within the market.

Chapter 1

Introduction

Since the publication of Satoshi Nakamoto's white paper[1] in 2008, which introduced a decentralized, peer-to-peer electronic cash system, cryptocurrencies have undergone a transformation from niche technologies to mainstream financial instruments. According to PwC's 2025 Annual Global Crypto Hedge Fund Report[2], more than half (55%) of traditional hedge funds surveyed reported exposure to digital assets, increased from 47% in 2024, signaling a marked upward trend in institutional adoption. As shown on CoinMarketCap[3], the total market capitalization of all cryptocurrencies reached approximately 3.3 trillion USD as of June 2025. Given the rapid expansion of the market and growing participation from both institutional and individual investors, accurate forecasting of cryptocurrency price movements has become increasingly critical for mitigating risk and optimizing returns in this highly volatile financial market.

Within this broader market, Cardano (ADA) has emerged as a prominent leader as an eco-friendly altcoin[1]. As of April 10, 2025, CoinMarketCap reports Cardano's market capitalization at $22 billion, positioning it among the top 10 cryptocurrencies[3]. Cardano is distinguished by utilizing a Proof-of-Stake (PoS) protocol, which offers substantial energy efficiency while maintaining robust scalability and security[4]. Its PoS model allows users to stake ADA tokens for network validation and

[1]Altcoins are all cryptocurrencies that are not Bitcoin. sourced from `https://www.investopedia.com/terms/a/altcoin.asp`

governance, cultivating a decentralized and energy efficient ecosystem that appeals to both individual and institutional investors. While some studies [5, 6] report that PoS cryptocurrencies exhibit higher volatility[2] than PoW assets (e.g Bitcoin (BTC), Litecoin(LTC)) attributing this to factors such as network upgrades and regulatory developments that generate systemic uncertainty across the cryptocurrency subsector[6], other research [8] suggests that consensus mechanism alone does not fully determine price dynamics. Instead, PoW and PoS cryptocurrencies exhibit distinct behaviors within the broader information transmission structure of the cryptocurrency market[8, 9]. While PoW coins are identified as the primary net exporters of systemic uncertainty, PoS coins export significantly less risk to PoW coins. However, ADA emerges as a significant outlier[10]. Unlike its PoS counterparts, ADA functions as a notable transmitter of volatility, demonstrating a risk exporting profile. A recent study[11] shows that ADA acts as net exporter of both skewness and kurtosis spillovers, suggesting that its return distributions significantly influence the broader market. These higher-moment effects exhibit high persistence, indicating that ADA's impact on market instability is long-lasting. This influence is reinforced by its strong correlation with major assets, including BTC, Ethereum (ETH), LTC, and Binance Coin (BNB). Therefore, analyzing the price dynamics of ADA provides critical predictive insights into the movements of other major cryptocurrencies, offering valuable implications for robust portfolio optimization and risk management. In addition, a recent study[12] shows that Cardano ecosystem exhibits extreme wealth concentration. As the staking rewards are paid in ADA and scale almost proportionally with stake. Entities with larger holdings accumulate ADA at a faster rate. Given the highly unequal distribution of wealth in Cardano, accurate price forecasts is relevant not only for trading but also for understanding how price dynamics propagate through a highly unequal economy and influence wealth distribution over time.

However, according to the recent survey[13] that covers 234 cryptocurrency fore-

[2]Financial market volatility denotes the extent of variation in asset prices, capturing the magnitude of price movements over a specified time horizon[7].

casting literature from 2016 to 2023, most existing cryptocurrency price forecasting studies concentrate on BTC and ETH, the two dominant cryptocurrencies by market capitalization, only 1.8% of deep learning-based forecasting studies include ADA, highlighting a substantial gap in the literature.

To address this gap, it is important to understand the methodological landscape of cryptocurrency forecasting. Substantial research has explored a wide range of forecasting techniques. According to a comprehensive survey by Yousaf et al.[13], existing approaches span several methodological categories, including traditional economic models such as Auto Regressive Integrated Moving Average (ARIMA)[14]; conventional machine learning(ML) algorithms such as Support Vector Machines (SVM)[15] and Linear Regression(LR)[16]; deep reinforcement learning models such as Deep Q-Networks (DQN)[17]; and deep learning(DL) architectures such as Convolutional Neural Networks (CNN)[18], Long Short-Term Memory (LSTM) network[19] and Transformer[20].

Notably, the taxonomy of the methods applied in cryptocurrency forecasting closely mirrors those used in traditional foreign exchange(FX) forecasting[21, 22]. In both markets, although traditional statistical models such as ARIMA remain widely used, they exhibit several limitations. First, these models are typically built on the assumptions of linearity and stationarity which limits their ability to effectively model real-world financial data that are inherently nonlinear, nonstationary and noisy[23, 24]. Second, they require domain expertise to manually configure components such as trend, differencing, and seasonality, making the modeling process both time consuming and error-prone[25]. To address these challenges, conventional ML and DL methods have been proposed and achieved impressive performance[26–29]. However, ML and DL models often exhibit substantial forecasting errors and many traditional neural networks are not well suited to the inherent complexities of cryptocurrency time series such as high volatility and regime[3] shifts which ulti-

[3]Cryptocurrency market regimes are defined based on volatility: Low volatility (price movement is stable and predictable), high volatility (risker but with greater return potential) and distress (sharp downward movements)[30].

mately can lead to poor predictions[22, 31], and vanishing and exploding gradients, which restrict their ability to capture long-term temporal dependencies [25, 32, 33].

Introduced by Vaswani et al.[20], Transformer marked a major shift in sequence modeling by replacing recurrence and convolution with a self-attention based architecture. Its ability to process all tokens in parallel significantly improves training efficiency, while self-attention enables the model to capture dependencies between elements regardless of their distance in the sequence. With the remarkable success of Transformer-based architectures in natural language processing[20, 34] and computer vision[35], researchers have begun exploring their potential in Time Series Forecasting(TSF)[25, 32, 36–39]. These studies highlight that the canonical Transformer architecture faces significant limitations when applied to time series data: its quadratic computational complexity and difficulty in capturing temporal patterns effectively. To address these challenges, several studies[39–43] propose patching-based representations that segment time series data into grouped time steps. By transforming raw data points into patch-level tokens, these approaches enhance local representation learning while reducing computational overhead by shortening the sequence length. Existing patching strategies can be broadly categorized into single-scale patching[40, 44] and multi-scale patching[41, 45]. Single-scale patching partitions a time series into segments of fixed length, thereby enabling the model to learn temporal dependencies at a predefined resolution. In contrast, multi-scale patching constructs multiple sets of representations from the original time series using varying patch scales, allowing the model to simultaneously capture temporal dependencies across multiple time scales. Rather than selecting patch scales empirically, Fast Fourier Transform(FFT) is used to adaptively determine patch scales based on the dominant frequency components of the time series, thereby aligning patch segmentation with the underlying spectral characteristics of the data[41]. By employing the patching technique, the Transformer's self-attention mechanism becomes more efficient in processing temporal sequences and more capable of capturing complex temporal dependencies across multiple time scales. However, despite the demon-

strated effectiveness of the patching based Transformer architectures, the adoption of Transformer-based methods in cryptocurrency forecasting remains extremely limited. As noted by John et al.[46], Transformers are not widely documented in the existing cryptocurrency forecasting literature, although their application has the potential to enhance feature extraction from time-series data. Furthermore, Yousaf et al. report that only 1% of the cryptocurrency forecasting articles they analyzed employed Transformer based models[13], underscoring a significant gap in the literature. This presents an opportunity to explore how Transformer architectures can be adapted and optimized for the cryptocurrency market.

Beyond various modeling innovations in cryptocurrency forecasting, from a data perspective, existing studies have also adopted diverse strategies. A substantial portion of the literature[31, 47, 48] focus on single currency forecasting, where historical price of the target asset is used as the primary input to predict its future value. In contrast, a growing body of research[49, 50] emphasizes the importance of modeling cross-asset dependencies by incorporating information from multiple cryptocurrencies simultaneously as cryptocurrency markets exhibit strong interconnectedness which leads to significant spillover effects across assets[11]. Several studies[51–53] have demonstrated that incorporating multiple coins historical prices can enhance forecasting performance by capturing these inter-market dynamics. Motivated by these studies, our work adopts a multivariate forecasting framework that leverages the historical price dynamics of multiple major cryptocurrencies to predict the future price of a target asset. Within this multivariate setting, each variate refers to the historical price time series of an individual cryptocurrency, which is treated as a distinct input channel in the TSF framework. In the context of this study, each cryptocurrency's historical price series is treated as a separate data channel. Building upon this definition, we investigate two alternative strategies for modeling multivariate time series inputs: channel-dependent (CD) processing and channel-independent(CI) processing. CD processing jointly models interactions across channels, enabling the model to explicitly learn cross-asset dependencies[54]. In contrast,

CI processing models each channel separately, allowing the model to focus on asset-specific temporal dynamics before aggregating the learned representations[40].

Furthermore, from a data perspective, an emerging direction in the broader machine learning community is *Retrieval-Augmented Context(RAC)* with Retrieval-Augmented(RA) techniques[55]. Inspired by RAG[56], RAC introduces external context by retrieving relevant historical information, thereby enhancing a model's predictive capability without increasing model size. In TSF literature, this technique is commonly referred as Retrieval-Augmented Forecasting (RAF)[57–60]. When large language models (LLMs) are incorporated to generate the retrieved historical information, the approach is often termed Retrieval-Augmented Generation (RAG) [61–63]. In this study, since retrieval is used specifically to construct an informative historical context for the forecasting model, we refer to this component as Retrieval-Augmented Context (RAC). While RAC has demonstrated effectiveness on benchmark datasets such as weather[58], exchange[64], and hotel service volume prediction[55], its application to cryptocurrency forecasting is unexplored. To address this research gap, RAC was incorporated during the later stage of this study to evaluate its effectiveness across multiple Transformer-based architectures in the context of cryptocurrency price forecasting. In the proposed framework, RAC is implemented as a plug-in module that enables the dynamic incorporation of contextual information, allowing it to adapt to different forecasting models and task requirements. The detailed implementation is illustrated in Section 4.5.

Given the aforementioned gaps, this thesis investigates the following central research question(RQ):

How should the Transformer-based architectures be adapted to the variable price dynamics and complex regime-dependent inter-relationships of the top 10 market capitalization cryptocurrencies in order to improve ADA price forecasting?

To systematically address this question, the thesis adopts a research framework that introduces a set of sub-RQs and hypotheses derived from the central RQ (Table1.1), and empirically evaluates these hypotheses through a series of experiments.

This research framework guides the methodological design of the thesis. Each exper-

Central Research Question (CRQ): How should the Transformer-based architectures be adapted to the variable price dynamics and complex regime-dependent inter-relationships of the top 10 market capitalization cryptocurrencies in order to improve ADA price forecasting?	
Sub-RQ1: How does the canonical Transformer architecture perform in forecasting ADA prices using multivariate historical price data from the top 10 market-cap cryptocurrencies, compared to benchmark model LSTM?	**H1:** Theoretically, leveraging the multi-head self-attention mechanism, the canonical Transformer will outperform LSTM in cryptocurrency price forecasting. Specifically, the attention mechanism enables Transformers to model long-range dependencies without the vanishing gradient limitations of Recurrent Neural Network (RNN)
Sub-RQ2: How do different patching strategies influence the predictive performance of the canonical Transformer model in multivariate cryptocurrency price forecasting?	**H2.1:** Single-scale patching: fixed length single-scale patching benefits Transformer model in two ways: 1. reducing computational complexity by reducing sequence length; 2. improved learning of local temporal patterns. **H2.2:** Multi-scale patching: comparing with single-scale patching, multi-scale patching enables the Transformer model to learn local patterns in multiple scales, improving the model's forecasting ability from learning various length dependencies. **H2.3:** FFT-based multi-scale patching: FFT-based frequency selection provides patch sizes aligned with dominant periodicities in the data, enhancing the model's ability to capture cyclical behaviors.
Sub-RQ3: How do CI and CD multivariate processing strategies affect the forecasting performance of Transformer-based architectures?	**H3.1:** CD processing: with CD processing, Transformer model can learn the relationship across all the channels in the dataset (e.g. BTC-ETH price movement relationship), therefore dependencies across channels are explicitly modeled. **H3.2:** CI processing: with CI processing, Transformer models each channel in the dataset separately while sharing weights across channels. By decoupling the representation learning process, the model avoids distortions that may arise from entangled inter-channel correlations. This separation helps preserver long-term feature dynamics within each channel.
Sub-RQ4: To what extent does RAC improve the forecasting performance of Transformer-based architectures in the ADA price prediction task?	**H4:** Integrating RAC-generated contextual embeddings with historical price data improves forecasting accuracy by enriching the model with complementary information that extends beyond the local input sequence, thereby enabling the capture of temporal patterns.

Table 1.1: Research questions and associated hypotheses

iment in Chapter 4 is explicitly constructed to test one or more of these hypotheses, thereby providing a structured pathway toward answering the central research question and formulating the proposed architecture.

The thesis is structured as follows: Chapter 2 reviews the relevant literature on statistical, machine learning, and deep learning approaches to cryptocurrency forecasting; identifies the limitations of canonical Transformers in TSF, including their quadratic computational complexity and their difficulty in effectively capturing temporal patterns; surveys techniques used by SOTA Transformer-based TSF models to address the limitations; and investigates prior work on retrieval-augmented methods for TSF. Chapter 3 provides an analysis of the dataset used in this study, formalizes the forecasting problem, and describes the proposed MSPRACFormer framework. Chapter 4 illustrates the experiments that validate each hypothesis and identify effective techniques for MSPRACFormer. Chapter 5 presents the empirical results of the proposed framework, systematically evaluating its performance against SOTA benchmark models. To ensure the generalization capability of the proposed method is robustly validated, the experiments are conducted not only on the primary ADA price forecasting task but are also extended to BTC and ETH forecasting. Finally, Chapter 6 concludes the thesis with discussion on limitations and future work. The overall structure of the thesis is illustrated in Figure 1.1.

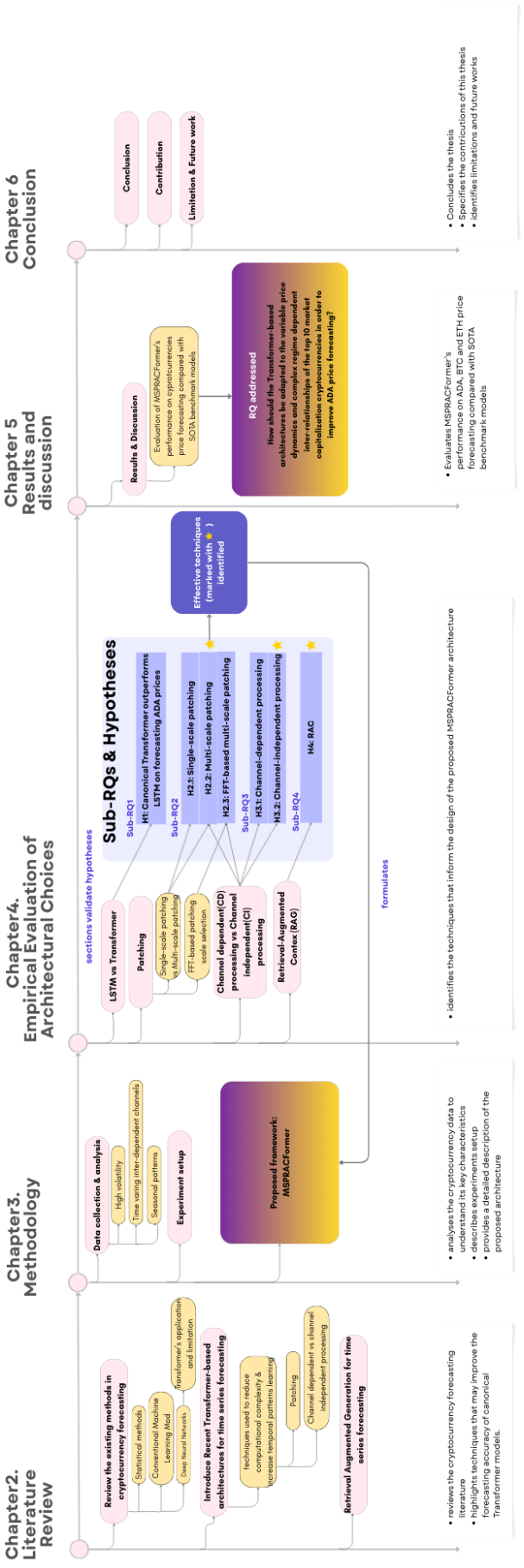


Figure 1.1: Overview of the thesis structure

Chapter 2

Literature Review

2.1 Introduction

This section provides a comprehensive review of approaches for cryptocurrency price forecasting, from conventional models to deep learning models, with a particular focus on Transformer-based architectures. The review is structured from both the model perspective and the data perspective, addressing SOTA advancements in both of the topics. From the model perspective, Section 2.2 reviews forecasting methodologies applied in traditional FX markets, establishing the methodological foundations commonly employed in conventional financial time-series analysis. Section 2.3 reviews the statistical models, conventional machine learning(ML), deep learning(DL) models and Transformer applied in cryptocurrency forecasting, and highlights the limitations of the canonical Transformer architecture within time-series settings. Section 2.4 reviews SOTA Transformer-based TSF models and discusses the techniques they employ to address existing limitations. Additionally, in Appendix B.1, we introduce a history of the canonical Transformer model in greater depth and traces its evolution from the Seq2Seq[65] framework, providing the conceptual foundation for Transformer-based forecasting methods. From the data perspective, motivated by the paradigm of RAG, this study adopts a novel approach to adaptively incorporate historical context information into Transformer-based models. The underlying concept and related studies are discussed in Section

2.5.

2.2 Predicting Foreign Exchange

The FX market is an international financial market where national currencies are traded against one another, and exchange rates are determined through continuous interactions among global market participants[66]. FX markets and cryptocurrency markets exhibit similar time series characteristics, including high volatility[67], non-stationarity[68] and regime-dependent dynamics[69]. Therefore, a review of forecasting methods developed in the traditional FX literature is essential to establishing the methodological grounding of cryptocurrency price prediction.

Before the widespread adoption of ML and DL techniques, FX rate forecasting was dominated by traditional statistical models such as ARIMA and Generalized Autoregressive Conditional Heteroskedasticity (GARCH)[23]. While these models provide valuable insights into linear dependencies and volatility dynamics, they rely on linearity assumptions that limit their ability to capture complex non-linear relationships in FX time series[70]. In practice, the FX market is influenced by a wide range of factors such as market sentiment and geopolitical events, which introduce non-linear and time-varying dynamics that cannot be adequately modeled using linear models[70]. Consequently, more recent studies have increasingly focused on ML and DL approaches to better capture the complex structure of FX markets.

Comparing with traditional statistic methods, ML methods can automatically identify and learn patterns from large amount of data without requiring manual parameter specification[71], as is typically needed in models such as ARIMA. These methods can adapt to new data to continuously improve their predictive capability. Deep learning methods are the models typically composed by multiple layers that learn hierarchal feature representations, making them more effective for processing unstructured data, pattern recognition and modeling complex, non-linear relationships[72].

To evaluate the effectiveness of ML models for forecasting the monthly USD/INR

exchange rate, Qureshi[71] compared Random Forest (RF) and SVM models with traditional statistical approaches, including ARIMA and the Random Walk (RW) model. The study utilized historical monthly USD/INR exchange rate data together with macroeconomic variables(interest rates, inflation rates, GDP growth, and foreign exchange reserves) covering the period from January 2015 to December 2023. The empirical results indicated that the RF model achieves the highest forecasting accuracy, with an R^2 of 0.79, Root Mean Square Error (RMSE) of 0.029, and Mean Absolute Error (MAE) of 0.021. In contrast, the RW model performs the worst, with an R^2 of 0, followed by the ARIMA model, which attains an R^2 of 0.43.

Madhulatha et al.[24] demonstrated the advantages of a deep learning model by comparing the forecasting performance of LSTM with traditional statistical methods, including ARIMA and Exponential Smoothing (ETS). Using 10 years of daily INR/USD exchange rate data together with relevant macroeconomic variables, including gold prices, crude oil prices, and inflation over from 2014 to 2024, their empirical results show that the LSTM model substantially outperforms both ARIMA and ETS across multiple evaluation metrics, including Mean Square Error (MSE), MAE, and RMSE.

To compare the forecasting accuracy of ML and DL methods, Niu[73] investigated the next day exchange rate forecasting performance of Linear Regression(LR), RF and LSTM. The study employs daily AUD/EUR exchange rate data from 2014 to 2024 with lagged features to capture temporal dependencies. MSE is adopted as the primary evaluation metric, as it is commonly regarded as a standard criterion for assessing point forecast accuracy[73]. The empirical results indicated that LSTM achieved the best predictive performance, yielding the lowest MSE value 0.0000254, followed by LR. In contrast, the RF model performed the worst with an MSE of 0.0000323. Similarly, Abouzaid et al.[70] compared the forecasting performance of a Multilayer Perception (MLP), RF and LSTM models for the monthly EUR/USD exchange rate. The study utilizes 10 years of monthly exchange rate data, together with macroeconomic indicators, including interest rates, inflation, GDP growth and

unemployment, spanning 2014 to 2024. Model performance is evaluated using R^2 to measure the proportion of variance explained, alongside RMSE and MAE to quantify the magnitude of prediction errors. The empirical results indicate that LSTM achieves the strongest performance, with an R^2 of 0.9234, RMSE of 0.0152 and MAE of 0.0112.

Somya et al.[74] compared the effectiveness of several deep learning models for forecasting the daily Thai Baht (THB) exchange rate against 5 major global currencies: USD, EUR, GBP, CNY and JPY. The dataset spans an 11-year period from 1 January 2013 to 31 December 2023. The models evaluated include Gated Recurrent Units (GRU), LSTM, Neural Basis Expansion Analysis for Interpretable Time Series (N-BEATS), and the Temporal Fusion Transformer (TFT). The empirical results indicate that the GRU model achieves the best overall performance across RMSE, MAE and MAPE, establishing it as the most accurate forecasting approach in this setting. The performance of the LSTM model is closely comparable, occasionally maching that of GRU in predictive accuracy.

Ayitey Junior et al.[21] conducted a systematic review of ML and DL applied to FX forecasting. Based on an analysis of 60 primary studies published between 2010 and 2021, the review identified LSTM as the most frequently adopted deep learning architecture, appearing in 19 studies. This prevalence was attributed to LSTM's strong ability to model non-linear temporal dependencies and to mitigate the vanishing gradient problem commonly encountered in traditional recurrent neural networks. Similarly, Vancsura et al.[22] reviewed of 100 studies published between 2014 to 2023 to identify widely used forecasting models across major financial markets. Focusing specifically on FX forecasting, the authors report that LSTM and GRU are the most frequently employed models due to their ability to capture sequential dynamics. Furthermore, the review finds that hybrid architectures, such as combined GRU-LSTM model, often achieve higher predictive accuracy than their standalone counterparts.

2.2.1 Summary

Recent studies have increasingly focused on ML and DL approaches for FX forecasting, as these methods are better suited to capturing the complex and non-linear dynamics of FX markets[22, 70]. Among these approaches, LSTM have emerged as one of the most widely adopted models due to their strong capability to model non-linear temporal dependencies[21]. Notebly, there is no standardised benchmark dataset in the FX forecasting literature. Existing studies examine different combinations of models across various currency pairs, often using datasets that span between 1 to 10 years[22]. In terms of evaluation metrics, commonly adopted performance metrics include MAE, MAPE, RMSE and MSE[21].

Building on this understanding of traditional currency market forecasting methods, it becomes evident that many of these approaches are also widely applied in cryptocurrency market forecasting. Accordingly, the following section presents a taxonomy of methods used in cryptocurrency price prediction.

2.3 Cryptocurrency Price Forecasting Methods

The rapid growth of the cryptocurrency market has stimulated substantial academic and practical interest in developing predictive methodologies for cryptocurrency price forecasting. As noted by Pečiulis et al.[75], academic output in this area has grown markedly since 2017, reflecting an increasing demand for robust analytical tools in this volatile domain.

The methodologies employed in cryptocurrency forecasting research can broadly be categorized into statistical approaches, conventional ML and DL based methods. Section 2.3.1 provides an overview of the statistical techniques commonly utilized in this domain and a discussion of their limitations. Section 2.3.2 and 2.3.3 reviews recent advancements ML in DL models respectively, highlighting their growing application and effectiveness in cryptocurrency price prediction. From the reviewed studies on cryptocurrency forecasting, the models can be grouped into categories

shown in Figure 2.1.

2.3.1 Statistical Models

Statistical models are mathematical frameworks that produce quantifiable predictions by leveraging statistical assumptions and probabilistic processes[76]. In the cryptocurrency price forecasting task, one of the most widely used statistical models has been ARIMA. The model utilizes its own lagged values and lagged forecast error to facilitate the forecasting of future values[76]. However, ARIMA struggles with datasets characterized by high volatility, sudden changes, and long-term forecasting[77, 78].

ARIMA

Azari's study[77] utilized BTC closing price data in USD from September 2015 to September 2018 to evaluate the performance of the ARIMA model in cryptocurrency forecasting. The findings indicated that ARIMA performs effectively in short-term (1-day) forecasting scenarios, especially when the time series is relatively stable[1]. However, during periods characterized by significant price fluctuations or for long-term[2] forecasting, the model was found to produce substantial errors. Similarly, Yang[78] conducted a comparative study of ARIMA models with varying hyperparameters, using daily BTC closing prices from January 2017 to September 2022. The results confirmed that ARIMA struggles with price fluctuations and is better suited for short-term (5-, 10-day) rather than long-term[3] forecasting. Latif et al.[79] conducted a comparative analysis of the ARIMA and LSTM model[19] using BTC closing price data at 10-minute intervals from December 21 2020, 00:00 to December 21, 2021, 16:00. Their findings revealed that the LSTM model significantly outper-

[1]Here, "stable" refers to a sub time-series exhibiting relatively constant statistical properties, predictable trends, and limited extreme volatility[77].

[2]The study does not provide a specific numerical value for "long-term", it uses the term to describe predictions that extend beyond the immediate next day.

[3]The study defines "long-term forecasting" by using the entire test set as the forecasting horizon, where the data span early 2021 to Sep 2022. Forecasting horizon here means how far ahead a model attempts to predict.

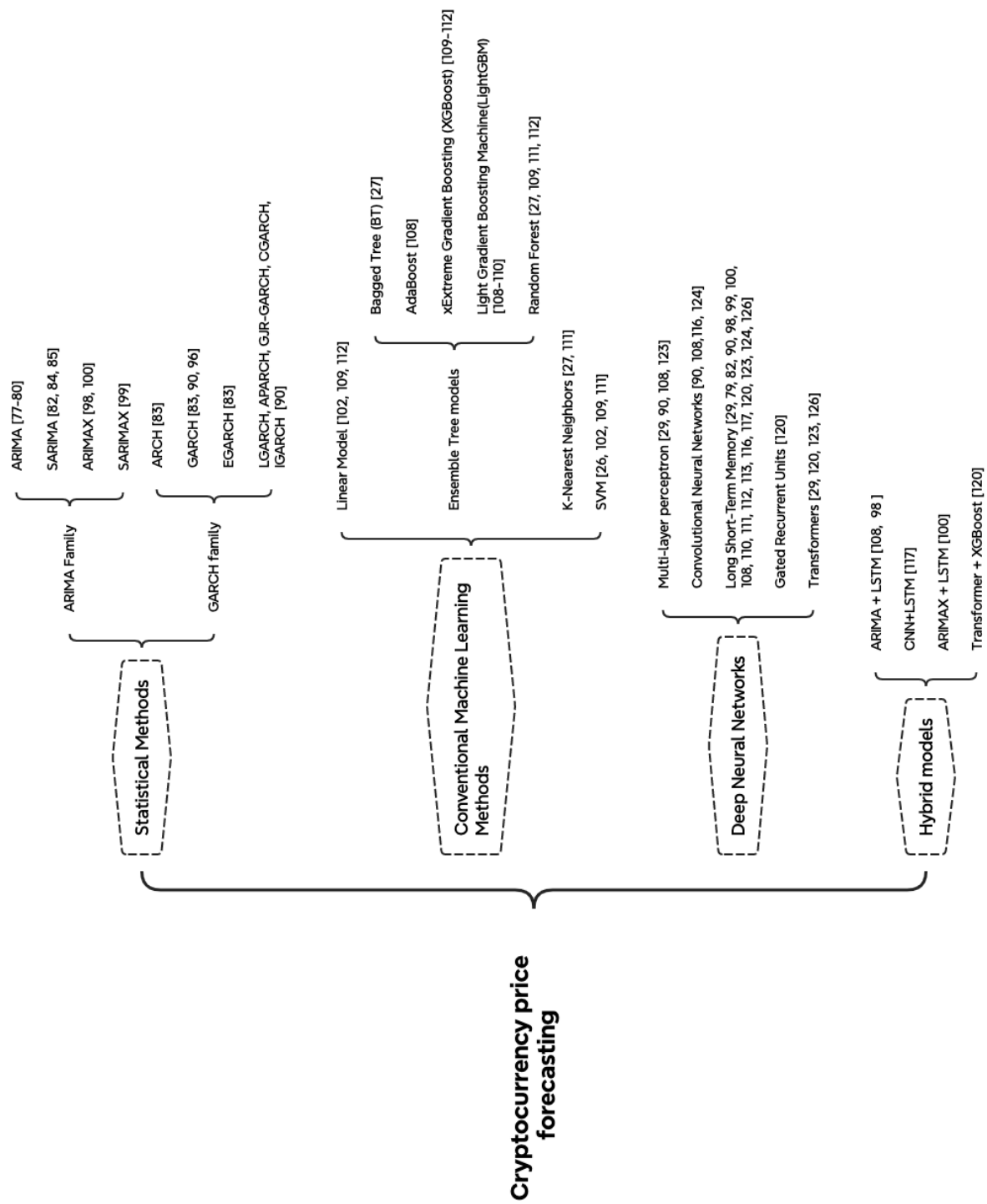


Figure 2.1: Models used in cryptocurrency forecasting.

formed ARIMA, achieving a forecasting accuracy of 99.73% on short-term forecasting[4]. Similarly, Ganesh et al.[80] compared ARIMA, LR, Random Forest Regression(RFR) and xExtreme Gradient Boosting(XGBoost)'s performance on predicting future BTC prices in USD using a BTC time series comprise by open, high, low, adjusted close and volume values, although the temporal coverage of the dataset was not specified. In their study, LR is employed to predict future BTC prices by modeling a linear relationship between past observations and future predicted values. ARIMA is selected for leveraging autocorrelations in historical observations and the residual errors from prior predictions. RFR was utilized for its capacity to model complex non-linear relationships with highly volatile cryptocurrency data through its robust ensemble architecture. XGBoost was adopted for its advances handling of diverse data types and its ability to uncover subtle dependencies via iterative gradient boosting. The results indicate that LR and RFR outperformed ARIMA model in forecasting accuracy. LR achieved the lowest errors, with an RMSE of 224.83, MAE of 158.04 and highest R^2 values of 0.9974 while RFR attained an R^2 of 0.99238. Although the authors confirms the stationary of the dataset before applying ARIMA model, the experiment results shows that ARIMA struggles with cryptocurrency datasets where high volatility and seasonal patterns exist[81].

SARIMA and GARCH for capturing seasonal patterns

To address the limitations of ARIMA model, statistical models such as Seasonal Autoregressive Integrated Moving Average (SARIMA) and GARCH have been employed in cryptocurrency price forecasting for extending ARIMA to capture seasonal patterns[82] and modeling time varying volatility variance[83] respectively.

In [82], Panigrahi compared the forecasting ability of SARIMA, LSTM and Facebook Prophet (FbProphet)[5] using daily USD prices of BTC, ETH, USDT, USDC, BNB, and XRP covering January 2019 to February 2024. The study identified dis-

[4]A prediction window contains data span nearly 20 hours. During this period, the models generate predictions for the next 261 periods, with each period representing a 10-minute interval

[5]FbProphet is an open source forecasting tool developed by Facebook for modeling time series data with strong seasonal effects approach. https://facebook.github.io/prophet/

tinct seasonal patterns in cryptocurrency markets during the COVID-19 pandemic period. Experimental results demonstrated that the LSTM model achieved the lowest prediction errors, outperforming both SARIMA and FbProphet, highlighting the advantage of DL in capturing nonlinear dependencies. Similarly, Zhu [84] compared ARIMA, SARIMA, and LR models for forecasting future BTC prices using historical price and volume data spanning 1st January 2012 to 31st March 2021. The results revealed that the LR model achieved the lowest forecasting errors in terms of MSE and MAE, whereas SARIMA produced the least accurate predictions. The author attributed this outcome to Bitcoin's weak seasonality, noting that BTC prices are largely non-seasonal compared to other financial time series. Since SARIMA is explicitly designed to model recurring seasonal patterns, its structure proved less suitable for capturing the stochastic and irregular nature of cryptocurrency price movements. Islam et al. [85] proposed EtherVoyant, a hybrid architecture that integrates ARIMA and SARIMA to forecast ETH prices. In this framework, the ARIMA component captures linear dependencies, while the SARIMA component models seasonal variations in ETH price dynamics. Using daily ETH open, high, low, close and volume data (OHLCV), and adjusted closing prices from 7 August 2015 to 18 October 2021, the authors compared ARIMA, SARIMA, and EtherVoyant models on the next day price forecasting task. The results showed that EtherVoyant achieved the best performance with MSE=0.009, MAPE=1.2, and RMSE=0.015, with SARIMA ranking second. The study concluded that, while ARIMA is effective for short-term(1-day) forecasting, it struggles with seasonality; conversely, SARIMA accounts for seasonal effects but may overfit under complex seasonal structures. The hybrid EtherVoyant framework successfully mitigates these limitations by combining these complementary statistical models to capture both linear and seasonal dynamics in cryptocurrency markets.

Hakan et al.[83] identified the GARCH model as the most effective framework for predicting the volatility of BTC returns by comparing the performance of the Autoregressive Conditional Heteroskedasticity (ARCH), GARCH, and Exponential

GARCH (EGARCH) models on logarithmic returns of weekly BTC prices from 11 November 2013 to 22 March 2020. The ARCH model captures volatility clustering by modeling conditional variance as a function of past squared residuals[86], while the GARCH model extends this framework by incorporating both autoregressive and moving average components in the conditional variance equation[83]. This extension mitigates the limitation of the ARCH model, where parameter estimates have been known to introduce negative variance[83]. However, GARCH assumes that volatility reacts symmetrically to positive and negative market shocks[87]. In cryptocurrency markets, this assumption may be unrealistic as downside shocks often exert a stronger impact on volatility than upside movements[88]. To address this limitation, the EGARCH model represents the logarithm of conditional variance[89], thereby ensuring positivity and capturing the asymmetric response of volatility to positive and negative shocks. The experimental results demonstrated that GARCH provided the best fit for Bitcoin return volatility, achieving the lowest Akaike Information Criterion (AIC) value among the evaluated models. Similarly, Khaldi et al.[90] performed a comprehensive evaluation of multiple GARCH-type models, including Log-GARCH (LGARCH)[6], EGARCH, Asymmetric Power ARCH (APARCH)[7], Threshold GARCH (TGARCH)[8], Glosten–Jagannathan–Runkle GARCH (GJR-GARCH)[9], Component GARCH (CGARCH)[10], and Integrated GARCH (IGARCH)[11]. These were benchmarked against ML models such as the Multilayer Perceptron (MLP) and RNN for Bitcoin volatility forecasting. Using daily Bitcoin price indices in USD from 1 October 2013 to 20 January 2019, the results indicated that TGARCH outperformed other GARCH-type models, while MLP surpassed both RNN and all

[6]Log-GARCH reformulates the GARCH variance equation on a logarithmic scale to ensure variance positivity [91].

[7]APARCH, proposed by Ding et al.[92], introduces a flexible power term to improve the model's ability to capture asymmetry in volatility.

[8]TGARCH, introduced by Zakoian[93], models conditional volatility directly, allowing positive and negative shocks to have asymmetric effects.

[9]GJR-GARCH [94] is a counterpart to TGARCH that models conditional variance rather than standard deviation.

[10]CGARCH [95] decomposes conditional variance into permanent and transitory components to examine long- and short-term volatility dynamics.

[11]IGARCH constrains the sum of model parameters to one, implying persistent shocks to volatility [90].

GARCH variants, demonstrating a superior capability to capture nonlinear volatility dynamics in Bitcoin returns for both short-term and long-term forecasting[12]. Shen et al.[96] conducted a comparative study involving GARCH, ARIMA, and several ML models including XGBoost and Light Gradient Boosting Machine (LightGBM) to forecast BTC, ETH, and Dogecoin(DOGE) prices using daily closing data from 1st January 2020, to 31st May 2022. Their findings revealed that both GARCH and ARIMA underperformed during periods of heightened market volatility, such as the COVID-19 pandemic, whereas ML models exhibited superior forecasting accuracy and robustness under extreme conditions.

ARIMAX and SARIMAX for exogenous variables

ARIMA with eXogenous feature (ARIMAX) and SARIMA with eXogenous feature (SARIMAX) is also widely used for cryptocurrency forecasting, as such data is known to be susceptible to exogenous shocks[97], particularly when incorporating heterogeneous data such as market sentiment as it extends ARIMA and SARIMA by modeling exogenous variables[98, 99].

Sembiring et al.[98] evaluated multiple model configurations combining traditional statistical and DL approaches for cryptocurrency price forecasting. The configurations included standalone ARIMAX and LSTM models, as well as several hybrid architectures: ARIMA-LSTM (single-split), ARIMAX-LSTM (single-split), ARIMA-LSTM (two-stage split), and ARIMAX-LSTM (two-stage split). The standalone ARIMAX model was employed to capture linear dependencies using lagged features, while the standalone LSTM model focused on modeling nonlinear relationships and long-range temporal dependencies. In the hybrid ARIMA-LSTM (single-split) framework, the ARIMA model first modeled the linear component of the series, and its residuals were subsequently learned by the LSTM network. The hybrid ARIMAX-LSTM (single-split) extended this by allowing ARIMAX to account for exogenous lagged features before modeling the residuals with LSTM.

[12]The study does not define fixed short- or long-term horizons, evaluating performance across 1-, 2-, 5-, 10-, and 20-day forecasts.

The two-stage split variants introduced a temporal partitioning strategy: ARIMA or ARIMAX was trained on an earlier time segment to model linear dynamics, and LSTM was trained on the residuals derived from forecasts of a later segment, enabling a temporal transfer of learned errors. The study utilized daily Bitcoin closing prices from 1 January 2015 to 31 December 2024 and evaluated model performance using MAE, RMSE, and Mean Absolute Percentage Error (MAPE). The results showed that the hybrid ARIMA-LSTM (two-stage split) achieved the best proportional accuracy with a MAPE of 2.6%, while the hybrid ARIMAX-LSTM (two-stage split) produced the lowest absolute errors, with MAE of 1583.69 and RMSE of 2056.47. These findings support the authors' hypothesis that decoupling the linear and nonlinear learning phases enhances forecasting accuracy by improving residual learnability and enabling the LSTM to capture complex, nonlinear structures more effectively. In [99], Abaji and Haraty compared the forecasting performance of SARIMAX, LSTM, and FbProphet models using two datasets: one containing Bitcoin's market price and volume features obtained from the Blockchain.com API, and another incorporating sentiment scores generated by a pre-trained Large Language Model (LLaMA-2) based on Bitcoin related news and tweets from Yahoo Finance and Twitter. The authors first evaluated the models using only the financial dataset, where SARIMAX achieved the best performance with the lowest MAPE of 1.328×10^{-1} and the highest prediction accuracy of 8.672×10^{-1}. When sentiment features were included, the LSTM model achieved superior performance, yielding the lowest MAPE of 1.275×10^{-1} and an accuracy of 8.725×10^{-1}. These results indicate that LSTM is more effective at integrating heterogeneous sentiment information and capturing subtle, nonlinear market dynamics. In contrast, SARIMAX struggled to translate sentiment-based features into improved predictive accuracy, highlighting its limitations in modeling complex unstructured data sources. Shrivastava et al.[100] conducted a comprehensive benchmarking study comparing a wide range of statistical, ML, and DL architectures for cryptocurrency price prediction. The statistical models included ARIMA and ARIMAX, while the ML and DL

models comprised Support Vector Regression (SVR), LSTM, and one-dimensional Convolutional Neural Networks (1D-CNN). In addition to these singular models, the authors evaluated ensemble architectures that combined CNN and LSTM layers, multi-modal configurations where CNN learned from market data and LSTM from sentiment data (and vice versa), and hybrid frameworks integrating statistical models (ARIMA or ARIMAX) with DL models (LSTM or CNN). The hybrid approach was designed to capture linear dependencies while DL models to learn complex nonlinear patterns. Two empirical studies were conducted. The first utilized hourly BTC OHLCV data combined with Twitter sentiment and a one-hour lag, covering January 2017 to August 2019. The second employed daily BTC and ETH OHLC data with Twitter sentiment and Google Trends, using one-day and five-day lags for BTC and ETH, respectively, spanning January 2019 to February 2021. Results from the first study demonstrated that LSTM achieved the best performance among singular and ensemble models, with the lowest RMSE of 4.56 $\times 10^{-3}$. In the second study, hybrid architectures integrating LSTM with ARIMA or ARIMAX consistently produced the lowest RMSE values, outperforming all singular, ensemble, and multi-modal alternatives. The findings indicate that hybrid statistical DL architectures are more effective for cryptocurrency forecasting, as they combine the linear modeling strength of ARIMA type models with the nonlinear feature learning capability of deep networks. Moreover, predictions for Ethereum exhibited greater stability and robustness across varying feature combinations compared to Bitcoin, while sentiment-based signals were found to be partially weakened by noise in social media data.

Summary

Table2.1 summarises the above literatuers employing statistical approaches for cryptocurrency forecasting. As demonstrated by studies [77, 78], the widely used ARIMA model performs effectively for short-term(1-, 5-, 10-day) forecasting horizons, but its performance deteriorates as the forecasting horizon extends. Compared with ML

and DL approaches, both ARIMA-type[82, 85, 99, 100] and GARCH-type[90, 96] models do not consistently outperform due to their limited capability in capturing complex temporal patterns within high volatile cryptocurrency markets[96] and integrating heterogeneous data sources such as social media sentiment[99].

Notably, the studies [79, 82, 98–100] demonstrate the strong performance of the LSTM, both as a standalone model and as a component within hybrid frameworks that integrate statistical and neural models. Together, these studies highlight the limitations of classical statistical models in capturing the nonlinear dynamics and extreme volatility characteristic of cryptocurrency markets, thereby underscoring the necessity of adopting more robust and adaptive modeling approaches to improve forecasting accuracy and stability.

Study	Method	Data	Data Source	Benchmark Model	Evaluation Metrics	Findings	Future Study
[77]	ARIMA	Daily closing BTC in USD from 1 Sep 2015 to 1 Sep 2018	N/A	N/A	Residual Sum of Squares (RSS), MSE	The ARIMA model is effective for short-term (1-day) forecasting but tend to produce substantial errors for long-term forecasting	More features to be used alongside price data for higher accuracy
[78]	ARIMA	Daily closing BTC in USD from Jan 2017 to Sep 2022	Yahoo Finance	N/A	RMSE, MAE, MAPE, MASE	ARIMA model demonstrates accurate forecasting for short-term (5-, 10-day) forecasting tasks, but performs poorly on long-term forecasting task; ARIMA is a basic statistical time series model, which may not capture the full complexity of Bitcoin's extreme price fluctuations	The study suggests that accuracy could be improved by integrating more advanced methods, specifically ML techniques
[79]	ARIMA	BTC closing price data at 10 minute intervals from December 21 2020, 00:00 to December 21, 2021, 16:00	Bloomberg Terminal	LSTM	MAE, Mean Absolute Deviation(MAD), Weighted Absolute Percentage Deviation(WAPE), Accuracy Percentage	LSTM outperformed ARIMA by accurately predicting both price direction and magnitude, whereas ARIMA only captured general trend movements.	The researchers suggest improving the LSTM model by utilizing minute-to-minute data to achieve more precise real-time results

Continued on next page

Study	Method	Data	Data Source	Benchmark Model	Evaluation Metrics	Findings	Future Study
[80]	ARIMA	OHLCV of BTC in USD	Yahoo Finance	LR, RFR, XGBoost	RMSE, MAE, R^2	ARIMA struggles with cryptocurrency datasets where high volatility and seasonal patterns exist and LR emerged as the top performer, showing the highest explanatory power and the lowest error rates	Developing frameworks that integrate the strengths of various of the models;Using rolling-window cross-validation to better evaluate a model's performance to market dynamics
[82]	SARIMA	Daily closing price in USD of BTC, ETH, USDT, BNB, USDC, and XRP span Jan 2017 to Feb 2024	Yahoo Finance	LSTM, FbProphet	MSE, RMSE	LSTM model consistently outperformed both SARIMA and FbProphet across all testing periods	Including a wider variety of cryptocurrencies; Investigating FbProphet's underperformance; Analysing inter-cryptocurrency connections
[84]	SARIMA	Daily BTC OHLC prices in USD and volume span 1 Jan 2012 to 31 Mar 2021	Kaggle	ARIMA, LR	MSE, MAE	SARIMA is the least accurate for Bitcoin because Bitcoin prices do not exhibit the strong seasonal patterns that SARIMA is designed to capture	Investigating more sophisticated ML models
[85]	EtherVoyant: an hybrid model(ARIMA + SARIMA)	Daily ETH OHLC prices in USD, volume and adjusted close span August 7, 2015 to October 18, 2021	Kaggle	ARIMA, SARIMA	MAE, MAPE, MSE, RMSE	EtherVoyant that decomposes price data into trend and seasonal components is superior to individual ARIMA and SARIMA models	Further enhancements on EtherVoyant to increase precision; Incorporating external factors such as macroeconomic data;

Continued on next page

Study	Method	Data	Data Source	Benchmark Model	Evaluation Metrics	Findings	Future Study
[83]	GARCH	Weekly BTC prices span 24 November 2013 to 22 March 2020	investing.com	ARCH, EGARCH	AIC	GARCH is identified as the best fit for Bitcoin return volatility	Expanding the study to include other cryptocurrencies such as ETH
[90]	7 GARCH-type models	Daily Bitcoin price indices in USD from 1 October 2013 to 20 January 2019	Coindesk.com	MLP, RNN	Metrics	MLP surpassed both RNN and all GARCH-type models	Applying more sophisticated models that incorporate external factors like Google Trends; Investigating modern RNNs such as LSTM; Exploring hybrid models;
[96]	GARCH, ARIMA	Daily close BTC, ETH, DOGE prices from 1 Jan 2020 to 31 May 2022	Yahoo Finance	XGBoost, LightGBM	MAE, MSE, MAPE, RMSE, R^2	Both GARCH and ARIMA underperformed during periods of heightened market volatility, whereas ML models exhibited superior forecasting accuracy and robustness	Adding larger variety of virtual currencies; Improving ML methods' accuracy by adding pandemic related factors;
[98]	Hybrid methods utilizing ARIMA, ARIMAX, LSTM	Daily Bitcoin closing prices in USD from 1 Jan 2015 to 31 Dec 2024	Yahoo Finance	Standalone ARIMA and LSTM	MAE, RMSE, MAPE	ARIMAX-LSTM achieved the lowest absolute errors; Decoupling linear and nonlinear learning improves forecasting accuracy by enhancing residual learning and capturing complex nonlinear patterns.	Incorporating more exogenous features such as macroeconomic indicators; Exploring DL methods; Implementing dynamic validation frameworks such as walk-forward validation;

Continued on next page

Study	Method	Data	Data Source	Benchmark Model	Evaluation Metrics	Findings	Future Study
[99]	SARIMAX	Daily BTC market price and volume; BTC news data	Blockchain.com; news-api.com	LSTM, FbProphet	MAPE, Accuracy	With financial data only, SARIMAX achieved the best performance; With sentiment features, LSTM achieved superior performance	Utilizing multi-years of data; Integrating the sentiment-augmented predictive models into broader portfolio management frameworks;
[100]	ARIMA, ARIMAX	Hourly BTC OHLCV with Twitter sentiment covering January 2017 to August 2019; Daily BTC and ETH OHLC data with Twitter sentiment and Google Trends spanning January 2019 to February 2021	Kaggle, Yahoo Finance, Google Trends	SVR, LSTM, 1D-CNN, ensemble models, hybrid models	RMSE, MAPE	LSTM outperformed other models on the first dataset, whereas LSTM–ARIMA/ARIMAX hybrids performed best on the second.	Shifting focus to return-based forecasting; Incorporating more sentiment data from various sources; Extending forecasting horizons

Table 2.1: Summary of literature on cryptocurrency TSF using statistical methods

2.3.2 Conventional Machine Learning Models

To address the limitations of statistical models, the application of ML models in cryptocurrencies research has increased significantly in recent years.[101]. Chaudhary and Saroj [102] conducted a comparative study evaluating the performance of LR, SVM, and Decision Tree (DT) in forecasting cryptocurrency prices. Their experiments aimed to predict the next five days' closing prices of BTC, ETH, DOGE, and Bitcoin Cash (BCH) using daily closing price data. The datasets covered the period from 31 December 2016 to 21 May 2022 for BTC, ETH, and DOGE, and from 23 July 2017 to 20 May 2022 for BCH. The authors employed LR due to its simplicity and effectiveness in modeling linear dependencies between variables, providing a benchmark for assessing the added value of more complex models. SVM was selected for its ability to map input features into higher dimensional spaces through kernel functions, allowing it to capture non-linear relationships that linear models might miss. DT was included for its intuitive structure and capacity to partition the feature space into non-linear, hierarchical decision boundaries without requiring extensive parameter tuning. The experimental results revealed that LR achieved the most consistent performance across all cryptocurrencies, attaining the highest accuracy of 98.61% for BTC. Although SVM demonstrated the worst results on DOGE (5.71% accuracy), it performed best on Bitcoin Cash (92.92% accuracy), indicating that its effectiveness varies with data characteristics. DT also exhibited relatively stable performance but did not outperform LR on any of the evaluated assets. Overall, the findings highlight that LR is generally more reliable and effective at handling the overall fluctuation and dynamism of the cryptocurrency market.

Similarly, Alsini et al.[27] examined the predictive capability of a Bagged Tree (BT) ensemble model for generating cryptocurrency buy signals. The study utilized 15-minute interval OHLCV data for BTC, ETH, ADA, and BNB spanning November 2018 to November 2021. After feature engineering, the authors constructed a dataset incorporating raw market variables (OHLCV), a range of technical indicators

(including RSI[13], MACD[14], Bollinger Bands[15]), and buy signal labels derived from the MACD, Bollinger Bands, and RSI indicators(MACD_buy_signal, BB_buy_signal and RSI_buy_signal). To evaluate the proposed model's forecasting performance, several benchmark models were selected: DT, K-Nearest Neighbors (KNN), Neural Network (NN), and RF. DT was employed for its interpretability and ability to capture non-linear decision boundaries through hierarchical partitioning of the feature space. KNN was included as a non-parametric model that predicts new observations based on their similarity to nearby training samples, making it effective for capturing local patterns in the data. Neural Networks were used for their capacity to model complex, non-linear interactions among multiple technical indicators, potentially uncovering deep representations of market dynamics. RF is an ensemble of randomized decision trees, served as a strong benchmark due to its proven robustness against overfitting and its effectiveness in handling noisy financial data[105]. The proposed BT model is constructed by training multiple decision trees on bootstrapped subsets of the data and aggregating their predictions. The core advantage is that merging these models' predictions improves overall accuracy while reducing the risk of overfitting. The experimental results demonstrated that the BT model achieved leading or near leading accuracy across all buy signal types, including 100% accuracy on RSI_buy_signal (tied with DT and RF), 95.73% on MACD_buy_signal (highest among all models), and 89.39% on BB_buy_signal (second only to RF). These findings suggest that the proposed method offers strong potential for trading signal generation in volatile cryptocurrency markets.

Zhou et al.[26] proposed a multi-source learning framework for cryptocurrency price movement prediction and portfolio optimization. The study utilized a SVM

[13]Relative Strength Index (RSI): It is often used to highlight the relative strength of a security in relation to the market on which it is traded or with a different security[103].

[14]Moving Average Convergence/Divergence (MACD): It is an indicator that measures price momentum, meaning the strength and direction of a trend in a stock's price[104].

[15]Bollinger Bands are a volatility-based technical indicator composed of three lines: a central line, typically defined as a moving average, and two outer bands positioned at an equal distance above and below it. They are commonly used to assess market volatility and to identify potential reversal points, as price movements toward or beyond the outer bands may signal overbought or oversold conditions[27].

model to forecast 1-, 3- and 5-day price movements (rise or fall) for seven major cryptocurrencies including BTC, ETH, XRP, ADA, DOGE, DOT, and LTC. The input features combined historical daily price data with social media sentiment indicators derived from Twitter and Google Trends, covering the period from 1 January 2021 to 31 August 2021. The SVM is used to find an optimal hyperplane makes it suitable for forecasting future price movements based on the multi-source data. To evaluate the economic value of the predictions, the authors integrated the SVM's directional forecasts with a Global Minimum Variance (GMV) portfolio optimization model, forming a hybrid trading strategy. Experimental results demonstrated that the SVM model achieved an average next day prediction accuracy of 65% across the seven cryptocurrencies when using multi-source data. Furthermore, the out-of-sample portfolio analysis revealed that the proposed SVM-GMV strategy consistently achieved higher Sharpe ratios[16], Sortino ratios[17] and Certainty Equivalent Returns[18] than traditional benchmark strategies across 1-, 3-, and 5-day holding periods. These findings highlight the benefit of integrating sentiment signals with market data to enhance short-term cryptocurrency investment decisions.

Bouteska et al.[108] conducted a comprehensive comparative study to evaluate the predictive performance of various ML and DL models for cryptocurrency price forecasting and to assess their effectiveness in trading applications through their simulated buy and sell[19], long and short trading strategies[20]. The study utilized daily log transformed USD prices of BTC, ETH, XRP and LTC covering 1 April 2016 to 31 August 2023. To capture potential structural differences before and after the COVID-19 shock, the dataset was divided into two sub-periods: a pre-

[16]Sharpe ratio expresses the excess return, or equivalently, the investor's reward per unit of (total) risk[106].

[17]Sortino ratio is a modification of the Sharpe ratio. It recognizes that investors prefer upside risk rather than downside risk and utilizes semi-standard deviation[107].

[18]Certainty Equivalent Return is the guaranteed (zero-risk) return that an investor would accept instead of a larger return associate with risk. Available at: `https://www.nasdaq.com/glossary/c/certainty-equivalent-return`

[19]This strategy replicates the action of buying a cryptocurrency for a fixed amount of money every day and selling it at the end of the day[108].

[20]A long strategy invests only when the predicted return is positive, and a short strategy invests only when the predicted return is negative[108].

COVID-19 period (April 2016-December 2019) and a post-COVID-19 period (January 2020–August 2023). The authors compared a diverse set of forecasting models, including ensemble learning algorithms (AdaBoost and LightGBM), deep neural architectures (GRU, Simple RNN, and LSTM). In addition, hybrid statistical and ML models such as ARIMA+MLP and ARIMA+LSTM were implemented to capture both linear temporal dependencies and nonlinear dynamics. As benchmarks, conventional models including ARIMA, Multilayer Perceptron (MLP), SVR, RW, and White Noise (WN) were used for performance comparison. Each model was evaluated using regression error metrics (MAPE and RMSE) to assess point forecast accuracy and investor oriented metrics (Sharpe Ratio, mean return, Mean Directional Accuracy(MDA)[21]) to evaluate trading profitability. The empirical findings revealed that LightGBM and AdaBoost consistently delivered superior performance for BTC and ETH across both sub-periods, achieving lower forecast errors and higher directional accuracy. For XRP and LTC, however, the best performing model varied by period, with relatively simpler models such as MLP and Simple RNN occasionally outperforming the more complex ensembles. The authors concluded that, in terms of point forecasts, less complex conventional models could sometimes be more accurate than highly complex models. When focusing on investor metrics, LightGBM showed superior performance and provided highly profitable trading strategies for BTC, ETH, and LTC. For the XRP, the Simple RNN model was recommended as the best forecasting model for investors.

Alnami et al.[109] presented an integrated framework combining close price forecasting and anomaly detection for BTC, ETH, BNB and LTC using daily OHLC price and volume data spanned 2015 until 2021. RF, Gradient Boosting(GB) and a feed-forward deep neural network(DL) are compared for the forecasting task using regression metrics including MSE, RMSE, MAE and R-squared. The RF model was selected for its robustness against overfitting and its capacity to effectively cap-

[21]Mean Directional Accuracy is a prediction accuracy measurement. It compares the forecast direction (upward or downward) to the actual realized direction. Available at: `https://en.wikipedia.org/wiki/Mean_directional_accuracy`

ture non-linear relationships within volatile cryptocurrency data. GB is used to detect complex patterns and achieve high accuracy in regression tasks by iteratively addressing the residual errors of preceding trees. DL is adopted to handle the complexities and non-linear trends present in the dataset. In addition to forecasting, the authors integrated a Z-score-based anomaly detection module to classify closing price observations as "normal" or "anomalous," aiming to identify extreme market fluctuations and improve the robustness of the predictive framework. Experimental results showed that both RF and GB achieved superior regression accuracy, whereas the DL model exhibited comparatively weaker performance. These results suggest that, on this dataset and setup, ensemble tree methods outperformed the neural network for price forecasting. The authors conclude that combining accurate forecasting with anomaly detection enhances the reliability of cryptocurrency modeling.

To identify an optimal predictive model for BTC prices, Kose et al. [110] conducted a comparative analysis of multiple ML models, including XGBoost, LightGBM, and SVR, alongside DL models such as LSTM, GRU, CNN, and Temporal Fusion Transformer (TFT). These approaches were benchmarked against conventional statistical models, including ARIMA, VARX, and GARCH. The study utilised daily BTC prices in USD along with macro financial indicators such as the Volatility Index (VIX), oil prices, the US Dollar Index (DXY), gold prices, the 5-year US Treasury yield (DGS5), and the Federal Funds Rate (DFF), covering the period from January 2012 to February 2024. The results indicated that the TFT model achieved the highest predictive accuracy, demonstrating superior capability in capturing complex nonlinear dependencies. Model performance was evaluated using MAE, MAPE, MSE, and R^2 metrics. Additionally, the authors identified gold prices as the most influential explanatory variable for BTC price movements, followed by the DXY.

Similarly, Tanrikulu and Pabuccu[111] conducted a comparative study of several ML and DL methods, including RF, KNN, XGBoost, SVM, Naive Bayes (NB), ANN and LSTM, for predicting the next day directional movement of BTC prices. The authors utilized two datasets: a continuous dataset an a trend dataset. The

continuous dataset comprised daily BTC OHLC prices, technical indicators such as RSI and Bollinger Bands, and sentiment related features including tweet counts and Google Search Trends. The trend dataset was constructed by transforming features from the continuous dataset into binary representations through a discretizations process. Both datasets covered a 7-year period from September 2014 to June 2021. For the continuous dataset, ANN and SVM achieved the best performance in terms of accuracy, F1-score, and area under the ROC curve (AUC). In contrast, LSTM demonstrated the highest predictive accuracy when applied to the trend dataset. The authors further observed that discretion features into trend-based representations improved models performance by simplifying pattern recognition. Additionally, experiments conducted using a reduced dataset spanning June 2020 to June 2021 revealed that larger datasets generally result in higher predictive accuracy, particularly for more complex models such as LSTM.

Hiskiawan et al.[112] also evaluated the predictive perforance of several ML and DL models for forecasting future BTC prices. The models examined include RF, XGBoost and LSTM, which were benchmarked against conventional statistical approaches: LR, ARIMA and FbProphet. Model performance was assessed using multiple evaluation metrics, namely MAE, RMSE, MAPE, R^2, and symmetric mean absolute percentage error (SMAPE). Daily BTC OHLC price and volume spanning January 2020 to December 2024 were collected, with a subset covering BTC daily close price from January 2024 to January 2025 used for training and evaluation to better capture short-term market volatility. A rolling-window forecasting framework was employed, in which the preceding 60 days of BTC prices were used to predict prices over the subsequent 30-day horizon. The results indicate that LR and ARIMA consistently achieved higher R^2 values and lower MAE and RMSE. In contrast, LSTM and XGBoost demonstrated superior capability in capturing short-term nonlinear dynamics and momentum. However, XGBoost and RF exhibited high errors and inability to capture temporal patterns inherent in the data.

Summary

This section reviews conventional ML methods applied to cryptocurrency price forecasting. Commonly employed are the three-based models include DT, RF, XGBoost and LightGBM, alongside other ML approaches such as LR, SVM, KNN and MLP. To evaluate the relative effectiveness of ML techniques, many studies[108–112] also incorporate DL models including LSTM, GRU, CNN and TFT, as well as statistical benchmarks such as ARIMA and GARCH. Although several studies[26, 27, 102] report that ML methods can achieve strong predictive performance, DL models such as LSTM[112] and TFT[110] are recognized for their robustness in capturing complex non-linear dynamics. Table 2.2 summaries the reviewed literature, highlighting the methodologies, datasets, and key findings of each study. To further examine the effectiveness of DL approaches in cryptocurrency forecasting, the following section reviews recent studies with a particular focus on recurrent architectures such as LSTM and Transformer-based frameworks.

Study	Method	Data	Data Source	Benchmark Model	Evaluation Metrics	Findings	Future Study
[102]	LR, SVM, DT	Daily closing BTC, ETH, DOGE, Bitcoin-Cash in USD from 31 Dec 2016 to 21 May 2022	Coin Metrics and Coin Codex	N/A	Accuracy	LR was found to be the most consistent and effective algorithm.	Incorporating social media sentiments.
[27]	BT	15-min OHLC BTC, ETH, ADA and BNB prices in USD and trading volumes anning November 2018 to November 2021; Technical indicators: RSI, MACD, Bollinger Bands	Binance exchange	DT, KNN, RF, NN	Accuracy, Percision, Recall, F1-score	The proposed BT model demonstrated leading or near leading accuracy across all buy signal types.	Incorporating more features such as sell signals to improve model's accuracy; Implementing reinforcement learning to capture dynamic market movements.
[26]	SVM with GMB portfolio optimization model	Daily OHLC prices and trading volume of BTC, ETH, XRP, ADA, DOGE, DOT and LTE from 1 January 2021 to 31 August 2021; Social media sentiment	cn.investing.com, Twitter, Google Trends	Minimum variance strategy, 1/N strategy, Black-Litterman approach	Sharpe ratios, Sortino ratios, Certainty Equivalent Returns	Incorporating social media sentiment data significantly improved the accuracy of SVM price movement predictions compared to using historical trading data alone.	Including more cryptocurrencies and developing asset evaluation and selection methods.

Continued on next page

Study	Method	Data	Data Source	Benchmark Model	Evaluation Metrics	Findings	Future Study
[108]	AdaBoost, LightGBM, RNN, LSTM, GRU, ARIMA-LSTM, ARIMA-MLP	Log transformed daily USD prices of BTC, ETH, XRP and LTC from 1 Apr 2016 to 31 Aug 2023	investing.com	RW, WN, ARIMA, MLP	MAPE, RMSE, Sharpe Ratio, Mean return, MDA	For investor focused metrics, LightGBM delivered the best performance and most profitable trading strategies for BTC, ETH, and LTC, while RNN performed best for XRP.	Incorporating high-frequency real-time data to better leverage deep learning methods; Construction of trading strategies for portfolio investors.
[109]	RF, GB, DL	Daily OHLCV of BTC, ETH, BNB and LTC spanned 2015 to 2021	Kaggle	LR, SVM, ARIMA, ARCH, GARCH, XGBoost, LightGBM, GRU	MSE, RMSE, MAE, R^2	Both RF and GB achieved superior regression accuracy, whereas the DL model exhibited comparatively weaker performance. These authors concluded that ensemble tree methods outperformed the neural network for price forecasting.	Incorporating additional data sources such as social media sentiment data.

Continued on next page

Study	Method	Data	Data Source	Benchmark Model	Evaluation Metrics	Findings	Future Study
[110]	XGBoost, LightGBM, SVR, LSTM, GRU, CNN, TFT	Daily BTC price and macro-financial indicators such as gold prices from Jan 2012 to Feb 2024	investing.com	ARIMA, VARX, GARCH	MAE, MSE, MAPE, R^2	TFT is identified as the most accurate predictive method; Gold prices is the most influential explanatory variable for BTC price movement;	Exploring alternative modeling techniques like hybrid models
[111]	RF, KNN, XGBoost, SVM, NB, ANN and LSTM	Daily BTC OHLC prices, 12 technical indicators such as RSI, tweet counts, Google search trends from Sep 2014 to Jun 2021	Yahoo Finance, bitinfocharts.com, Python technical analysis Library	KNN model's accuracy	accuracy, F1-score, AUC	For the continuous dataset, ANN and SVM achieved best performance; For the trend dataset, LSTM achieved best accuracy; Almost all the models' performance increased when applied to the trend dataset due to the simplicity of the data; Larger dataset results in higher predictive accuracy.	Utilizing deep learning, Transformers, and large language models (LLMs) to achieve higher accuracy; Incorporating higher frequency data.

Continued on next page

Study	Method	Data	Data Source	Benchmark Model	Evaluation Metrics	Findings	Future Study
[112]	RF, XGBoost, LSTM	Daily closing prices of BTC from Jan 2024 to Jan 2025	Yahoo Finance	ARIMA, LR, FbProphet	MAE, RMSE, R^2, MAPE, SMAPE	LR and ARIMA achieved the best performance, while LSTM and XGBoost were more effective at capturing short-term nonlinear dynamics and momentum. However, XGBoost and RF exhibited high errors.	Incorporating macroeconomic variables and sentiment factors; Applying the study to more volatile assets such as ETH; Developing ensemble methods to capture diverse strengths of different models.

Table 2.2: Summary of literature on cryptocurrency TSF using machine learning methods

2.3.3 Deep Neural Networks

Previous studies[79, 82, 98–100] have consistently shown that DL models, especially LSTM networks, excel at capturing non-linear and temporal dependencies in cryptocurrency price dynamics, outperforming conventional statistical models. In addition, more researchers have explored a range of alternative DL architectures tailored to cryptocurrency forecasting. For example, Seabe et al. [113] conducted a comparative analysis of LSTM, GRU, and Bi-directional LSTM (Bi-LSTM) models for forecasting the prices of BTC, ETH and LTC. Both LSTM and GRU are designed to to alleviate the vanishing gradient problem in traditional RNNs[114]. Unlike LSTM, which uses multiple gates and an internal memory cell to regulate the data flow, GRU employs only a single update date and reset gate to decide which information to retain and discard[113]. This architecture makes GRU computationally more efficient while still achieving comparable performance. The Bi-LSTM extends the conventional LSTM by processing input sequences in both forward and backward directions, enabling the model to leverage contextual information from both past and future temporal directions[115]. Their study utilized daily closing prices in USD from January 1, 2018 to December 31, 2021 for training, and data from January 1, 2022 to January 1, 2023 for testing. Their findings presents that the Bi-LSTM model achieved superior performance, with the lowest RMSE and MAPE across all cryptocurrencies, followed by the GRU model. Similarly, Nair et al.[116] compared LSTM, RNN GRU, Conv1D and Bi-LSTM models for predicting Bitcoin prices. Their dataset comprised daily open, high, low, close, adjusted close prices, and trading volume from September 17, 2014, to February 1, 2022. Their results revealed that the LSTM model provided the most accurate forecasts, followed by Bi-LSTM and GRU models.

Kiram et al.[28] conducted a comparative study between the Temporal Convolutional Network (TCN) and the GRU to forecast ETH daily closing prices. The dataset is obtained from Yahoo Finance, covering a 4-year period. The TCN, a convolutional model employing causal and dilated convolutions, was utilized to cap-

ture long-range temporal dependencies. In contrast, the GRU is employed to model both short and medium-term dependencies. The models were evaluated using metrics including MAE, MAPE, and MSE. Experimental results demonstrated that TCN achieved lower MAE and MAPE values, indicating higher average predictive accuracy and and a stronger ability to capture overall market patterns. Conversely, GRU obtained a slightly lower MSE, suggesting superior robustness in handling large deviations and and abrupt price swings during high volatility periods.

Omole and Enke[117] explored the predictive power of CNN-LSTM, Long- and Short-Term Time Series Network(LSNet) and TCN on predicting future BTC price direction. The dataset comprises BTC prices and on-chain metrics[22] covering the period from 6 February 2013 to 18 February 2023, spanning 3665 daily observations. The authors highlight that on-chain data better reflect the pricing dynamics and helping in providing better forecasting ability. The CNN–LSTM model employed in the study is a hybrid architecture, where the CNN extracts local temporal patterns, while the LSTM layers learn long-term sequential dependencies in the data. The LSTNet model is designed with CNN and RNN to extract short-term and long-term dependencies among multiple variables[119]. All deep learning models were benchmarked against the ARIMA model using directional accuracy as the evaluation metric. The experimental results revealed that the CNN–LSTM model consistently outperformed the other configurations, demonstrating superior capability in identifying complex temporal relationships in BTC movement forecasting.

Bourday et al.[120] conducted comparative analysis of hybrid Transformer-based deep learning architectures to predict Bitcoin prices. Four hybrid models are explored with daily BTC prices from 01 January 2016 to 31 December 2023: Transformer+XGBoost, Transformer+ANN, Transformer+LSTM, and Transformer+SVR. Through extensive experiments using MAE, RMSE, and MSE as performance metrics the authors found that the Transformer+XGBoost configuration achieved the

[22]On-chain metrics are data points derived from information generated by the blockchain network, such as the size of the blockchain, the number of blocks attached to it, or the difficulty of mining blocks[118].

highest forecasting accuracy with MAE of 0.011 and RMSE of 0.018. The study concludes that combining the feature extraction capability of Transformers with the robust ensemble learning of XGBoost effectively captures both nonlinear and long-range temporal dependencies in cryptocurrency data.

Mazinani et al.[29] conducted an extensive comparative analysis of deep learning algorithms for forecasting the prices of BTC, ETH and XRP. The dataset consists of 4-hour interval OHLC price data, spanning August 2020 to November 2022 for ETH and XRP, and November 2020 to March 2023 for BTC. The study evaluated a diverse set of DL architectures, including MLP, SimpleRNN, LSTM, BiLSTM, GRU, Bidirectional-GRU (Bi-GRU), CNN, Transformer, and two hybrid models (CNN–RNN and CNN–GRU). Performance was assessed using standard regression metrics (RMSE, MAE, MAPE, and R^2) and computational complexity metrics such as Multiply-and-Accumulate (MAC) operations, parameter count, and flash memory size. The results demonstrate that for short-term forecasting (3-step ahead), the Transformer model achieved the lowest RMSE and MAE, and the highest R^2 across all cryptocurrencies. For long-term horizons (up to 20 steps ahead), the Transformer maintained its dominance, yielding the most accurate predictions for BTC and ETH. However, the study also highlighted the computational cost of the Transformer, which exhibited the highest MAC operations and parameter count. In contrast, CNN–RNN and CNN–GRU models offered the best trade-off between prediction accuracy and computational efficiency. The authors concluded that, while Transformer architectures provide the highest predictive accuracy, hybrid CNN–RNN and CNN–GRU models are more suitable for resource-constrained environments where computational efficiency is a key consideration.

Both studies [29, 117] have consistently shown the Transformer model outperforms conventional DL architectures by effectively capturing long-range temporal dependencies in cryptocurrency price movements. The next section further explores more Transformer-based approaches and their specialized adaptations in this domain.

Transformer Applications in Cryptocurrency Price Forecasting

As has been noted earlier, Transformer has been adopted across domains such as natural language processing (NLP)[121], image processing[122] and TSF[40]. Despite their relatively limited adoption in cryptocurrency forecasting, emerging research suggests that Transformer-based models exhibit strong potential in this domain.

Singh and Bhat[123] conducted a comparative study evaluating the performance of Transformer against benchmark models including LSTM, MLP, and ANN models for forecasting ETH prices. The study utilized a dataset comprising historical closing prices and trading volumes of ETH, ADA and DOT, social media sentiment from Twitter and Reddit, and on-chain indicators such as block size from October 2021 to January 2024. The models were evaluated under three data configurations: 1. ETH closing price data only; 2. a combination of ETH price, trading volume, and sentiment data; 3. an integrated dataset incorporating ETH price, trading volume, sentiment data and prices of correlated cryptocurrencies including ADA and DOT. The experimental results indicated that the Transformer model achieved the lowest MSE, while LSTM demonstrated superior performance in terms of RMSE and MAPE with the ETH price only dataset. However, model performance declined when additional data sources were introduced in the second and third configurations. The authors attributed this decline to the limited predictive contribution of sentiment data. They suggested social media sentiment exhibits relatively weak explanatory power for ETH price movements.

Yildirim et al.[124] proposed the *Optuna-Based Optimized Transformer*(OBOT) for BTC price forecasting. They employed an Optuna-based[23], hyperparameter optimization process before applying the Transformer model on the prediction task. The OBOT model was benchmarked against a range of approaches, including statical models (ARIMA and ARIMAX), ML methods (XGBoost, ANN, RNN), and DL methods (LSTM, Bi-LSTM and TCN) for forecasting BTC prices over 6- and 12-step forecasting horizons. The study utilized a dataset comprising BTC closing

[23]Optuna is an open source Python library designed for hyperparameter optimization[125]

prices sampled at 4-hour intervals from early 2020 to early 2024. Multiple train-test splits (0.5-0.5, 0.7-0.3 and 0.8-0.2) were evaluated to determine the optimal experimental configuration. The results indicated that the proposed method consistently outperformed benchmark models across all train-test splits in terms of MAE and RMSE. Optimal performance was achieved using a 0.7-0.3 train-test split with a 6-step forecasting horizon. The authors further reported that ARIMA and ARIMAX demonstrated limited effectiveness, while Bi-LSTM and TCN achieved competitive performance in certain scenarios. Overall, models exhibited stronger predictive performance for shorter forecasting horizons (6-step) compared with longer horizons (12-step).

To leverage the Transformer's strength in capturing long range dependencies and the GRU's capability to model short-term sequential patterns, Mahdi et al. [126] proposed a hybrid Transformer-GRU model. This model was evaluated against several benchmark models, including Radial Basis Function Network (RBFN), General Regression Neural Network (GRNN), Bi-LSTM, and Bi-GRU. RBFN and GRNN are non-sequential, feedforward neural network models often applied in financial time series analysis and regression tasks. They are included as benchmarks to demonstrate the advantages of deep and sequential architectures over basic feedforward networks for TSF. Bi-LSTM and Bi-GRU are adopted due to their ability in handling long-term and bidirectional dependencies in sequential data. The evaluation utilized a dataset comprising daily closing prices in USD and trading volumes for Bitcoin from 17 September 2014, to 28 February 2025, and Ethereum from 9 November 2017 to 28 February 2025. Additionally, the authors incorporated a Fear and Greed Index (FGI) derived from social media sentiment and Google Trends data related to BTC and ETH. Experimental results demonstrate that the hybrid Transformer-GRU model outperformed the benchmark models, achieving superior performance in terms of MSE, MAE, RMSE, and MAPE.

Summary

Table 2.3 summarize the aforementioned studies that employ DL models including Transformer-based models for cryptocurrency price forecasting. The reviewed literature indicates that LSTM is one of the most frequently utilized DL architectures in this domain. This observation is consistent with the survey conducted by John et al.[46], which analysed 93 high quality studies published between 2014 and 2024 and identified LSTM as the most commonly adopted method in cryptocurrency forecasting research.

Although the above studies shows that academic research has begun to explore the use of Transformer architectures in cryptocurrency forecasting[120, 124, 126], their adoption remains relatively limited compared to more established DL models such as LSTM. As noted by Zhang el al.[127] and Kim et al.[128], the Transformer represents a promising successor to LSTM for TSF due to its superior ability to capture long-term dependencies (although Transformer's efficiency in long-term TSF tasks is a subject of debate [129]). However, despite this potential, the use of Transformer-based models in cryptocurrency forecasting remains underexplored within the current academic literature. One possible reason for this limited adoption is the inherent limitations of the Transformer architecture. Section2.4 examines these limitations in detail and provides a comprehensive review of SOTA approaches that employ various techniques to address them.

Study	Method	Data	Data Source	Benchmark Model	Evaluation Metrics	Findings	Future Study
[113]	LSTM, GRU, Bi-LSTM	Daily closing price in USD of BTC, ETH, and LTC span 1 Jan 2018 to 31 Dec 2023	Yahoo Finance	N/A	RMSE, MAPE	The Bi-LSTM model achieved the lowest RMSE and MAPE across all cryptocurrencies, indicating superior predictive performance, followed by the GRU model.	Exploring external factors such as social media sentiment data and trading volumes.
[116]	RNN, LSTM, GRU, Bi-LSTM, Conv1D	Daily BTC OHLC prices, adjusted close price and volume from 17 Sep to 1 Feb 2022.	Kaggle	N/A	MSE, MAE, RMSE, R^2	LSTM model exhibited superior performance than all other models.	Applying more complex DL methods or hybrid DL methods; Incorporating sentiment data such as tweets; Expanding the research to more cryptocurrencies.
[28]	TCN, GRU	4 years of daily ETH closing price	Yahoo Finance	N/A	MAE, MSE, MAPE	TCN model excelled in average accuracy in terms of MAE and MAPE while GRU performed better in terms of MSE, suggesting GRU is more robust in reducing the impact of large error.	Developing hybrid architectures that combines TCN and GRU; Extending the predictin horizon beyond daily forecasts.

Continued on next page

Study	Method	Data	Data Source	Benchmark Model	Evaluation Metrics	Findings	Future Study
[117]	CNN-LSTM, LSNet, TCN	BTC prices and on-chain metrics covering the period from 6 Feb 2013 to 18 Feb 2023	Glassnode	ARIMA	Accuracy, percision, recall, F1-score, Market exposure	The CNN-LSTM model achieved the highest overall predictive accuracy, whereas the ARIMA model exhibited comparatively weaker performance, suggesting that its relatively simple structure is insufficient to capture the complex dynamics of Bitcoin price movements.	Incorporating sentiment data and technical indicators.
[120]	Hybrid methods with Transformer and XGBoost, ANN, LSTM, SVR	Daily BTC close prices in USD, blockchain technical metrics such as block size, sentiment data such as tweets and Google Trends from 1 Jan 2016 to 31 Dec 2023	bitinfocharts.com	ARIMA, LSTM, RFR	MAE, MSE, RMSE	The Transformer+XGBoost model outperformed the other hybrid models for 7-, 30- and 90-days forecasting; Combining the feature extraction capability of Transformers with XGBoost can capture both non-linear and long-range temporal dependencies in cryptocurrency data.	Incorporating wider range of predictive models such as Graph Neural Networks (GNN); Exploring additional features beyond the ones used in this study.

Continued on next page

Study	Method	Data	Data Source	Benchmark Model	Evaluation Metrics	Findings	Future Study
[29]	MLP, RNN, LSTM BiLSTM, GRU, Bi-GRU, CNN, Transformer, CNN-RNN, CNN-GRU	OHLC prices of ETH and XRP data cover Aug 2020 to Nov 2022, while BTC data covers Nov 2020 to Mar 2023	12 cryptocurrency trading platforms	ARIMA, ARMA	RMSE, MAE, MAPE, R^2, MAC	The Transformer model achieved the lowest RMSE and MAE and the highest R^2 across all cryptocurrencies for both short- and long-term forecasting. However, due to its high computational cost, hybrid models such as CNN–RNN and CNN–GRU may be more suitable when efficiency is a key concern.	Incorporating sentiment analysis based approaches; Considering macroeconomic factors.
[123]	Transformer	Daily closing prices and volume of ETH, DOT and ADA; Social media data; On-chain data from Oct 2021 to Jan 2024	Yahoo Finance, Twitter, Reddit, CoinMarketCap	LSTM, MLP, ANN	RMSE, MSE, MAPE	Transformer model achieved the best performance with ETH price data only, however, it performance worse than LSTM on RMSE and MAPE; Adding sentiments and cross-correlation data increased the error rates.	Expanding sentiment data to incorporate news articles and full content of Reddit posts and comments; Exploring methods such as Autoformer and TimeGPT-1

Continued on next page

Study	Method	Data	Data Source	Benchmark Model	Evaluation Metrics	Findings	Future Study
[124]	OBOT	BTC prices in USD with 4-hour intervals span the beginning of 2020 to the beginning of 2024	Binance	ARIMAX, ARIMA, XGBoost, ANN, RNN, LSTM, Bi-LSTM, TCN	MAE, RMSE	OBOT consistently outperformed other models across all train–test splits, with optimal performance achieved using a 0.7–0.3 split and a six-step forecasting horizon; ARIMA and ARIMAX failed to capture trend and seasonal patterns, while deep learning models such as Bi-LSTM and TCN performed well in specific cases but were generally less consistent; Shorter forecasting horizons (6 steps) yielded better performance than longer horizons (12 steps).	Expanding the experiments to different cryptocurrencies; Exploring hybrid models; Incorporating macroeconomic indicators and social media analysis.

Continued on next page

Study	Method	Data	Data Source	Benchmark Model	Evaluation Metrics	Findings	Future Study
[126]	Transformer-GRU	Daily closing prices in USD and trading volumes for BTC (17 Sep 2014 - 28 Feb 2025) and ETH (9 Nov 2017 - 28 Feb 2025); FGI related to BTC and ETH	coinmarketcap.com; Alternative.me; Twitter; Google Trends	RBFN, GRNN, Bi-LSTM, Bi-GRU	MAE, MSE, MAPE, RMSE	The proposed method consistently outperformed all other models;	Applying the proposed methods to more digital assets and stock markets; Exploring other hybrid architectures such as Transformer+LSTM and Transformer+BiLSTM; Incorporating macroeconomic data or on-chain indicators.

Table 2.3: Summary of literature on cryptocurrency TSF using deep learning methods

2.3.4 Synthesis of Cryptocurrency Forecasting Methods: Data, Strengths and Limitations, Challenges and Research Gaps

The preceding sections reviewed statistical, conventional ML, DL and Transformer-based approaches applied to cryptocurrency forecasting. This section consolidates the current status of SOTA and provides a comprehensive review of the current status of the cryptocurrency forecasting field from a list of aspects: data, strength and limitations of existing forecasting approaches, challenges and research gaps.

Data

The previous reviewed articles indicate that cryptocurrency forecasting research lacks standardized benchmark datasets. Current studies employ heterogeneous data sources, including exchange-specific price data[27, 90, 102], on-chain indicators[123], social media sentiment[26, 111], and macroeconomic indicators[27, 110]. These datasets are typically collected from multiple independent sources, such as cryptocurrency exchanges, blockchain networks and social media platforms. Furthermore, most studies focus on one or a limited number of cryptocurrencies while employing different historical time spans and data granularity. This heterogeneity presents significant challenges for the consistent comparison of forecasting models across studies. Variants in dataset and evaluation metrics often lead to inconsistent model performance rankings. For instance, both studies [96, 100] evaluated the effectiveness of ARIMA model for BTC forecasting. However, [99] employs daily BTC price data spanning January 2020 to May 2022, whereas [100] utilizes hourly BTC OHLC data covering January 2017 to August 2019. Such differences in data type, historical time coverage, and granularity introduce substantial variability in experimental outcomes.

This observation is consistent with prior survey studies[13, 46], which highlight substantial heterogeneity in cryptocurrency forecasting research, including variations in data sizes, data source consistency, cryptocurrency selection, forecasting

horizons and metrics. Such methodological diversity introduces challenges for reproducibility and fair model comparison. To mitigate these limitations, the authors recommended the adoption of standardized frameworks that transform dynamic real-world market data into controlled and reproducible experimental environments, along side promoting open-source code and dataset to facilitate validation and further investigation.

Following these recommendations, this study adopts the TSF framework[24] originally proposed by Zhou et al.[38] and subsequently adopted by a substantial number of TSF studies[32, 36, 40, 41, 44]. This framework provides a standardized pipeline for data processing, model configuration, and performance evaluation. Furthermore, to ensure fair and widely comparable model evaluation, this study employs commonly used performance metrics, including MSE, MAE, RMSE, and MAPE, which are widely adopted in both cryptocurrency forecasting[13, 46] and broader TSF literature[36, 38, 40, 44]. In addition, to enhance reproducibility, the implementation and experimental pipeline developed in this research are open-source for replication and evaluation.

Strength and limitation of existing forecasting approaches

The existing studies[77, 78] have demonstrated that statistical models exhibit strong performance in short-term (1-, 5-, 10-day) forecasting tasks, however, their forecasting accuracy deteriorates as the prediction horizon increases. Furthermore, the statistical methods are relying on the assumption of a linear data generation process, which is often unrealistic for highly volatile cryptocurrency data[29]. Consequently, these models frequently struggle to adapt to abrupt market shifts and are limited in their ability to capture the non-linear relationships inherent in cryptocurrency price movements[110].

Unlike traditional statistical models, ML models are capable of extracting complex non-linear patterns from large datasets[120]. These models can process data

[24] `https://github.com/zhouhaoyi/Informer2020`

without requiring strong prior assumptions about the underlying data structure and can adapt to evolving market dynamics without explicit reprogramming[108]. Several studies[80, 90, 96] demonstrate that ML models generally outperform conventional economic methods in cryptocurrency price forecasting. Nevertheless, ML methods remain highly susceptible to overfitting, particularly when trained on limited or noisy datasets[46, 120]. Moreover, their effectiveness often relies on extensive data preprocessing and manual feature engineering, which increases implementation complexity and may limit scalability across different market environments[46].

DL methods further extend the modeling capability of ML by identifying hidden patterns and non-linear dependencies[46]. Recurrent neural network architectures, such as LSTM, have demonstrated strong performance in financial forecasting due to their ability to model temporal dependencies across sequential observations[13]. Additionally, DL models generally exhibit greater robustness to noise and do not require strict assumptions regarding data stationarity or distributional properties[116, 120]. However, conventional DL architectures, particularly recurrent-based models, suffer from several limitations. Their sequential processing structure restricts parallel computation, resulting in high computational costs and longer training times[126]. Furthermore, these models often struggle to capture very long-range temporal dependencies due to gradient vanishing problem[108, 126].

To address these limitations, recent research[46] has increasingly focused on Transformer-based architectures, which leverage self-attention mechanisms to model temporal dependencies more efficiently. Unlike recurrent models, Transformers process entire sequences simultaneously, enabling improved parallelization and enhanced capability in capturing long-range dependencies[29, 126]. In addition, the attention mechanism allows the model to dynamically weigh the importance of different tokens in the input sequence, prioritizing the most informative market signals[46]. This architectural design has demonstrated superior performance in long sequence time-series forecasting tasks[40, 44] and has become a prominent direction in modern financial forecasting research.

Challenges and research gaps

Despite the rapid growth of cryptocurrency forecasting research, several challenges and research gaps remain unresolved. First, although the conventional financial markets such as FX markets are considered highly volatile[21], cryptocurrency markets exhibit significantly greater price volatility[22], which makes accurate forecasting particularly challenging. Moreover, cryptocurrency price movements exhibit strong interdependency structures, where relationships among digital assets evolve dynamically over time[130]. These interdependencies indicate that cryptocurrency price behavior is influenced by broader market interactions.

Although widely studied cryptocurrencies such as BTC and ETH dominate existing research[13], relatively limited attention has been given to other cryptocurrencies. For instance, while prior studies suggest that ADA price movements are strongly associated with broader cryptocurrency market dynamics[11], less than 2% of recent forecasting studies explicitly include ADA in their empirical analysis[13]. Existing research further demonstrates that cryptocurrency markets exhibit dynamic interdependency structures, where significant price movements in major cryptocurrencies often propagate across other digital assets[113, 126]. This interconnected behavior suggests that incorporating historical price information from multiple cryptocurrencies may improve forecasting accuracy by capturing cross-market information spillovers. However, the majority of current studies focus predominantly on BTC and ETH, with comparatively few investigations examining multivariate forecasting frameworks that incorporate other cryptocurrencies.

Second, while Transformer-based architectures have demonstrated strong performance in TSF[40, 41, 44] and have shown promising results in cryptocurrency price prediction[123, 124, 126], the application and systematic evaluation of SOTA Transformer variants within cryptocurrency forecasting remain limited[13, 131]. Consequently, the generalizability and robustness of these advanced architectures within cryptocurrency markets remain insufficiently explored.

Third, the literature lacks standardized benchmark datasets, evaluation met-

rics and a numerical definition of short- and long-term forecasting. Current studies employ heterogeneous datasets and inconsistent evaluation protocols, which complicates cross-study comparison. Due to the variance of the datasets, the definition of short-term and long-term forecasting horizons remain inconsistent across studies. Although Kong et al.[132] suggest that short-term forecasting typically ranges from hours to weeks, while long-term forecasting spans from months to years, these definitions often vary depending on the dataset characteristics. For instance, when using datasets containing 4-hour interval OHLC price data, [29] defines 3-step ahead forecasting(12 hours) as short-term prediction and categorizes 10-, 15-, and 20-step ahead predictions as long-term forecasting(40-, 60-, and 80-hour). In contrast, when daily price data are employed, [133] defines 1-day ahead prediction as short-term forecasting and classifies 30- and 90-day ahead predictions as long-term forecasting. These variations demonstrate that forecasting horizon definitions are strongly influenced by data sampling frequency and experimental design choices, which limits the comparability of forecasting models across studies.

To address these identified gaps, this study proposes a multivariate cryptocurrency forecasting framework that incorporates historical price information from multiple dominant cryptocurrencies to forecast target asset prices, with a particular focus on ADA as a representative altcoin. In addition, this study systematically evaluates multiple SOTA Transformer-based architectures within a unified experimental pipeline to provide a consistent and reproducible benchmarking environment. To enhance comparability, standardized evaluation metrics and multiple forecasting horizons, explicitly defined using fixed numerical prediction steps, are adopted throughout the experimental design. Furthermore, this study utilizes a structured data preprocessing and evaluation framework to promote transparency and reproducibility, and the implementation is made publicly available to facilitate further validation and extension by the research community.

2.4 Recent Transformer-based architectures for TSF

2.4.1 Limitations of canonical Transformers in TSF

Zhang et al.[127] noted that the canonical Transformer model requires quadratic time and memory with respect to the input sequence length, which is a key reason it has not been as widely adopted as recurrent architectures such as LSTM in TSF. Similarly, in the past, studies like[25, 32, 36–39] focus on improving Transformer-based TSF models have highlighted that the canonical Transformer architecture encounters substantial challenges when applied to financial sequences due to its high computational cost and limited ability to capture temporal dependencies in time series data.

These limitations stem from the Transformer's self-attention mechanism. Specifically, the dot product attention operation computes pairwise interactions between all positions in the input sequence, resulting in a computational cost that scales quadratically with respect to sequence length[134]. Moreover, this point-wise dot product self-attention in canonical Transformer architecture is insensitive to local context, which can make the model prone to anomalies in time series[25]. Consequently, the vanilla[25] Transformer struggles to learn intrinsic temporal information, especially in long sequences[38]. Zhou et al.[32] further illustrate these limitations by comparing the prediction outputs of the canonical Transformer with the ground truth data from a real world electricity-transformer temperature dataset[38]. Their analysis highlights the Transformer's difficulty in capturing long-term temporal patterns in time series data,

As a mitigation, a variety of Transformer-based architectures have been proposed, such as LogSparse Transformer[25], Informer[38], Autoformer[36], FEDformer[32], Pyraformer[135], Crossformer[39], PatchTST[40], MSGNet[41], TimeXer[42], iTransformer[44] and MSPatch [45]. These models introduce architectural innovations aimed at reducing computational complexity and enhancing the ability to capture

[25]In this thesis, the term "vanilla" refers to the original or canonical version of a model architecture, without task-specific modifications or enhancements.

temporal dependencies. The following sections presents SOTA Transformer-based TSF models along with the techniques they employ to address the limitations of the canonical Transformer architecture.

2.4.2 Computational complexity reduction

To tackle the issue of quadratic computational complexity inherent in the vanilla Transformer, the SOTA Transformer-based TSF architectures have introduced structural modifications. LogSparse[25] incorporates sparsity into the attention mechanism by allowing each position to attend to only a strategically selected subset of previous positions, rather than all preceding data points. This sparsity has been shown to significantly reduces the computational burden, reducing the overall complexity from $\mathcal{O}(L^2)$ to $\mathcal{O}(L(\log L)^2)$. Informer[38] further reduces the computational complexity to $\mathcal{O}(L(logL))$ by introducing so-called *ProbSparse attention*. Instead of computing query-key pairs, *ProSparse* selects a limited number of most important queries to participate in the attention computation.

Autoformer[36] also reduces the computational complexity to $\mathcal{O}(L(logL))$ by introducing two key innovations: a series decomposition block and an Auto-Correlation mechanism. The decomposition module separates the input time series into trend and seasonal components. The Auto-Correlation mechanism then aggregates information across similar sub-series. Leveraging the natural sparsity in time series and operating at the sub-series level, this mechanism enhances both computational efficiency and temporal pattern extraction.

Fedformer[32] and Pyraformer[135] further achieve reduction in computational complexity with $\mathcal{O}(L)$. Like Autoformer, Fedformer decomposes the time series into seasonal and trend components. In addition, it incorporates a frequency domain strategy by randomly selecting a fixed number of Fourier components, allowing the model to focus on dominant periodic patterns. This combination enables Fedformer to maintain high predictive accuracy while operating with significantly reduced computational overhead. Pyraformer, on the other hand, introduces the Pyramidal

Attention Module (PAM), which constructs a pyramidal graph where each node corresponds to a different temporal resolution. Attention scores are computed only among connected nodes, capturing both inter-scale (across different resolutions) and intra-scale (within the same resolution) dependencies. This hierarchical attention structure allows Pyraformer to efficiently model long-range temporal patterns while preserving fine-grained local information.

2.4.3 Patching - Enhancing temporal information modeling

The application of canonical attention mechanisms to time series data often involves computing dot products at individual time steps. However, comparing with words in a sentence, a single point in a time series typically lacks semantic meaning. Thus, capturing local contextual patterns becomes essential for accurately modeling dependencies within temporal sequences[40].

Motivated by the success of the patching strategy in the Vision Transformer (ViT)[35], which segments images into fixed size patches and treats them as tokens, recent research has adapted patching for time series data. Several Transformer-based architectures[39–43] divide the input time series into patches and treat each patch as a token. Figure 2.2 illustrates the patching process which groups individual tokens (i.e. $t1$ to $t12$) to patches ($p1$ to $p4$). The patching-based Transformers not only enables the model to better capture temporal patterns, but also reduces the attention computation complexity from $\mathcal{O}(L^2)$ to $\mathcal{O}((L/P)^2)$, where P is the patch size. iTransformer[44] is an extreme case of the patching strategy. Rather than dividing the time series into smaller temporal segments, it treats each entire variable as a distinct token, referred to as a *variable token*, thereby enabling the model to capture global dependencies across variables more effectively. The attention computational complexity is $\mathcal{O}(M)^2$ where M is the number of variables.

These models have been extensively evaluated on real world benchmark datasets. The results show that they outperform earlier architectures such as Autoformer and achieve SOTA forecasting accuracy. Notably, the patching strategy has also been

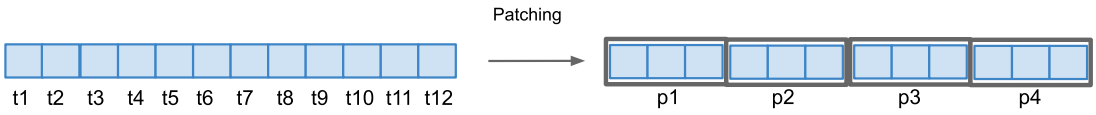


Figure 2.2: The Process of Patching

successfully extended beyond Transformer-based models. For example, MLP-Mixer, a model built on MLP, incorporates patching to achieve comparable improvements in capturing temporal patterns[136].

Table 2.4 summarizes the previously-mentioned Transformer-based models, highlighting their key techniques, attention computation complexity, and the datasets used for performance evaluation. Detailed descriptions of each dataset are provided in Table 2.5. While these SOTA models have been extensively validated across a variety of real-world datasets, it is worth noting that relatively few have been evaluated on cryptocurrency price data.

Patch scale selection

The Transformer-based architectures leveraging patching techniques differ in their strategies for selecting patch sizes, employing either a fixed patching scale or a set of multiple scales. Models such as CrossFormer[39], PatchTST[40], and InjectTST[43] adopt a single exogenous patch size, typically selected based on domain expertise and prior knowledge of the dataset. In contrast, iTransformer[44] redefines the notion of a patch by treating each entire variate as a single token. This design enables the model to simultaneously capture both inter-variate and temporal dependencies.

To exploit multiscale temporal patterns, models like CrossFormer[39], MLP-Mixer[136], MSPatch[45], and MSGNet[41] adopt multiple patching scales. Among them, MSGNet introduces an adaptive mechanism using the Fast Fourier Transform (FFT) to select patch sizes based on the dominant frequencies present in the data, allowing it to better align with the underlying periodicity of different time series.

Model	Method	Attention Complexity	Dataset (see Table 2.5)
LogSparse[25]	Sparse attention	$\mathcal{O}(L(LogL)^2)$	electricity-f(fine), traffic-f(fine), traffic-c(coarse), solar, wind
Informer[38]	Sparse attention with ProbSpare	$\mathcal{O}(L(logL))$	ETT, ECL, Weather
Autoformer[36]	Sparse attention with auto-correlation	$\mathcal{O}(L(logL))$	ETT, Electricity, Exchange, Traffic, Weather2, Illness
Fedformer[32]	Frequency enhanced attention	$\mathcal{O}(L)$	ETT, Electricity, Exchange, Weather2, Illness
Pyraformer[135]	Pyramidal attention module	$\mathcal{O}(L)$	Wind2, App Flow, Electricity2, ETT
PatchTST[40]	Patching	$\mathcal{O}(L/P)^2$	Weather2, Traffic, Electricity ETT
Crossformer[39]	Two-Stage Attention(TSA) with patching	$\mathcal{O}(L/P)^2$	ETT, Weather, ECL, Illness, Traffic
MSGNet[41]	Patching, Graph convolution	$\mathcal{O}(L/P)^2$	Flight, Weather, ETT, Electricity, Exchange
TimeXer[42]	Patching, Cross attention on variables	$\mathcal{O}(L/P)^2$	ECL, Weather, ETT, Traffic
iTransformer[44]	Channel-depended attention and Patching. Patch scale=sequence length L. M is number of variables	$\mathcal{O}(M)^2$	ETT, Electricity, Exchange, Traffic, Weather2, Solar, PEMS
MSPatch[45]	Multi-scale patching, Patch linear attention, Hybrid patch convolution	$\mathcal{O}(L/P)^2$	ETT, Weather, Solar, Electricity, Traffic

Table 2.4: Transformer-based models. This table is organized chronologically-based on the publication date of each model and ranked by the computational complexity of their attention mechanism.

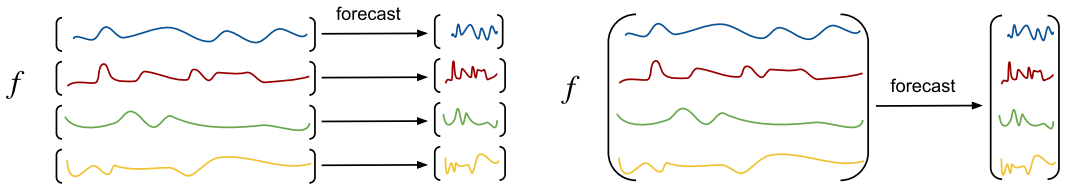
2.4.4 Channel-independence (CI) vs Channel-dependence (CD)

In the TSF literature[40, 42, 54, 139], there is an ongoing discussion regarding CI vs CD paradigms. Both CI and CD represents distinct methodologies for processing input sequences. As Figure 2.3 illustrates, under the CI paradigm, a model processes channels separately. The weights are shared across all the channels. This contrasts with CD approaches, where all channels are modeled jointly, the across channel relationships are captured. Both methods have inherent strengths and weaknesses, yet their comparative application to cryptocurrencies remains an under-explored area.

Both PatchTST[40] and TimeXer [42] introduce CI strategy into its Transformer-based architecture to get benefits including improved generalization, faster conver-

Data	Description	Reference
Synthetic dataset	Data generated using a set of sinusoidal signals.	[25]
Electricity-f (fine)	15 min electricity usage data from 370 customers.	[25]
Electricity-c (coarse)	As Electricity-c but for hourly intervals.	[25]
Traffic-f (fine)	20 min occupancy data for 963 freeways in SF, CA.	[25]
Traffic-c (coarse)	As Traffic-f (fine) but for hourly intervals.	[25]
Solar	Hourly solar power data from 137 PV plants in AL from 01/2006-08/2006.	[25]
Wind	Daily energy potential estimates from 28 countries 1986-2015, data as a percent of plant max output.	[25]
M4	Consists of 414 hourly time series from the M4 competition.	[137]
ETT (Electricity-transformer Temperature)	2 years of data from two Chinese counties, comprises: ETTh1&ETTh2(1 hour frequency). Data points comprise oil temperature and 6 associated power load features.	[38]
ECL (Electricity Consuming Load)	Derived from electricity-c and electricity-f datasets (above, due to missing values in these) ECL has 2 years hourly electricity consumption data.	[38]
Weather	Contains hourly observations of local climate data for approx 1600 US locations from 2010-2013.	[38]
Electricity	Hourly consumption data from 321 customers 2012-2014.	[36]
Exchange	Daily data for 8 countries (Aus, NZ, UK, Canada, Switzerland, China, Japan, Singapore) 1990-2016.	[119]
Traffic	Hourly road occupancy freeway sensor data from SF Bay Area (from Caltrans).	[36]
Weather2	10 min meteorological data e.g. air temperature, humidity etc from 2020	[36]
Illness	Weekly CDC data for influenza-like illness (ILI) for 2002-2021.	[36]
Wind2	Hourly wind data from 1986-2015 as a percentage of a power plant's max output	[135]
App Flow	4 months of hourly max traffic flow for 128 systems across 16 data centers.	[135]
Electricity2	15 min electricity consumption data.	[135]
Flight	OpenSky data, granularity not explicitly stated.	[41]
PEMS	Consists of 4 sub-datasets recorded at 5 minute intervals with 358, 307, 883 and 170 time series.	[138]

Table 2.5: Dataset description. This table presents detailed descriptions of the datasets utilized by the Transformer-based models.



(a) CI processing (b) CD Processing

Figure 2.3: Comparison of CI and CD processing for training multivariate TSF(reproduced from Han et al.[54]). In CI processing, each channel is learned separately using a shared set of model parameters across all channels. In contrast, CD processing learns all channels jointly, explicitly modeling inter-channel dependencies.

gence, and reduced overfitting. As observed by Han et al.[139], the CI paradigm provides several important advantages, outlined below, which have led to its adoption in many SOTA TSF models.

1. **Improved generalization**[40, 139]: Nie et al.[40] emphasized that CI enables Transformer-based models to generate similar attention maps for unrelated but similar time series. This behavior allows the model to better generalize across diverse datasets. Han et al.[139] also point out that CI can preserve long-term feature information by preventing its distortion through entangled inter-channel correlations.

2. **Faster convergence**: Empirical results from Nie et al. [40] show that models trained under the CI paradigm tend to converge more quickly than those using CD when trained on the same amount of data.

3. **Robustness**: Through a capacity–robustness framework, Han et al.[54] demonstrated that the CI strategy exhibits lower capacity but higher robustness under distributional drift, making it less sensitive to noise and less prone to overfitting compared with the CD strategy.

While CI offers notable benefits, it is also inferior to CD in that its processing of variates independently can hinder the model's ability to learn inter channel information[139]. Additionally, this separation may lead to longer training time and

reduced scalability. Thus, several recent works [39, 44] keep using CD with their Transformer-based architectures.

Comparing with the canonical Transformer model, the recent Transformer-based models have been redesigned to show significantly enhanced TSF capabilities. The empirical experiments on real world data have shown the effectiveness of the techniques including: sparse attention [36], patching [40], CI[40] and CD [44]. Although these Transformer-based architectures have been extensively validated on real-world datasets including traffic [25], electricity consumption [36], weather [38], and currency [44], we notice very few of them[31, 131] have been used on cryptocurrency price forecasting.

This study addresses this gap by systematically evaluating the effectiveness of single-scale patching, multi-scale patching (with, and without, FFT), and CI versus CD processing in enhancing Transformer-based forecasting performance under the highly volatile dynamics of cryptocurrency markets. Through a step-by-step experimental design, we rigorously assess the contribution of each technique, verify their individual impact on prediction accuracy, and ultimately identify multi-scale patching combined with CI processing as the most effective configuration for cryptocurrency price forecasting.

2.5 Retrieval Augmented(RA) Techniques for TSF

RA can be traced back to Retrieval Augmented Generation (RAG), which was firstly introduced for knowledge-intensive natural language processing (NLP) tasks by Lewis et al.[56], since then, it has been adapted as a key component in the large language model (LLM) pipelines in NLP tasks[140, 141]. For a specific query, RAG retrieves semantically relevant external knowledge or documents from a vector database to provide up-to-date context information to LLMs. The generated dynamic context enables LLMs to produce more informed and accurate responses. Additionally, RAG reduces the reliance on retraining or domain specific fine tuning by enabling models to access task relevant information through retrieval[57]. As

noted in Chapter1, depending on the underlying TSF architecture, this technique is referred to as either RAF or RAG. In our case, we refer to it as *Retrieval-Augmented Context (RAC)* since we are using RA for creating an informal historical context. The following paragraphs review existing RA architectures and recent innovations that incorporate RA techniques into TSF models.

Jing et al.[142] highlighted a significant limitation in current DL methods in TSF. They identified that when there are insufficient historical observations, particularly when the ratio of the forecasting horizon to the sum of input and output lengths is high (e.g., 0.8), this leads to a significant increase in forecasting uncertainty[142]. The authors proposed a ReTime framework that retrievess relevant time series as references to reduce this uncertainty. ReTime is a two-stage method that includes relational retrieval and content synthesis. In the first stage, ReTime identifies and retrieves top-k relevant historical time series from a database based on established relationships among time series. For example, for a traffic dataset, the relation is defined by whether two speed sensor stations are adjacent. In the second stage, ReTime synthesises the future values by combining the content of the target time series with the retrieved references. This stage includes an input module to map time series into an embedding space, an aggregation module with content and temporal attention to address discrepancies and temporal dependencies, and an output module for final prediction. In this work, two datasts are evaluated: traffic ((PeMS)[26]) and temperature[143]. For both forecasting and imputation tasks, ReTime achieved RMSE scores when compared against traditional RNN-based methodologies.

Wang et al.[55] addressed the challenges faced by existing RNNs and Transformer models in effectively capturing intricate temporal patterns within time series data. Inspired by RAG, they proposed retrieval-augmented Time Series Forecasting (RATSF). This innovative framework is designed to identify and leverage similar historical data to guide the forecasting process, thereby enhancing the model's ability to learn from past analogous scenarios. RATSF is composed of two key components:

[26]https://dot.ca.gov/programs/traffic-operations/mpr/pems-source

the Time Series Knowledge Base (TSKB) and retrieval-augmented Cross-Attention (RACA). TSKB segments and archives historical time series data, establishing an indexing system that allows fast retrieving hostorical context. RACA, a specialized variant of the cross-attention mechanism, is integrated into the decoder layers of vanilla Transformer architectures. For a given forecasting input, the TSKB retrieves N related historical sequences using either Dynamic Time Warping (DTW)[144] or embeddings similarity measured by Euclidean Distance. Subsequently, RACA merges these retrieved sequences with the current input. RATSF is evaluated across several datasets, including the Fliggy Hotel Service Volume (FHSV), ETTh1 (electricity-transformer temperature)[38], exchange[145] and traffic data[27]. It was integrated with and compared against prominent Transformer variants such as Fedformer[32], Autoformer[36], and NS-Transformer[146], as well as the foundational vanilla Transformer. Furthermore, its performance was contrasted with ReTime, the aforementioned RAG-based TSF architecture proposed by Jing et al.[142]. The experimental results demonstrated that equipping Transformer-based models with RATSF led to significant reductions in both MSE and MAE for the specific dataset tested. Specifically, for the FHSV dataset, RATSF facilitated an MAE loss reduction of 18% for the vanilla Transformer, 14% for NS-Transformer, 5% for Autoformer, and 4% for Fedformer. In contrast, ReTime yielded MAE reductions of 5.34%, 5.9%, 2.8%, and 1.7% for the respective models. Thus, the authors conclude that the Retrieval Augmented approach can help improving the Transformer-based time series models' performance and RATSF demonstrates superior performance when comparing with ReTime.

Tire et al.[57] proposed a novel Retrieval Augmented Forecasting (RAF) framework. RAF is designed to enhance the capabilities of time series foundation models, such as the Chronos[147], by leveraging external knowledge and similar historical patterns to guide the forecasting process. The first step of RAF is to build a database from various time series across domains. Given a query in the form of a historical

[27]http://pems.dot.ca.gov/

time series, RAF then retrieves similar time series from this database. The similarity is quantitatively determined by the $L2$ norm between the embeddings of the query time series and those stored in the database. Subsequently, the retrieved time series, along with their corresponding future segments, are then combined and appended to the original query to form an augmented input for the subsequent TSF model. RAF has been evaluated on two real world benchmark datasets encompassing both long-context/long-horizon and short-context/short-horizon forecasting scenarios. Utilizing Chronos Mini and Chronos Base as the underlying forecasting models, the experimental results consistently demonstrated that RAF significantly outperformed the baseline Chronos models in terms of Weighted Quantile Loss (WQL)[28] and Mean Absolute Scaled Error (MASE). Specifically, RAF achieved average WQL scores of 0.823 and MASE scores of 0.939 for Chronos Mini, and further improved performance with WQL scores of 0.772 and MASE scores of 0.908 when employing Chronos Base.

To address the problem of Time Series Foundation Models (TSFMs) struggling with zero shot forecasting on unseen or specific domains, Zhang et al.[58] proposed a TimeRAF (Retrieval-Augmented Forecasting) model. TimeRAF is structured as a two-stage method involving knowledge retrieval and knowledge integration. For knowledge retrieval, it builds a knowledge base, a database that contains multiple datasets from various domains. A learnable retriever is designed to identify the top-k relevant historical time series using dot product similarity and embeds both the input query and the retrieved relevant historical time series. In the knowledge integration stage, the embeddings of the input query and the retrieved historical data will be concatenated to form a context. Followed the concatenation, an MLP layer will process the context to extract the most relevant information. A residual layer is subsequently applied to keep the original input. Finally, the "knowledge-enhanced input embedding" is fed into the TTM-Base(TTMB) model[148] which is the TSF backbone of the TSFMs for prediction. TimeRAF is evaluated on several

[28]WQL is used to measure how closely the predicted distribution aligns with the actual observed values across various quantile level[57].

widely used benchmark datasets including ETT (electricity-transformer temperature), weather, traffic, and electricity which are commonly adopted in prior TSF studies[149, 150]. The framework is tested under zero-shot forecasting settings and across multiple backbone forecasting models, including the large-scale foundation model Timer-1B[151]. The ablation study shows that with TimeRAF the backbone model's prediction accuracy enhanced across all the datasets. .

Retrieval Augmented Time series Diffusion (RATD) model is proposed by Liu et al.[64] to address the problem that traditional diffusion models are unstable and struggle with complex patterns in prediction tasks. Such limitations were found to have been caused by limited size and imbalanced datasets in model training. Additionally, time series data does not seem to have had semantic meaning or label correspondence which leads to insufficient guidance to diffusion models[64]. RATD tackled the problems by an embedding-based retrieval process and a reference-guided diffusion model. The embedding-based retrieval process retrieves the most similar time series to historical data from a database. The similarity is quantified by the Euclidean distance ($L2$ norm) between the embeddings of the historical data and the retrieved time series. The retrieved time series are then utilized as guidance in the denoising process of the diffusion model. RATD is evaluated on several real world datasets including electricity[29], wind[152], exchange[119], weather[30] and MIMIC-IV-ECG Health[153]. Comparing with various TSF methods, RATD has achieved superior forecasting accuracy in terms of MSE, MAE and Continuous Ranked Probability Score (CRPS) on exchange, wind, and weather dataset, and comparable results on electricity dataset. For the MIMIC-IV-ECG dataset, RATD showed significant improvement in addressing rare cases, outperforming other models on these particularly difficult predictions.

The aforementioned innovations all employ RA to provide contextual information that enhances the forecasting capabilities of time series models. Similarly, RA

[29]Hourly electricity consumption data from `https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014`

[30]The Weather dataset from the Max Planck Institute (https://www.bgc-jena.mpg.de/wetter/), contains 21 meteorological indicators recorded at 10-minute intervals from 2020 to 2021.

has been combined with LLMs to achieve the same objective. Ravuru et al.[63] proposed Agentic RAG framework leverages a hierarchical, multi-agent architecture to tackle complex time series tasks. In the framework, a master agent analyses the incoming user request, and orchestrates specialized sub-agents dedicated to tasks like forecasting, anomaly detection, imputation, and classification. The sub-agents are pre-trained language models (e.g. Gemma[154] and Llama 3[155]) that are fine-tuned for time series tasks. Each sub-agent maintains distinct prompt pools as internal knowledge bases, which store domain- and task-specific knowledge as key-value pairs about historical patterns and trends. A dynamic prompting mechanism allows sub-agents to retrieve the most relevant k prompts based on cosine similarity with the input time series. The retrieved prompts are then concatenated with the original input time series to form an augmented input embedding for time series tasks including forecasting, classification, anomaly detection and imputation. The proposed method is tested on various datasets such as ETT(electricity-transformer temperature)[38] and PeMSD traffic data[156]. Overall, the proposed framework outperforms baseline methods across benchmark datasets, showing significant improvements on the forecasting and anomaly detection tasks.

Xiao et al.[62] reported that traditional RAG architectures are designed to retrieve and integrate discrete textual data, which inherently differ from continuous temporal signals. The authors highlighted that, using traditional RAG, it is difficult to align retrieved information with forecasting tasks. TimeRAG is proposed to address the problem. The framework operates in two stages: Firstly, it optimizes retrieval by utilizing LLM's feedback to extract relevant historical sequences. The feedback is defined as the probability that the LLM produces a correct output when conditioned on a query sequence and a candidate retrieved sequence. Secondly, it integrates this information into LLMs by formatting time series data in JSON for better comprehension and embedding it into the LLM's input context. This approach, described as the first RAG method tailored for continuous temporal sequences, significantly outperforms conventional context learning LLMs and exist-

ing RAG methods in predicting stock movements across various real world datasets. TimeRAG has been applied to the task of predicting future stock movement directions using four benchmark datasets of high trade volume stock in US markets: ACL18[157], BigData22[158], CIKM18[159] and Stock23 that contains 50 stock historical prices from 3 January 2022 to 31 December 2023. The experiment results show that TimeRAG notably outperformed all evaluated approaches across all four datasets in terms of the Matthews Correlation Coefficient (MCC)[31]. It achieved MCC scores of 0.140 on ACL18, 0.145 on BIGDATA22, 0.197 on CIKM18, and 0.219 on Stock23, significantly surpassing baselines.

In [59], Han et al. proposed RAFT, a method that incorporates similar historical patterns retrieved from the training corpus into the forecasting pipeline. The core idea of RAFT is that the input with future values from the most similar historical segments can provide an explicit inductive bias at inference time. To account for the heterogeneous temporal behaviors at different periodicities, RAFT constructs multiple downsampled versions of the full series using periods 1, 2, 4. For each downsampled series, consecutive time frames are structured as key patches and their corresponding future "value" patches via a sliding-window approach. The retrieval module computes similarity with Pearson correlation between the input query and the keys. The top-k most relevant key patches and their corresponding future segments are aggregated through a softmax weighted attention mechanism. The aggregated retrieval results from all periods are linearly projected into a shared latent space, summed, and then concatenated with the original preprocessed input. Finally, a simple linear predictor produces the forecast. Empirically, RAFT demonstrates strong and consistent performance across ten widely adopted benchmark datasets (ETT[38], electricity[161], exchange rate[119], illness[32], solar[138], traffic[156], and weather[33]). The experiment results show that RAFT achieves an average win ratio of

[31]MCC is a correlation coefficient between the observed and predicted binary classifications[160]. In the reviewed study, a positive MCC indicates that the model is effectively retrieving valuable historical sequences to assist the LLM in analyzing and predicting stock trends.

[32]weekly ratio of patients with influenza-like illness from 2002 to 2021) https://gis.cdc.gov/grasp/fluview/fluportaldashboard.html

[33]The same weather data used in RATD[64]

86%, outperforming transformer-based architectures (Autoformer[36], Informer[38], FedFormer[32], PatchTST[40]), MLP-based models (DLinear[162], TimeMixer[163], TSMixer[164]), and convolutional approaches (MICN, TimesNet).

Li (2025) introduces Retrieval-Augmented Forecasting(RAF)[60], a retrieval enhanced framework designed specifically for tabular TSF, addressing the persistent challenges of schema rigidity, temporal myopia, and pipeline sub-optimality that affect existing models. Schema rigidity arises because conventional forecasters cannot adapt to structural data schema changes[165]. RAF overcomes this through dynamic schema hashing, which embeds column metadata so that semantically related variables such as "revenue" and "sales" are aligned despite naming differences. Temporal myopia refers to a forecaster's inability to model relevant distant events[60]. RAF addresses this using temporal cross-attention, enabling the model to jointly attend to the current table and a set of retrieved, schema-aligned historical table slices. Finally, RAF resolves pipeline by training the retriever and forecaster jointly via Gumbel Softmax-based optimization[166], ensuring that the retriever learns to return predicatively useful historical tables rather than merely semantically similar ones. RAF is evaluated on three large and heterogeneous real-world benchmarks: FRED-MD[167](107 monthly US macroeconomic indicators from 1959-2023), Yahoo Finance Volatility (daily stock returns and 10-K textual filings for S&P 500 firms from 2010–2023), and World Bank Development Indicators (more than 50 years of cross country development indicators with frequent schema change). Across these datasets, RAF is compared against a comprehensive set of six strong baselines: DeepAR[168], N-BEATS[169], TSMixer[164], Temporal Fusion Transformer(TFT)[170], TAPAS-RAG[171], and a Schema-Adaptive GNN[172]. Using sMAPE as the main evaluation metric, RAF was found to deliver consistent performance gains with up to 26% lower sMAPE on FRED-MD and strong robustness under schema shifts. On Yahoo Finance, RAF also reaches 65.4% directional accuracy for tech stocks, outperforming TAPAS-RAG and TFT.

Yang et al.[61] propose TimeRAG, a retrieval-augmented framework designed to

enhance the forecasting capability of LLMs for TSF tasks. The framework consists of three key stages. 1. Time Series Knowledge Base (KB) Construction: The raw time series is segmented using a sliding window, and K-means clustering is applied to extract representative subsequences, forming a diverse knowledge base of historical patterns. 2. Retrieval of Reference Sequences: Given a forecasting query, TimeRAG retrieves the most similar subsequences from the KB using DTW chosen for its robustness to temporal misalignment and ability to match sequences with similar waveforms despite local distortions. The top-k most similar sequences are selected as reference patterns. 3. LLM Augmentation and Prediction: The query sequence and retrieved reference sequences are reformatted into a unified textual prompt and fed into a frozen LLM[34] for forecasting, allowing the model to leverage retrieved historical analogues without modifying LLM parameters. TimeRAG is evaluated on the M4 dataset[137], which includes $100,000$ series across diverse domains and frequencies. The framework is compared against SOTA Transformer-based forecasting models (iTransformer[44], FEDformer[32], Pyraformer[135], Autoformer[36], Informer[38], Reformer[37]) and competitive alternatives (Time-LLM[174], DLinear[162], TSMixer[164], MICN[175], FiLM[176], LightTS[177]). Experimental results show that incorporating retrieval improves LLM forecasting accuracy by an average of 2.97%, with TimeRAG found to be achieving top-3 performance in 14 out of 18 evaluations and outperforming Time-LLM across all major metrics (SMAPE, MASE, Overall Weighted Average (OWA))[61].

To summarize the RA-based approaches in TSF, the reviewed literature reveals a common three-stage workflow: First, constructing a time series knowledge base; Second, retrieving historically similar patterns; Third, integrating the retrieved information into the forecasting model. A range of similarity metrics has been employed for retrieval, with Euclidean distance (L2 norm) and DTW being the most frequently-adopted due to their effectiveness in distance and shape similarities in temporal dynamics. In all methods, the top-k most similar historical segments are

[34]A frozen LLM refers to a pre-trained large language model with parameters not updated[173].

Study	**Similarity calculation method**	**Integration method**
ReTime[142]	Relational retrieval: Random Walk with Restart (RWR) computes relational proximity between the input time series (x_{input}) and historical sequences (x_h)	[$x_{1...k}$, x_{input}]
RATSF[55]	Dynamic Time Warping (DTW) and Euclidean distance	Cross attention applied between $x_{1...k}$ and x_{input}
RAF[57]	L_2 norm	[$x_{1...k}$, $y_{k+1...j}$, x_{input}]
TimeRAF[58]	Dot product	Pairwise concatenation: each retrieved $x_h \in x_{1...k}$ is paired with x_{input}, concatenated, averaged to form z, and z is concatenated with x_{input} for downstream prediction
RATD[64]	L_2 norm	[$x_{1...k}$] and [x_{input}] used as separate inputs to a diffusion model
Agentic RAG framework[63]	Cosine similarity	[$x_{1...k}$, x_{input}]
TimeRAG[62]	LLM-guided retrieval	$x_{1...k}$ and x_{input} formatted as JSON and fed into an LLM for prediction
RAFT[59]	Pearson correlation	[$y_{k+1...j}$, x_{input}]
RAF[60]	Embedding similarity (similarity algorithm not mentioned)	Cross attention applied between $x_{1...k}$ and x_{input}
TIMERAG[61]	DTW	$x_{1...k}$ and x_{input} are reformatted into a unified textual prompt

Table 2.6: Summary of the methods employed to measure similarity between the current input x_{input} and historical sequences x_h across RA-based time series studies. It also details the approaches used to integrate the retrieved historical sequences $x_{1...k}$ into downstream forecasting models.

retrieved and subsequently used to enrich the model's contextual understanding. The nature of the retrieved information differs across studies. Tire et al.[57] retrieve both the historical analogues sequences $x_{1..k}$ and the subsequent future values $y_{k+1..j}$, thereby providing the forecaster with richer information. In contrast, Han et al. [59] rely solely on retrieved future values $y_{k+1..j}$, using them as direct analogues to guide forecasting. Once retrieved, $x_{1..k}$ (or $y_{k+1..j}$) are feed into the prediction model through concatenation or cross-attention. For LLM forecasting methods, the retrieved information along with the input query are normally transformed to JSON for forecasting.

Table 2.6 summarizes the similarity measures and integration mechanisms used across the surveyed studies. In the table, x_{input} denotes the current input sequence, $x_{1..k}$ the retrieved historical segments, and $y_{k+1..j}$ their corresponding future values.

The notation $[x_{1..k}, x_{input}]$ indicates simple concatenation of retrieved and current inputs.

As far as we were able to ascertain from extensive research on the subject, the application of RA for constructing auxiliary informational contexts in cryptocurrency TSF remains largely unexplored. Following the methodological workflow identified in the reviewed literature, this study investigates the feasibility of incorporating RA into cryptocurrency forecasting in Chapter 4.5.

2.6 Summary

This chapter reviews the existing forecasting methodologies applied to cryptocurrency markets, the SOTA Transformer-based methods, and RA techniques for TSF . It begins with Section 2.2, which establishes the methodological foundations by examining forecasting techniques developed for traditional FX markets. Given that FX and cryptocurrency price data share several important characteristics, including high volatility, non-stationarity and regime-dependent dynamics[31, 51, 67–69, 178], insights from FX forecasting provide a valuable methodological basis for cryptocurrency price prediction. Section 2.3 then surveys forecasting methods employed in recent cryptocurrency studies.

Across both domains, the reviewed literature indicates that forecasting approaches can be broadly categorized into three groups: statistical methods, conventional ML techniques and DL models. Widely used statistical approaches include the ARIMA and GARCH families[77, 82, 83, 90, 96], which are effective for modeling linear dependency and volatility dynamics. Conventional ML methods commonly applied in this context include LR, SVM, KNN and tree-based methods[26, 27, 102, 109–111]. DL approaches frequently employed include MLP, CNN, GRU and LSTM[28, 29, 113, 116, 117]. Among these, LSTM has demonstrated strong predictive performance and remains one of the most widely adopted models in cryptocurrency forecasting[46, 112].

Each class of methods exhibits distinct strengths and limitations. Statistical

models have been shown to perform well for shot-term forecasting (1-, 5-, and 10-day) under relatively stable conditions[77, 78]. However, their ability to capture complex nonlinear dependencies is limited, particularly during periods of heightened market volatility[96]. Compared with statistical models, conventional ML approaches can better accommodate evolving market dynamics without requiring explicit model reconfiguration[46]. Nevertheless, ML methods may be prone to overfitting, especially when trained on limited and nosily datasets[46, 120]. DL models extend these capabilities by automatically learning nonlinear representations from data[46]. RNN architectures, particularly LSTM, have demonstrated strong predictive power in cryptocurrency forecasting tasks[82, 99, 100]. However, RNN-based models suffer from high computational costs due to their sequential processing nature and may struggle to capture very long-range dependencies because of vanishing gradients[108, 126].

To address these limitations, recent studies have explored Transformer-based architectures. Leveraging self-attention mechanisms that enable parallel computation and more effective modeling of long-term dependencies[29, 126], Transformer have achieved superior performance in many TSF tasks[**2025**, 31, 44, 45, 131, 139]. Despite these advantages, their adoption in cryptocurrency forecasting remains limited. Section 2.4 reviews the SOTA Transformer variants developed for TSF, highlighting both the shortcomings of the canonical architecture and recent SOTA techniques proposed to mitigate them. In particular, the discussion of patching strategies and CI and CD paradigms illustrates their potential to address key limitations of the canonical Transformer, including quadratic computational complexity and challenges in modeling long temporal patterns[128].

From a data-centric perspective, Section 2.5 examines recent developments in RAC, which employs RA techniques to construct informative contextual inputs for forecasting models. The essential components and workflow of the studies are summarized. Although RAC has been investigated in TSF[55, 57, 60–62], its effectiveness in cryptocurrency forecasting has not yet been studied.

This literature review indicates that cryptocurrency forecasting remains a highly challenging task due to the inherent characteristics of the data, including extreme volatility and complex non-linear dynamics[31, 51, 178]. Several important research gaps emerge from existing studies. First, the majority of previous work focus predominately on BTC and ETH[79, 85, 110–112, 116, 120], with limited attention to other assets such as ADA. Although empirical evidence suggests that cryptocurrency markets exhibit strong interdependence and spillover effects[11, 126], relatively few studies investigate multivariate forecasting frameworks that explicitly leverage cross-asset information. Second, while Transformer-based architectures have demonstrated superior performance in TSF task, their application to cryptocurrency forecasting remains limited. Third, RAC approaches have shown effectiveness in TSF by providing informative contextual information for forecasting models, yet their application in cryptocurrency prediction has been largely unexplored. Finally, the existing literature lacks standardized benchmark datasets, consistent evaluation metrics and undefined definition of forecasting horizons (e.g. short- and long-term forecasting). These inconsistencies hinder comparisons across studies and make it difficult to asses the effectiveness of proposed methods.

To address these gaps, this thesis proposes a multivariate forecasting framework that leverages historical price information from dominant cryptocurrencies to predict future ADA prices using a Transformer-based architecture augmented with an adaptive RAC module. Building upon the unified experimental framework introduced by Informer, the proposed approach employs standardized evaluation metrics and multiple forecasting horizons to enhance comparability, transparency, and reproducibility of results.

Chapter 3

Methodology

3.1 Introduction

This chapter introduces the proposed framework, MSPRACFormer (Multi-Scale Patching with Retrieval-Augmented Context Transformer), which is designed for the ADA price forecasting task by leveraging the historical price movements of dominant cryptocurrencies. By integrating configurable and adaptive modules, this architecture supports single- and multi-scale patching, as well as retrieval-augmented context. Prior to discussing the architectural design, Section 3.2 provides a comprehensive overview of the dataset employed in this study, facilitating an understanding of its key statistical properties. In Section 3.3, we formally define the forecasting problem and describe the experimental setups and evaluation metrics. Section 3.4 introduces MSPRACFormer with a detail description on the key techniques and the adaptive layers.

3.2 Dataset

3.2.1 Data collection

From CoinMarketCap [1], we observed that the top 10 cryptocurrencies by market capitalization account for approximately 88.6% of the total market capitalization, which amounts to 3.9 trillion in USD as of 9th September 2025. Table 3.1 shows the market capitalization and relative share of these leading cryptocurrencies. Within this group, Bitcoin and Ethereum can be seen as the dominant assets with 63.8% and 15.0% of the top 10 market capitalization respectively. Focusing on this subset allows the study to capture the dominant market dynamics, as these leading assets have been found to exert disproportionate impact on market-wide volatility[179].

Cryptocurrency	**Symbol**	**Market Cap (USD)**	**Market Share (%)**
Bitcoin	BTC	$2,205,475,731,102	56.53
Ethereum	ETH	$518,453,969,869	13.29
Tether	USDT	$168,866,854,547	4.33
XRP	XRP	$167,196,276,509	4.29
BNB	BNB	$119,849,964,213	3.07
Solana	SOL	$109,757,605,107	2.81
USD Coin	USDC	$72,545,673,238	1.86
Dogecoin	DOGE	$32,333,898,293	0.83
TRON	TRX	$31,346,593,255	0.80
Cardano	ADA	$29,372,649,526	0.75

Table 3.1: Top 10 cryptocurrencies by market capitalization

We analysed the daily closing prices of the top 10 market capitalization cryptocurrencies using a dataset that obtained from Yahoo Finance[2]. Daily closing prices were selected because they provide a widely-accepted and informative representation of market valuation while reducing the noise generated by hourly or minutes fluctuations and helping detecting sustained market trends rather than temporary blips[180]. In addition, the closing price reflects the final consensus valuation after

[1] https://api.coinmarketcap.com/data-api/v3/cryptocurrency/listing data accessed 9th September 2025

[2] `https://finance.yahoo.com`, accessed 9th September 2025

a full trading day and is widely used in both academic research[98, 108, 109, 113, 126] for forecasting.

The dataset covers a five-year period from 1st January 2020 to 1st January 2025. This time frame encompasses several episodes of heightened volatility in the cryptocurrency market, driven by major global events such as the COVID-19 pandemic[181], escalating China-US trade tensions[182, 183], Terra-LUNA crash[184], and the Russian invasion of Ukraine[181]. Empirical studies have shown that such crises substantially affect cryptocurrency volatility, returns, and inter-asset dependencies. For example, research examining the COVID-19 pandemic and the Russia-Ukraine war reports significant impacts on cryptocurrency volatility and spillover dynamics across markets[185]. Additionally, the Terra-Luna collapse in 2022 triggered systemic instability within the digital asset ecosystem, wiping out $50 billion in valuation in 3 days[186, 187]. By capturing both extreme stress periods and relatively stable phases, the chosen time frame provides a comprehensive representation of high- and low-volatility regimes, thereby enabling more robust evaluation of forecasting models under diverse market conditions.

Figure 3.1 illustrates the historical log-transformed price patterns of the top 10 market capitalization cryptocurrencies from 2020 to 2025. Because BTC's price level is substantially higher than that of the other assets, plotting raw daily prices obscures cross-asset comparisons. To enable a clearer visual examination of relative movements and co-trends across assets, we present the log-transformed historical prices.

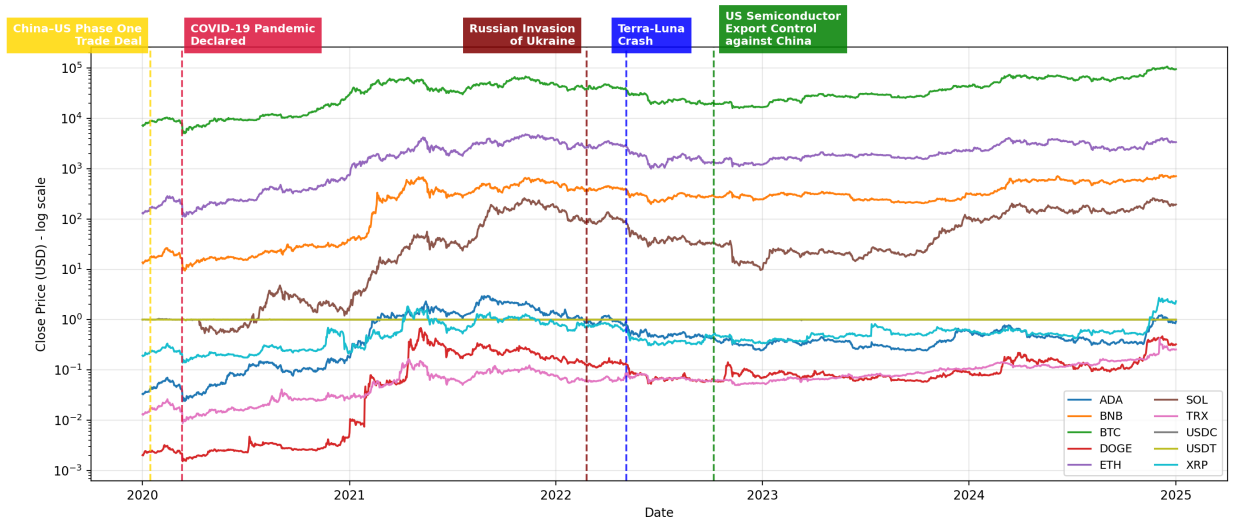


Figure 3.1: Historical price patterns of the top 10 market capitalization cryptocurrencies from 1st January 2020 to 1st January 2025. The panel displays the log-price (in USD) series to highlight relative growth dynamics.

Our empirical analyses in the subsequent sections reveal that cryptocurrency markets exhibit high volatility(Section 3.2.2), strong inter-correlations among major assets(Section 3.2.3), and temporal patterns(Section 3.2.4). These observations align with prior research demonstrating the extreme volatility of cryptocurrencies[188], time-varying inter-correlations across assets[31], the presence of seasonal patterns[189], and evidence of long-term trends [178].

3.2.2 Extreme Values of Volatility

To provide a scale-independent comparison of volatility across the top 10 cryptocurrencies, Table 3.2 reports descriptive statistics computed using daily log-returns rather than raw prices. Figure 3.2 offers a visual illustration of cross-asset differences in dispersion and overall volatility. It can be seen that assets such as TRX, ADA, and DOGE exhibit wider inter-quartile ranges and longer whiskers, indicating higher volatility and extreme return fluctuations, while BTC and ETH display comparatively lower variability despite their large market capitalization. Stablecoins[3] (i.e., USDC and USDT) show highly compressed boxes centered near zero. This behavior is expected, as stablecoins are explicitly designed to maintain a fixed value.

[3]Stablecoins are digital currencies that are pegged to non-volatile assets. [190]

Overall, these statistics highlight the high volatility and heterogeneity in the crypto market.

The "count" column in Table 3.2 reflects missing transactions corresponding to missing observations for SOL. This is because SOL was launched in March 2020,[4] with the earliest available trading data starting on April 10, 2020. Consequently, the dataset for SOL does not span the full five-year period covered for the other cryptocurrencies.

Symbol	Min	Max	Mean	Std. Dev.	Count	Annualized Volatility (30-day)
ADA	-0.5036	1.5163	0.00181	0.05174	1827	0.82784
BNB	-0.5431	0.5292	0.00216	0.04670	1827	0.59860
BTC	-0.4647	0.1718	0.00141	0.03407	1827	0.37681
DOGE	-0.5151	1.5163	0.00278	0.07368	1827	0.74576
ETH	-0.5507	0.2307	0.00178	0.04408	1827	0.52793
SOL	-0.5496	0.3872	0.00308	0.06729	1727	0.60545
TRX	-0.5231	0.6592	0.00162	0.04673	1827	2.19852
USDC	-0.0372	0.0424	-0.00000	0.00268	1827	0.00271
USDT	-0.0526	0.0534	-0.00000	0.00255	1827	0.00665
XRP	-0.5505	0.5486	0.00136	0.05530	1827	0.86434

Table 3.2: Descriptive statistics of daily log-returns for the top 10 cryptocurrencies

3.2.3 Cross-Asset Correlation of Cryptocurrency Markets

One of our goals of this thesis is to predict future ADA price using historical top market cap cryptocurrencies prices. To investigate the inter-relationships between the coins, we conducted a correlation analysis using both Pearson correlation coefficient and Spearman's rank correlation coefficient. Pearson correlation measures the strength and direction of linear relationships between two time series. Spearman correlation captures monotonic relationships. It has been found to be more robust measuring non-linear dynamics[191]. Both of the methods are commonly used in cryptocurrencies prices forecasting[192, 193].

[4]See Agatha Ozog, "Solana (SOL) was launched in March 2020 with an initial price of around $0.22 per SOL," *Binance*, 2020. Available at: `https://www.binance.com/en/square/post/21381405543082`.

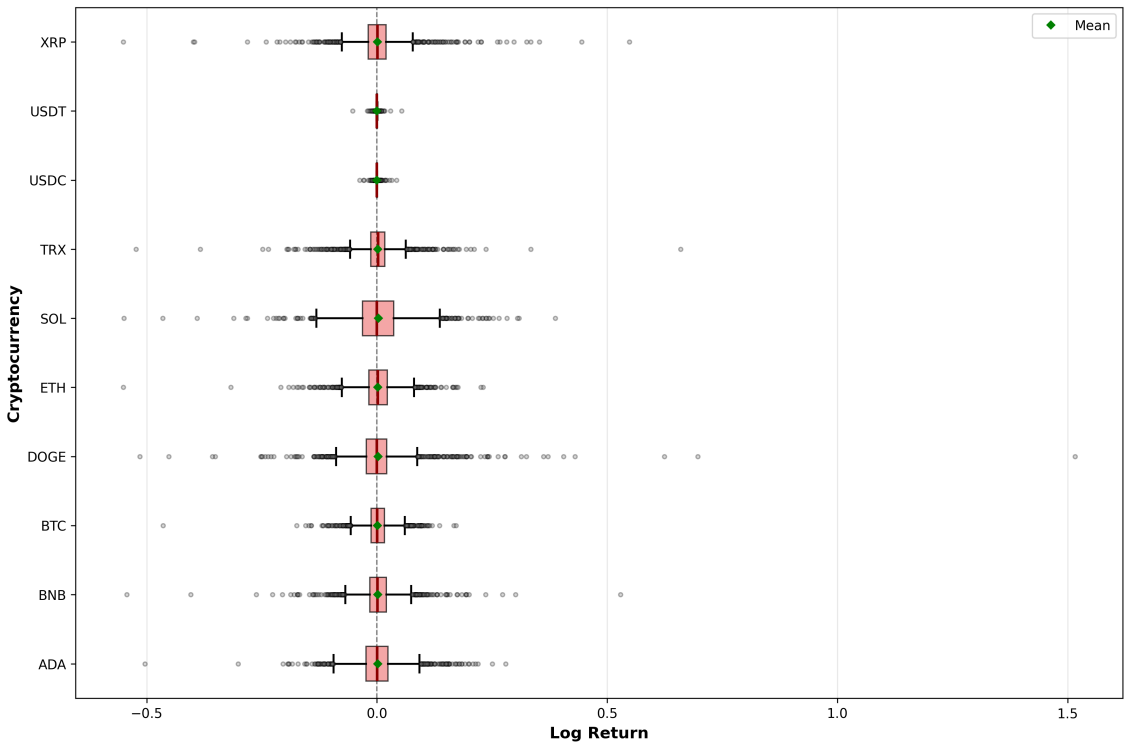


Figure 3.2: Boxplots of daily log-returns for the top 10 cryptocurrencies

Following previous studies[194, 195], we applied these two methods within rolling windows of 30, 60, 90, and 180 days on the detrended log price[5], thereby capturing the dynamic evolution of the inter-dependencies of the coins over short-, medium- and long-horizons. We observe that while ADA exhibits varying degrees of correlation with most major cryptocurrencies across time, stablecoins such as USDT and USDC consistently show near zero correlation with ADA. Our finding is consistent with what was reported by Rudkin et al. [196], who examined the top 15 cryptocurrencies by market capitalization and found that correlations between USDT and other coins remain close to zero. This result is expected, as stablecoins are explicitly designed to maintain a fixed value relative to fiat currencies so their price movements are insulated from the volatility that characterizes other cryptocurrencies. Consequently, and following prior studies[197, 198], USDT and USDC are removed from our dataset as they are intended to maintain a constant price.

The rolling Pearson and Spearman correlation results are plotted in Figure sets:

[5]The detrended price is computed by subtracting the rolling mean from the log price. An Augmented Dickey Fuller (ADF) test is applied to verify that the resulting series is stationary for Pearson and Spearman correlation analysis.

A.1 - A.4 and A.6 - A.9 in Appendix A. The average Pearson correlation heatmap (Figure A.5) and the Spearman correlation heatmap (Figure A.10) provide a more direct visualization of this finding.

3.2.4 Revealing Recurring Cycles: The Fast Fourier Transform (FFT)

The FFT is a widely-used technique in time series analysis that decomposes a series into its constituent frequency components to reveal recurring cycles and periodic patterns[199]. To discover the underlying seasonal patterns in the price movements, we conducted an FFT analysis on ADA, BTC and ETH log returns. Table 3.3 summaries the top 10 dominant Fourier frequencies for ADA, BTC, and ETH, and Figure 3.3 provides the associated power spectral density (PSD) plots. It can be observed that all three assets exhibit short-term cycles, with dominant periods clustered between 5 and 15 days, whereas ADA additionally displays distinct medium-term cycles in the 30- to 70-day range. It highlights that ADA, BTC and ETH exhibit identifiable periodic behaviors rather than purely stochastic fluctuations. Such frequency domain insights can be effectively leveraged to improve forecasting performance under the context of highly volatile cryptocurrency prices. For instance, Akbar [199] demonstrated that incorporating FFT-derived seasonal variations and periodic patterns enhanced the predictive accuracy of an XGBoost model for Bitcoin prices. Similarly, Cai et al.[41] employed FFT to select top-k periods as patch scales for their model, MSGNet, enabling it to capture temporal patterns across different time scales. Following the approach of Cai et al, our study assessed if FFT could help determine patch scales for multi-scale patching module in our proposed method. With FFT, the dominant frequencies and their corresponding periods are extracted. The periods are used as patching scales in the multi-scale patching mechanism, allowing the model to adjust its patch sizes adaptively, thereby capturing the underlying cyclic patterns in cryptocurrency price movements. The details of the experiment is recorded in Chapter 4.3.2.

Rank	Frequency (cycles/day)	Period (days)	Amplitude
1	0.023	44.282	4.328
2	0.116	8.635	4.321
3	0.116	8.592	4.221
4	0.023	43.175	4.038
5	0.099	10.099	3.992
6	0.089	11.288	3.935
7	0.027	36.745	3.932
8	0.014	69.080	3.760
9	0.121	8.263	3.713
10	0.195	5.125	3.644

(a) ADA

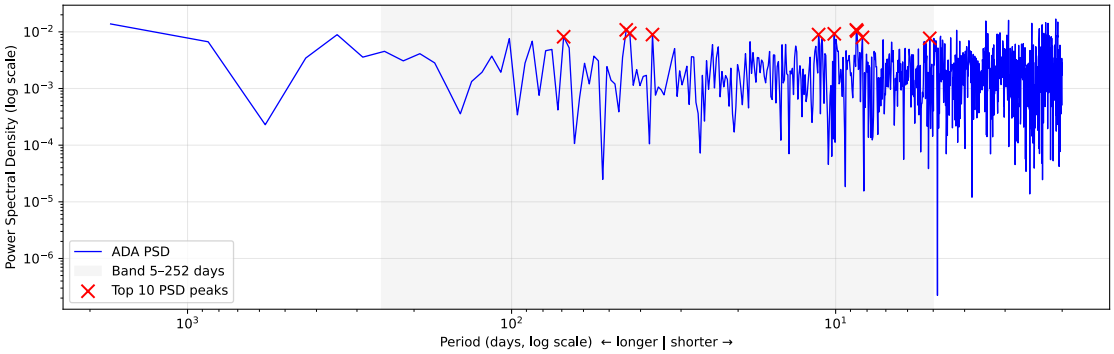
Rank	Frequency (cycles/day)	Period (days)	Amplitude
1	0.108	9.285	2.842
2	0.036	27.413	2.791
3	0.076	13.183	2.771
4	0.121	8.263	2.765
5	0.089	11.288	2.714
6	0.075	13.285	2.694
7	0.124	8.033	2.664
8	0.189	5.298	2.641
9	0.025	39.250	2.621
10	0.073	13.706	2.596

(b) BTC

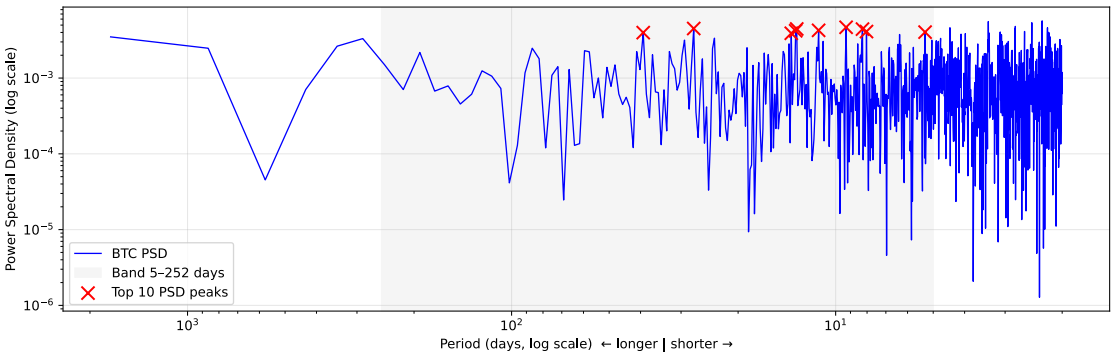
Rank	Frequency (cycles/day)	Period (days)	Amplitude
1	0.089	11.288	4.021
2	0.149	6.720	4.007
3	0.108	9.285	3.951
4	0.076	13.183	3.766
5	0.191	5.233	3.723
6	0.121	8.263	3.449
7	0.009	115.133	3.333
8	0.036	27.413	3.297
9	0.033	30.298	3.287
10	0.034	29.271	3.270

(c) ETH

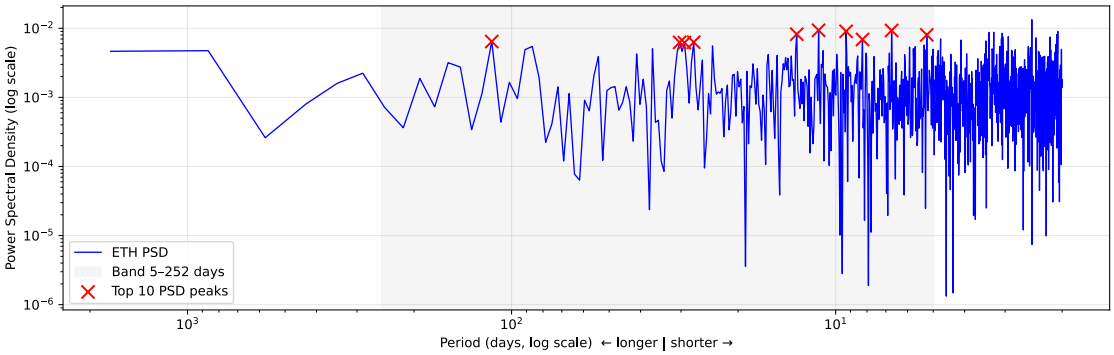
Table 3.3: Top 10 Fourier frequencies by amplitude for ADA, BTC and ETH.



(a) ADA



(b) BTC



(c) ETH

Figure 3.3: Power spectral density (PSD) of ADA, BTC, and ETH log-returns, restricted to periods between one trading week (5-day) and one trading year (252-day).

3.2.5 Dataset for experiments

Using data from CoinMarketCap and Yahoo Finance covering the 2020–2025 period, we characterized the crypto market as highly heterogeneous, with BTC and ETH collectively dominating market capitalization. The rolling correlation analysis using both Pearson and Spearman coefficients revealed dynamic and time varying

interdependencies among major cryptocurrencies. ADA displayed fluctuating correlations with leading market assets such as BTC and ETH, suggesting that its price formation is influenced by the market condition. The FFT analysis on the closing prices of ADA, BTC, and ETH revealed dominant periodicities between approximately 5 to 15 days, corresponding to weekly and bi-weekly market cycles. These findings demonstrate that, despite their volatility, cryptocurrencies appear to exhibit recurrent seasonal behavior. With the assumption that using these periodic components may further improve model's prediction ability, we conducted an experiment that utilizing FFT to determine patching scales for our multi-scale patching model. Experiment details and results are presented in the following chapter 4.3.2.

From the data insights, all observations prior to 10 April 2020 were removed due to missing data for SOL. After these adjustments, the final dataset for the following experiments comprises 1728 daily time steps for ADA, BNB, BTC, DOGE, ETH, SOL, TRX, and XRP, spanning the period from 10 April 2020 to 1 January 2025. The forecasting task leverages the interdependencies across these eight cryptocurrencies to improve predictions of future ADA, BTC and ETH prices. Table 3.4 illustrates the sample data used for the experiments in this study.

Date	ADA_close	BNB_close	BTC_close	DOGE_close	ETH_close	SOL_close	TRX_close	XRP_close
2020-04-10	0.033215	13.737724	6865.493164	0.001968	158.412445	0.951053	0.012462	0.187463
2020-04-11	0.033353	13.804287	6859.083007	0.001990	158.216018	0.776818	0.012482	0.188062
2020-04-12	0.033805	14.375110	6971.091796	0.001990	161.142425	0.882507	0.012901	0.190135
⋮								
2025-01-01	0.917591	706.512939	94419.757812	0.324306	3353.504150	193.873733	0.255313	2.322340

Table 3.4: Sample of multivariate daily closing prices for selected cryptocurrencies used in the forecasting experiments

3.3 Experimental Setup

3.3.1 Forecasting problem definition

Formally, let $x_{1:t} \in R^{t \times N}$ denote the multivariate time series input, where each time step includes N features, and t is the number of past time steps. The forecasting

task is to predict the next l values $\hat{\mathbf{y}}_{t+1:t+\ell}$, representing the price of the crypto coin we are forecasting at the following time steps.

3.3.2 Experimental pipeline

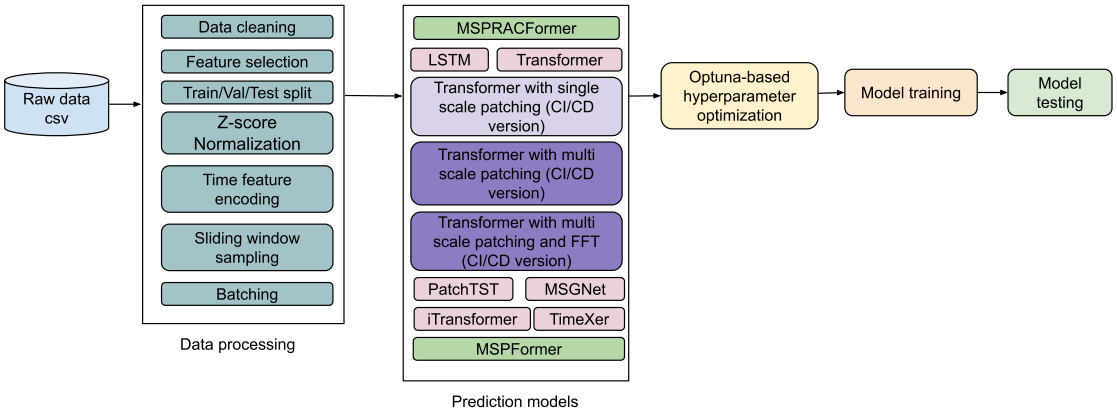


Figure 3.4: Workflow of the proposed experimental pipeline, from raw data preprocessing to model testing.

Figure 3.4 illustrates the pipeline of the experimental setup, which consists of raw data input (see data sample in Table 3.4), data processing, model training and testing stages. At data processing stage, the raw data are cleaned to address missing values. As noted earlier, all observations prior to 10 April 2020 are removed due to missing values for SOL, resulting in a final dataset containing 1728 time steps. The remaining dataset is then partitioned into training, validation, and test sets using a ratio of 70%, 10%, 20% respectively, resulting in 1188 samples for training, 153 samples for validation, and 324 samples for testing. All numerical features are standardized using Z-score normalization[6] to stabilize optimization and improve model convergence. The correspond date value is incorporated via time feature encoding, allowing the model to capture periodic structures and calendar-related effects. Table 3.5 illustrates the normalized cryptocurrency price data and encoded time features. The standardized series are subsequently converted into supervised learning samples using a sliding window mechanism that constructs input-output

[6]Z-score normalization is a data scaling technique that transforms data so it has $mean = 0$ and standard deviation = 1. This process centers the data around zero and scales it to have a unit standard deviation[200].

sequences from historical observations by a fixed-length window moving sequentially along the timeline[201]. The fixed-length window (input) is a historical sequence that contains the most recent observations used as predictors (refereed as *look-back window* in this thesis), while the corresponding future values (output) constitute the prediction targets. These sequences are organized into batches to support efficient gradient-based optimization[202]. The model is then trained using the training data, tuned based on validation performance, and ultimately evaluated on the unseen test set to measure its generalization capability.

time_features	XRP	BNB	BTC	DOGE	ETH	SOL	TRX	prediction_target
0.1667, -0.2000, -0.2260	1.3738e+01	6.8655e+03	1.9680e-03	1.5841e+02	9.5105e-01	1.2462e-02	1.8746e-01	3.3215e-02
0.3333, -0.1667, -0.2233	1.3804e+01	6.8591e+03	1.9900e-03	1.5822e+02	7.7682e-01	1.2483e-02	1.8806e-01	3.3354e-02
0.5000, -0.1333, -0.2205	1.4375e+01	6.9711e+03	1.9900e-03	1.6114e+02	8.8251e-01	1.2902e-02	1.9014e-01	3.3806e-02
...								

Table 3.5: Sample of the processed data before training

Before training, the hyperparameters[7] of the prediction models were optimized through a grid search implemented via Optuna, ensuring a balanced trade-off between model complexity and predictive accuracy. During training, the forecasting models parameters[8] are iteratively updated by minimizing the loss function, MSE, which measures prediction accuracy by penalizing deviations from the ground truth. Parameter updates are performed using the Adam optimizer[207], an adaptive gradient based algorithm. The gradients of the loss with respect to model parameters are computed via backpropagation. These gradients indicate how the parameters should be adjusted to reduce prediction error[208]. Backpropagation computes these gradients efficiently by propagating the error signal backward through the network[209]. The optimizer then uses the computed gradients to iteratively adjust the parameters toward minimizing the loss[210]. Adam is widely used in deep learning and is often the default choice for training neural networks due to its combination of momen-

[7]Hyperparameters are predefined configuration variables that control the structure of a model and are not updated during learning[203]. Typical examples include learning rate, batch size, and number of layers[204]. In Transformer architectures, the important hyperparameters include the number of attention heads, embedding dimension and feed-forward layer size[20].

[8]Model parameters are the weights and biases learned from data during training[205]. In Transformer architectures, trainable parameters include the project matrices (Q, K and V) used in multi-head self-attention, the feed-forward network weights, and embedding matrices. As opposed to hyperparameters, which are fixed prior to training, these parameters are optimized through training process[206].

tum and per-parameter adaptive learning rates[211]. The SOTA Transformer-based time-series forecasting models, including PatchTST[40] and DeformableTST[212], consistently rely on Adam, where it has demonstrated stable and efficient convergence, particularly in long-sequence settings. This makes Adam a well-suited and empirically validated optimizer for our Transformer-based forecasting experiments.

All the models employed in this study utilize a relatively large number of parameters compared with the size of the available dataset, the high model complexity increases the risk of overfitting[213], particularly when working with relatively small datasets such as ours. To prevent overfitting and ensure robust model selection, we employed an early stopping strategy that monitors validation performance and halts training when improvement ceases. In addition, a relatively high dropout rate was applied to improve regularization.

After training is completed, the final model is evaluated on the unseen test dataset, which is not used during training process. This evaluation provides an unbiased estimate of the model's ability to generalize to new data using a list of evaluation metrics that are explained in section 3.3.3.

This experiment pipeline is built upon the TSF framework built by Zhou et al.[38]. As aforementioned in section 2.3.4, this framework has been adopted by numerous recent SOTA Transformer-based TSF studies to facilitate fair and widely comparable model evaluation. The framework employs widely used ML libraries, such as PyTorch, Numpy, and Optuna, to support data processing, model development and hyperparameter optimization. Detailed information on the software environment and library versions is provided in the accompanying open-source repository[9].

3.3.3 Evaluation Metrics

MSE and MAE are the most commonly adopted metrics in SOTA Transformer-based TSF models [40–42, 44]. To be consistent with these SOTA and our benchmarks, we

[9]Available at `https://github.com/cmajorsolo/MSPRACFormer`

employed MSE and MAE. To complement these measures and provide additional insights into relative and scale invariant performance, we further report Root Relative Squared Error (RRSE), RMSE, and MAPE (report below). Each of these metrics quantifies forecasting accuracy from a distinct perspective, and their combined use ensures a more balanced and robust assessment of predictive performance.

As shown in equations 3.1 and 3.2, MSE quantifies the average squared differences between predicted values and ground truth, while MAE measures the average absolute differences. Lower MSE and MAE values indicate superior model performance. Notably, MSE penalizes extreme errors more heavily due to the squaring of residuals and MAE reflects the overall average error[214].

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{3.1}$$

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{3.2}$$

As Equation 3.3 shows, RMSE is the root over the MSE. By taking the square root, RMSE expresses the error in the same units as the target variable, making it more interpretable and preferable to MSE [215].

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{3.3}$$

MAPE expresses the average absolute error as a percentage of the actual values, Equation 3.4. It is intuitive and explainable, as it shows how far the predictions deviate from the ground truth in percentage on average. However, it can be unstable when the ground truth is close to zero[216].

$$\text{MAPE} = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{3.4}$$

Equation 3.5 shows that RRSE is the square root of RSE. RSE normalizes the sum of squared errors by the variance of the ground truth series. The metric com-

pares the actual forecast error to the forecast error of a simple baseline, the mean of the actual data[217]. It provides a scale independent measure that facilitates comparison across datasets or variables with different magnitudes, therefore it can be used in models whose errors are measured in different units[218]. When RRSE equals to 0, it means the model's prediction output exactly matched the observed values. When $0 < RRSE < 1$, the model performs better than the baseline that simply predicts the mean of the observed data. Conversely, when it is greater than 1, it suggests that the model underperforms the mean predictor, and may not capable to capture underlying temporal patterns[219].

$$\text{RRSE} = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{3.5}$$

Together, these metrics provide a robust and comprehensive evaluation framework. Scale dependent metrics such as MSE, RMSE, and MAE quantify absolute prediction accuracy, with RMSE offering the additional advantage of being expressed in the same units as the target variable, thereby making errors more interpretable in practical terms. In contrast, scale independent metrics such as RRSE and MAPE facilitate relative comparisons across different settings by normalizing errors with respect to the underlying scale of the data.

3.4 MSPRACFormer: Multi-Scale Patching with Retrieval-Augmented Context Transformer

The proposed architecture, MSPRACFormer, illustrated in Figure 3.5 and Algorithm 1, integrates four core components:

1. A RA-based contextual (RAC) module that retrieves relevant historical patterns to enrich the model's input representation;

2. A channel-independent multi-scale patching (MSP) module that captures temporal patterns at various time scales;

3. A Transformer encoder;

4. An aggregation layer with learnable weights;

The RAC module provides supplementary contextual information, particularly when the available input history is limited. The contextual information is obtained by incorporating historically similar data, which may provide additional signals about future price movements. Similarity is measured using the $L2$ distance. Within the MSP module, the input sequence is patched with various patching scales. Each patched sequence is subsequently processed by the Transformer encoder, producing a set of scale specific representations. Drawing on the insights from Section 4.4, where the CI approach was shown to outperform the CD alternative for cryptocurrency data with time varing interdependencies, the MSP module is therefore designed following the CI paradigm. Under the CI paradigm, each input channel is treated as a separate sequence and processed independently by a shared Transformer encoder. All channels use the same model parameters θ, with no inter-channel attention or feature mixing. During training gradients computed from each channel are aggregated to update the shared parameters in a single step. Since gradient aggregation is commutative, the processing order of channels does not influence the final parameter. Finally, the aggregation layer learns the contribution of each patching scale, determining how much each should influence the final prediction.

The RAC module is an adaptive and model-agnostic plugin. It can be activated or disabled by setting the corresponding hyperparameter *use_rac* to either `true` or `false`. Moreover, the multi-scale patching layer can be configured as a single-scale variant by setting the hyperparameter *top_k* to 1. This flexibility allows integration with a broad range of downstream forecasting architectures, enabling the model to dynamically incorporating external temporal context under data scarce conditions and modeling temporal patterns. Collectively, MSPRACFormer works as a framework that unifies single- and multi-scale temporal modeling, and RAC contextual information within a Transformer architecture.

Based on the findings reported in Section 4.4, FFT-based patch scale selection

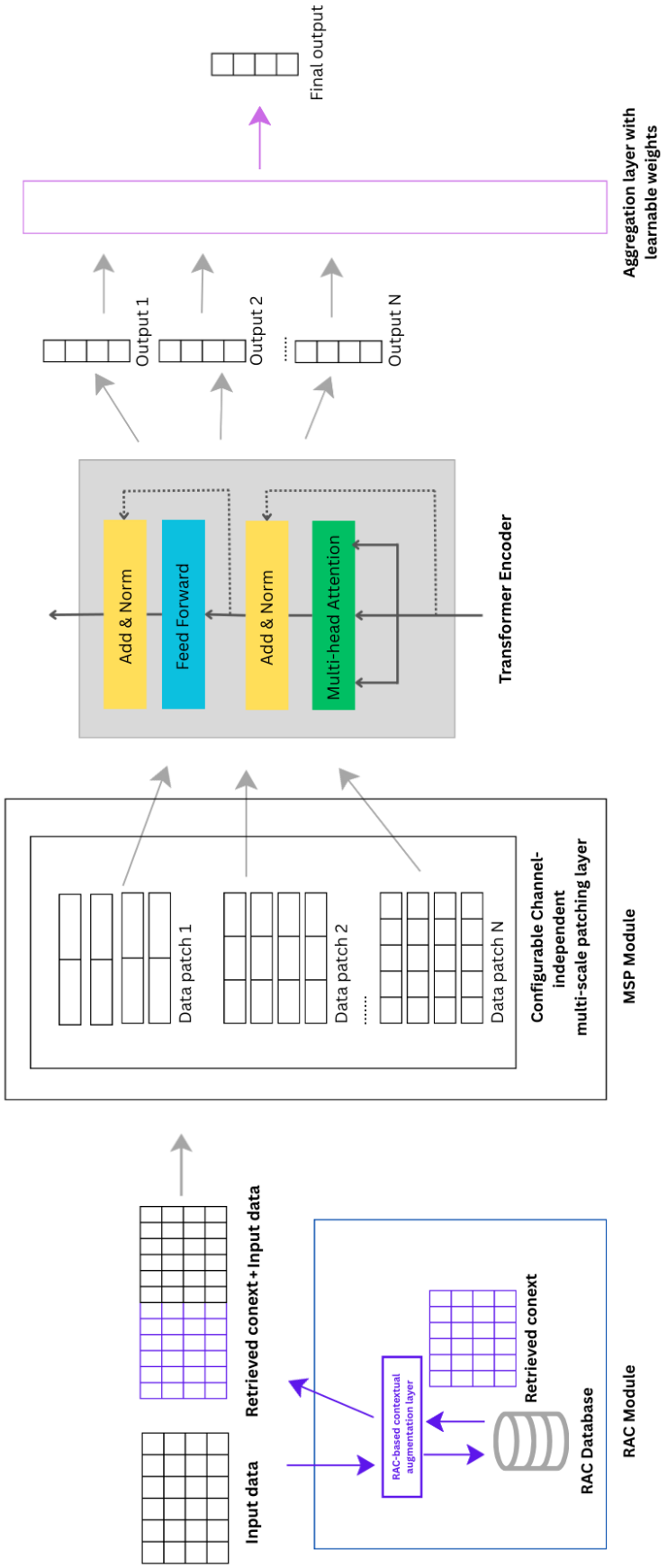


Figure 3.5: MSRACFormer architecture

Algorithm 1 MSRACFormer: Training and Inference (CI, Multi-Scale, optional RAC)

Require: Training set $\mathcal{D}_{\text{train}} = \{(X^{(i)}, Y^{(i)})\}_{i=1}^{N}$, Eval set $\mathcal{D}_{\text{eval}}$
Require: Horizon H, flag `use_rac`, historical context m, top-k patch scales k
Require: Empirical scale list $\mathcal{S}_{\text{emp}}$
Ensure: Trained parameters $(\theta^\star, \phi^\star)$ and predictions $\hat{Y}$, where:
θ denotes the shared Transformer encoder parameters responsible for temporal representation learning under CI setting;
ϕ denotes the parameters of the multi-scale fusion and output projection layers.

Procedure BuildRACDatabase($\mathcal{D}_{\text{train}}$) ▷ Index training windows only to avoid leakage
  return database $\mathcal{B}$
Function Retrieve($\mathcal{B}, X, m$)
  return X_h ▷ retrieve m historical information chunks by L2

Procedure Train($\mathcal{D}_{\text{train}}$)
  if `use_rac` **then**
    $\mathcal{B} \leftarrow$ BuildRACDatabase($\mathcal{D}_{\text{train}}$)
  for each epoch **do**
    for each minibatch (X, Y) from $\mathcal{D}_{\text{train}}$ **do**
      if `use_rac` **then**
        $X_h \leftarrow$ Retrieve($\mathcal{B}, X, m$)
        $Z \leftarrow [X_h + X]$ ▷ prepend along time
      else
        $Z \leftarrow X$
      $\mathcal{S} \leftarrow \mathcal{S}_{\text{emp}}$
      for each scale $s \in \mathcal{S}$ **do**
        $\mathcal{P}_s \leftarrow$ MakePatches(Z, s)
        for channel $c = 1$ to C **do** ▷ CI: process each channel with shared weights θ
          $\hat{H}_{s,c} \leftarrow$ EncoderCI($\mathcal{P}_s[c]; \theta$)
        $\hat{H}_s \leftarrow$ Stack($\hat{H}_{s,1}, \ldots, \hat{H}_{s,C}$)
      $\hat{Y} \leftarrow$ Aggregate($\{\hat{H}_s\}_{s\in\mathcal{S}}; \phi$) ▷ learnable scale-wise fusion
      $\mathcal{L} \leftarrow$ Loss($\hat{Y}, Y$)
      $(\theta, \phi) \leftarrow$ Update($\theta, \phi, \nabla\mathcal{L}$) ▷ update the parameters θ, ϕ using the gradients of the loss function $\nabla\mathcal{L}$

Procedure Infer($\mathcal{D}_{\text{eval}}$)
  if `use_rac` **then**
    $\mathcal{B}$ exists from training
  for each (X, Y) in $\mathcal{D}_{\text{eval}}$ **do**
    if `use_rac` **then**
      $X_h \leftarrow$ Retrieve($\mathcal{B}, X, m$); $Z \leftarrow [X_h, X]$
    else
      $Z \leftarrow X$
    $\mathcal{S} \leftarrow \mathcal{S}_{\text{emp}}$
    for each $s \in \mathcal{S}$ **do**
      $\mathcal{P}_s \leftarrow$ MakePatches(Z, s)
      for channel $c = 1$ to C **do**
        $\hat{H}_{s,c} \leftarrow$ EncoderCI($\mathcal{P}_s[c]; \theta^\star$)
      $\hat{H}_s \leftarrow$ Stack($\hat{H}_{s,1}, \ldots, \hat{H}_{s,C}$)
    $\hat{Y} \leftarrow$ Aggregate($\{\hat{H}_s\}_{s\in\mathcal{S}}; \phi^\star$)

has been shown to provide performance improvements within CD settings rather than CI settings. Because the proposed MSP module is developed under the CI paradigm, the FFT component is not included in the proposed method. Similarly, the findings in Section 4.5 demonstrate that RAC does not provide performance gains for ADA forecasting. Consequently, the parameter *use_rac* is set to `False`. However, given that the RAC module can be effective under certain conditions, such as a forecasting task with short input sequence, long forecasting horizons or when the underlying forecasting model is designed for handling exogenous data, it is designed as an adaptive module in the proposed framework for future inference tasks.

Prior to training, hyperparameter tuning was conducted through Optuna to identify optimal model configurations. For the MSP module, candidate patching scales and the parameter *top_k* were explored to determine the most effective combinations. For the Transformer encoder, key architectural hyperparameters such as the number of attention heads were also optimized through the search process. During training, the Transformer parameters (denoted as θ in Algorithm 1) such as the query, key and value projection matrices, are iteratively updated via backpropagation. In addition, the parameters of the aggregation layer (denoted as ϕ in Algorithm 1) are learnt to determine the optimal contribution of each patching scale to the final prediction.

We employ several SOTA Transformer-based TSF benchmark models, including PatchTST[40], MSGNet[41], iTransformer[44] and TimeXer[42], alongside the widely used RNN baseline LSTM[19], and the linear decomposition model DLinear [129] to conduct a comprehensive performance comparison. In addition to ADA forecasting, we further assess the robustness and generalization capability of MSPRACFormer by extending the forecasting task to BTC and ETH, the two largest cryptocurrencies by market capitalization.

Chapter 4

Empirical Evaluation of Architectural Options

4.1 Introduction

Following the established research framework, we conducted a set of experiments to systematically validate the underlying hypothesis and identify the effective techniques for the proposed method. This chapter details the experiments designed to test each hypothesis to identify the key components that ultimately shape the proposed architecture, MSPRACFormer. To answer Sub-RQ1 and validate H1, Section 4.2 compares the performance of canonical Transformer and LSTM on forecasting ADA price utilizing historical dominant cryptocurrency prices. Section 4.3 subsequently examines the effectiveness of various of patching techniques to address Sub-RQ2 and hypotheses H2.1, H2.2 and H2.3. Section 4.4 compares CI and CD processing, addressing Sub-RQ3 and hypotheses H3.1 and H3.2. Finally, the effectiveness of RAC is demonstrated in Section 4.5 through a comparative study of the RAC-enhanced SOTA Transformer-based TSF architectures against their respective baselines. Sub-RQ4 and H4 are addressed in the section. The insights from the experiments collectively inform the architectural decisions leading to the development of MSPRACFormer. Figure 4.1 maps the sub-sections to their corresponding

hypotheses.

4.2 LSTM vs Transformer

Following the research framework, the first stage involves answering Sub-RQ1, which hypothesizes that the canonical Transformer achieves superior forecasting performance to LSTM model in cryptocurrency markets.

Model design

LSTM Guided by the hyperparameter configurations obtained from Grid Search and Optuna's optimization process, the LSTM architecture is constructed using the layer specifications reported in Table 4.1.

Layer	**Param #**
Data embedding	768
LSTM × 2	8,192
Sequence projection	22
Feature projection	264
Total	**17,438**

Table 4.1: Architecture of the LSTM model.

Transformer Instead of the full canonical Transformer mode, we employed an encoder-only Transformer for this experiment. Unlike sequence-to-sequence tasks such as machine translation, where an encoder-decoder structure is required to generate the next token based on the encoder input and previously generated token, our cryptocurrency price forecasting task involves learning a mapping from a fixed lengths of past observations to a fixed horizon of future values. In this setting, the decoder becomes unnecessary, since a mapping function that projects encoded historical representations onto the prediction horizon in a single step can produce a prediction sequence. Therefore, an encoder-only Transformer is well suited for our forecasting task as it retains the self-attention mechanism to capture long-range temporal dependencies while avoiding the additional computational overhead of a decoder. The previous SOTA time series Transformer architectures (e.g. PatchTST[40]

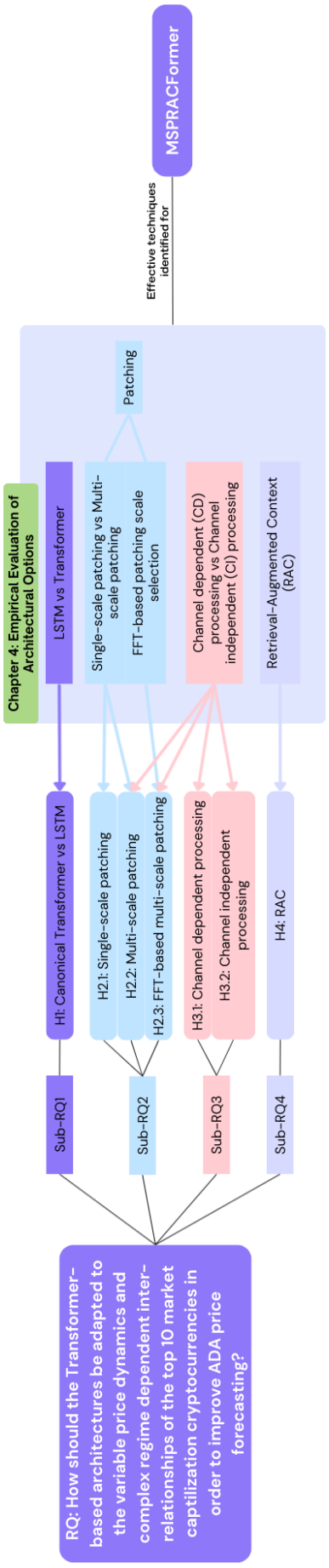


Figure 4.1: Methods chart: This chart links each sub-section of Section Empirical Evaluation of Architectural Choices to its corresponding hypothesis, highlighting the key findings that shape the proposed MSPRACFormer architecture.

and iTransformer[44]) have adopted the design and demonstrated the effectiveness of the encoder-only architecture. Our Transformer model is composed with the layers outlined in Table 4.2.

Component	**Layer**	**Param #**
Embedding	Token embedding	768
Embedding	Positional embedding	0
Embedding	Temporal embedding	96
Transformer × 5	Multi-head self-attention	4,224
	Residual connection	0
	Layer normalization	64
	Feed-forward layer × 2	2,112
	Residual connection	0
	Layer normalization	64
Transformer	Final layer normalization	64
Projection	Sequence projection	22
	Feature projection	264
	Total	**33,534**

Table 4.2: Encoder-only Transformer architecture used for cryptocurrency forecasting, as recommended by Optuna.

Input sequence length The hyperparameter optimization process with Optuna suggested an input sequence length of 21 days (corresponding to 3 weeks data) results in optimal performance from LSTM and Transformer models. As our experiments were designed to assess both short-term and long-term predictive capabilities (especially long-term capability because the existing literature suggests that Transformer is superior on capturing long-term dependencies[220]), we conducted a different set of experiments with input sequence 60 (2-month data). The prediction horizon was set to 1, 7, 15, and 21 days for the 21-day input experiment, and the 7, 15, 30, 45, and 60 days for the 60-day input experiment.

Experimental results and findings

Tables 4.3 and 4.4 present the numerical results across different experimental settings, while the accompanying figures provide a visual illustration of model performance under varying input lengths and forecasting horizons. Specifically, Figure 4.2 show the test performance of the LSTM and Transformer models when the input sequence length is 21. Similarly, Figures 4.3 depict the corresponding results for both

models when the input sequence length is extended to 60. From the combined

Horizon	Model	MSE	RMSE	MAE	RRSE	MAPE
1	Transformer	0.00402	0.06345	0.04384	0.5344	0.08025
	LSTM	**0.00196**	**0.04429**	**0.02844**	**0.4467**	**0.05134**
7	Transformer	0.00882	0.09395	0.06167	0.6538	0.10846
	LSTM	**0.00466**	**0.06826**	**0.04412**	**0.5577**	**0.07911**
15	Transformer	0.01370	0.11705	0.07673	0.7357	0.13802
	LSTM	**0.01332**	**0.11539**	**0.06927**	**0.7302**	**0.11752**
21	Transformer	0.02081	0.14427	0.08914	0.8390	0.15212
	LSTM	**0.02026**	**0.14229**	**0.08149**	**0.8047**	**0.13055**
Avg (Transformer)		0.01184	0.10468	0.06785	0.6909	0.11921
Avg (LSTM)		**0.01055**	**0.09256**	**0.05583**	**0.6499**	**0.09463**

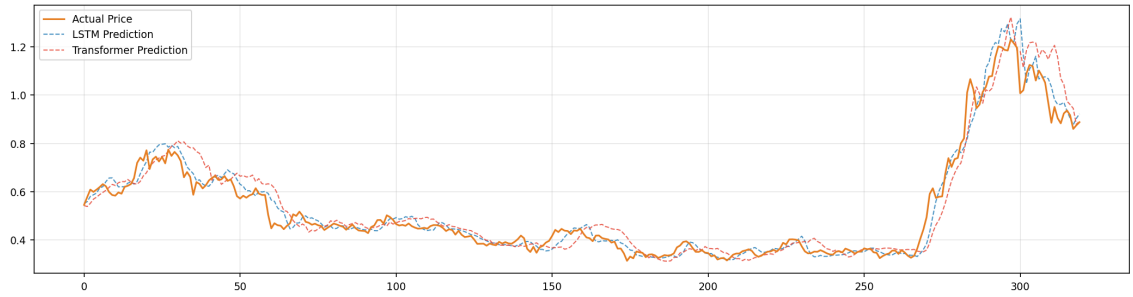
Table 4.3: Prediction results for LSTM and Transformer with input sequence length of 21. Best results for each prediction horizon are highlighted in bold.

Horizon	Model	MSE	RMSE	MAE	RRSE	MAPE
7	Transformer	**0.01460**	**0.12083**	**0.07931**	**0.73246**	**0.14088**
	LSTM	0.03261	0.18060	0.10887	0.89554	0.17647
15	Transformer	0.02656	0.16299	0.10802	0.86123	0.19228
	LSTM	**0.01754**	**0.13245**	**0.08006**	**0.77650**	**0.14143**
30	Transformer	**0.03459**	**0.18599**	**0.11075**	**0.92592**	**0.19206**
	LSTM	0.05218	0.22840	0.12763	1.02508	0.19786
45	Transformer	**0.04679**	**0.21627**	**0.12132**	**1.05626**	0.20566
	LSTM	0.04933	0.22200	0.12141	1.07167	**0.20037**
60	Transformer	**0.04037**	**0.20093**	**0.10964**	**1.07260**	**0.19391**
	LSTM	0.04101	0.20252	0.11407	1.07670	0.20770
Avg (Transformer)		**0.03258**	**0.17760**	**0.10501**	**0.93839**	**0.18416**
Avg (LSTM)		0.03833	0.19399	0.11081	0.97609	0.18437

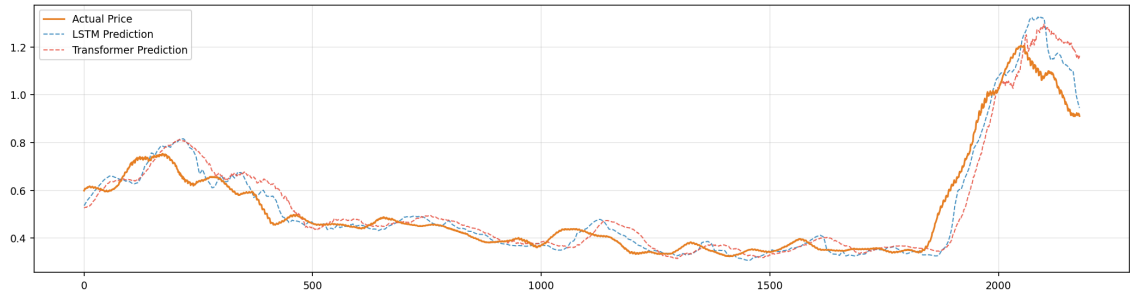
Table 4.4: Prediction results for LSTM and Transformer with input sequence length of 60. Best results per horizon are highlighted in bold.

experiments results listed in Tables 4.3 and 4.4, several clear patterns emerge that shed light on the relative strengths and limitations of the Transformer and LSTM models in cryptocurrency forecasting.

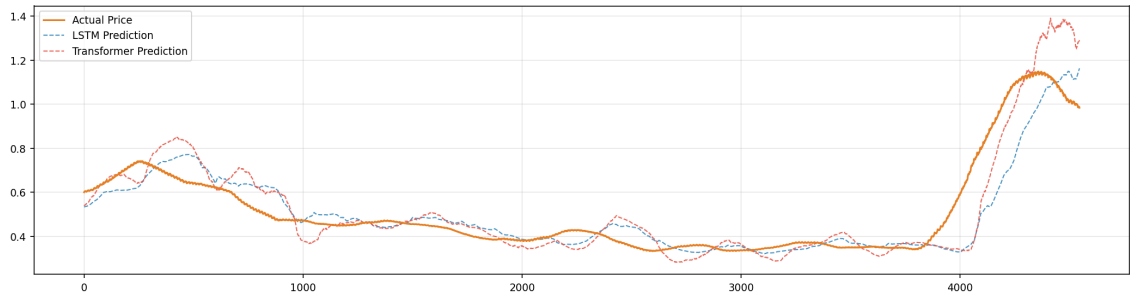
First, our experimental results support the conclusion that the input sequence length is a critical hyperparameter for TSF models. Consistent with Minotti[221],



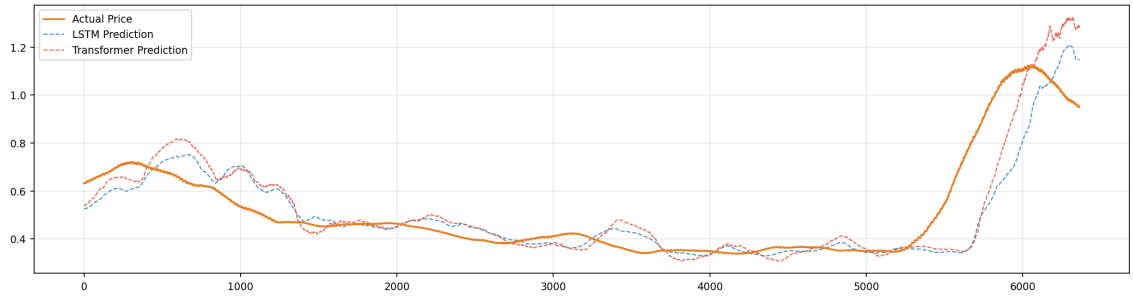
(a) prediction horizon=1 day



(b) prediction horizon=7 days

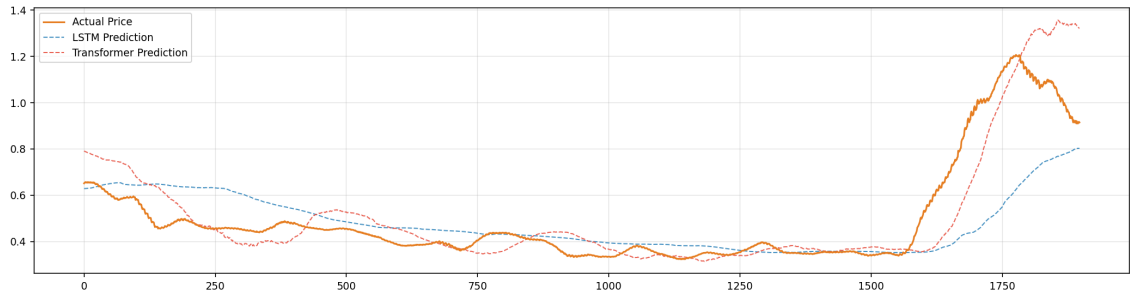


(c) prediction horizon=15 days

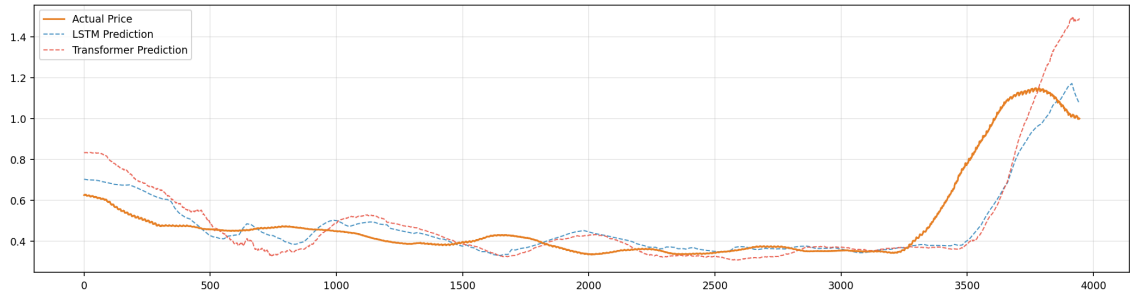


(d) prediction horizon=21 days

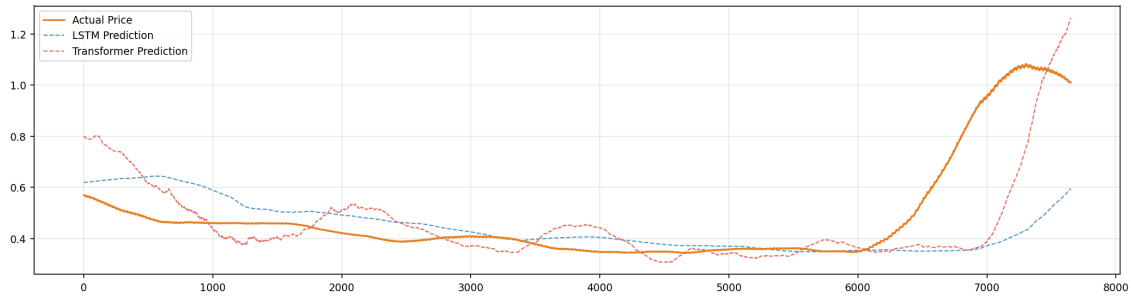
Figure 4.2: LSTM and Transformer prediction results comparison with input sequence length of 21 days for different prediction horizons



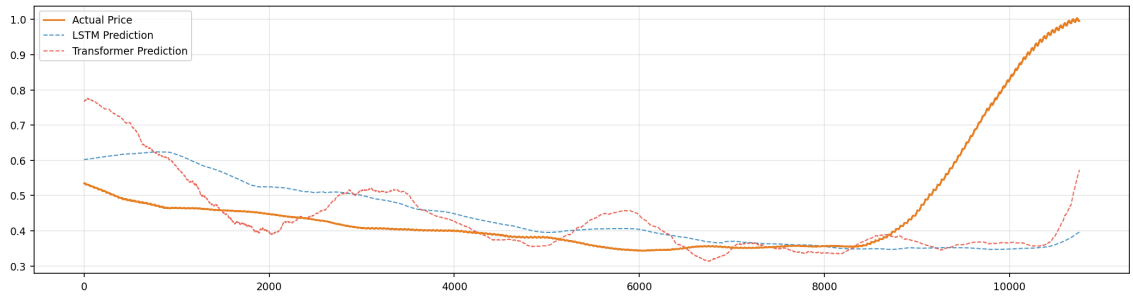
(a) prediction horizon=7 days



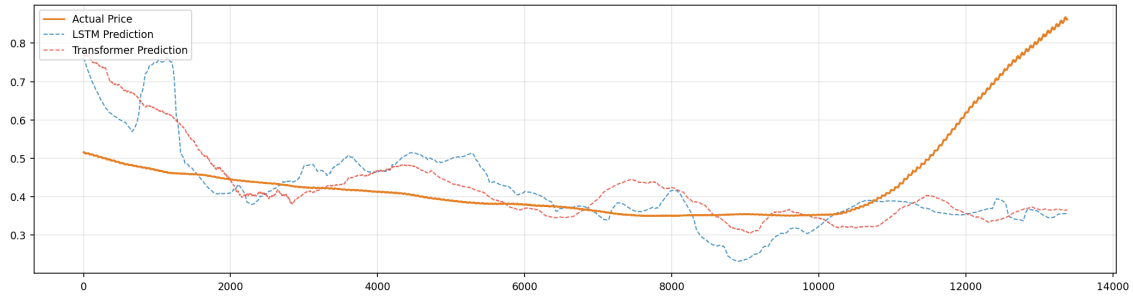
(b) prediction horizon=15 days



(c) prediction horizon=30 days



(d) prediction horizon=45 days



(e) prediction horizon=60 days

Figure 4.3: LSTM and Transformer prediction results comparison with input sequence length of 60 days for different prediction horizons

who shows that excessively long windows introduce noise and reduce forecasting model's accuracy when forecasting on cryptocurrency, our findings indicate that a moderate window (i.e. 21-day) provides the best balance between information content and model performance. In particular, setting the input sequence length to 21 days (suggested by our hyperparameter optimization process) enabled both the LSTM and Transformer models to achieve their lowest forecasting errors. Compared with experiments using a 60-day window, the 21-day configuration consistently yielded lower errors for both models. This outcome suggests that a 21-day look-back horizon is sufficiently long to capture temporal dependencies relevant for cryptocurrency dynamics. Additionally, as the forecasting horizon increases (15- and 21-day predictions), the performance gap between the Transformer and LSTM narrows, reflecting the Transformer's growing advantage in longer-term forecasting. This pattern is consistent with the findings of Kişmiroğlu and İşik[134], who report that LSTM models perform better for short-term predictions (1-week horizon), whereas Transformer models become superior as the prediction horizon extends to 6 months.

Second, when using a longer input length (input sequence length=60 days), the Transformer outperforms the LSTM across most of the forecasting horizons (1-, 30-, 45- and 60-day). In contrast, when the input length is short (input sequence length=21 days), the Transformer's advantage diminishes. This behavior is consistent with the findings of Wu et al.[36], who indicate that recurrent models (LSTM, GRU) are effective for short-term dependencies, but their ability degrades when the input sequence becomes long. In contrast, attention-based models benefit from long input sequences and capture long-range dependencies more effectively. Third, as the prediction horizon increases, both models exhibit marked degradation in performance metrics. In particular, at horizons of 45 and 60 steps, the RRSE values exceed 1.0, indicating that neither model is outperforming a naïve baseline of forecasting the mean. This phenomenon indicates that as forecasting horizons increase, the prediction accuracy decreases due to accumulation of errors and increased un-

certainty[222]. This challenge is amplified in cryptocurrency markets, where daily price fluctuates 5% and 10% occurs, and the price movement can be affected by macroeconomic trends and technological updates[223]. Consequently, models that rely solely on historical price trajectories become increasingly inadequate at longer horizons, underscoring the importance of incorporating external information and advanced modeling techniques to capture the complex dynamics of these markets.

These findings provide a direct response to Sub-RQ1 and H1. Our experiments show that the Transformer does not consistently outperform the LSTM. Instead, its advantage emerges primarily when longer input sequences are provided, enabling it to leverage self-attention to capture distant dependencies. In contrast, with shorter input sequences, LSTM appears more robust.

Furthermore, our results validate the outcomes of the hyperparameter optimization process, which indicated that an input sequence length of 21 days yields the strongest performance for both models. This finding suggests that in cryptocurrency forecasting, the recent historical data carry the highest predictive value, whereas extending the sequence length beyond this point seems introduces additional noise or irrelevant information. Under such conditions, the Transformer does not surpass the LSTM in capturing short-term dependencies. This observation raises an important path for future research: how might Transformer architectures be modified or augmented to better capture local temporal dynamics, thereby narrowing the gap with the recurrent model LSTM in short-sequence settings? The question is answered in the subsequent sections in this chapter.

4.3 Patching

4.3.1 Single-scale patching vs Multi-scale patching

Recent advances in Transformer-based time series models [41][39][44][40] have introduced the concept of *patching*, which aims to enhance the model's ability to represent local temporal patterns. The concept was illustrated in Chapter 2.4. In-

stead of treating each time step as a single token, the patching approach groups consecutive time steps into compact segments and encodes each segment as a single token. By bundling time steps, this approach not only captures local contextual information but also reduces computational complexity from $\mathcal{O}(L^2)$ to $\mathcal{O}((L/P)^2)$ where P is the patch size[224].

Among patch-based Transformer architectures, Crossformer[39] and MSGNet[41] employ multi-scale patching mechanisms to capture temporal dependencies across different time horizons. By aggregating information extracted from multiple temporal resolutions, these models can simultaneously attend to both fine-grained and coarse-grained temporal structures, thereby enhancing their forecasting capability.

To evaluate whether patching can improve the canonical Transformer's performance in forecasting future ADA prices, we developed a patch-based Transformer model. Two additional hyperparameters were introduced: *scale_list* and *top_k*. The *scale_list* defines a set of patch lengths, representing different temporal scales. Before encoding the input time series, the raw time steps are first grouped into patches according to the scales specified in *scale_list*; each patch is then treated as an input token to the Transformer model. The *top_k* parameter determines how many scales from the *scale_list* are used in the experiment. When $top_k = 1$, the model is a single-scale patching Transformer. When $top_k > 1$, the model operates in a multi-scale setting. In the multi-scale configuration, the Transformer independently generates predictions for each selected patch scale. These predictions are subsequently combined by a trainable aggregation layer that dynamically adjusts the relative importance of each temporal scale in forming the final output. Figure 4.4 illustrates the overall architecture of the proposed Transformer model incorporating the patching mechanism.

Experimental results and findings

The prediction performance of single-patch, multi-patch, the canonical Transformer architectures, and the LSTM baseline across multiple forecasting horizons is pre-

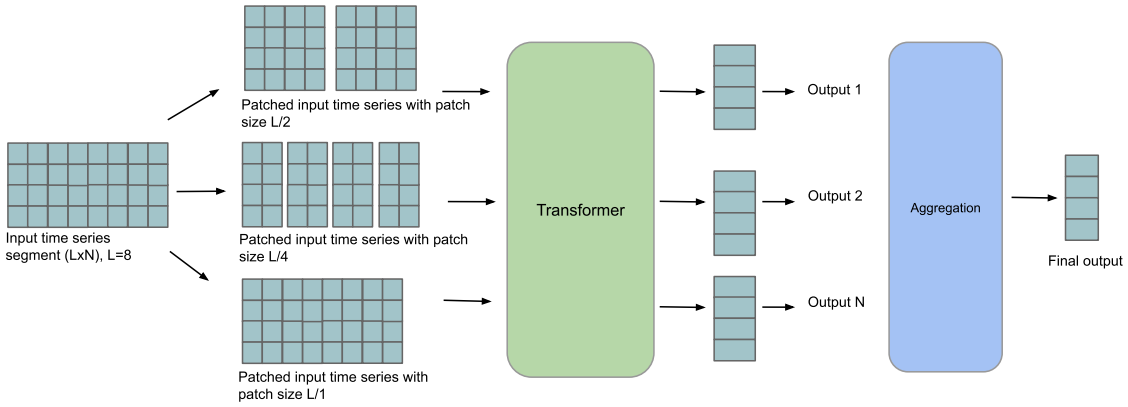


Figure 4.4: Proposed multi-scale patching architecture (CD). A historical time series window of length L ($L = 8$ in the example) and N ($N = 4$ in the example) features is transformed into multiple patch representations at different scales, processed in a CD way through the Transformer, and aggregated into the final forecast.

sented in Table 4.5. To conduct a fair comparison, we set the input sequence length to 21, which is identical to the previous LSTM vs Transformer experiment. From the table, it can be observed that, at the shortest forecasting horizon (1-day), the LSTM achieves the best overall performance across all metrics, indicating its strong ability to capture short-term temporal dependencies. The single-patch Transformer performs competitively, slightly underperforming the LSTM but outperforming the canonical Transformer, suggesting that patching contributes to improved feature representation even over very short time frames.

As the forecasting horizon increases, the relative performance ranking changes. For the 7-day horizon, the single-patch model maintains the lowest MSE, MAE and RMSE, outperforming both the canonical Transformer and the LSTM, although LSTM remains superior in relative error metrics (RSE, MAPE). This indicates that patching helps stabilize Transformer predictions over moderately extended horizons.

Beyond the 15- and 21-day horizons, the single-patch Transformer achieves superior forecasting performance, outperforming the benchmark LSTM across all evaluation metrics. In contrast, the multi-scale patching variant performs better than the LSTM but still falls short of the single-patch Transformer. This result suggests that, under our multi-scale patching configuration, certain patch scales may be introducing redundant features that do not translate into improved forecast accu-

Horizon	Model	MSE	RMSE	MAE	RRSE	MAPE
1	Single-scale patching Transformer	0.00210$^{(*)}$	0.04583$^{(*)}$	0.03250	0.62350	0.06945
	Multi-scale patching Transformer	0.00241	0.04909	0.03147$^{(*)}$	0.47003$^{(*)}$	0.05579$^{(*)}$
	Transformer	0.00402	0.06345	0.04384	0.53432	0.08025
	LSTM	**0.00196**	**0.04429**	**0.02844**	**0.44677**	**0.05134**
7	Single-scale patching Transformer	**0.00311**	**0.05578**	**0.04140**	0.68843	0.08917
	Multi-scale patching Transformer	0.00586	0.07651	0.04837	0.60356$^{(*)}$	0.08502$^{(*)}$
	Transformer	0.00882	0.09395	0.06167	0.65383	0.10846
	LSTM	0.00466$^{(*)}$	0.06826$^{(*)}$	0.04412$^{(*)}$	**0.55758**	**0.07911**
15	Single-scale patching Transformer	**0.00424**	**0.06513**	**0.04771**	0.74723	**0.10181**
	Multi-scale patching Transformer	0.01254$^{(*)}$	0.11200$^{(*)}$	0.06767$^{(*)}$	**0.71927**	0.11344$^{(*)}$
	Transformer	0.01370	0.11705	0.07673	0.73539	0.13802
	LSTM	0.01332	0.11539	0.06927	0.73027$^{(*)}$	0.11752
21	Single-scale patching Transformer	**0.00533**	**0.07300**	**0.05270**	**0.79486**	**0.11260**
	Multi-scale patching Transformer	0.01985$^{(*)}$	0.14095$^{(*)}$	0.08192	0.80061$^{(*)}$	0.13194
	Transformer	0.02081	0.14427	0.08914	0.81019	0.15212
	LSTM	0.02026	0.14229	0.08149$^{(*)}$	0.80474	0.13055$^{(*)}$

Table 4.5: Prediction performance comparison among single-scale patching Transformer, multi-scale patching Transformer, canonical Transformer, as well as the LSTM baseline, using an input sequence length of 21. The best results for each prediction horizon are highlighted in bold, the second best results are marked with the superscript $^{(*)}$

racy. Similar issues have been documented in the literature [225], where multi-scale methods are shown to suffer from redundant features[1] that can jeopardize prediction performance. To examine whether dynamically adjusting patch scales using FFT could mitigate this issue, we conducted an additional experiment comparing the performance of the multi-scale patching Transformer with and without FFT-based adaptive patch-scale selection.

Overall, the findings indicate that patching enhances the canonical Transformer's

[1]According to Liu et al.[225], a feature is redundant if it is useful for prediction but correlated with another useful feature in the data.

predictive accuracy across all forecasting horizons, particularly for the longer horizons of 7, 15, and 21 days. The LSTM remains a strong baseline for short-term forecasting, while the single-patch Transformer demonstrates greater robustness and an improved ability to capture richer temporal dynamics over extended horizons. However, the effectiveness of the multi-scale patching configuration appears limited. To investigate whether the constrained performance of the multi-scale patching Transformer arises from the empirically selected patch scales, we conducted an additional experiment in which patch sizes were determined dynamically using the FFT. The FFT enables the model to identify dominant frequency components within the dataset, allowing patch scales to be adaptively defined in accordance with the underlying spectral characteristics of the time series.

4.3.2 FFT-based patching scale selection

FFT has been widely applied in the TSF domain to identify periodic patterns. In the patching approach, MSGNet [41] leverages FFT to align the patch sizes with the underlying frequency characteristics, enabling the model to perform multi-scale patching that reflects the underlying frequency characteristics of the signal.

Building on this idea, we extend our multi-scale patching Transformer by integrating FFT analysis to determine patch sizes from the frequency spectrum of each input sequence dynamically. This adaptive mechanism allows the model to adjust its patching scales in response to the dominant temporal frequencies present in the data, thereby improving its ability to capture seasonal patterns across different patching scales.

Experimental results and findings

From Table 4.6, it can be observed that, when the prediction horizon is 1-day, the FFT-based approach does not yet surpass the empirically defined multi-scale patching model across all evaluation metrics. However, at the 7-day horizon, the FFT-enhanced Transformer begins to outperform the fixed multi-scale variant, indicating

that frequency-guided patch selection becomes more effective as the forecast window lengthens. When the horizon is further extended to 15 and 21 days, the FFT determined patching remains the second best performer, trailing only the single-patch Transformer while consistently outperforming the baseline LSTM. These results suggest that adapting patch scales to the dominant frequency components of the input series helps reduce the noise introduced by manually chosen patch sizes. However, the method can not fully eliminate the redundant features problem. The single-scale patching Transformer still achieves the best overall performance, indicating that a fixed scale representation remains the most stable and effective configuration in this forecasting task.

From the above experiments, it can be concluded that patching is an effective technique for enhancing the Transformer model in our forecasting task, leading to improved forecasting accuracy across all forecasting horizons and leading the performance at longer horizons (7, 15, and 21 days). This finding contributes to the central research question by answering the Sub-RQ2 "How do different patching strategies influence the predictive performance of the canonical Transformer model in multivariate cryptocurrency price forecasting?" and addressing hypotheses H2.1, H2.2. and H2.3.

1. Patching (H2.1): From Table 4.5, it can be observed that applying single-scale patching can substantially increase the Transformer forecasting performance on the ADA price prediction task. By grouping consecutive time steps into patches prior to encoding, the model achieves a dual benefit: reduced computational complexity by shortening the input sequence length and enhanced prediction accuracy by modeling local temporal information.

2. Multi-scale patching (H2.2): Table 4.5 reveals that when inter-channel information is explicitly modeled under CD setting, multi-scale patching may introduce additional cross-channel noise compared to the single-scale configuration. This may suggest that the combination of multi-scale patching and CD

[2]Multi-scale patching Transformer with with FFT selected patch scales

Horizon	Model	MSE	RMSE	MAE	RRSE	MAPE
1	Single-scale patching Transformer	0.00210(*)	0.04583(*)	0.03250	0.62350	0.06945
	Multi-scale patching Transformer	0.00241	0.04909	0.03147(*)	0.47003(*)	0.05579(*)
	Multi-scale patching with FFT Transformer [2]	0.00256	0.05059	0.03312	0.47758	0.05918
	LSTM	**0.00196**	**0.04429**	**0.02844**	**0.44677**	**0.05134**
7	Single-scale patching Transformer	**0.00311**	**0.05578**	**0.04140**	0.68843	0.08917
	Multi-scale patching Transformer	0.00586	0.07651	0.04837	0.60356	0.08502
	Multi-scale patching with FFT Transformer	0.00549	0.07411	0.04609	0.59373(*)	0.08230(*)
	LSTM	0.00466(*)	0.06826(*)	0.04412(*)	**0.55758**	**0.07911**
15	Single-scale patching Transformer	**0.00424**	**0.06513**	**0.04771**	0.74723	**0.10181**
	Multi-scale patching Transformer	0.01254	0.11200	0.06767	0.71927(*)	0.11344
	Multi-scale patching with FFT Transformer	0.01246(*)	0.11163(*)	0.06697(*)	**0.71821**	0.11221(*)
	LSTM	0.01332	0.11539	0.06927	0.73027	0.11752
21	Single-scale patching Transformer	**0.00533**	**0.07300**	**0.05270**	0.79486(*)	**0.11260**
	Multi-scale patching Transformer	0.01985	0.14095	0.08192	0.80061	0.13194
	Multi-scale patching with FFT Transformer	0.01711(*)	0.13084(*)	0.07840(*)	**0.77144**	0.13016(*)
	LSTM	0.02026	0.14229	0.08149	0.80474	0.13055

Table 4.6: Prediction performance comparison among Transformer-based models with single-scale, multi-scale patching, multi-scale patching with FFT selected patch scales and the baseline LSTM, using an input sequence length of 21. The best results for each prediction horizon are highlighted in bold, the second best results are marked with the superscript (*)

attention can amplify spurious correlations among assets, leading to less stable representations. The experiments in the following chapter empirically support this assumption by comparing the performance of the multi-scale patching Transformer against its single-scale patching variant under a CI configuration.

3. FFT-based multi-scale patching (H2.3): Table 4.5 shows that under the CD configuration, the FFT-based multi-scale patching approach consistently out-

performs the empirically selected multi-scale patching variant for the 7-, 15-, 21- day forecasting horizons. This finding suggests that when cross channel information is modeled, adapting patch sizes to dominant frequency components allow the Transformer to capture seasonal dependencies shared across the correlated cryptocurrencies. The adaptive frequency selection appears beneficial because the resulting patch sizes align with the dominant frequencies that are shared across multiple dominant cryptocurrencies.

Although the best-performed patching Transformer, the single-patch Transformer, outperforms the baseline LSTM in longer forecasting (7-, 15-, and 21-day), its advantage diminishes when the prediction horizon is short (1-day). To further investigate potential improvements to the Transformer's forecasting capability, we subsequently conducted a comparative study between CD and CI modeling approaches.

4.4 Channel-Dependent (CD) vs Channel-Independent (CI)

Our data analysis in Section 3.2.3 reveals that the cryptocurrency data is inter-correlated. Motivated by this empirical evidence, the models developed in the preceding experiments adopt a CD architecture to exploit potential inter-channel interactions among the major cryptocurrencies directly.

However, prior research suggests that CI modeling can often be more robust. PatchTST [40], evaluated on the ETT (electric transformer temperature), electricity, traffic, and weather benchmarks [36], reports that CI architectures consistently provide better generalization, faster convergence, and reduced overfitting compared to CD models. Similarly, MC-Former [139], tested on common multivariate forecasting datasets such as ETT, electricity, weather [36], solar energy [226], and traffic [146], also finds that CI modeling yields improved stability and predictive performance.

Additionally, Abdelmalak et al. [227] demonstrated that, when datasets lack

strong inter-channel coupling, CI models frequently outperform CD ones. In the context of cryptocurrency markets, Wątorek et al. [228] further showed that cross-asset correlations fluctuate across market regimes, e.g. bubbles or crashes. This implies that inter-channel relationships learned in one period may become irrelevant or even noisy in another. Motivated by these findings, we therefore developed CI variants of our models to assess if the CI approach can further improve Transformer's forecasting performance across all the forecasting horizons. Figure 4.5 illustrates the workflow of the CI configuration of our architecture. In contrast to the CD setup, where all features (i.e., all cryptocurrency price series) are concatenated and jointly processed by a single Transformer, the CI configuration treats each channel as an independent univariate time series. Specifically, each channel is patched separately and passed through the Transformer encoder, ensuring that temporal patterns are learned independently for each asset. This means that the model focuses solely on the temporal dynamics within a single channel, without mixing information across different assets during the encoding stage.

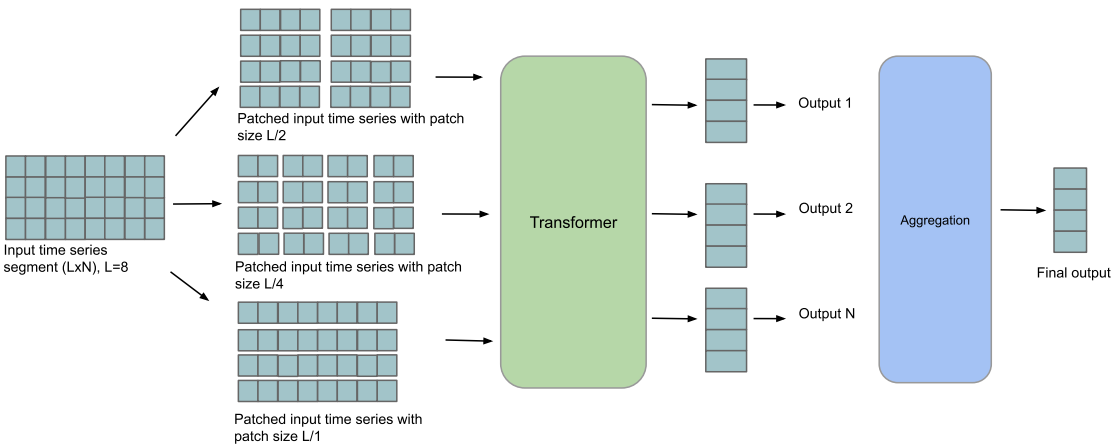


Figure 4.5: Proposed multi-scale patching architecture (CI). A historical time series window of length L ($L = 8$ in the example) and N features ($N = 4$ in the example) is transformed into multiple patch representations at different scales, processed in a CI way through the Transformer, and aggregated into the final forecast.

4.4.1 Experimental results and findings

Table 4.7 presents a comparative performance result of CI and CD version of our Transformer variants' and the benchmark LSTM's performance across four prediction horizons (1, 7, 15, 21 days). To be consistent with the previous experiments, the input sequence length is set to 21. The results show that the CI version of multi-scale patching Transformer consistently achieves the best overall performance across almost all metrics, except for the RRSE, where the LSTM baseline remains slightly superior. Additionally, comparing with the CI version of single-scale patching Transformer, these results validate our hypothesis (H2.2) that incorporating multiple temporal scales enables the Transformer to capture local dependencies of varying lengths. By learning from multiple resolutions simultaneously, the model develops a richer temporal hierarchy that translates into improved predictive accuracy. In the table, the CI verions of single-scale patching Transformer ranks second in performance. Across all prediction horizons, the CI variants of the Transformer architectures outperform their CD counterparts, suggesting that explicit cross channel interactions do not necessarily benefit forecasting accuracy in this context. This observation validates the hypothesis H3.2, and aligns with recent findings [227][228] that channel independence can yield better generalization when inter-asset correlations are weak or time-varying, as is often the case in cryptocurrency markets. Furthermore, the FFT rows in Table 4.7 reveal that, under the CI configuration, the FFT-based multi-scale patching approach underperforms comparing with the empirically determined multi-scale patch size variant. This result contradicts the findings in Section 4.3.2, where the FFT approach outperfroms the multi-scale patching variant under the CD configuration. This difference in outcome might stem from the specific manner in which the FFT technique is integrated into the modeling process. Under the CI setting, each asset is modeled independently, however the patch sizes are defined by the same FFT function that aggregate spectral information across all channels. While this is consistent with the CD setting, it become misaligned with the CI configuration, where each channel is modeled independently. As a result,

the patch sizes determined from global frequency aggregation may not reflect the dominant temporal structures of individual assets, thereby reducing performance in the CI model.

Notably, the best performing multi-scale CI Transformer achieves substantial improvements over the LSTM baseline, reducing MSE by approximately 25–40% and MAE by 20–30%, depending on the prediction horizon. These results confirm that patching-based Transformer architectures can effectively model non-stationarity in cryptocurrency time series, and CI enhances their robustness and generalization across different market regimes.

The empirical results from above experiments have answered the Sub-RQ3: "How do CI and CD multivariate processing strategies affect the forecasting performance of Transformer-based architectures?" The results demonstrate that among the proposed architectural modifications, multi-scale patching and CI can enhance the Transformer's capacity to model the non-stationary and regime dependent nature of cryptocurrency markets.

The hypotheses H3.1 and H3.2 have been systematically examined and validated as follows:

1. CI vs CD processing (H3.1 and H3.2): The results of the comparative experiments between the CI and CD versions of our patching Transformer models, as reported in Table 4.7, demonstrate that the CI approach consistently outperforms the CD approach. This finding suggests that explicitly modeling inter-cryptocurrency correlations can introduce additional noise and instability. The superior results of the CI models imply that independently learning temporal dynamics can lead to better understanding of asset specific patterns and better forecasting results. These observations are consistent with a prior study [228], which reported that correlations among major cryptocurrencies are time-varying and regime-dependent. Under such environments, the CD modeling approach may lead the Transformer to overfit to transient inter-asset

[3]Multi-scale patching Transformer with with FFT selected patch scales

Horizon	Model	Version	MSE	RMSE	MAE	RRSE	MAPE
1	Single-scale patching Transformer	CI	0.00039(*)	0.01975(*)	0.01367(*)	0.40883(*)	0.02955(*)
		CD	0.00210	0.04583	0.03250	0.62350	0.06945
	Multi-scale patching Transformer	CI	**0.00035**	**0.01871**	**0.01309**	**0.39813**	**0.02813**
		CD	0.00241	0.04909	0.03147	0.47003	0.05579
	Multi-scale patching with FFT Transformer [3]	CI	0.00088	0.02966	0.01984	0.50235	0.04279
		CD	0.00256	0.05059	0.03312	0.47758	0.05918
	LSTM	–	0.00196	0.04429	0.02844	0.44677	0.05134
7	Single-scale patching Transformer	CI	0.00144(*)	0.03795(*)	0.02651(*)	0.56811	0.05804(*)
		CD	0.00311	0.05578	0.04140	0.68843	0.08917
	Multi-scale patching Transformer	CI	**0.00142**	**0.03768**	**0.02610**	0.56581(*)	**0.05704**
		CD	0.00586	0.07651	0.04837	0.60356	0.08502
	Multi-scale patching with FFT Transformer	CI	0.00249	0.04990	0.03606	0.65136	0.07817
		CD	0.00549	0.07411	0.04609	0.59373	0.08230
	LSTM	–	0.00466	0.06826	0.04412	**0.55758**	0.07911
15	Single-scale patching Transformer	CI	0.00239(*)	0.04888(*)	0.03468(*)	0.64722(*)	0.07570(*)
		CD	0.00424	0.06513	0.04771	0.74723	0.10181
	Multi-scale patching Transformer	CI	**0.00236**	**0.04858**	**0.03439**	**0.64541**	**0.07505**
		CD	0.01254	0.11200	0.06767	0.71927	0.11344
	Multi-scale patching with FFT Transformer	CI	0.00372	0.06100	0.04467	0.72312	0.09540
		CD	0.01246	0.11163	0.06697	0.71821	0.11221
	LSTM	–	0.01332	0.11539	0.06927	0.73027	0.11752
21	Single-scale patching Transformer	CI	0.00332(*)	0.05763(*)	0.04069(*)	0.70582(*)	0.08830(*)
		CD	0.00533	0.07300	0.05270	0.79486	0.11260
	Multi-scale patching Transformer	CI	**0.00330**	**0.05745**	**0.04039**	**0.70473**	**0.08776**
		CD	0.01985	0.14095	0.08192	0.80061	0.13194
	Multi-scale patching with FFT Transformer	CI	0.00486	0.06974	0.05012	0.77639	0.10711
		CD	0.01711	0.13084	0.07840	0.77144	0.13016
	LSTM	–	0.02026	0.14229	0.08149	0.80474	0.13055

Table 4.7: Prediction performance comparison among CI and CD version of Transformer-based models with single-scale, multi-scale patching, multi-scale patching with FFT selected patch sizes, as well as the LSTM baseline, using an input sequence length of 21 days. The best results for each prediction horizon are highlighted in bold, the second best result are marked with the superscript (*)

relationships, capturing co-movements specific to one market regime (e.g. bubble time) that fail to generalize to others (e.g. crash time). Conversely, CI modeling mitigates this issue by decoupling channel representations while still sharing weights across assets, enabling robust learning of intrinsic temporal structures within each cryptocurrency.

Based on the preceding experiments, two architectural strategies were identified as consistently improving Transformer performance: multi-scale patching and CI processing. These findings motivate the initial design of our proposed architecture

showing in Figure 4.5: a multi-scale patching (MSP) Transformer with CI processing. From this point onward, we refer to this architecture as **MSPFormer**. Within the architecture, the input data $x_{1:t} \in R^{t \times N}$ is first divided into multiple temporal patches. In this way, each token corresponds to a segment of observations that encapsulate local temporal dynamics instead of a single time step. By adopting a multi-scale patching strategy, the model constructs representations at different temporal resolutions enabling it to capture both short- and long- range dependencies within the time series. Following the patching, the data is then processed by the Transformer model in a CI way. Instead of modeling the inter-channel dependencies, the data channels are modeled independently and the weights in the Transformer layer is shared. For each patching scale, the Transformer produces an independent forecast representation. Finally, a learnable fusion layer adaptively aggregates these scale-specific outputs, assigning dynamic importance weights to each temporal scale to generate the final prediction.

4.5 Retrieval-Augmented Context

As highlighted in the literature review, several recent studies[55][57][64] in TSF have explored the use of Retrieval-Augmentation(RA) techniques to build contextual information provider to further improve model performance. Motivated by these developments, we designed a model-agnostic RAC module that functions as a plug-in layer and can be integrated into the SOTA Transformer-based TSF models including Informer[38], Autoformer[36], FedFormer[32], PatchTST[40], MSGNet[41], TimeXer[42], and iTransformer[44] without altering its core structure. To assess its effectiveness in the cryptocurrency domain, we applied this RAC layer to MSPFormer and conducted a comparative evaluation between the baseline, MSPFormer, and its RAC-enhanced counterpart on the same forecasting task.

Furthermore, to assess the general applicability of RAC beyond a single architecture, we extended the evaluation to include several SOTA Transformer-based models including TimeXer[42], iTransformer[44], MSGNet[41] and PatchTST[40]. This

broader investigation allows us to determine whether the integration of retrieval-based contextual information yields consistent performance gains across diverse Transformer architectures for our ADA price forecasting task. To the best of our knowledge, this work represents one of the first attempts to operationalize a RAC framework for cryptocurrency price forecasting.

Figure 4.6 illustrates the RAC approach and compares the training and inference processes of models with and without RAC integration, showing how retrieval-augmented information enriches the contextual input used for forecasting. In the RAC-based approach, a dedicated database containing historical cryptocurrency price sequences is first constructed. The database is built exclusively with training data and is maintained as an in-memory dictionary that supports efficient similarity retrieval. Limiting the repository to training data ensures that no information from validation or test sets is introduced into the retrieval process, thereby preventing data leakage and preserving the integrity of out-of-sample evaluation. To facilitate reproducibility and reuse, the database can be serialized as a pickle file[4] and reloaded when needed to retrieve analogous historical patterns. Before the actual forecasting, the system identifies a similar historical sequence via $L2$ distance and prepends it to the original sequence. FAISS[5] is utilized to construct an efficient index and perform retrieval based on the $L2$ metric. During retrieval, FAISS produces a list of candidate sequences ordered by increasing L_2 distance to the query. A temporal constraint is subsequently enforced to discard candidates whose indices succeed that of the input sequence, preventing data leakage. The top-k remaining sequences are then selected as auxiliary context and concatenated to the input prior to prediction. In this work, the value k=1 was determined via grid search before training.

[4]A pickle file is a binary serialization format provided by Python's pickle module that encodes in-memory objects into a byte stream for persistent storage. `https://docs.python.org/3/library/pickle.html`

[5]FAISS (Facebook AI Similarity Search) is a high-performance open source library for similarity search over embeddings of multimedia documents that are similar to each other `https://ai.meta.com/tools/faiss/`

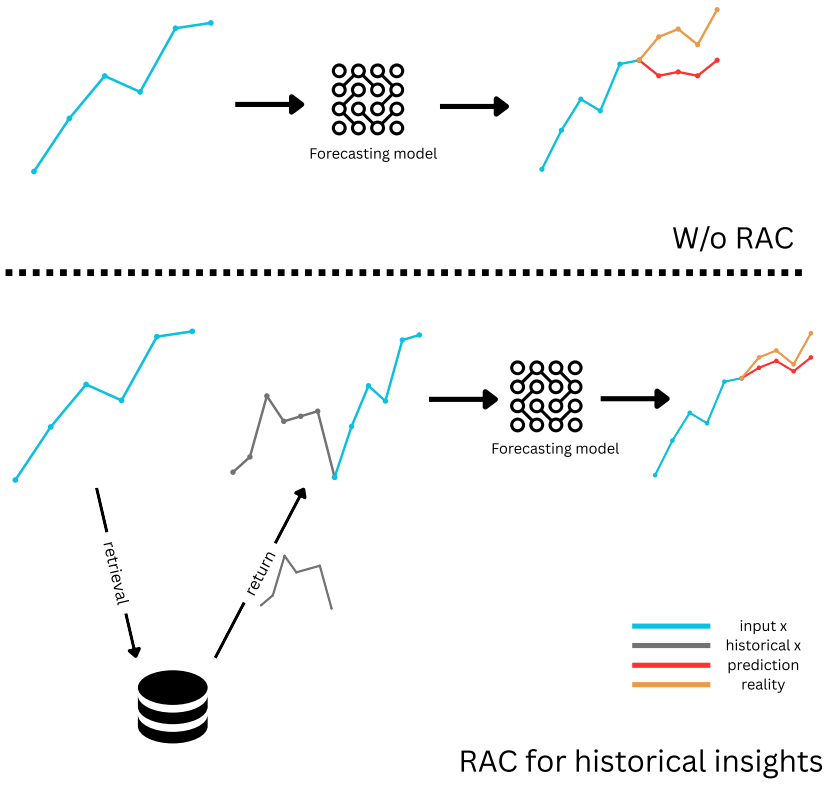


Figure 4.6: Comparison of time series forecasting before and after RAC. It can be observed that an informative historical sequence is retrieved and concatenated with the input sequence prior to its processing by the forecasting model. The augmented input sequence is hypothesized to enable the forecasting model to yield more accurate predictions.

Algorithm 2 further details the procedure for RAC database construction and context retrieval. Consider a training sequence $\mathcal{D}_{\text{train}} = [t_1, t_2, \ldots, t_{10}]$ with chunk size $m = 2$. Following the procedure *BuildRACDatabase*, the resulting database comprises three components: the set of data chunks $\texttt{Chunks}(D_{\text{train}}, m)$, which in this example yields $[[t_1, t_2], [t_3, t_4], \ldots, [t_9, t_{10}]]$; metadata associated with each chunk (e.g. start index in the original raw data); and a FAISS index constructed over the chunk representations to support efficient similarity search. During the retrieval, suppose the input sequence is $X = [t_5, t_6]$. According to the procedure *RetrieveContext*, FAISS is first used to identify candidate chunks in the database that are most

similar to X. The retrieved chunk indices are then filtered to retain only those that precede the temporal position of X, preventing look-ahead bias. From the filtered candidates, the top-k most similar chunks are selected to provide contextual information for forecasting. In this study, $k = 1$ was chosen based on extensive grid search experiments. If the selected chunk is $[t_1, t_2]$, it is prepended to the input sequence prior to model inference. Consequently, the final augmented input becomes $[t_1, t_2, t_5, t_6]$.

Algorithm 2 RAC: database building and data retrieval

Require: Training raw data $D_{\text{train}} = \{X^{(i)}\}_{i=1}^{N}$:
Require: Chunk size m

Procedure BUILDRACDATABASE($\mathcal{D}_{\text{train}}$) ▷ Training data only (no leakage)
    Initialize in-memory dictionary/database $\mathcal{B}$;
    Split D_{train} into fixed-length chunks of length m: $\texttt{Chunks}(D_{\text{train}}, m)$
    Add metadata $\mathcal{M}_{\text{meta}}$ for each of the chunks including starting_index, end_index, starting_time_stamp, end_time_stamp
    Build FAISS indexes over $\texttt{Chunks}(D_{\text{train}}, m)$ with L_2 distance: $\mathcal{I} \leftarrow \texttt{FAISSIndex}(\texttt{Chunks}(D_{\text{train}}, m))$
    Store the chunks and the corresponding index metadata in $\mathcal{B}$
    return $\mathcal{B} = (\texttt{Chunks}(D_{\text{train}}, m), \mathcal{M}_{\text{meta}}, \mathcal{I})$

Procedure RETRIEVECONTEXT($X, \mathcal{B}, index$) ▷ Retrieve only past chunks
    Retrieve candidate indices via FAISS: $\texttt{idx} \leftarrow \texttt{Search}(\mathcal{I}, X, index')$ ▷ $index' \geq index$
    Select top-k: $\texttt{idx}_k \leftarrow \texttt{TopK}(\texttt{idx}_{\text{past}}, k)$
    Retrieve chunks: $\{x_i\}_{i \in \texttt{idx}_k} \leftarrow \texttt{Chunks}(\mathcal{D}_{\text{train}}, \updownarrow)[\texttt{idx}_k]$
    Concatenate and prepend retrieved context: $X' \leftarrow \texttt{Concat}(x_i, X)$
    return augmented input X'

This augmentation enriches the model's contextual information by introducing temporally analogous historical patterns that may provide additional clues about forthcoming price dynamics. Behind this design, our hypothesis is that incorporating retrieval-augmented context will improve the model's predictive performance by allowing it to condition on a more informative input sequence with complementary information.

To be consistent with the earlier experimental configuration, the comparative evaluation was initially conducted using an input sequence length of 21 days, which

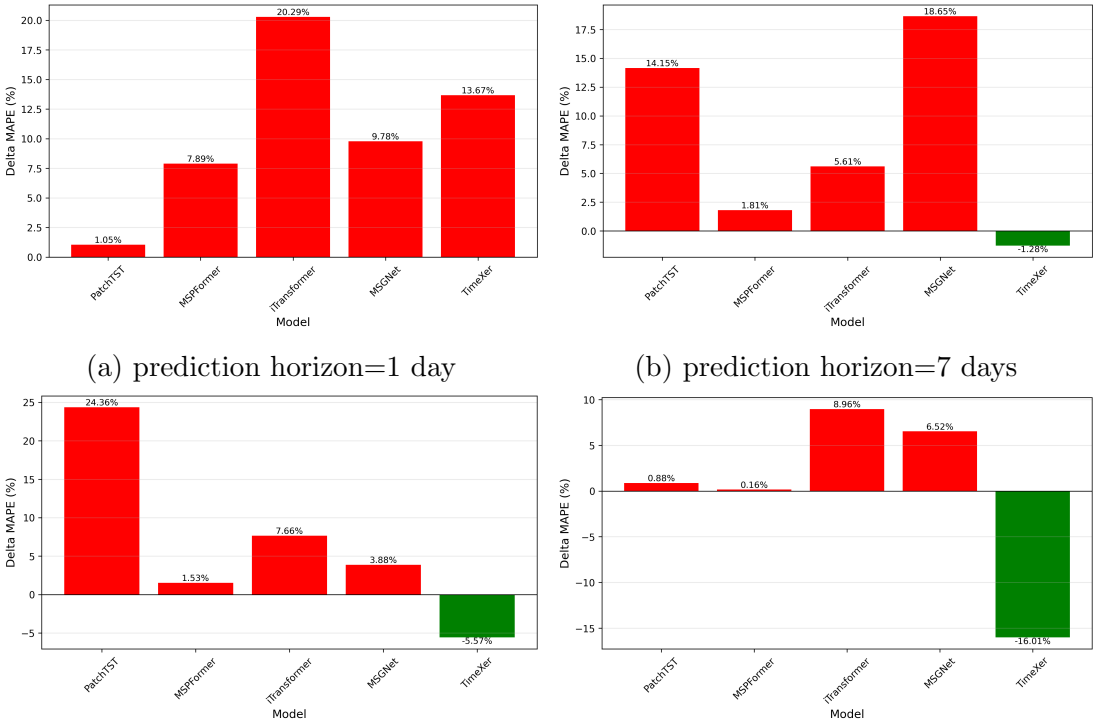
was identified as the optimal hyperparameter setting based on the grid search results. To additionally assess the effectiveness of the proposed RAC module under conditions of limited historical information, a second set of experiments was performed using a shorter input sequence of 7 days. This allows us to examine whether RAC can compensate for data scarcity by enriching the input context, and thereby improving forecasting performance.

4.5.1 Experimental results and findings

Table 4.8 presents the comparison results across different forecasting horizons under the 21-day input sequence setting, while Table 4.9 reports the corresponding results under the 7-day input sequence setting. To clearly view the effect of RAC on model performance, we report the percentage change ($\Delta\%$) in error metrics relative to the no RAC baseline. In this thesis, Δ denotes the change after introducing RAC, where a negative $\Delta\%$ indicates an improvement (lower error) and a positive $\Delta\%$ indicates a degradation in performance. Under the 21-day input sequence setting, Figures 4.7 and 4.8 visualize the $\Delta\%$ change in MAPE and RMSE across forecasting horizons for all models. The equivalent results for the 7-day input sequence setting are shown in Figures 4.9 and 4.10.

The experiments results show that RAC is not universally beneficial. It can be observed that RAC yields the most benefit when the input sequence length is short, forecasting horizon is mid- or long-term, and when applied to architectures capable of integrating external context (e.g., TimeXer). On the other hand, patch-based models (e.g., PatchTST, iTransformer) are more prone to performance degradation.

Under the 21-day input sequence setting, most models exhibit performance degradation after incorporating RAC, with positive $\Delta\%$ values indicating increased error. This effect is particularly evident for PatchTST and iTransformer, where substantial increases in both RMSE and MAPE are observed across multiple horizons. For instance, PatchTST experiences a sharp deterioration at horizon 15 (+93.72% RMSE and +24.36% MAPE), indicating that the retrieved sequences may introduce



(a) prediction horizon=1 day

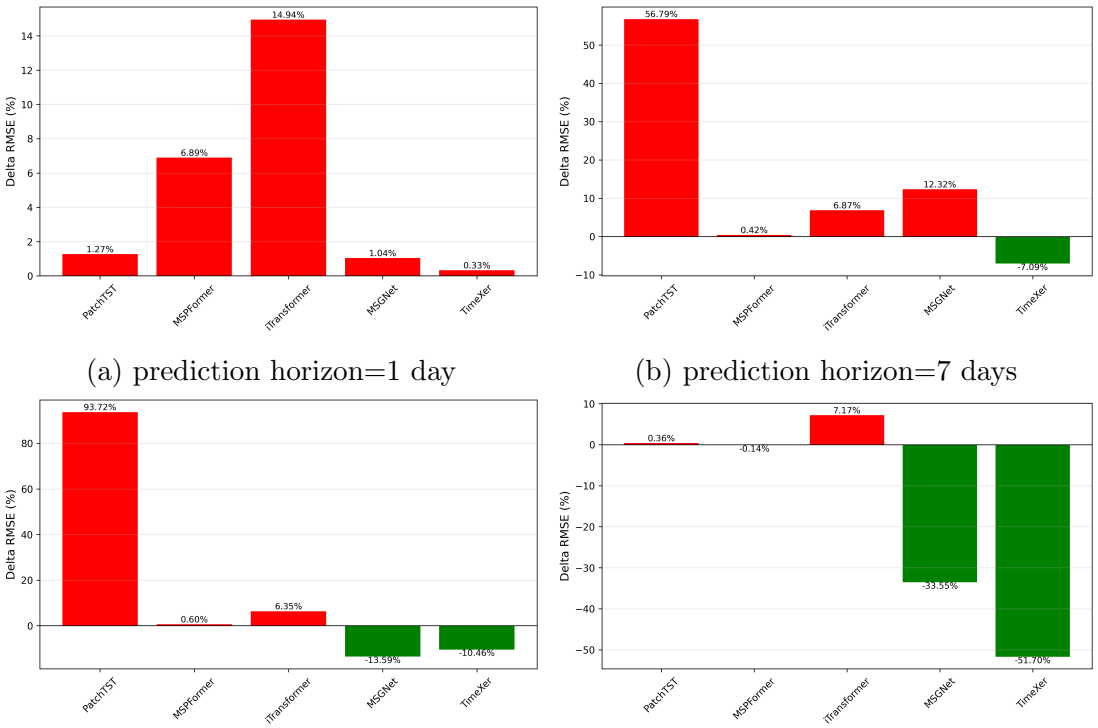
(b) prediction horizon=7 days

(c) prediction horizon=15 days

(d) prediction horizon=21 days

Figure 4.7: Percentage change (Δ%) in MAPE for each model after applying RAC across different forecasting horizons (input sequence length of 21). Negative values indicate improved performance (lower error), while positive values indicate performance degradation relative to the no-RAC baseline.

more noise than useful signal. These results demonstrate that, when the model already has access to a sufficiently long historical input window the incorporation of externally retrieved sequences offers limited additional benefit and may introduce misalignment with the model's learned temporal representations. This observation is consistent with the findings of Zhang et al.[229], who report that the performance gains from retrieval augmentation diminish as the in-window history increases and the model already encodes the necessary temporal structure. Similarly, Cui et al. [55] indicate that when the retrieved segment length grow beyond an optimal point, performance declined because the representation became increasingly mixed and complex. Under the shorter 7-day input setting, the impact of RAC becomes more balanced, and improvements are more frequently observed. This pattern is expected because a short input window provides only limited recent price dynamics, restricting the model's ability to infer broader trends or regime information. By



(a) prediction horizon=1 day

(b) prediction horizon=7 days

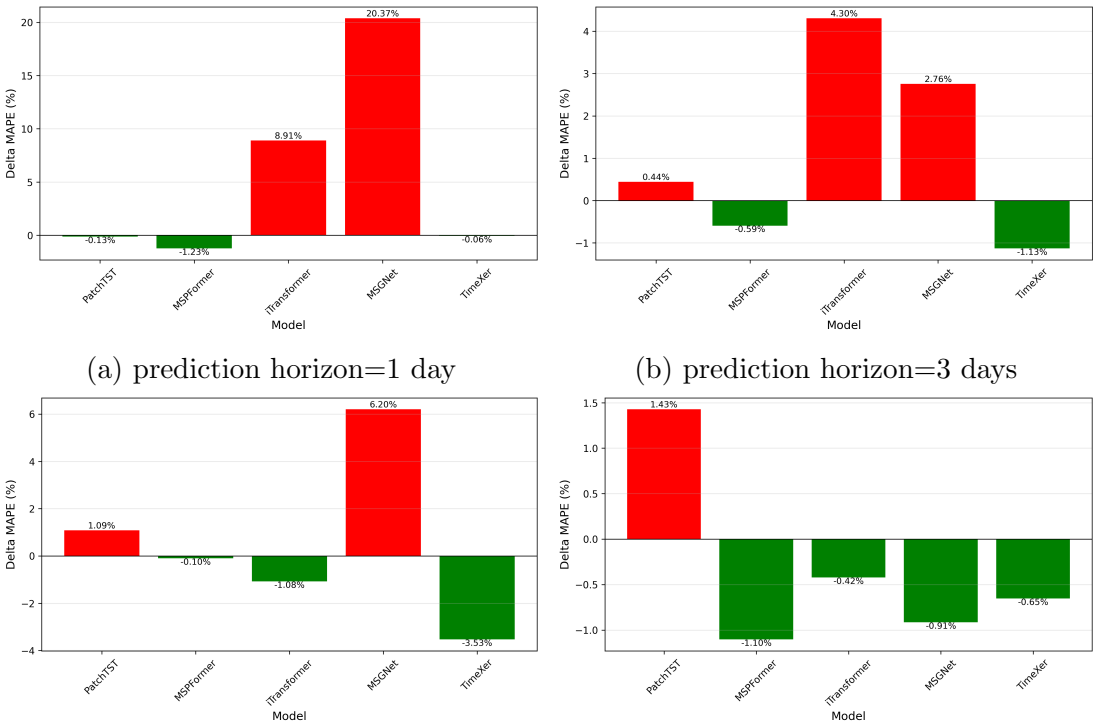
(c) prediction horizon=15 days

(d) prediction horizon=21 days

Figure 4.8: Percentage change (Δ%) in RMSE for each model after applying RAC across different forecasting horizons (input sequence length of 21). Negative values indicate performance improvement, while positive values indicate degradation relative to the no-RAC baseline.

retrieving historically similar sequences, RAC effectively extends the temporal context beyond the fixed 7-day window. In doing so, it supplements the model with longer-horizon structural patterns that would otherwise be unavailable. As a result, RAC partially compensates for the reduced input length, leading to more negative % values, particularly for TimeXer and MSPFormer. This suggests that RAC is especially beneficial when the original input lacks sufficient historical context to capture medium- or long-term dependencies. This finding aligns with the work of Jing et al.[142], who observed that the historical values of a target time series become increasingly unreliable when the input window is short. To address this limitation, they proposed ReTime, a retrieval-based framework that enhances forecasting accuracy by retrieving reference sequences from a database using relational similarity prior to prediction.

Across both input settings, RAC tends to provide relatively more benefit for



(a) prediction horizon=1 day

(b) prediction horizon=3 days

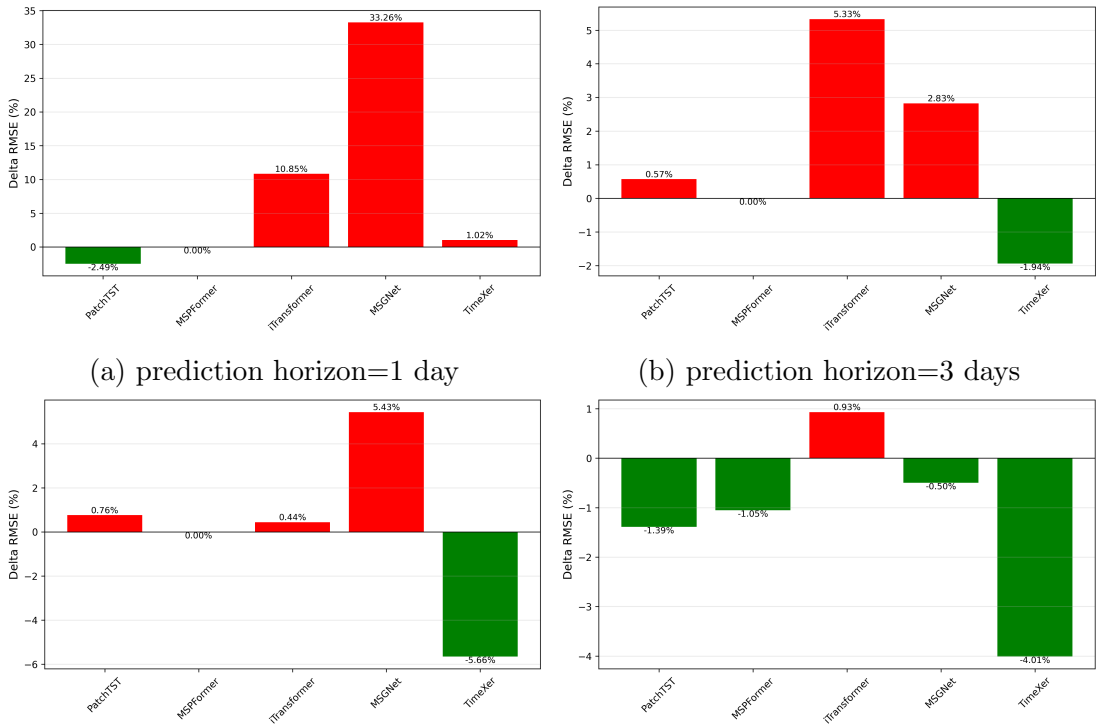
(c) prediction horizon=5 days

(d) prediction horizon=7 days

Figure 4.9: Percentage change (Δ%) in MAPE for each model after applying RAC across different forecasting horizons (input sequence length of 7). Negative values indicate performance improvement, while positive values indicate degradation relative to the no-RAC baseline.

longer prediction horizons. Short-term forecasting (e.g., 1-day horizon) relies more heavily on recent temporal patterns, and retrieved sequences often add limited new information or cause distraction. For mid- to long-horizons (5 days or more), the model increasingly depends on broader temporal context to predict future movement[230], making the retrieved patterns more relevant and beneficial (when model architecture permits effective integration, e.g. TimeXer).

In addition, the degree to which RAC influences performance is closely linked to the architectural design of the forecasting model. Among all models tested, TimeXer benefits the most from RAC, especially under longer forecasting horizons. Unlike the CI (PatchTST and MSPFormer) and CD (iTransformer and MSGNet) architectures in our model group, TimeXer integrates exogenous information through a cross attention mechanism, allowing the model to learn selective dependencies between the endogenous feature (ADA price) and exogenous features (the other cryptocur-



(a) prediction horizon=1 day

(b) prediction horizon=3 days

(c) prediction horizon=5 days

(d) prediction horizon=7 days

Figure 4.10: Percentage change (Δ%) in RMSE for each model after applying RAC across different forecasting horizons (input sequence length of 7). Negative values indicate performance improvement, while positive values indicate degradation relative to the no-RAC baseline.

rency prices in our feature selection). Cross attention enables TimeXer to attend to only the most relevant exogenous signals at each time step, effectively acting as a learned noise filter. As a result, when RAC introduces additional historical patterns, TimeXer is better positioned to extract useful contextual information while suppressing irrelevant or misaligned information, explaining why it benefits more consistently from retrieval-augmented input compared to the other models. The effectiveness of cross attention in integrating information from diverse source by filtering noise and focusing on signals has been demonstrated in prior work by Xi et al.[231].

In contrast, PatchTST and iTransformer consistently underperform with RAC. PatchTST relies on fixed length patch embeddings to model local temporal dependencies [40]. Concatenating retrieved sequences disrupts the patch boundary alignment and temporal continuity, leading to misalignment between the learned

and retrieved temporal representations. iTransformer embeds each channel as an independent token (variate token) and performs attention across the channel dimension rather than across time steps[44]. However, the concatenated historical sequence adds more time steps, not more variates. Since iTransformer's attention mechanism does not attend across the time dimension, the prepended context is only weakly integrated into the learned representations. Thus, RAC provides little useful signal and may introduce temporal noise that the architecture is not designed to filter. Similary, RAC complements MSGNet only under short input sequence and long forecasting horizon condition, because MSGNet's multi-scale FFT-based graph learning[41] naturally captures inter series correlations when enough historical data is available. RAC provides value only when the internal multi-scale representation is under-informed and can benefit from external analogues of longer term cross series dynamics.

Our multi-scale patching Transformer, MSPFormer, only got a marginal gain when there is scares input information. Comparing with PatchTST which is a single scale patching Transformer, MSPFormer can exploit the additional historical patterns across multiple scales, therefore the temporal patterns learnt from different resolutions, allowing it to align retrieved segments with relevant temporal structures. This explains the marginal gains observed under short input settings.

From the experiment results, the Sub-RQ4 "To what extent does RAC improve the forecasting performance of Transformer-based architectures in the ADA price prediction task?" and hypothesis H4 can be addressed from several perspectives. The impact of RAC is conditional rather than universal, depending strongly on the model architecture and the amount of historical information available. In general, RAC provides marginal improvements when there is no sufficient temporal information (short input window or medium- and long- forecasting horizon). Under this conditions, the retrieved historical sequences enrich the input representation, enabling models that is designed for incorporating external contextual information with noise filtering functions, such as TimeXer, to effectively leverage additional

contextual information and improve predictive accuracy. For models equipped with modules capable of capturing multi-scale temporal patterns, MSPFormer and MSGNet, the inclusion of RAC yields only marginal gains. In contrast, models with single scale design, PatchTST and iTransformer, rely heavily on predefined temporal boundaries. For these models, appending external historical sequences tends to be redundant and can even introduce misaligned or noisy information, likely because the retrieved context disrupts their local pattern representations.

More broadly, RAC relies on the assumption that historically similar patterns exist within the database and are informative for future dynamics[59]. However, this assumption may not hold in the highly non-stationary cryptocurrency markets, as past conditions may not adequately reflect future regimes[178]. Consequently, the benefits of RAC are depend on the representativeness and usefulness of the stored data. Therefore, the benefits of RAC on cryptocurrency forecasting should be interpreted as conditional.

4.6 Conclusion

Through a series of experiments, this chapter assesses all the hypotheses pertaining to the RQs and identifies the critical techniques that formulate the proposed architecture, MSPRACFormer. Table 4.10 summarizes the findings from the empirical experiments and identifying the conditional advantages of each technique, informing the rationale for the proposed architectural design.

-

Horizon	Model	MSE	RMSE	MAE	RRSE	MAPE
1	PatchTST (no-RAC)	0.00039	0.01975	0.01367	0.40883	0.02955
	PatchTST (RAC)	0.00040 ↑	0.02000 ↑	0.01385 ↑	0.41037 ↑	0.02986 ↑
	MSPFormer (no-RAC)	**0.00035**	**0.01871**	**0.01309**	**0.39813**	**0.02813**
	MSPFormer (RAC)	0.00040 ↑	0.02000 ↑	0.01405 ↑	0.41223 ↑	0.03035 ↑
	iTransformer (no-RAC)	0.00053	0.02302	0.01600	0.44209	0.03465
	iTransformer (RAC)	0.00070 ↑	0.02646 ↑	0.01932 ↑	0.47363 ↑	0.04168 ↑
	MSGNet (no-RAC)	0.00149	0.03859	0.02387	0.41679	0.04265
	MSGNet (RAC)	0.00152 ↑	0.03899 ↑	0.02402 ↑	0.52328 ↑	0.04682 ↑
	TimeXer (no-RAC)	0.00134	0.03662	0.02328	0.40667	0.04280
	TimeXer (RAC)	0.00135 ↑	0.03674 ↑	0.02402 ↑	0.50684 ↑	0.04865 ↑
7	PatchTST (no-RAC)	0.00144	0.03795	0.02651	0.56811	0.05804
	PatchTST (RAC)	0.00354 ↑	0.05950 ↑	0.03459 ↑	0.62098 ↑	0.06625 ↑
	MSPFormer (no-RAC)	**0.00142**	**0.03767**	**0.02610**	**0.56581**	**0.05704**
	MSPFormer (RAC)	0.00143 ↑	0.03783 ↑	0.02640 ↑	0.56671 ↑	0.05807 ↑
	iTransformer (no-RAC)	0.00176	0.04195	0.03039	0.59719	0.06679
	iTransformer (RAC)	0.00201 ↑	0.04483 ↑	0.03224 ↑	0.61728 ↑	0.07054 ↑
	MSGNet (no-RAC)	0.00410	0.06403	0.03948	0.64445	0.07571
	MSGNet (RAC)	0.00517 ↑	0.07192 ↑	0.04645 ↑	0.68274 ↑	0.08983 ↑
	TimeXer (no-RAC)	0.00424	0.06512	0.04131	0.64985	0.07998
	TimeXer (RAC)	0.00366 ↓	0.06050 ↓	0.03969 ↓	0.62638 ↓	0.07896 ↓
15	PatchTST (no-RAC)	0.00239	0.04887	0.03468	0.64722	0.07570
	PatchTST (RAC)	0.00896 ↑	0.09467 ↑	0.05268 ↑	0.73872 ↑	0.09414 ↑
	MSPFormer (no-RAC)	**0.00236**	**0.04858**	**0.03439**	**0.64541**	**0.07505**
	MSPFormer (RAC)	0.00239 ↑	0.04887 ↑	0.03478 ↑	0.64728 ↑	0.07620 ↑
	iTransformer (no-RAC)	0.00267	0.05167	0.03798	0.66518	0.08316
	iTransformer (RAC)	0.00302 ↑	0.05495 ↑	0.04038 ↑	0.68647 ↑	0.08953 ↑
	MSGNet (no-RAC)	0.01035	0.10173	0.06059	0.76596	0.10991
	MSGNet (RAC)	0.00772 ↓	0.08791 ↓	0.05837 ↓	0.71194 ↓	0.11417 ↑
	TimeXer (no-RAC)	0.01044	0.10218	0.05864	0.76763	0.10454
	TimeXer (RAC)	0.00837 ↓	0.09149 ↓	0.05310 ↓	0.72633 ↓	0.09872 ↓
21	PatchTST (no-RAC)	0.00332	0.05761	0.04069	0.70582	0.08830
	PatchTST (RAC)	0.00334 ↑	0.05782 ↑	0.04105 ↑	0.70720 ↑	0.08908 ↑
	MSPFormer (no-RAC)	0.00330	0.05745	**0.04039**	0.70473	**0.08776**
	MSPFormer (RAC)	**0.00329** ↓	**0.05737** ↓	0.04044 ↑	**0.70430** ↓	0.0879 ↑
	iTransformer (no-RAC)	0.00353	0.05940	0.04330	0.71685	0.09443
	iTransformer (RAC)	0.00405 ↑	0.06366 ↑	0.04656 ↑	0.74176 ↑	0.10289 ↑
	MSGNet (no-RAC)	0.01421	0.11917	0.06781	0.80300	0.11900
	MSGNet (RAC)	0.00627 ↓	0.07919 ↓	0.05763 ↓	0.82762 ↑	0.12676 ↑
	TimeXer (no-RAC)	0.01582	0.12578	0.06939	0.82473	0.11882
	TimeXer (RAC)	0.00369 ↓	0.06075 ↓	0.04530 ↓	0.72510 ↓	0.09980 ↓

Table 4.8: Forecasting performance across horizons with input sequence 21 (21 days of cryptocurrencies daily prices). For each model and horizon, RAC vs. no-RAC changes are indicated after the RAC values with arrows : ↓ means improvement (lower error), ↑ means degradation. The best result for each prediction horizon are highlighted in bold.

Horizon	Model	MSE	RMSE	MAE	RRSE	MAPE
1	PatchTST (no-RAC)	0.00081	0.02846	0.01723	0.37462	0.03142
	PatchTST (RAC)	0.00077 ↓	0.02775 ↓	0.01701 ↓	**0.37046** ↓	0.03138 ↓
	MSPFormer (no-RAC)	**0.00034**	**0.01844**	0.01307	0.37730	0.02761
	MSPFormer (RAC)	0.00034	0.01844	**0.01297** ↓	0.39784 ↓	**0.02727** ↓
	iTransformer (no-RAC)	0.00035	0.01871	0.01328	0.40172	0.02817
	iTransformer (RAC)	0.00043 ↑	0.02074 ↑	0.01454 ↑	0.42280 ↑	0.03068 ↑
	MSGNet (no-RAC)	0.00107	0.03271	0.01966	0.40168	0.03578
	MSGNet (RAC)	0.00190 ↑	0.04359 ↑	0.02476 ↑	0.46354 ↑	0.04307 ↑
	TimeXer (no-RAC)	0.00098	0.03130	0.01894	0.39298	0.03438
	TimeXer (RAC)	0.00100 ↑	0.03162 ↑	0.01875 ↓	0.39470 ↑	0.03436 ↓
3	PatchTST (no-RAC)	0.00178	0.04219	0.02478	**0.45314**	0.04500
	PatchTST (RAC)	0.00180 ↑	0.04243 ↑	0.02511 ↑	0.45438 ↑	0.04520 ↑
	MSPFormer (no-RAC)	**0.00067**	**0.02588**	0.01812	0.47205	0.03887
	MSPFormer (RAC)	**0.00067**	**0.02588**	**0.01809** ↓	0.47151 ↓	**0.03864** ↓
	iTransformer (no-RAC)	0.00073	0.02702	0.01922	0.48302	0.04137
	iTransformer (RAC)	0.00081 ↑	0.02846 ↑	0.02014 ↑	0.49527 ↑	0.04315 ↑
	MSGNet (no-RAC)	0.00225	0.04743	0.02906	0.48086	0.05259
	MSGNet (RAC)	0.00238 ↑	0.04877 ↑	0.02998 ↑	0.48731 ↑	0.05404 ↑
	TimeXer (no-RAC)	0.00210	0.04583	0.02701	0.47264	0.04875
	TimeXer (RAC)	0.00202 ↓	0.04494 ↓	0.02660 ↓	0.46774 ↓	0.04820 ↓
5	PatchTST (no-RAC)	0.00263	0.05128	0.03091	**0.49692**	0.05614
	PatchTST (RAC)	0.00267 ↑	0.05167 ↑	0.03150 ↑	0.49900 ↑	0.05675 ↑
	MSPFormer (no-RAC)	**0.00107**	**0.03271**	**0.02283**	0.53041	**0.04897**
	MSPFormer (RAC)	**0.00107**	**0.03271**	0.02287 ↑	0.53011 ↓	0.04892 ↓
	iTransformer (no-RAC)	0.00115	0.03391	0.02414	0.54039	0.05191
	iTransformer (RAC)	0.00116 ↑	0.03406 ↑	0.02406 ↓	0.54080 ↑	0.05135 ↓
	MSGNet (no-RAC)	0.00313	0.05595	0.03434	0.51934	0.06224
	MSGNet (RAC)	0.00348 ↑	0.05899 ↑	0.03689 ↑	0.53328 ↑	0.06610 ↑
	TimeXer (no-RAC)	0.00318	0.05639	0.03342	0.52116	0.05982
	TimeXer (RAC)	0.00283 ↓	0.05320 ↓	0.03196 ↓	0.50630 ↓	0.05771 ↓
7	PatchTST (no-RAC)	0.00365	0.06042	0.03678	0.53732	0.06589
	PatchTST (RAC)	0.00355 ↓	0.05958 ↓	0.03716 ↑	**0.53367** ↓	0.06683 ↑
	MSPFormer (no-RAC)	0.00144	0.03795	0.02666	0.57069	0.05714
	MSPFormer (RAC)	**0.00141** ↓	**0.03755** ↓	**0.02635** ↓	0.56799 ↓	**0.05651** ↓
	iTransformer (no-RAC)	0.00158	0.03975	0.02857	0.58451	0.06151
	iTransformer (RAC)	0.00161 ↑	0.04012 ↑	0.02868 ↑	0.58709 ↑	0.06125 ↓
	MSGNet (no-RAC)	0.00492	0.07014	0.04207	0.57898	0.07434
	MSGNet (RAC)	0.00487 ↓	0.06979 ↓	0.04190 ↓	0.57771 ↓	0.07366 ↓
	TimeXer (no-RAC)	0.00420	0.06481	0.03881	0.55670	0.06894
	TimeXer (RAC)	0.00387 ↓	0.06221 ↓	0.03807 ↓	0.54545 ↓	0.06849 ↓

Table 4.9: Forecasting performance across forecasting horizons (1-, 3-, 5-, 7-day) with input sequence 7 (7 days of cryptocurrencies daily prices). For each model and horizon, RAC vs. no-RAC changes are indicated after the RAC values with arrows : ↓ means improvement (lower error), ↑ means degradation. The best result for each prediction horizon are highlighted in bold.

Research Question	Hypothesis Number	Hypothesis Description	Section	Findings
Sub-RQ1: How does the canonical Transformer architecture perform in forecasting ADA prices using multivariate historical price data from the top 10 market-cap cryptocurrencies, compared to benchmark model LSTM?	H1	Transformer vs LSTM: Theoretically, leveraging the multi-head self-attention mechanism, the canonical Transformer will outperform LSTM in cryptocurrency price forecasting. Specifically, the attention mechanism enables Transformers to model long-range dependencies without the vanishing gradient limitations of Recurrent Neural Network (RNN)	4.2	The Transformer's advantages become evident when supplied with longer input sequences where their self-attention mechanism can fully exploit long-term relationships in the data. In contrast, when provided with short input sequence, LSTM tends to be more robust.
Sub-RQ2: How do different patching strategies influence the predictive performance of the canonical Transformer model in multivariate cryptocurrency price forecasting?	H2.1	fixed length single-scale patching benefits Transformer model in two ways: 1. reducing computational complexity by reducing sequence length; 2. improved learning of local temporal patterns.	4.3.1	Implementing of single-scale patching can lead to a significant increase in the canonical Transformer's forecasting performance. By grouping adjacent time steps, the model achieves a more efficient extraction of localized temporal patterns while simultaneously mitigating the quadratic computational overhead inherent to the standard attention mechanism.
	H2.2	Multi-scale patching: comparing with single-scale patching, multi-scale patching enables the Transformer to learn local patterns in multiple scales, improving the model's forecasting ability from learning various length dependencies.	4.3.1,4.4	Under CD setting, due to the time-varying inter-dependence in our dataset, multi-scale patching may introduce more noise than informative signal comparing with single-scale patching; Under CI setting, as each channel is processed independently, applying multi-scale patching enables the Transformer model to capture temporal dependencies across various scales.
	H2.3	FFT-based multi-scale patching: FFT-based frequency selection provides patch sizes aligned with dominant periodicities in the data, enhancing the model's ability to capture cyclical behaviors.	4.3.2, 4.4	Under CD setting, FFT help aligning patching scales with dominant frequency components, allowing the Transformer model to capture seasonal patterns shared across the channels; Under CI setting, all the channels are processed independently. Therefore, aggregating spectral information across all channels may introduce misalignment.

Continued on next page

Research Question	Hypothesis Number	Hypothesis Description	Section	Findings
Sub-RQ3: How do CI and CD multivariate processing strategies affect the forecasting performance of Transformer-based architectures?	H3.1	CD processing: Transformer model can learn the relationship across all the channels in the dataset (e.g.BTC-ETH price movement relationship), therefore dependencies across channels are explicitly modeled.	4.4	Given the time varying inter-dependence inherent in our dataset, modeling the inter-relationship explicitly through the CD configuration may introduce noise rather than beneficial signals. The empirical results support this notion, indicating that CI processing represents the optimal strategy for handling the time varying dependencies within this specific dataset.
	H3.2	Transformer models each channel in the dataset separately while sharing weights across channels. By decoupling the representation learning process, the model avoids distortions that may arise from entangled inter-channel correlations. This separation helps preserver long-term feature dynamics within each channel.	4.4	
Sub-RQ4: To what extent does RAC improve the forecasting performance of Transformer-based architecture in the ADA price prediction task?	H4	Integrating RAC-generated contextual embeddings with historical price data improves forecasting accuracy by enriching the model with complementary information that extends beyond the local input sequence, thereby enabling the capture of temporal patterns.	4.5	RAC is beneficial only under certain conditions. The experiment results identify three conditions under which RAC provides a performance gain: 1. Short input sequence; 2. Medium- or long-term forecasting horizons; 3. Application alongside forecasting models that are specifically designed for incorporating external information with integrated noise filtering function, such as TimeXer. In addition, RAC assumes that historical patterns are predictive of future dynamics; however, in highly volatile cryptocurrency markets this assumption may not hold due to frequent regime changes, so its benefits are conditional on the representativeness of the stored data.

Table 4.10: This table lists all the hypotheses and their corresponding sections where they are assessed. The last column demonstrates the key findings derived from each section.

Chapter 5

Results and discussion

5.1 Introduction

This chapter presents the empirical results obtained by applying the MSPRACFormer framework to the ADA price forecasting task by leveraging the historical price data of top dominant cryptocurrencies. To verify the generalization capabilities of MSPRACFormer, we further applied the framework to BTC and ETH price forecasting. As detailed in Section 3.2, these coins are the top two market dominant cryptocurrencies, making them critical choices for assessing transferability.

A comprehensive set of benchmark models was included in the experiments. This set encompasses recent SOTA Transformer-based architectures, specifically PatchTST, iTransformer, MSGNet, and TimeXer. Furthermore, the most adopted method in cryptocurrency forecasting[98–100], LSTM and the DLinear model-which relies on linear methods[162]-are also incorporated for performance comparison.

From the grid search result, we identified that an input sequence length of 5 days is optimal for MSPRACFormer when forecasting both ADA and ETH prices, whereas an input sequence length of 3 days was found to be optimal for BTC forecasting. Using these settings, experiments are conducted across forecasting horizons to evaluate the model's predictive capability over different temporal spans for the coins.

The implementation of the proposed framework has been released as open-source

software[1], enabling the research community to reuse the developed methods. The repository contains all necessary artefacts for reproduction, including the source code, configuration files, and example execution scripts. Comprehensive documentation is provided to guide users through data preparation, parameter configuration, model training, evaluation, and extension to new models. Detailed explanations of the project structure and all configurable parameters are presented in Appendix D, which serves as a readme style implementation guide.

5.2 Experimental results and analysis

Table 5.1, 5.2 and 5.3 reports the full set of evaluation results across all metrics for ADA, BTC and ETH respectively. To facilitate a clearer comparison between our model and the baseline architectures, Figure 5.1, 5.2, and 5.3 visualizes the RMSE and MAE values for ADA, BTC and ETH forecasting across the various prediction horizons. A set of prediction versus true value figures located in Appendix C, illustrating the comparison between all the model's prediction and the actual values.

From the experiment result tables 5.1, 5.2, and 5.3, it can be observed that across all forecasting horizons, MSPRACFormer outperforms all the benchmark models in terms of MSE, RMSE, MAE and MAPE for ADA and ETH forecasting. For BTC forecasting, it demonstrates superior performance across MSE, RMSE, and MAE, and achieves near optimal results on MAPE (securing 3rd place for 1-day forecasting, and 2nd place for 5-day forecasting). MSPRACFormer consistently delivers the strongest overall accuracy for all the coins, achieving the lowest error values in nearly all evaluation metrics for short (1-3 days), medium (5 days), and long horizons (10-15 days). Notably, at longer horizons where the prediction length is two to three times the input sequence window for ADA and ETH and three to five times for BTC, MSPRACFormer continues to outperform all baseline models. These results highlight the model's ability to retain predictive accuracy even as

[1]Available at: `https://github.com/cmajorsolo/MSPRACFormer`

Horizon	Model	MSE	RMSE	MAE	MAPE	RRSE
1	MSPRACFormer	**0.00035**	**0.01871**	**0.01316**	**0.02773**	0.40125
	PatchTST	0.00073	0.02702	0.01653	0.03069	**0.37038**
	iTransformer	0.00037(*)	0.01924(*)	0.01359(*)	0.02856(*)	0.40744
	MSGNet	0.00103	0.03209	0.01947	0.03569	0.40293
	TimeXer	0.00080	0.02828	0.01730	0.03227	0.37912(*)
	LSTM	0.00104	0.03225	0.01953	0.03424	0.38601
	DLinear	0.00129	0.03592	0.02269	0.03838	0.40692
3	MSPRACFormer	**0.00067**	**0.02588**	**0.01817**	**0.03889**	0.47346
	PatchTST	0.00172	0.04147	0.02452	0.04478	**0.45540**
	iTransformer	0.00069(*)	0.02627(*)	0.01861(*)	0.03981(*)	0.47636
	MSGNet	0.00212	0.04607	0.02764	0.04997	0.47973
	TimeXer	0.00180	0.04243	0.02509	0.04573	0.46048
	LSTM	0.00205	0.04528	0.02735	0.04808	0.45637(*)
	DLinear	0.00325	0.05701	0.03450	0.05612	0.51207
5	MSPRACFormer	**0.00104**	**0.03225**	**0.02260**	**0.04839**	0.52783
	PatchTST	0.00260	0.05100	0.03050	0.05544	**0.50150**
	iTransformer	0.00109(*)	0.03302(*)	0.02335(*)	0.05017(*)	0.53302
	MSGNet	0.00329	0.05736	0.03527	0.06350	0.53172
	TimeXer	0.00276	0.05253	0.03142	0.05698	0.50896
	LSTM	0.00318	0.05642	0.03428	0.06022	0.50814(*)
	DLinear	0.00458	0.06768	0.04268	0.06980	0.55679
10	MSPRACFormer	**0.00186**	**0.04312**	**0.03056**	**0.06540**	0.60901
	PatchTST	0.00534	0.07307	0.04365	0.07650	**0.59311**
	iTransformer	0.00195(*)	0.04416(*)	0.03176(*)	0.06802(*)	0.61570
	MSGNet	0.00610	0.07810	0.04714	0.08296	0.61300
	TimeXer	0.00548	0.07405	0.04425	0.07817	0.59690(*)
	LSTM	0.00664	0.08149	0.05018	0.08687	0.61437
	DLinear	0.01006	0.10030	0.05871	0.09462	0.68145
15	MSPRACFormer	**0.00251**	**0.05010**	**0.03628**	**0.07727**	0.65521(*)
	PatchTST	0.00810	0.09000	0.05262	0.08993	**0.65348**
	iTransformer	0.00263(*)	0.05128(*)	0.03741(*)	0.07985(*)	0.66258
	MSGNet	0.00941	0.09702	0.05803	0.10022	0.67754
	TimeXer	0.00909	0.09533	0.05567	0.09550	0.67154
	LSTM	0.00962	0.09810	0.05742	0.09825	0.68130
	DLinear	0.01460	0.12083	0.06863	0.10825	0.75608

Table 5.1: ADA forecasting performance across horizons with input sequence 5 (5 days of cryptocurrencies daily prices). The best result for each prediction horizon are highlighted in bold, and the second best result is marked with (*).

Horizon	Model	MSE	RMSE	MAE	MAPE	RRSE
1	MSPRACFormer	**3033215.25**	**1741.61290**	**1266.66577**	0.02041	0.50283
	PatchTST	3453086.5	1858.24823	1337.30823	0.02029(*)	0.40033
	iTransformer	3064911.0(*)	1750.68872(*)	1283.82861(*)	0.02068	0.50415
	MSGNet	3549588.0	1884.03503	1366.69373	0.02074	0.40309
	TimeXer	3430206.5	1852.08167	1327.56372	**0.02017**	0.39966
	LSTM	3796280.5	1948.40460	1417.16272	0.02095	0.38019(*)
	DLinear	3642669.25	1908.57781	1369.97681	0.02030	**0.37643**
3	MSPRACFormer	**5654948.0**	**2378.01346**	**1757.10876**	**0.02841**	0.59247
	PatchTST	6676372.5	2583.86774	1874.68628	0.02845(*)	0.47044
	iTransformer	5688715.5(*)	2385.10283(*)	1767.82861(*)	0.02857	0.59334
	MSGNet	7002625.5	2646.24744	1921.62915	0.02911	0.47597
	TimeXer	6780823.5	2604.00144	1883.70398	0.02856	0.47279
	LSTM	7242295.0	2691.15124	1975.69043	0.02917	0.44787 (*)
	DLinear	7219201.0	2686.85709	1959.61963	0.02895	**0.44695**
5	MSPRACFormer	**8419458.0**	**2901.63023**	**2178.04272**	0.03523(*)	0.66042
	PatchTST	9999629.0	3162.21900	2318.66186	**0.03517**	0.51797
	iTransformer	8438550.0(*)	2904.91824(*)	2187.31860(*)	0.03536	0.66073
	MSGNet	10584048.0	3253.31339	2396.49146	0.03625	0.52577
	TimeXer	10188113.0	3191.88236	2341.78467	0.03547	0.52064
	LSTM	10584296.0	3253.35150	2401.63770	0.03546	0.49108(*)
	DLinear	10446525.0	3232.10844	2385.32275	0.03531	**0.48949**
10	MSPRACFormer	**15274842.0**	**3908.30424**	**2945.90430**	**0.04762**	0.78194
	PatchTST	18950676.0	4353.23742	3166.91113	0.04768(*)	0.60137
	iTransformer	15399014.0(*)	3924.15774(*)	2961.33276(*)	0.04785	0.78371
	MSGNet	20410892.0	4517.84152	3292.83569	0.04938	0.61267
	TimeXer	19192436.0	4380.91726	3189.98682	0.04797	0.60328
	LSTM	20136080.0	4487.32437	3297.73364	0.04883	**0.58842**
	DLinear	20458770.0	4523.13719	3292.00244	0.04880	0.59072(*)
15	MSPRACFormer	**21730692.0**	**4661.61903**	**3538.47290**	**0.05710**	0.87306
	PatchTST	28613458.0	5349.15489	3878.87158	0.05781	0.66167
	iTransformer	21928460.0(*)	4682.78336(*)	3556.82764(*)	0.05737(*)	0.87521
	MSGNet	30268318.0	5501.66502	4009.68994	0.05958	0.67116
	TimeXer	28956840.0	5381.15601	3903.09033	0.05810	0.66402
	LSTM	30427666.0	5516.12781	4034.85376	0.05926	**0.65024**
	DLinear	30987506.0	5566.64226	4018.63403	0.05901	0.65322(*)

Table 5.2: BTC forecasting performance across horizons with input sequence 3 (3 days of cryptocurrencies daily prices). The best result for each prediction horizon are highlighted in bold, and the second best result is marked with (*).

Horizon	Model	MSE	RMSE	MAE	MAPE	RRSE
1	MSPRACFormer	**10711.38281**	**103.49581**	**72.92900**	**0.02383**	**0.46235**
	PatchTST	11249.22656	106.06237	74.81820(*)	0.02434(*)	0.46545
	iTransformer	10890.30176(*)	104.35661(*)	74.86322	0.02446	0.46427(*)
	MSGNet	11674.15918	108.04702	76.71465	0.02489	0.46978
	TimeXer	12150.38477	110.22878	78.29439	0.02553	0.47450
	LSTM	12637.21582	112.41537	81.54581	0.02632	0.47770
	DLinear	11551.41211	107.47750	75.67057	0.02441	0.46710
3	MSPRACFormer	**20983.02930**	**144.85520**	**102.14274**	**0.03351**	**0.54741**
	PatchTST	22258.44141	149.19263	107.02551	0.03491	0.55195
	iTransformer	21047.50977(*)	145.07760(*)	102.73936(*)	0.03367(*)	0.54782(*)
	MSGNet	23833.87695	154.38224	109.53932	0.03560	0.56147
	TimeXer	22847.91992	151.15528	107.69183	0.03517	0.55557
	LSTM	23854.11719	154.44778	111.26074	0.03590	0.56098
	DLinear	22794.54297	150.97862	108.16687	0.03503	0.55464
5	MSPRACFormer	**32563.41602**	**180.45336**	**126.71838**	**0.04157**	**0.61147**
	PatchTST	34351.44922	185.34144	132.25584	0.04309	0.61491
	iTransformer	32618.69921(*)	180.60648(*)	127.62386(*)	0.04185(*)	0.61173(*)
	MSGNet	37663.27344	194.07028	138.36876	0.04493	0.62923
	TimeXer	34890.06641	186.78883	133.13887	0.04336	0.61731
	LSTM	39778.67969	199.44593	143.63248	0.04659	0.63859
	DLinear	35529.63672	188.49307	134.60396	0.04353	0.62081
10	MSPRACFormer	**62947.24219**	**250.89289**	**181.13423**	**0.05907**	0.72342
	PatchTST	65234.85156	255.41114	185.55011	0.05994	0.72256(*)
	iTransformer	64224.41797(*)	253.42537(*)	183.20963(*)	0.05973(*)	0.72706
	MSGNet	69126.89063	262.91993	192.20313	0.06194	0.73310
	TimeXer	65132.66016	255.21101	185.50674	0.05992	**0.72227**
	LSTM	70911.47656	266.29209	196.07245	0.06282	0.73816
	DLinear	66924.52344	258.69775	188.14034	0.06059	0.72756
15	MSPRACFormer	**92621.92969**	**304.33851**	**223.62500**	**0.07266**	0.79760(*)
	PatchTST	96042.31250	309.90694	230.17645	0.07381	0.79812
	iTransformer	93712.75781(*)	306.12539(*)	225.57832(*)	0.07330(*)	0.80182
	MSGNet	101220.25000	318.15130	237.27612	0.07600	0.80867
	TimeXer	96578.06250	310.77011	230.57230	0.07398	0.79923
	LSTM	101848.89063	319.13773	239.56828	0.07678	0.80992
	DLinear	95562.53125	309.13190	231.23286	0.07429	**0.79713**

Table 5.3: ETH forecasting performance across horizons with input sequence 5 (5 days of cryptocurrencies daily prices). The best result for each prediction horizon are highlighted in bold, and the second best result is marked with (*).

temporal uncertainty increases. The consistency of these improvements suggests that the proposed architecture's robustness to the highly non-stationary and noisy dynamics characteristic of cryptocurrency markets.

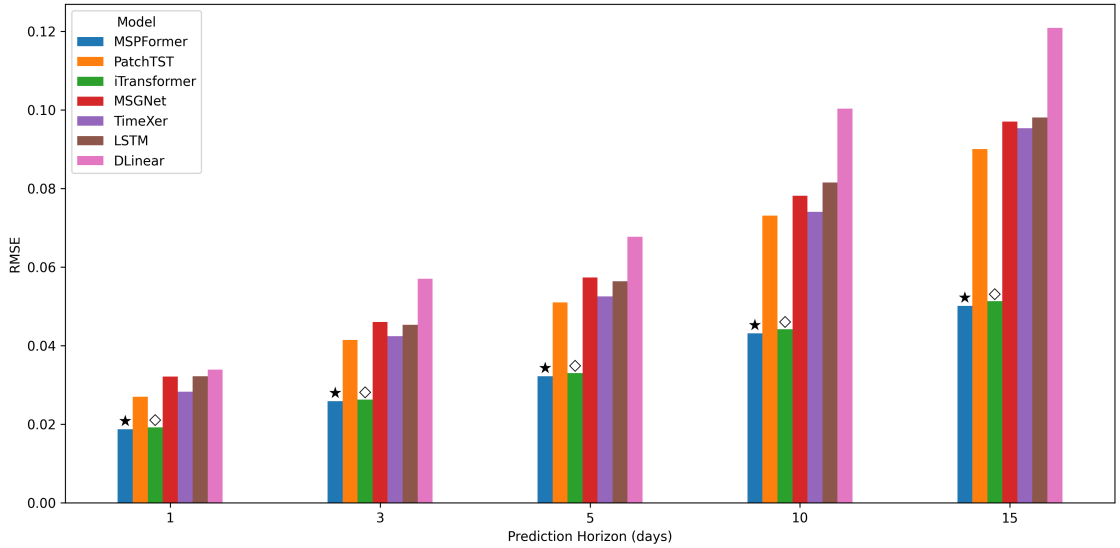
In terms of RRSE, MSPRACFormer exhibits highly competitive performance for both ADA and BTC forecasting, achieving metrics similar to the SOTA Transformer architecture, iTransformer. For ETH forecasting, the proposed model leads the performance across the short-(1–3 days) and medium-(5-day) forecasting horizons, while maintaining strong results in the long-forecasting horizons (10-days and 15-days), ranking 3rd and 2nd places respectively. The results indicate that for ADA and BTC, our model maintains its efficiency relative to the mean baseline, even when applied to different data scales. For ADA, the model provides the most substantial improvement over the mean baseline compared to all other benchmark models. Overall, the results confirms the generalization capability of MSPRACFormer across the cryptocurrency datasets, confirming its superior predictive efficiency.

Comparing with other Transformer-based baselines, MSPRACFormer remains the top performer across almost all horizons for all coins. Although the iTransformer frequently attains the second-best results, MSPRACFormer consistently outperforms it. This performance gap suggests that channel-independent multi-scale patching captures temporal structures that global self-attention alone cannot, enabling the model to learn both fine-grained and coarse-scale dependencies more effectively.[2] In contrast, PatchTST relies on fixed-scale patching[40], which limits its ability to represent temporal patterns that evolve across multiple scales. MSGNet and TimeXer explicitly model inter-channel dependencies through channel-dependent attention[41] and cross-attention[42] respectively. However, in cryptocurrency markets, cross-asset correlations are highly time-varying and regime-dependent[31]. Explicitly enforcing strong inter-channel interactions at every layer may therefore amplify noise and hinder performance during volatile periods. MSPRACFormer avoids this issue by operating in a CI manner, extracting stable asset specific temporal

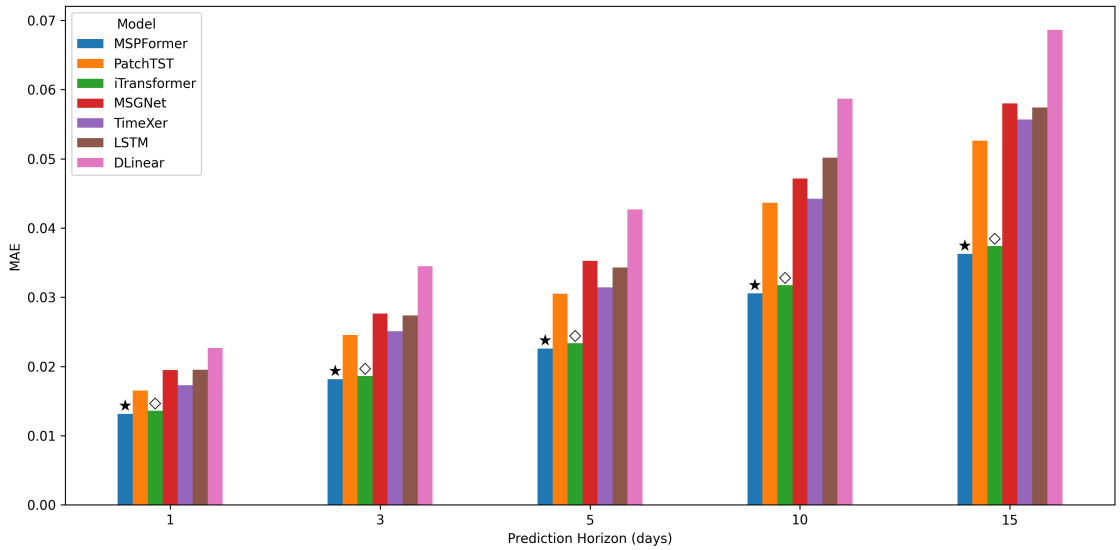
[2]iTransformer[44] treats each entire variable as a single "variable token," enabling the model to capture global inter-variable dependencies through self-attention.

features before aggregation, which leads to more robust generalization. Beyond Transformer architectures, LSTM and DLinear show clear limitations. LSTM, despite being a historically strong sequential model, struggles to capture long-range dependencies and exhibits pronounced error accumulation[134] as the forecasting horizon increases. Although DLinear is competitive, it remains unable to represent complex temporal patterns in cryptocurrency, causing its performance to deteriorate significantly at medium and long horizons.

Together, these comparisons demonstrate that MSPRACFormer's architectural design provides superior flexibility and robustness for cryptocurrency price forecasting, allowing it to outperform both Transformer-based and classical forecasting models across all prediction horizons.

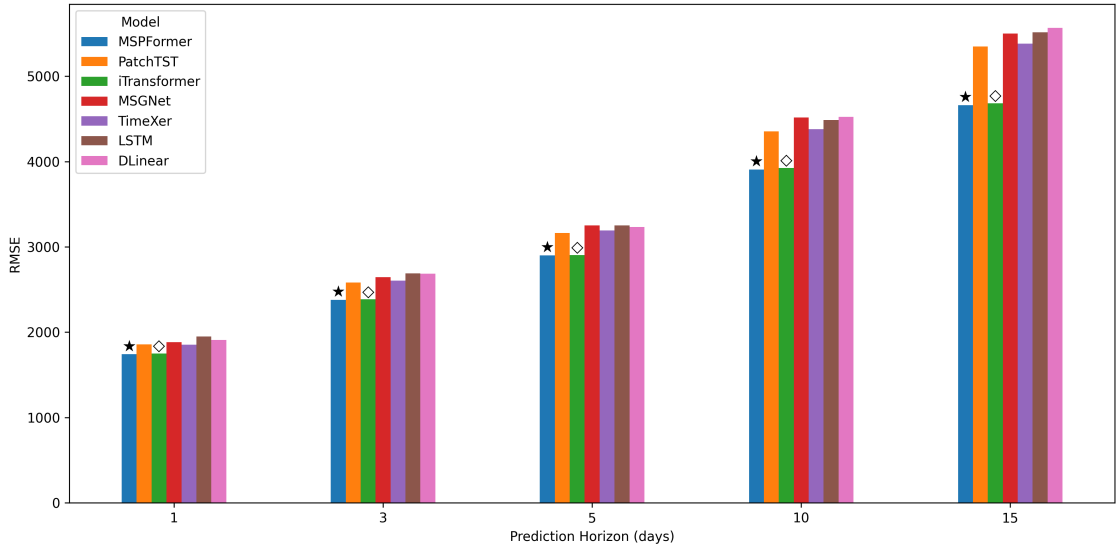


(a) RMSE of ADA forecasting results across different prediction horizons.

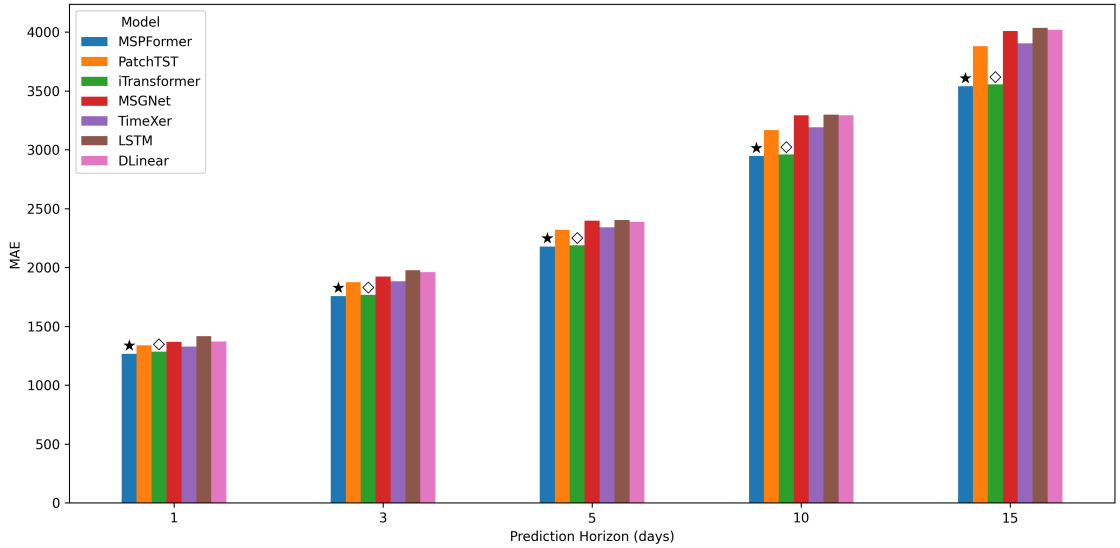


(b) MAE of ADA forecasting results across different prediction horizons.

Figure 5.1: RMSE and MAE comparison across different horizons for ADA price forecasting using MSPRACFormer, PatchTST, iTransformer, MSGNet, TimeXer, LSTM and DLinear. Lower values indicate better performance. The best performing model for each horizon is highlighted with a star, and the second best performer is denoted with a diamond.

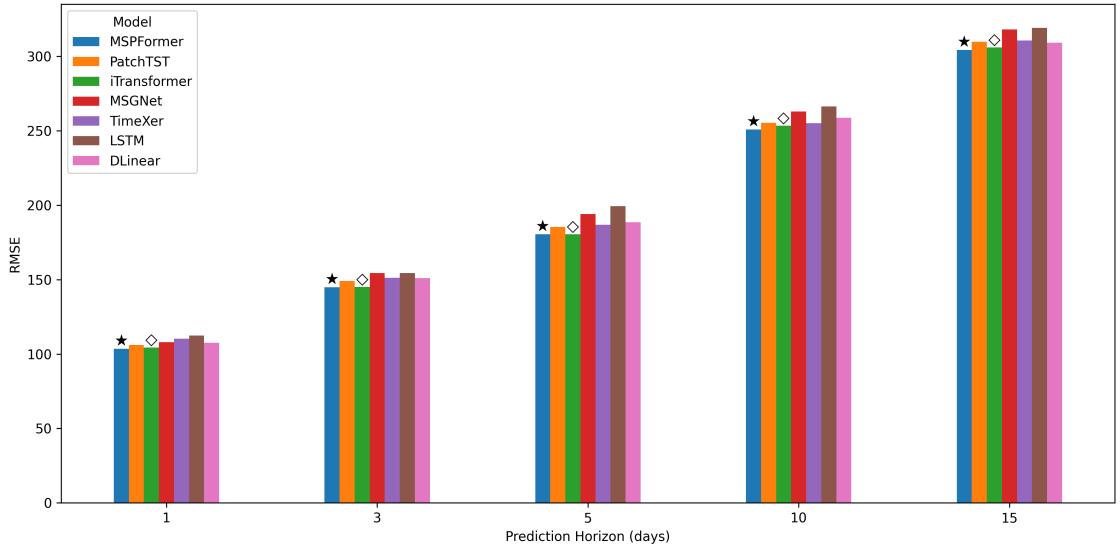


(a) RMSE of BTC forecasting results across different prediction horizons.

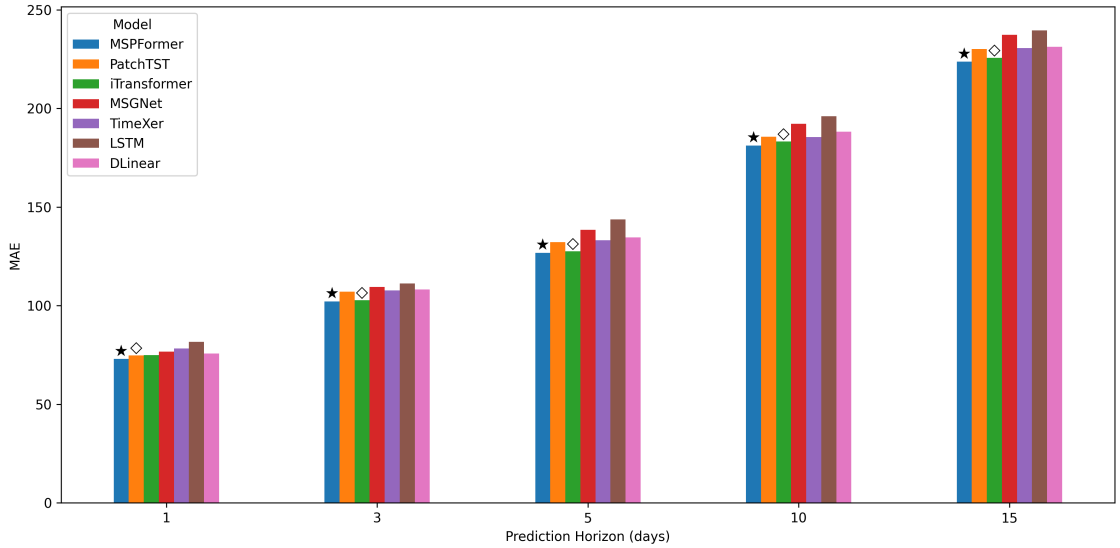


(b) MAE of BTC forecasting results across different prediction horizons.

Figure 5.2: RMSE and MAE comparison across different horizons for BTC price forecasting using MSPRACFormer, PatchTST, iTransformer, MSGNet, TimeXer, LSTM and DLinear. Lower values indicate better performance. The best performing model for each horizon is highlighted with a star, and the second best performer is denoted with a diamond.



(a) RMSE of ETH forecasting results across different prediction horizons.



(b) MAE of ETH forecasting results across different prediction horizons.

Figure 5.3: RMSE and MAE comparison across different horizons for ETH price forecasting using MSPRACFormer, PatchTST, iTransformer, MSGNet, TimeXer, LSTM and DLinear. Lower values indicate better performance. The best performing model for each horizon is highlighted with a star, and the second best performer is denoted with a diamond.

Chapter 6

Conclusion

Accurately predicting future cryptocurrency prices is widely regarded as a difficult task due to that market's high volatility[232], time-varying inter-dependencies[233], seasonal effects[234] and unpredictable behaviors[235]. In recent years, an expanding literature has examined cryptocurrency forecasting using statistical models, machine learning methods and deep learning techniques. However, most studies focus primarily on the dominant assets (BTC and ETH), leaving less attention to altcoins[13], such as ADA. Moreover, although Transformer-based models[40, 41, 44] have shown their effectiveness in TSF tasks, a limited number of studies have explored the effectiveness of Transformer architectures for cryptocurrency price forecasting[13]. This study addresses these gaps via answering the central RQ: "How should the Transformer-based architectures be adapted to the variable price dynamics and complex regime dependent inter-relationships of the top 10 market capitalization cryptocurrencies in order to improve ADA price forecasting?" by proposing MSPRACFormer, a novel Transformer-based architecture for cryptocurrency forecasting, and by conducting a systematic assessment of the techniques comprising the proposed architecture. Section 6.1 provides a detailed discussion of the study's contributions and novelty, whereas section 6.2 outlines the limitations and suggests avenues for future research.

6.1 Contribution

Through the established research framework, this study proposed MSPRACFormer (Multi-Scale Patching with Retrieval-Augmented Context Transformer). The proposed framework's efficacy was rigorously verified through a temporal hold-out validation on the task of predicting future ADA prices, and its generalization capability was further demonstrated on dominant cryptocurrency datasets, including BTC and ETH. The empirical results confirm the effectiveness and superiority of the proposed methodology.

In addition, this study systematically identifies and integrates several effective techniques designed to enhance the predictive accuracy of the canonical Transformer model for forecasting future cryptocurrency prices utilizing the historical daily price series of dominant cryptocurrencies. The integrated techniques include various patching mechanisms, CI strategy and RAC which have been recently introduced in the broader TSF literature but remain largely unexplored within the context of cryptocurrency forecasting. These techniques are identified through the defined research framework. A rigorous comparative analysis of various patching techniques and CI versus CD approaches was conducted for the ADA price forecasting task. This comparison yields valuable empirical insights into the effectiveness of these techniques relative to the characteristics of cryptocurrency datasets. We also quantified the impact of RAC on a list of SOTA Transformer-based architectures by comparing the RAC-enhanced model against its correspond baseline. Our results indicate that the benefits of RAC are not universal, and we identified the conditions under which RAC becomes effective.

We believe that we are among the first to rigorously verify the efficacy of these technologies and to integrate them into a novel, learnable, and configurable framework. The effective components, specifically single-scale patching, multi-scale patching, and RAC are implemented as configurable modules. This design allows for agile framework configuration based on the specific features of the input dataset, offering a robust and adaptable approach for diverse future forecasting tasks.

Specifically, our contributions are listed as following:

1. To answer the central RQ, we proposed MSPRACFormer, a Multi-Scale Patching(MSP) Transformer augmented with an adaptive Retrieval-Augmented Context(RAC) module for time series forecasting. The MSP module is designed utilizing a Channel Independent (CI) configuration. This design offers a dual advantage: the MSP component enables the model to effectively capture temporal patterns across various scales, while the CI configuration simultaneously mitigates the introduction of noise caused by the time varying interdependence across data channels. The integrated RAC module retrieves historically similar subsequences to provide additional contextual information for forecasting models. The proposed architecture is evaluated on the ADA price forecasting task, with its generalization capability further validated through application to BTC and ETH forecasting. The empirical results show that it consistently outperforms SOTA Transformer-based TSF models, including PatchTST, MSPFormer, iTransformer, MSGNet, and TimeXer, as well as DL and ML baselines, LSTM and DLinear, across multiple forecasting horizons.

2. To validate the hypotheses and answer the sub-RQs in Table1.1, a systematic evaluation of techniques used in SOTA Transformer-based TSF models for ADA forecasting utilizing top dominant historical cryptocurrency prices is conducted. Specifically, to address Sub-RQ1 and H1, we firstly compared the forecasting performance of LSTM and the canonical Transformer model. Sub-RQ2, along with hypotheses H2.1–H2.3, is examined by equipping the canonical Transformer with different patching mechanisms and evaluating their performance. Sub-RQ3 and H3.1-3.2 are answered and validated through a CI versus CD processing strategies comparison and Sub-RQ4 and H4 are explored by comparing RAC-enhanced SOTA Transformer architectures performance vis-à-vis its baseline Transformer version in the forecasting task. Our analysis identifies the conditions under which Transformer architectures outperform LSTM model and highlights the architectural components that deliver the

most robust performance.

- Transformer vs LSTM (Sub-RQ1): Intuitively, the attention mechanism in Transformer architectures enables more effective modeling of long-term dependencies than recurrent models such as LSTM[20, 25, 36, 236]. From this perspective, the canonical Transformer should be better suited for forecasting future ADA prices when leveraging the historical prices of major dominant assets. However, our experiments show that this advantage only becomes apparent when the model is provided with sufficiently long historical sequences (i.e. 60-day). Under shorter input windows, LSTM tends to perform better. The experimental results indicate that for cryptocurrency forecasting where near-term historical data carries greater predictive relevance[237], the canonical Transformer architecture might not be optimal.
- Patching (Sub-RQ2): patching has been tested and adopted in multiple SOTA Transformer-based TSF architectures[40, 41, 44]. Our experimental results confirm the utility of the patching mechanism by comparisons of Transformers equipped with single-scale patching , multi-scale patching, FFT-based multi-scale patching, the canonical Transformer and LSTM baseline. The results show that under the Channel Dependent (CD) setting, patching enhances the canonical Transformer's performance, allowing it to outperform the LSTM baseline for medium and long forecasting horizons. It can be observed that FFT-based multi-scale patching offers marginal benefits over standard multi-scale patching for long-term forecasts. However, single-scale patching remains the best-performing approach, outperforming both multi-scale methods with and without FFT enhancement. This result is counterintuitive, as multi-scale patching is expected to help the Transformer capture temporal patterns across different time scales. As indicated in [139], the multi-scale patching could amplify cross-channel distortions and introduce noise that ad-

versely affects the model's ability to learn stable representations from the cryptocurrency data. To validate this hypothesis, we conduct a series of additional experiments, which collectively address Sub-RQ3.

- CI vs CD (Sub-RQ3): For all the techniques used in the above experiments, we introduced CI variants to assess whether CI processing can further improve forecasting accuracy. The results indicate that the CI setting substantially enhances the performance of both single-scale and multi-scale patching Transformers. Under this configuration, the multi-scale patching Transformer achieves the best overall performance, surpassing its single-scale counterpart. This result indicates that CI can alleviate the cross-channel distortions[139] while satisfactorily modeling the cryptocurrency signal behind the time varying noise[228]. However, this is not a universal panacea: the FFT-based multi-scale patching model exhibits a decline in performance under the CI configuration. This may be attributed to the FFT operation that aggregates spectral information across all channels under the CI setting where channels are modeled independently. Therefore, the FFT-derived patching scales may no longer correspond to the dominant temporal structures of individual assets. As a result, the CI-based FFT model underperforms its CD counterpart.
- RAC (Sub-RQ4): To assess whether RAC created historical context can improve the predictive performance of both SOTA Transformer architectures and our multi-scale patching Transformer model, we conducted experiments comparing each model with its corresponding RAC-enhanced variant. The results show that RAC-enhanced historical context is not beneficial across all settings. It can be effective when the input sequence length is short, the forecasting horizon is medium or long, and the underlying architecture is designed to incorporate external contextual information (e.g., TimeXer). To our knowledge, this is the one of the early studies to apply RAC to cryptocurrency forecasting.

Through these contributions, this thesis advances the understanding of Transformer-based and retrieval-augmented strategy for cryptocurrency forecasting, offering new insights into modeling the complex and highly dynamic behavior of cryptocurrency markets.

6.2 Limitations and Future Work

Although the proposed architecture demonstrates superior performance compared with SOTA benchmark models, several limitations should be acknowledged. First, the empirical analysis focuses primarily on forecasting major cryptocurrency assets such as ADA, BTC and ETH prices using historical data from the dominant cryptocurrencies by market capitalization. To keep the study's scope focused, we do not incorporate alternative asset classes such as Exchange Traded Funds and stocks, nor do we incorporate information from market sentiment indicators. While this design captures major market dynamics, the findings may not generalize to lower liquidity altcoins. Second, the experiments rely solely on daily closing prices leaving potentially useful indicators such as on-chain data (e.g. size of the blockchain) and technical indicators (e.g. Bollinger Bands) not incorporated. These features have been shown in prior studies[27, 117] to provide valuable predictive information and may further enhance forecasting accuracy. Third, more similarity measurement methods such as DTW[61] or dot-product[58] can be utilized to generate historical context information with RAC. Finally, the proposed architecture is evaluated exclusively on cryptocurrency data. Its applicability to broader financial markets remains to be investigated.

As a result of above, future work will extend this study by pursuing the following directions:

1. Expanding the analysis to include broader set of assets and financial markets.
2. Leveraging more indicators such as technical indicator and on-chain data.
3. Incorporating additional auxiliary data such as cryptocurrency related finan-

cial news[238] and the Cryptocurrency Fear & Greed Index[239] to enrich the contextual understanding of market dynamics.

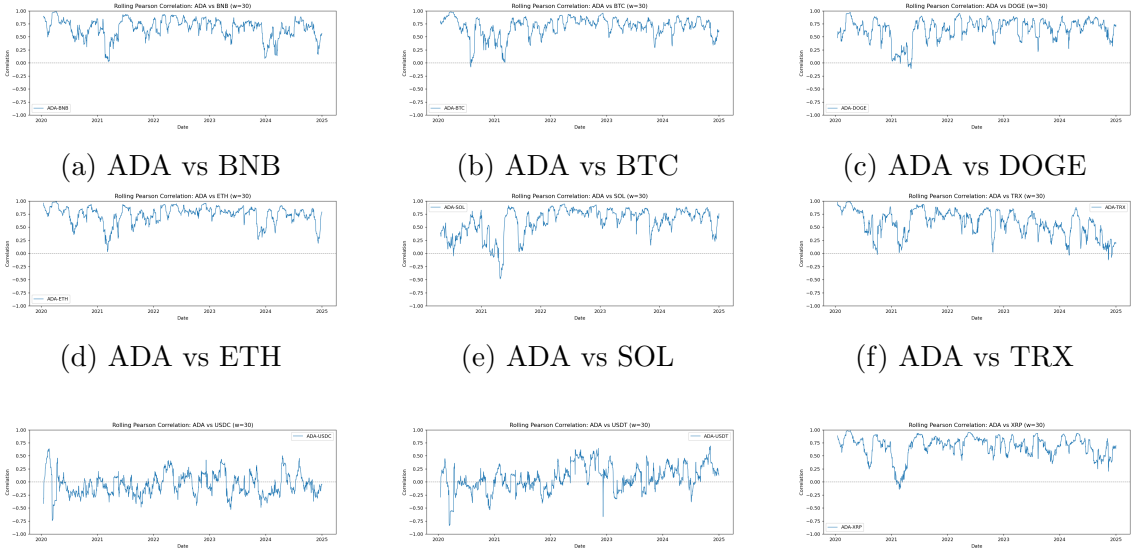
4. Exploring more retrieval methods such as DTW[61] and dot-product[58] for RAC.

5. Exploring Time-VLM ([240]) style approach into MSPFormer framework to allow it to process visual chart data (e.g., candlesticks) alongside textual sentiment and price data. This could enrich the model's understanding of structural patterns and support multi-source reasoning.

Appendix A

Data analysis

The rolling Pearson and Spearman correlation charts, as well as the average correlation heatmaps empirically demonstrate the existence of inter-correlation among the assets of the top market capitalization cryptocurrencies, with the notable exceptions of stablecoins (i.e. USDT and USDC). This exception is expected as stablecoins are engineered to maintain a stable parity with a specified fiat currency. These findings align with existing literature [196, 228] that highlights inter-correlation among dominant coins and the unique nature of stablecoin behavior.

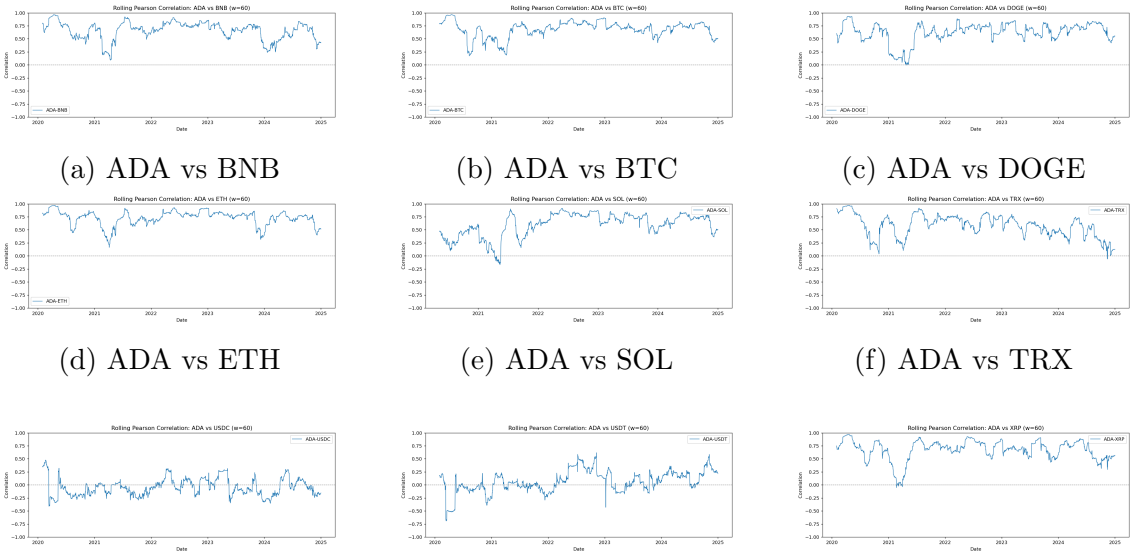


(a) ADA vs BNB (b) ADA vs BTC (c) ADA vs DOGE

(d) ADA vs ETH (e) ADA vs SOL (f) ADA vs TRX

(g) ADA vs USDC (h) ADA vs USDT (i) ADA vs XRP

Figure A.1: Rolling Pearson correlation between ADA and the other top nine cryptocurrencies using a 30 days window.

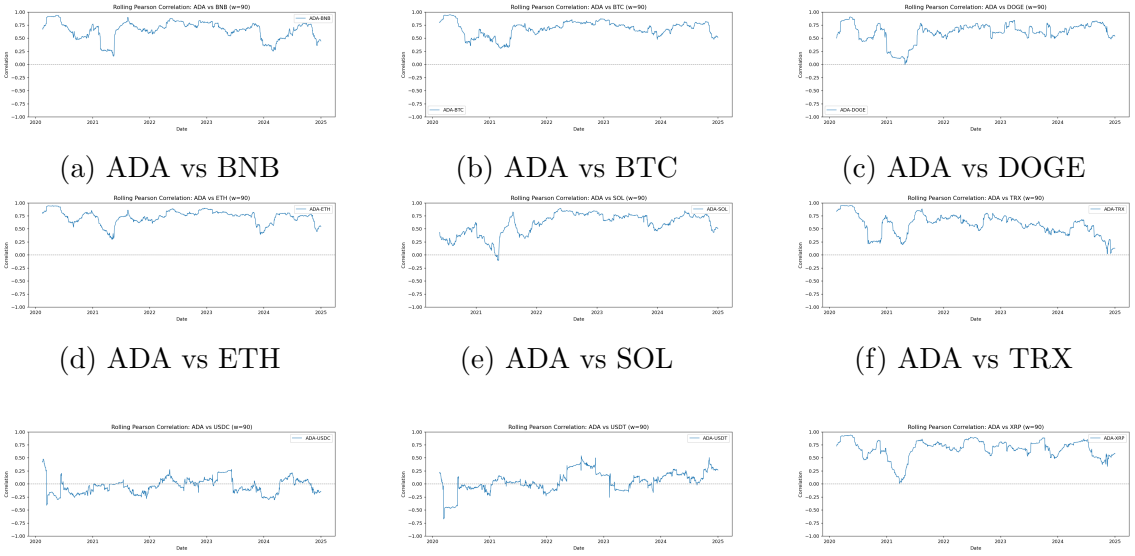


(a) ADA vs BNB (b) ADA vs BTC (c) ADA vs DOGE

(d) ADA vs ETH (e) ADA vs SOL (f) ADA vs TRX

(g) ADA vs USDC (h) ADA vs USDT (i) ADA vs XRP

Figure A.2: Rolling Pearson correlation between ADA and the other top nine cryptocurrencies using a 60 days window.

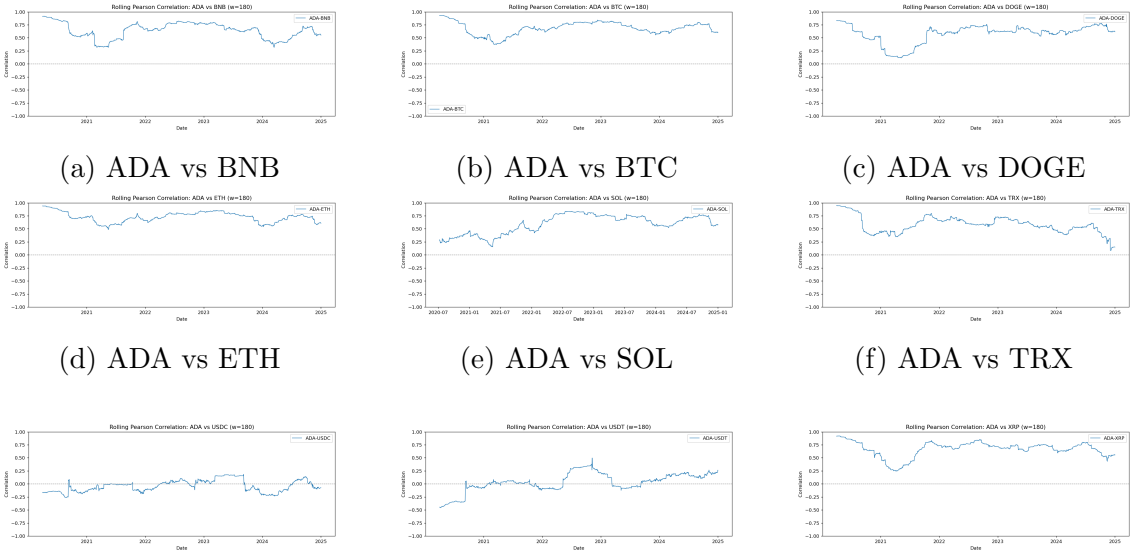


(a) ADA vs BNB (b) ADA vs BTC (c) ADA vs DOGE

(d) ADA vs ETH (e) ADA vs SOL (f) ADA vs TRX

(g) ADA vs USDC (h) ADA vs USDT (i) ADA vs XRP

Figure A.3: Rolling Pearson correlation between ADA and the other top nine cryptocurrencies using a 90 days window.

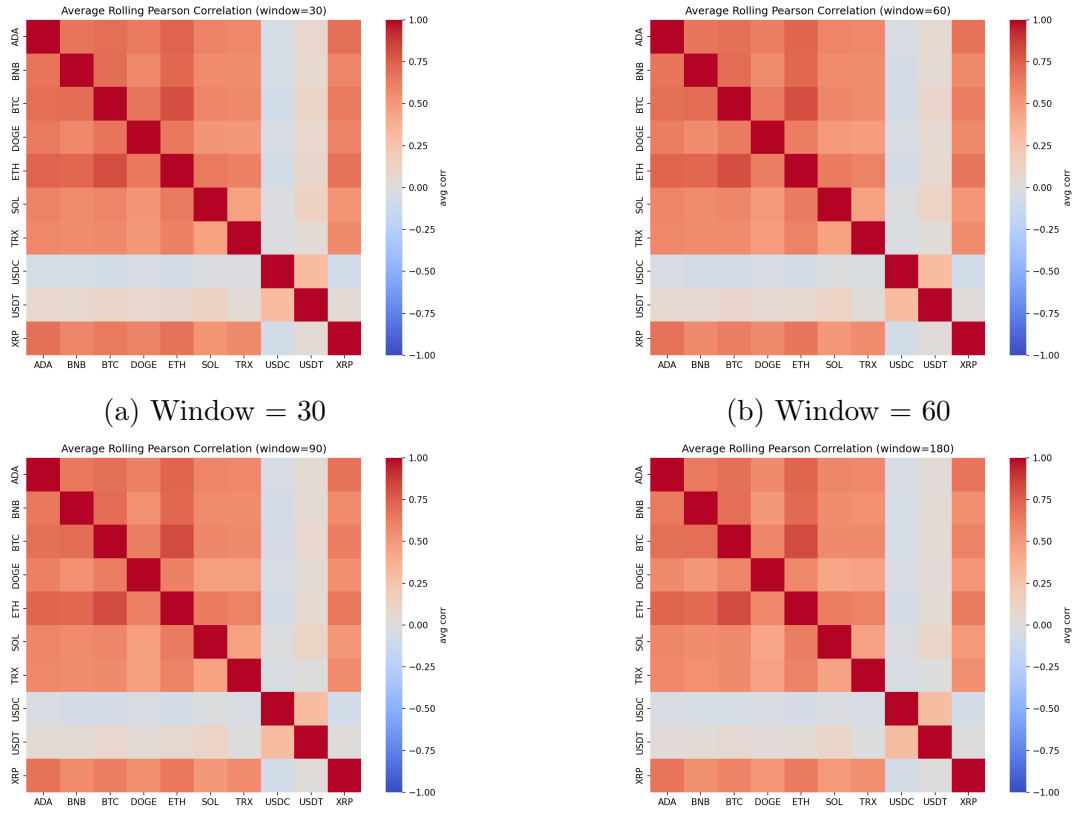


(a) ADA vs BNB (b) ADA vs BTC (c) ADA vs DOGE

(d) ADA vs ETH (e) ADA vs SOL (f) ADA vs TRX

(g) ADA vs USDC (h) ADA vs USDT (i) ADA vs XRP

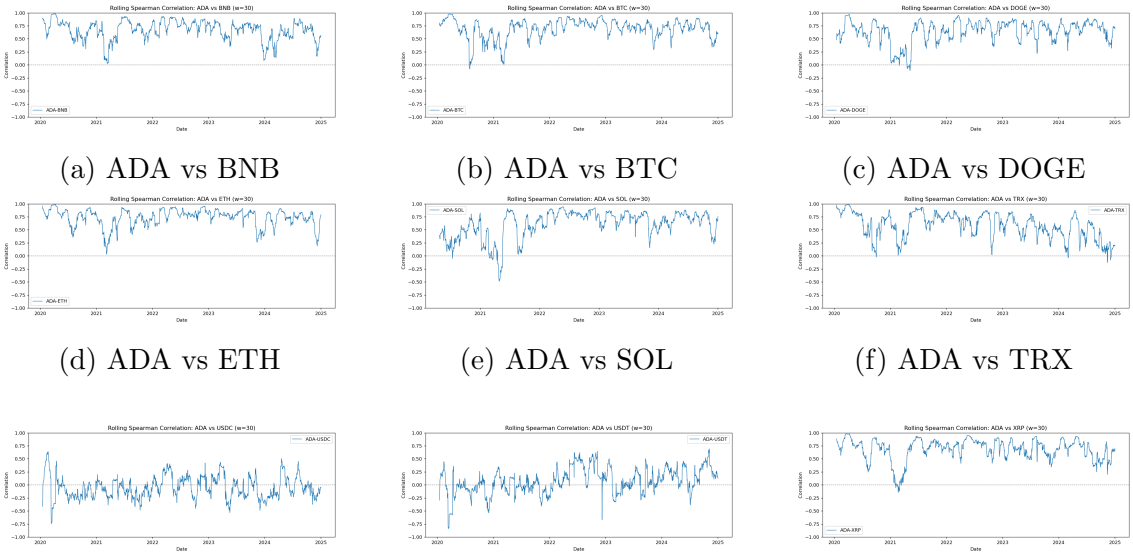
Figure A.4: Rolling Pearson correlation between ADA and the other top nine cryptocurrencies using a 180 days window.



(a) Window = 30 (b) Window = 60

(c) Window = 90 (d) Window = 180

Figure A.5: Average rolling Pearson correlation heatmaps with window sizes of 30, 60, 90, and 180 days.

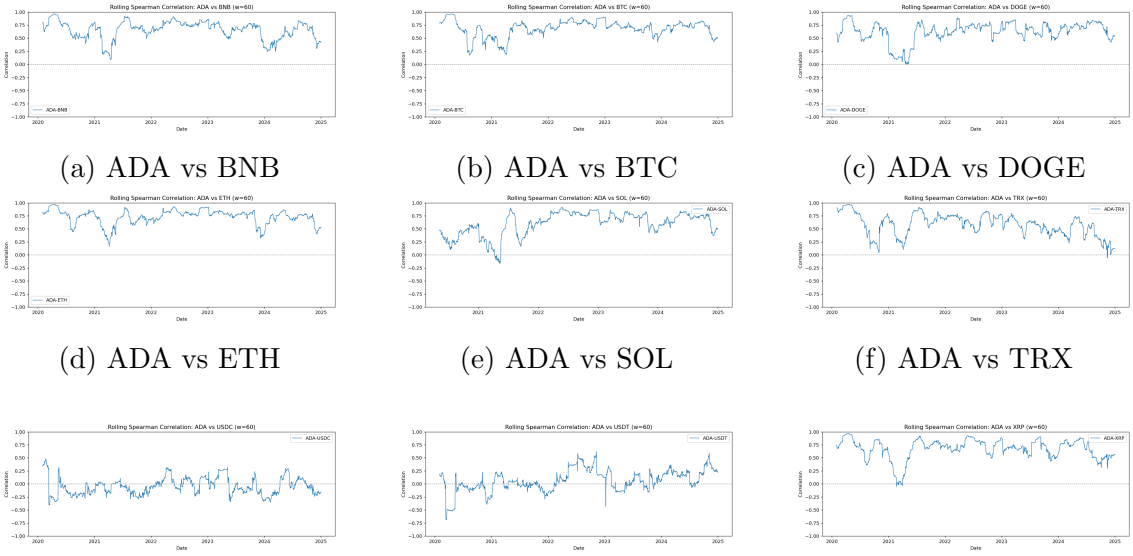


(a) ADA vs BNB (b) ADA vs BTC (c) ADA vs DOGE

(d) ADA vs ETH (e) ADA vs SOL (f) ADA vs TRX

(g) ADA vs USDC (h) ADA vs USDT (i) ADA vs XRP

Figure A.6: Rolling Spearman correlation between ADA and the other top nine cryptocurrencies using a 30 days window.

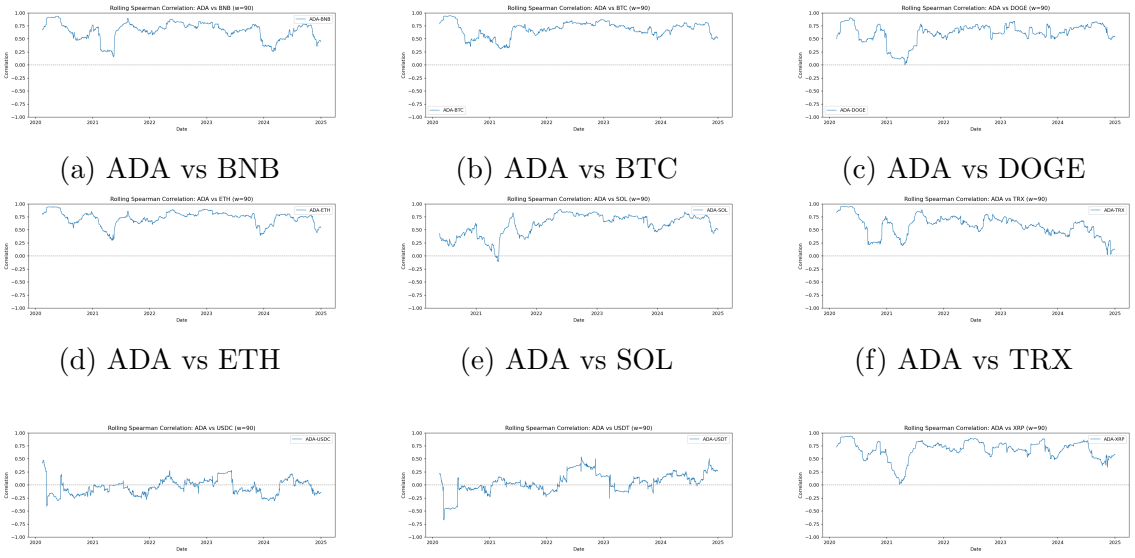


(a) ADA vs BNB (b) ADA vs BTC (c) ADA vs DOGE

(d) ADA vs ETH (e) ADA vs SOL (f) ADA vs TRX

(g) ADA vs USDC (h) ADA vs USDT (i) ADA vs XRP

Figure A.7: Rolling Spearman correlation between ADA and the other top nine cryptocurrencies using a 60 days window.

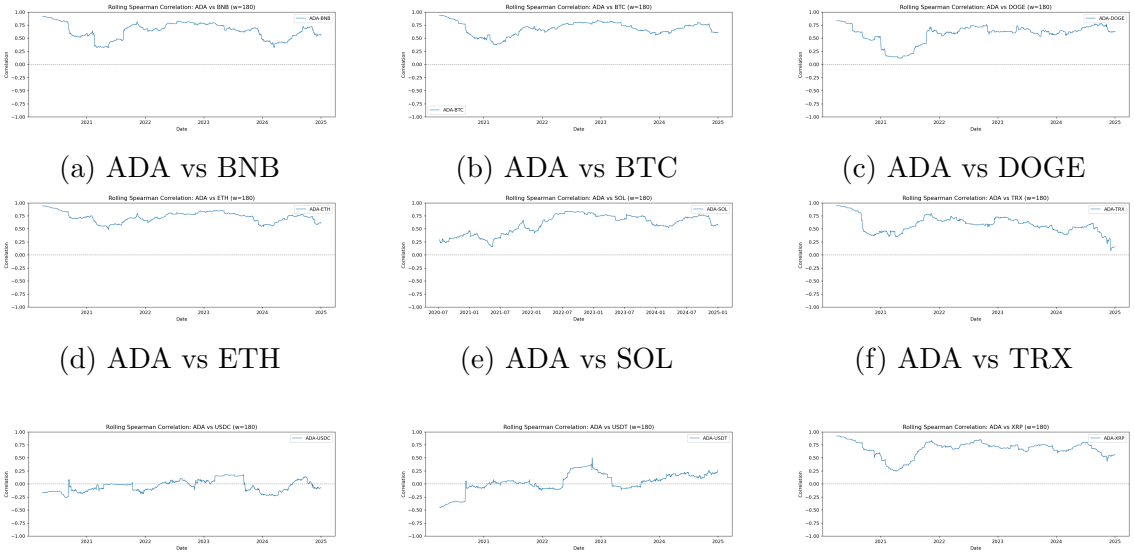


(a) ADA vs BNB (b) ADA vs BTC (c) ADA vs DOGE

(d) ADA vs ETH (e) ADA vs SOL (f) ADA vs TRX

(g) ADA vs USDC (h) ADA vs USDT (i) ADA vs XRP

Figure A.8: Rolling Spearman correlation between ADA and the other top nine cryptocurrencies using a 90 days window.

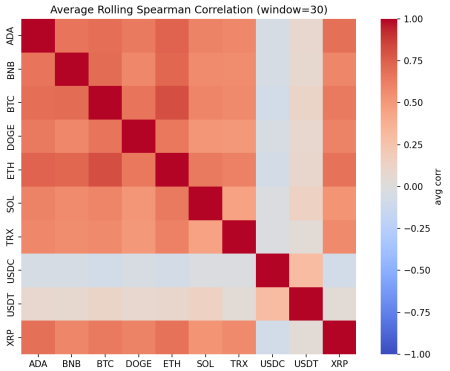


(a) ADA vs BNB (b) ADA vs BTC (c) ADA vs DOGE

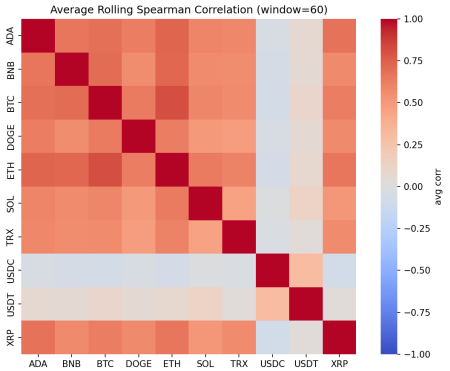
(d) ADA vs ETH (e) ADA vs SOL (f) ADA vs TRX

(g) ADA vs USDC (h) ADA vs USDT (i) ADA vs XRP

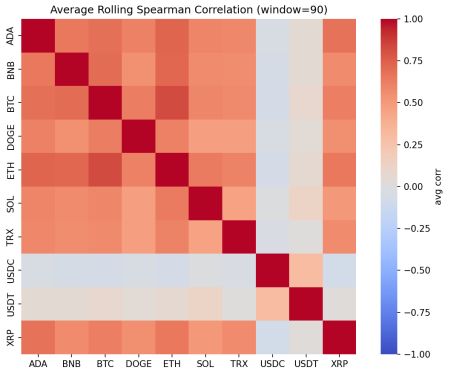
Figure A.9: Rolling Spearman correlation between ADA and the other top nine cryptocurrencies using a 180 days window.



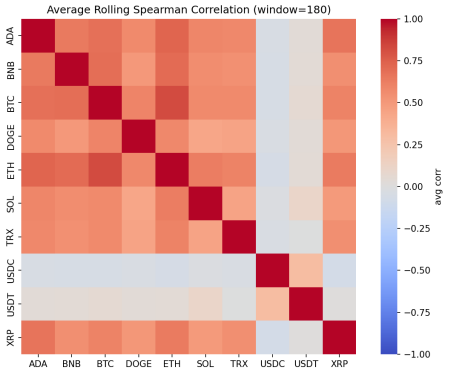
(a) Window = 30



(b) Window = 60



(c) Window = 90



(d) Window = 180

Figure A.10: Average rolling Spearman correlation heatmaps with window sizes of 30, 60, 90, and 180 days.

Appendix B

A history of Transformer

B.1 Transformer

The Transformer model, originally introduced by Vaswani et al. in the seminal work "Attention is All You Need"[20], has fundamentally transformed sequential data modeling by eliminating recurrence and instead relying entirely on a self-attention mechanism[20].

Unlike RNNs and their variants such as LSTM networks, which process input sequences in a step-by-step manner and often suffer from vanishing gradient issues, the Transformer processes entire sequences in parallel[241]. This parallelization not only accelerates training but also enables the model to capture long range dependencies more effectively[241], although, in TSF, studies[38, 127, 128] indicates that Transformer's effectivenss is constrained because it is less effective in modeling inherent sequence ordering and temporal continuity compared with RNN-based models. By contrast, LSTMs, despite mitigating some limitations of vanilla RNNs by retaining information over extended time lags, struggle with very long sequences and remain computationally inefficient due to their sequential nature[242]. Transformers address both of these challenges through the use of multi-head self-attention, which allows the model to weigh the relevance of each input element to every other element in the sequence[243, 244]. This mechanism provides a global view of the input data at each layer, facilitating the learning of both short-term and long-term dependen-

cies in time series[31]. Furthermore, the incorporation of positional encoding with sinusoidal functions compensates for the lack of inherent sequence ordering in Transformer[20]. It makes Transformers particularly well suited for temporal forecasting tasks such as the prediction of volatile financial time series such as cryptocurrency prices.

To fully understand the Transformer architecture and its evolution from the Sequence-to-Sequence (Seq2Seq) framework[65], it is essential to first examine the development of the attention mechanism, which forms the core of the Transformer's design. The following section traces the historical progression of attention mechanisms, beginning with their introduction in the Seq2Seq model.

B.1.1 A history of attention: from fixed vector to Transformer's scale-dot attention

Seq2Seq

While the Transformer architecture has become a popular approach and serves as the foundational architecture behind large language models, its foundation can be traced back to earlier models based on the **encoder–decoder** paradigm. One of the most influential among these is the Sequence-to-Sequence (Seq2Seq) model, introduced by Sutskever et al.[65] in 2014. In this framework, both the encoder and decoder are multi-layered LSTMs. The encoder processes an input sequence $(x_1, \ldots, x_T)$ and encodes it into a **fixed-length context vector** v. The decoder then generates the output sequence $(y_1, \ldots, y_{T'})$ from this vector and the previously generated tokens.

Attention

The fixed size vector in the original Seq2Seq model facilitates sequence modeling without requiring fixed input or output lengths. However, when it comes to long sequences it is a bottleneck for improving performance[245], as it compresses all contextual information into a single vector representation, often leading to the loss of information, particularly from the earlier parts of the input sequence. Bahdanau

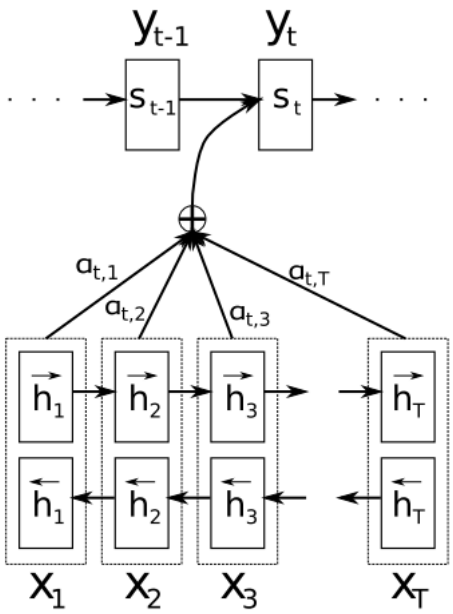


Figure B.1: Attention architecture proposed by Bahdanau et al.[245]

et al.[245] proposed an **attention mechanism** that replaces the fixed size context vector with a dynamic, learnable alignment between the encoder and decoder. This attention-based approach allows the decoder to selectively focus on relevant parts of the input sequence at each decoding step, resulting in improved performance.

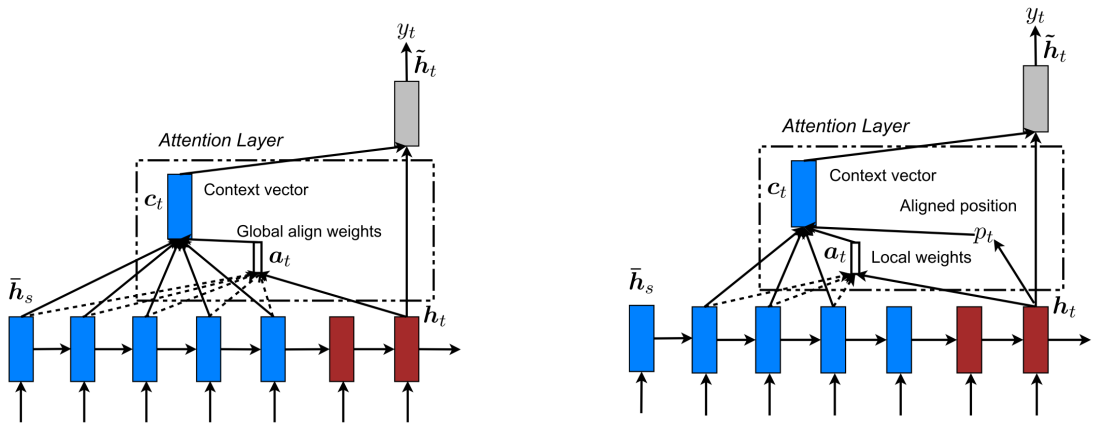
As illustrated in Figure B.1, the encoder transforms the input sequence x into a set of annotation vectors h using a bi-directional RNN. These annotations are then sum-weighted into a context vector c using equation $c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j$ where i is the i-th word the decoder is generating, and j is the j-th word that was encoded by the encoder. h_j is the hidden state produced by the encoder at the j-th position of the input sequence. α_{ij} are attention weights (alignment scores) that indicate how relevant annotation h_j is when predicting output token y_i. The decoder generates each output token y_t by leveraging the context vector c, the previous decoder hidden state s_{t-1}, and the previously generated output y_{t-1}. The use of a bi-directional RNN in the encoder enables the model to compute hidden states in both forward and backward directions, enriching each annotation h_j with contextual information from both inputs before and after[245].

This architecture defines the attention mechanism within the decoder, wherein

the next hidden state s_t and output y_t depends not only on the previous hidden state s_{t-1} but also on a dynamically computed context vector that reflects the most relevant portions of the input sequence. By eliminating the reliance on a fixed-size context vector, the attention mechanism enables the model to more effectively handle long and complex sequences[245].

Global attention vs local attention. Depending on whether all encoder hidden states h are considered in computing the context vector c_t, attention mechanisms can be broadly categorized into global attention and local attention[246]. As illustrated in Figure B.2a, global attention computes the alignment vector a_t at each decoding step t by attending to all encoder hidden states. The context vector c_t is then obtained as a weighted sum of these hidden states using the alignment scores.

Using all the hidden states can be expensive and can potentially render it impractical for tasks with long sequences, e.g. translating paragraphs or documents[246]. To mitigate this issue, local attention restricts the computation to a fixed size window of encoder states centered around a predicted aligned position p_t, as shown in Figure B.2b, the alignment vector a_t is computed only over encoder hidden states within the window [p_t-D, p_t+D], where D is a predefined window size. The resulting context vector c_t is a weighted average over this localized subset, making a_t a fixed-length vector.



(a) Global attention (b) Local attention

Figure B.2: Global and local attention mechanisms[246]

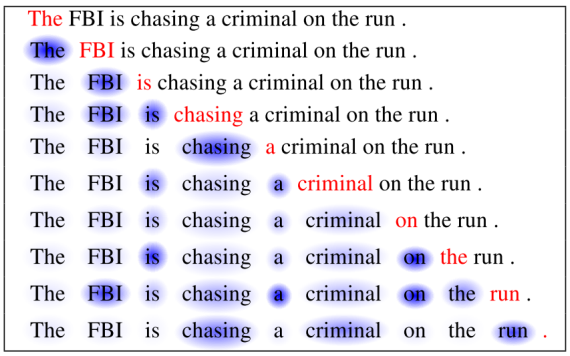


Figure B.3: Demonstration of Self-Attention in action[247]

Self-attention. While traditional attention mechanisms like global and local attention operate between separate encoder and decoder sequences, self-attention extends the idea of global attention within a single sequence. Self-attention is also referred as *intra-attention*[247]. It is a mechanism that computes representations of a sequence by attending to different positions within the same sequence. In this approach, each token is encoded by incorporating contextual information from other relevant tokens in the sequence, allowing the model to capture long-range dependencies and internal relationships. As illustrated in Figure B.3, self-attention enables the model to dynamically assign different levels of importance to previous tokens when processing the current token. This is particularly useful in tasks such as machine reading comprehension, where understanding a word often depends on its broader context. The model processes text incrementally while learning which past tokens in the memory and to what extent they relate to the current token being processed[247].

Variations of the Attention Mechanism There are several variants of attention mechanisms, distinguished primarily by how the alignment score a is calculated. The alignment score is a measure of how well a hidden state from the source sequence aligns with a hidden state from the target sequence[248].

In Bahdanau et al.'s work[245], the alignment score is calculated using a feed-forward neural network. This formulation is referenced as concat in Luong's research[246] and additive addition in Vaswani's Transformer work[20]. Luong pro-

posed the dot product attention written in $score(h_t, h_s) = h_t^T \bar{h_s}$ where h_t^T represents target hidden state and $\bar{h_s}$is the source hidden state. This computes "how well-aligned" the two vectors are via their raw inner-product. Their experiments highlighted dot product attention for its computational efficiency and suitability for large scale tasks comparing with the concat attention. Vaswani et al. also pointed out that dot production attention is much faster and more space efficient in practice, since it can be implemented using highly optimized matrix multiplication code[20].

Building on these prior works, Vaswani et al.[20] introduced the scaled dot product attention mechanism as a core component of the Transformer architecture. This method modifies standard dot product attention by scaling the raw dot product of the *query* and *key* vectors by the inverse square root of the key dimension d_k (*query*, *key* vectors will be described in the following section), resulting in the following scoring function: $score(h_t, h_s) = \frac{h_t^T \bar{h_s}}{\sqrt{n}}$. This scaling addresses the issue of extremely large dot product values when the dimensionality of the input is high, which can lead to vanishing gradients after the softmax operation. The scaled dot product formulation not only stabilizes training but also leverages optimized matrix multiplication routines, enabling the Transformer to achieve both high accuracy and computational efficiency.

B.1.2 Transformer architecture

The overall architecture of the Transformer is illustrated in Figure B.4. Similar to the traditional Seq2Seq model[65], the Transformer follows an encoder–decoder structure. In the architecture, the input tokens, such as words in a sentence or time steps in a time series, are first converted into input embeddings, which are dense vector representations that capture semantic meaning. To preserve the orders of the tokens, positional encodings are added to these embeddings. This allows the model to understand the relative and absolute positions of tokens in the sequence[249]. For each input token, the model computes three vectors: *query (Q)*, *key (K)*, and *value*

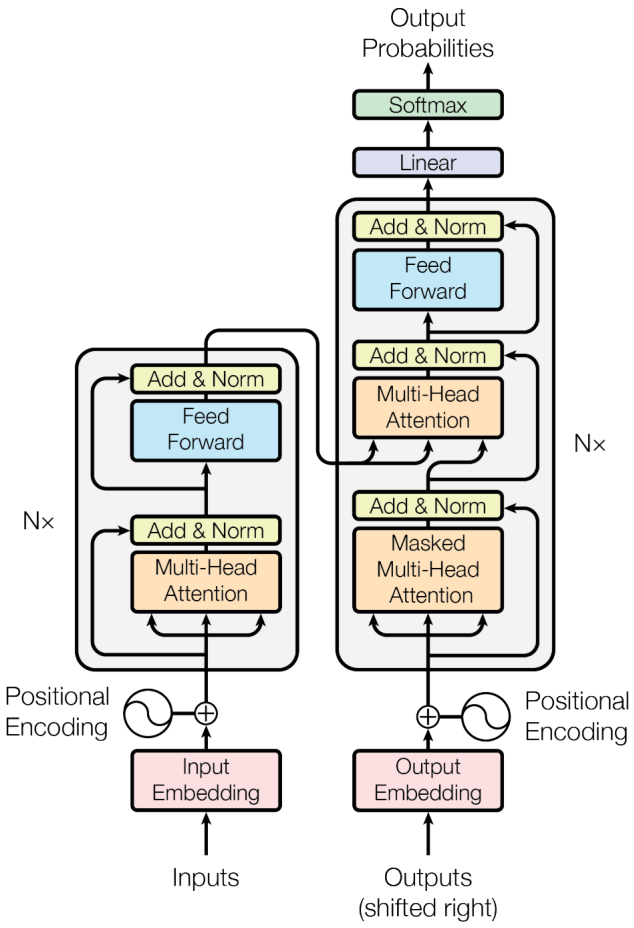


Figure B.4: The canonical Transformer architecture[20]

(V).[1] The resulting vectors are then passed into the multi-head self-attention layer, which enables the model to compute relationships between each token and every other token in the sequence. This mechanism allows the model to dynamically weigh the importance of other tokens when generating a representation for a given token[20]. The output of the self-attention layer is then passed through a position wise feed forward neural network, which is applied independently to each token. Its role is to transform and project the contextualized representation into a new space, allowing the model to learn higher level features for each token after incorporating context via self-attention[250].

The multi-head self-attention mechanism projects the queries, keys, and values h times using distinct learned linear projections. Each projected set is passed through an independent scaled dot product attention operation, Equation B.2 and B.3. The outputs from each attention head are then concatenated and projected again to form the final output, EquationB.1. This allows the model to jointly attend to information from different representation subspaces.

The decoder mirrors the encoder's structure with a key difference: it contains an additional multi-head attention mechanism that attends to the encoder's output representations. This enables the decoder to condition its predictions on the encoded source sequence. Finally, the decoder output is passed through a linear transformation followed by a softmax layer to project the representation into a probability distribution over the target vocabulary.

$$MultiHead(Q, K, V) = Concat(head_1, ..., head_h)W^o \tag{B.1}$$

where

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V) \tag{B.2}$$

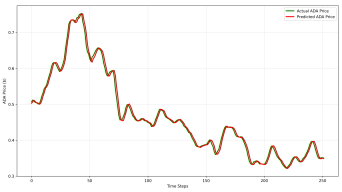
[1]The concept of key, value, and query in the attention mechanism is analogous to information retrieval. Consider a library scenario: if you are searching for books on computers, your query is "computer books." The library's index cards, which list topics such as ML, data mining, and system design, serve as keys. The actual books corresponding to these indexed topics represents the values.

$$Attention(Q, K, V) = Softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{B.3}$$

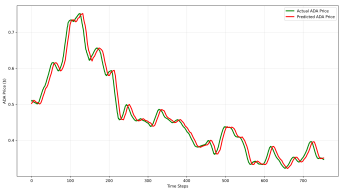
Appendix C

Predicted vs. True Value Plots for ADA, BTC and ETH forecasting results

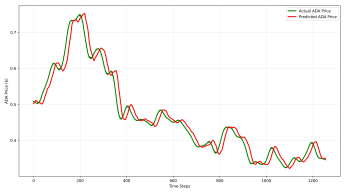
The figures in this section illustrate a comparison of MSPRACFormer's and the benchmark model's (PatchTST, iTransformer, MSGNet, TimeXer, LSTM, and DLinear) prediction and actual values for predicting ADA, BTC and ETH across various forecasting horizons.



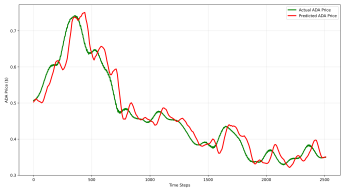
(a) Prediction horizon=1 day



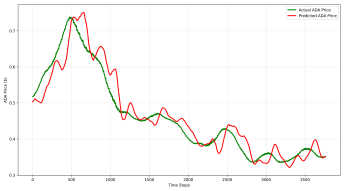
(b) Prediction horizon=3 days



(c) Prediction horizon=5 days

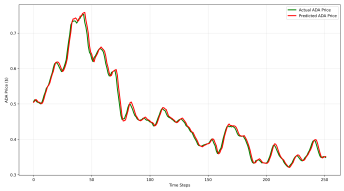


(d) Prediction horizon=10 days

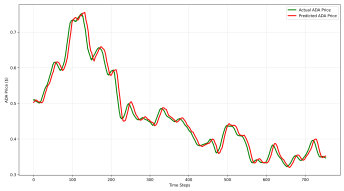


(e) Prediction horizon=15 days

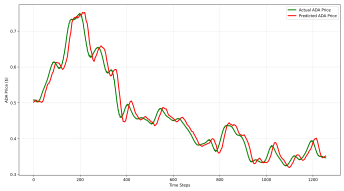
Figure C.1: ADA prediction with MSPRACFormer



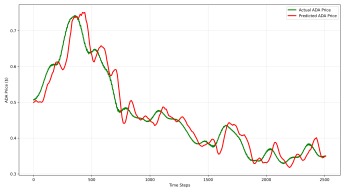
(a) Prediction horizon=1 day



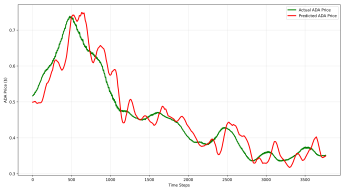
(b) Prediction horizon=3 days



(c) Prediction horizon=5 days

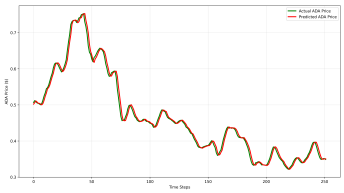


(d) Prediction horizon=10 days

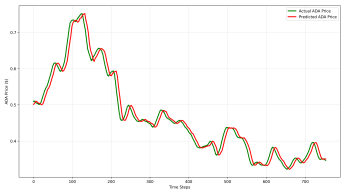


(e) Prediction horizon=15 days

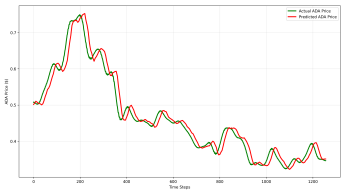
Figure C.2: ADA prediction with iTransformer



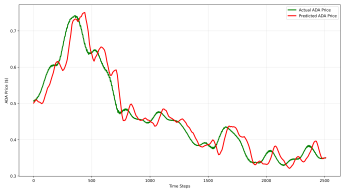
(a) Prediction horizon=1 day



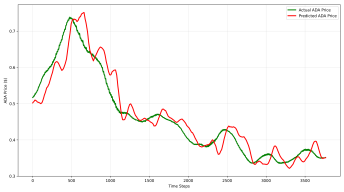
(b) Prediction horizon=3 days



(c) Prediction horizon=5 days

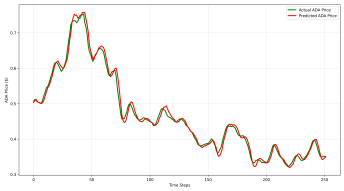


(d) Prediction horizon=10 days

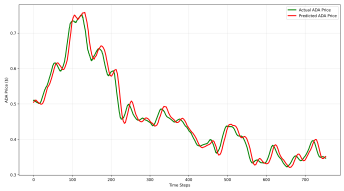


(e) Prediction horizon=15 days

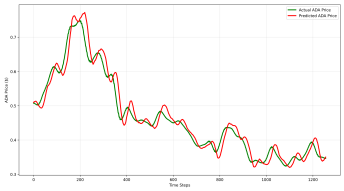
Figure C.3: ADA prediction with PatchTST



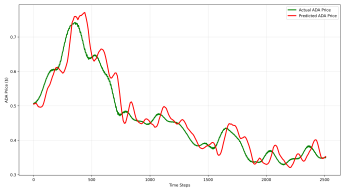
(a) Prediction horizon=1 day



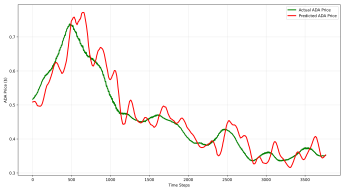
(b) Prediction horizon=3 days



(c) Prediction horizon=5 days

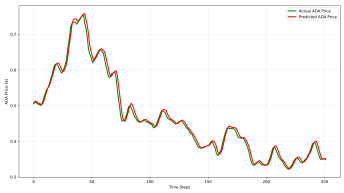


(d) Prediction horizon=10 days

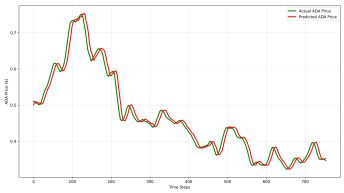


(e) Prediction horizon=15 days

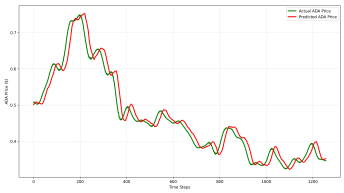
Figure C.4: ADA prediction with MSGNet



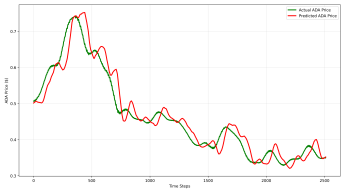
(a) Prediction horizon=1 day



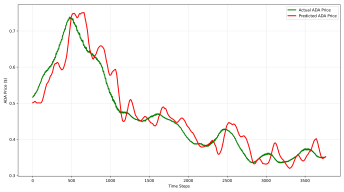
(b) Prediction horizon=3 days



(c) Prediction horizon=5 days

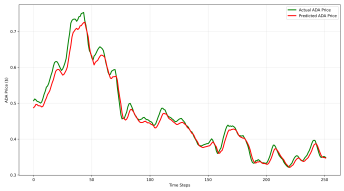


(d) Prediction horizon=10 days

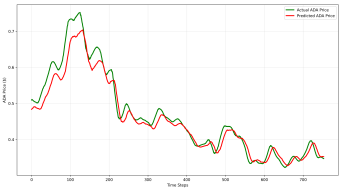


(e) Prediction horizon=15 days

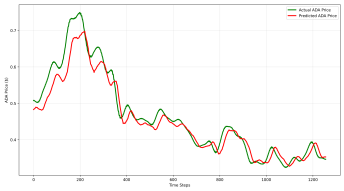
Figure C.5: ADA prediction with TimeXer



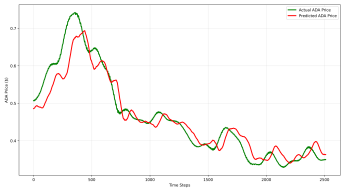
(a) Prediction horizon=1 day



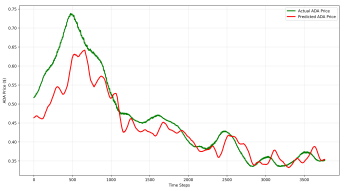
(b) Prediction horizon=3 days



(c) Prediction horizon=5 days

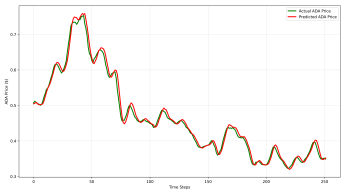


(d) Prediction horizon=10 days

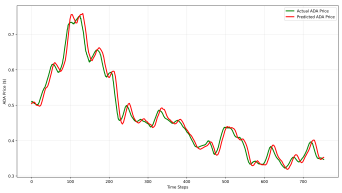


(e) Prediction horizon=15 days

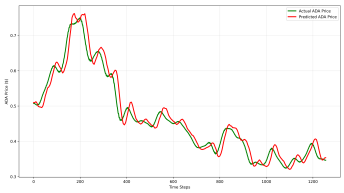
Figure C.6: ADA prediction with DLinear



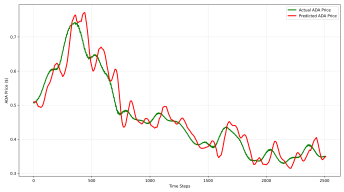
(a) Prediction horizon=1 day



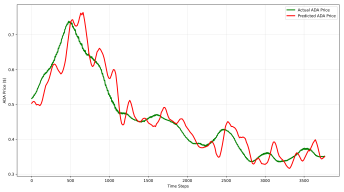
(b) Prediction horizon=3 days



(c) Prediction horizon=5 days

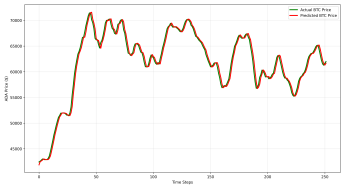


(d) Prediction horizon=10 days

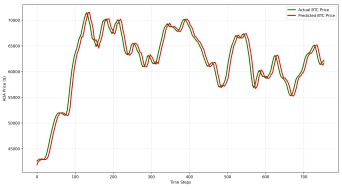


(e) Prediction horizon=15 days

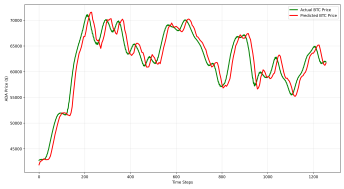
Figure C.7: ADA prediction with LSTM



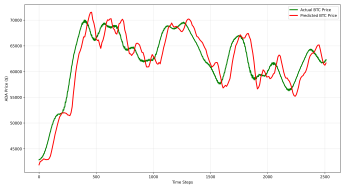
(a) Prediction horizon=1 day



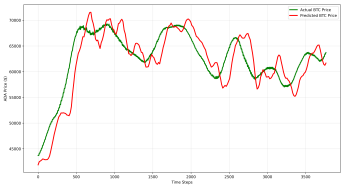
(b) Prediction horizon=3 days



(c) Prediction horizon=5 days

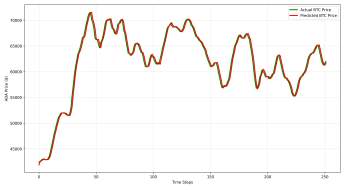


(d) Prediction horizon=10 days

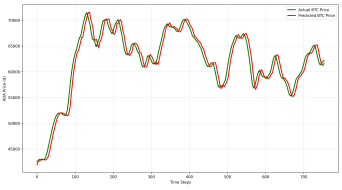


(e) Prediction horizon=15 days

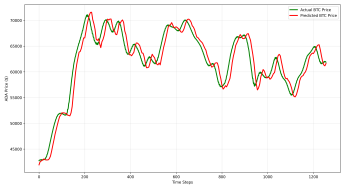
Figure C.8: BTC prediction with MSPRACFormer



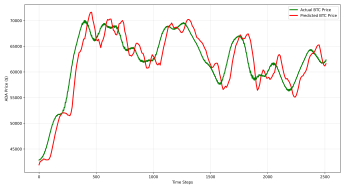
(a) Prediction horizon=1 day



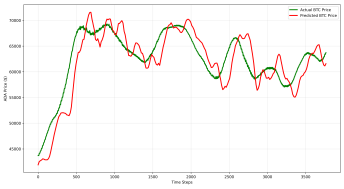
(b) Prediction horizon=3 days



(c) Prediction horizon=5 days

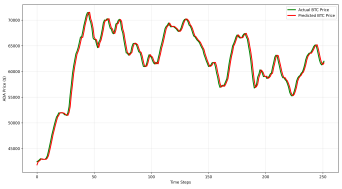


(d) Prediction horizon=10 days

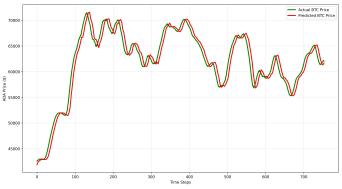


(e) Prediction horizon=15 days

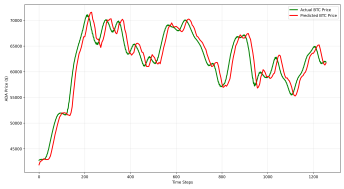
Figure C.9: BTC prediction with iTransformer



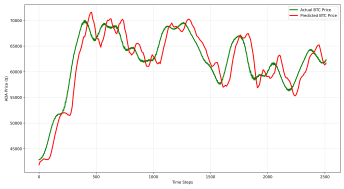
(a) Prediction horizon=1 day



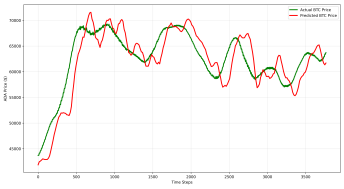
(b) Prediction horizon=3 days



(c) Prediction horizon=5 days

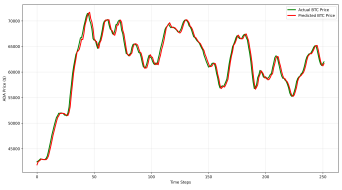


(d) Prediction horizon=10 days

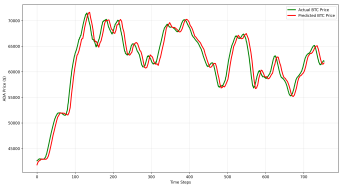


(e) Prediction horizon=15 days

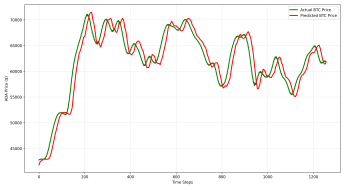
Figure C.10: BTC prediction with PatchTST



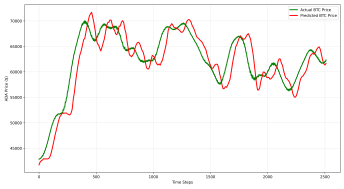
(a) Prediction horizon=1 day



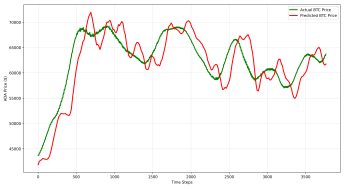
(b) Prediction horizon=3 days



(c) Prediction horizon=5 days

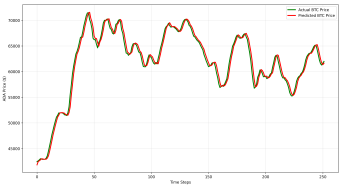


(d) Prediction horizon=10 days

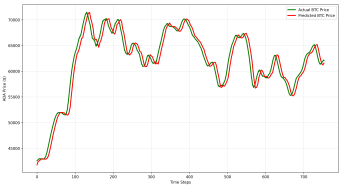


(e) Prediction horizon=15 days

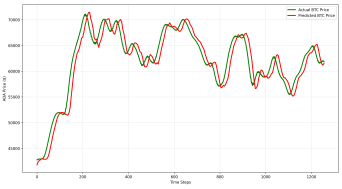
Figure C.11: BTC prediction with MSGNet



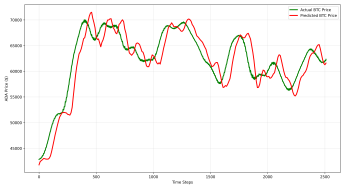
(a) Prediction horizon=1 day



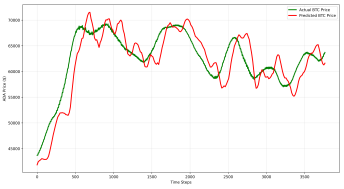
(b) Prediction horizon=3 days



(c) Prediction horizon=5 days

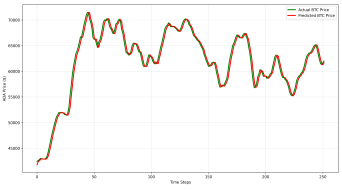


(d) Prediction horizon=10 days

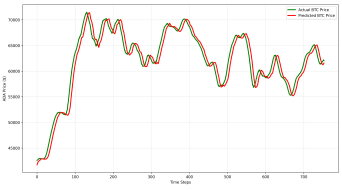


(e) Prediction horizon=15 days

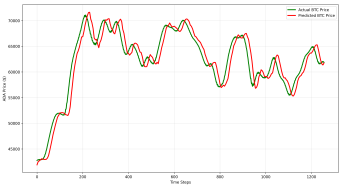
Figure C.12: BTC prediction with TimeXer



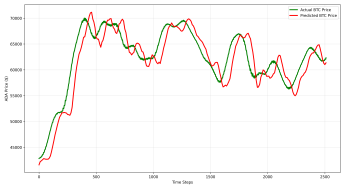
(a) Prediction horizon=1 day



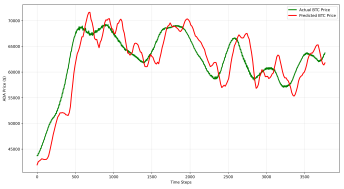
(b) Prediction horizon=3 days



(c) Prediction horizon=5 days

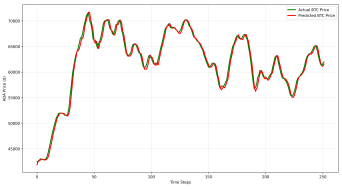


(d) Prediction horizon=10 days

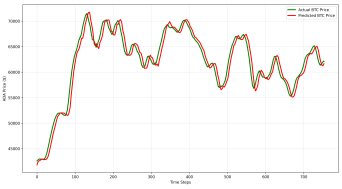


(e) Prediction horizon=15 days

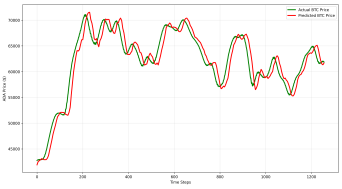
Figure C.13: BTC prediction with DLinear



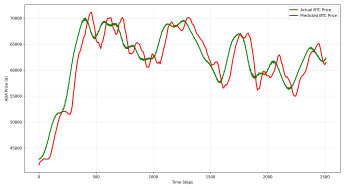
(a) Prediction horizon=1 day



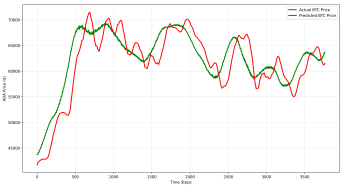
(b) Prediction horizon=3 days



(c) Prediction horizon=5 days

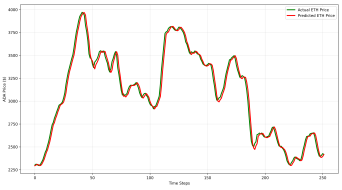


(d) Prediction horizon=10 days

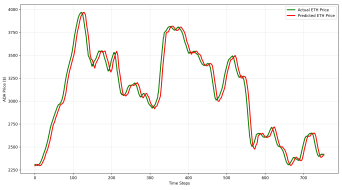


(e) Prediction horizon=15 days

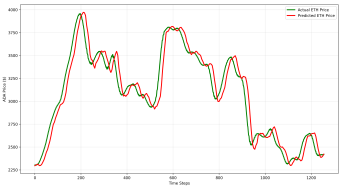
Figure C.14: BTC prediction with LSTM



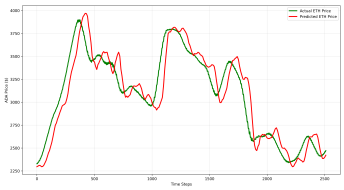
(a) Prediction horizon=1 day



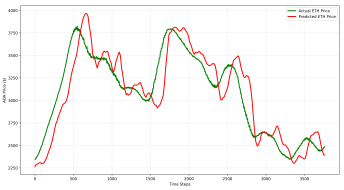
(b) Prediction horizon=3 days



(c) Prediction horizon=5 days

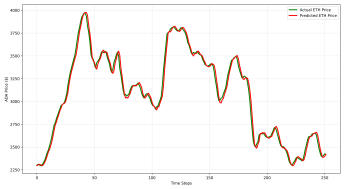


(d) Prediction horizon=10 days

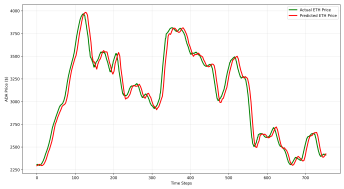


(e) Prediction horizon=15 days

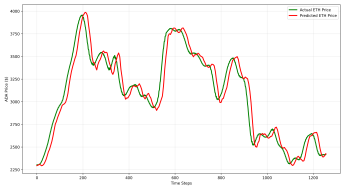
Figure C.15: ETH prediction with MSPRACFormer



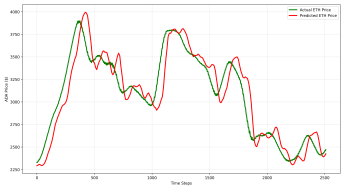
(a) Prediction horizon=1 day



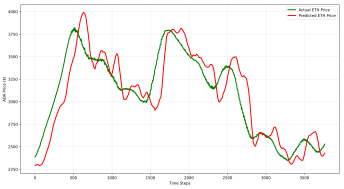
(b) Prediction horizon=3 days



(c) Prediction horizon=5 days

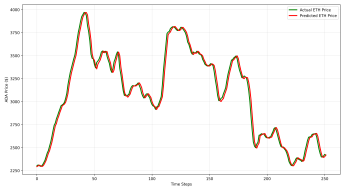


(d) Prediction horizon=10 days

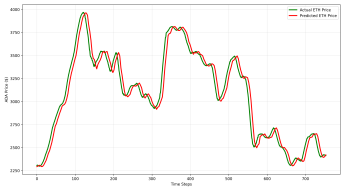


(e) Prediction horizon=15 days

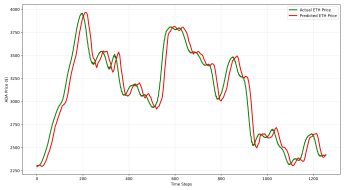
Figure C.16: ETH prediction with iTransformer



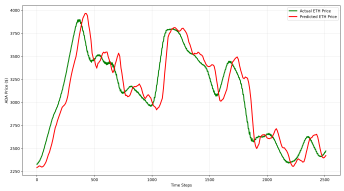
(a) Prediction horizon=1 day



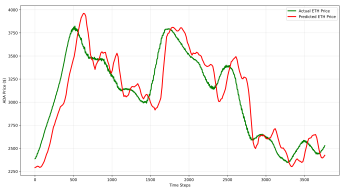
(b) Prediction horizon=3 days



(c) Prediction horizon=5 days

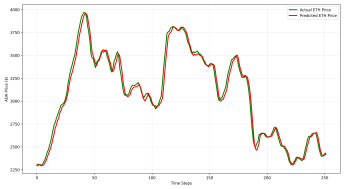


(d) Prediction horizon=10 days

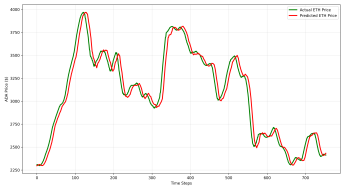


(e) Prediction horizon=15 days

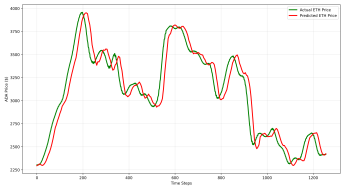
Figure C.17: ETH prediction with PatchTST



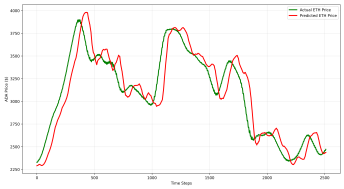
(a) Prediction horizon=1 day



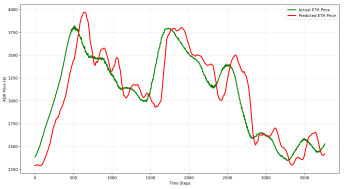
(b) Prediction horizon=3 days



(c) Prediction horizon=5 days

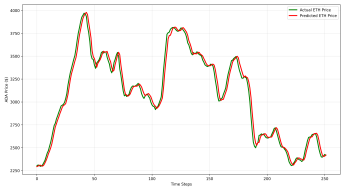


(d) Prediction horizon=10 days

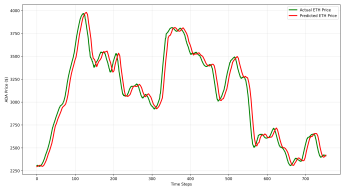


(e) Prediction horizon=15 days

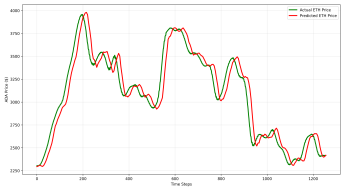
Figure C.18: ETH prediction with MSGNet



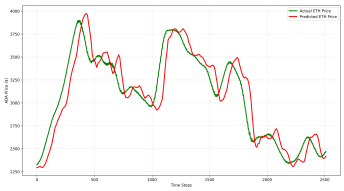
(a) Prediction horizon=1 day



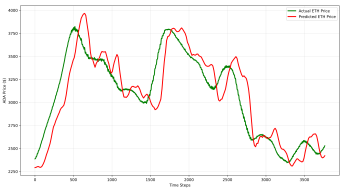
(b) Prediction horizon=3 days



(c) Prediction horizon=5 days

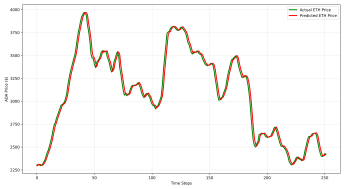


(d) Prediction horizon=10 days

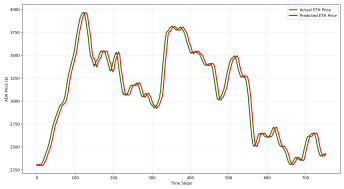


(e) Prediction horizon=15 days

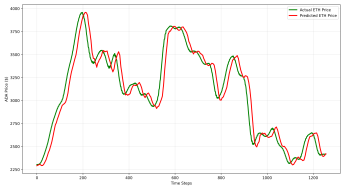
Figure C.19: ETH prediction with TimeXer



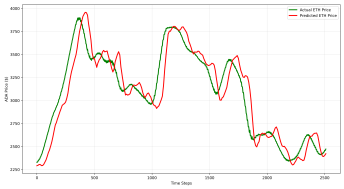
(a) Prediction horizon=1 day



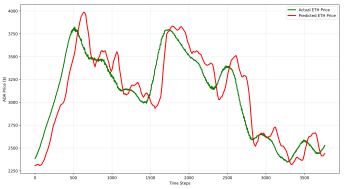
(b) Prediction horizon=3 days



(c) Prediction horizon=5 days

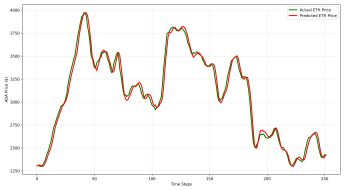


(d) Prediction horizon=10 days

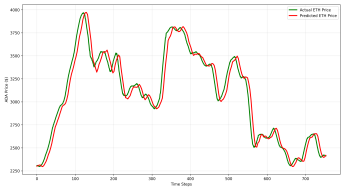


(e) Prediction horizon=15 days

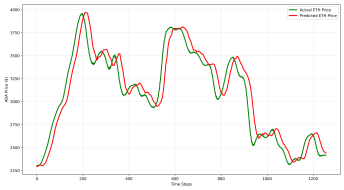
Figure C.20: ETH prediction with DLinear



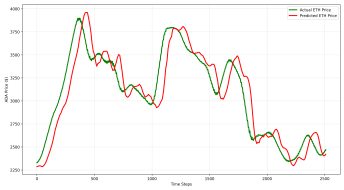
(a) Prediction horizon=1 day



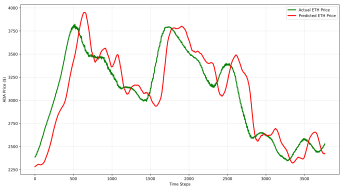
(b) Prediction horizon=3 days



(c) Prediction horizon=5 days



(d) Prediction horizon=10 days



(e) Prediction horizon=15 days

Figure C.21: ETH prediction with LSTM

Appendix D

Artefects

The implementation of the proposed framework is available at `https://github.com/cmajorsolo/MSPRACFormer`. As noted earlier, to facilitate fair and widely comparable model evaluation, this project is built upon the widely adopted TSF based framework created by Zhou et al.[38]. Figure D.1 overview the project structure. The repository is organized into modular structure to facilitate usability and extensibility. The *data* directory stores the raw input datasets. Model definitions such as MSPRACFormer architecture implemented in Python are located in the *models* directory. The *layers* directory contains reusable neural network components shared across different models. Data loading and preprocessing routines are implemented in *data_provider* directory. The implementation of the RAC mechanism, including database building and historical data retrieval, is provided in the *rac* directory. The *exp* directory contains the code defines training, validation and testing experiments. Finally, the utility functions such as evaluation metrics are contained in *util* directory.

```
MSPRACFormer/
├── data/
├── models/
│   ├── MSPRACFormer.py
│   ├── LSTMModel.py
│   └── Transformer.py
├── layers/
├── data_provider/
├── rac/
├── exp/
├── utils/
├── requirements.txt
└── run_longExp.py
```

Figure D.1: Directory structure of the MSRACFormer framework

This project is implemented in Python using the PyTorch deep learning framework and a set of supporting libraries that provide essential functionalities, such as NumPy for numerical computation. A complete list of required dependencies and their corresponding versions is specified in the `requirements.txt` file. The prerequisites can be installed by executing the command `pip install -r requirements.txt`.

The entry point of this project is `run_longExp.py`, in which all hyperparameters are specified and configured. The key hyperparameters of MSPRACFormer are summarized in Table D.1. To launch model training, the main script `run_longExp.py` is executed with the desired hyperparameters supplied as command line arguments. An example command is: `python run_longExp.py -model MSPRACFormer -data rag -use_rag 1 -chunk_size 7 -k_retrieval 1`. Additional configuration details are documented in the repository and the associated README file. The implementation is structured to be self explanatory, with descriptive comments provided throughout the code to support comprehension and reuse.

Hyperparameter	Description
–model	Specifies the model used for training or testing.
–data	Specifies the dataset type.
–root_path	Root directory containing the dataset files.
–data_path	File name or path of the dataset to be loaded.
–chunk_size	Length of each chunk used to construct the RAC database.
–k_retrieval	Number of RAC chunks retrieved to form contextual information.
–use_rag	Flag indicating whether the RAC component is enabled.
–target	Name of the target variable to be forecast.
–seq_len	Length of the input sequence provided to the model.
–pred_len	Length of the prediction horizon.
–scale_list	List of patching scales used for multi-scale representation.
–top_k	Number of patching scales selected for training.
–enc_in	Number of input channels to the encoder.
–c_out	Number of output channels produced by the model.
–n_vars	Total number of channels in the dataset.
–n_heads	Number of attention heads in the Transformer encoder.
–e_layers	Number of encoder layers.
–d_ff	Dimensionality of the feed-forward network in the Transformer encoder.
–is_training	Flag indicating whether the model in training model.
–dropout	Dropout rate applied during training.
–activation	Type of activation function used in the model.
–train_epochs	Number of training epochs.
–batch_size	Number of samples per training batch.
–learning_rate	Learning rate of the optimizer.
–patience	Patience parameter for early stopping.
–loss	Loss function used for model optimization.

Table D.1: Key hyperparameters of MSPRACFormer

Bibliography

[1] Satoshi Nakamoto. *Bitcoin: A Peer-to-Peer Electronic Cash System*. 2008. URL: https://bitcoin.org/bitcoin.pdf.

[2] PWC. *Annual Global Crypto Hedge Fund Report 2025*. PWC, Nov. 2025. URL: https://caymanfinance.ky/wp-content/uploads/2025/11/7th-Annual-Global-Crypto-Hedge-Fund-Report.pdf.

[3] CoinMarketCap. *Cryptocurrency Prices, Charts and Market Capitalizations*. Retrieved from https://coinmarketcap.com/. Accessed: 10th Apr 2025. Oct. 2025.

[4] Athira Jayavarma, Preetha Parakkat Kesava Panikker, and Manjula G Nair. "Revolutionizing the energy sector: exploring diversified blockchain platforms for a sustainable future". In: *Frontiers in Blockchain* 8 (2025), p. 1544770.

[5] Fabian E. Eska et al. "Do design features explain the volatility of cryptocurrencies?" In: *Finance Research Letters* 66 (2024), p. 105536. ISSN: 1544-6123. DOI: https://doi.org/10.1016/j.frl.2024.105536. URL: https://www.sciencedirect.com/science/article/pii/S154461232400566X.

[6] Son D. Pham et al. *Exploring Common Volatility in Cryptocurrencies: A Subsectoral Perspective*. SSRN Working Paper. Available at SSRN. 2025. DOI: 10.2139/ssrn.5373305. URL: https://ssrn.com/abstract=5373305.

[7] Qing Zeng et al. "Stock market volatility and economic policy uncertainty: New insight into a dynamic threshold mixed-frequency model". In: *Finance Research Letters* 59 (2024), p. 104714. ISSN: 1544-6123. DOI: https://doi.

org/10.1016/j.frl.2023.104714. URL: https://www.sciencedirect.com/science/article/pii/S1544612323010863.

[8] Thijs Donders. *Blockchain consensus mechanisms and price volatilities*. 2022.

[9] George Milunovich. "Assessing the connectedness between Proof of Work and Proof of Stake/Other digital coins". In: *Economics Letters* 211 (2022), p. 110243. ISSN: 0165-1765. DOI: https://doi.org/10.1016/j.econlet.2021.110243. URL: https://www.sciencedirect.com/science/article/pii/S0165176521004778.

[10] George Milunovich. "Assessing the connectedness between Proof of Work and Proof of Stake/Other digital coins". In: *Economics Letters* 211 (2022), p. 110243. ISSN: 0165-1765. DOI: https://doi.org/10.1016/j.econlet.2021.110243. URL: https://www.sciencedirect.com/science/article/pii/S0165176521004778.

[11] Bin-xia Chen and Yan-lin Sun. "Risk characteristics and connectedness in cryptocurrency markets: New evidence from a non-linear framework". In: *The North American Journal of Economics and Finance* 69 (2024), p. 102036. ISSN: 1062-9408. DOI: https://doi.org/10.1016/j.najef.2023.102036. URL: https://www.sciencedirect.com/science/article/pii/S1062940823001596.

[12] Mostafa Chegenizadeh, Sheng-Nan Li, and Claudio J Tessone. "Towards a deeper understanding of the Cardano macro-economics". In: *2024 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. IEEE. 2024, pp. 585–593.

[13] Mahmood Yousaf et al. "A Comprehensive Survey of Cryptocurrency Forecasting: Methods, Trends, and Challenges". In: *SSRN* (2024). DOI: 10.2139/ssrn.5234539. URL: https://doi.org/10.1007/s10462-024-11044-2.

[14] George E. P. Box, Gwilym M. Jenkins, and Gregory C. Reinsel. *Time Series Analysis: Forecasting and Control*. Holden-Day, 1970.

[15] Corinna Cortes and Vladimir Vapnik. “Support-vector networks”. In: *Machine Learning* 20.3 (1995), pp. 273–297.

[16] Douglas C. Montgomery, Elizabeth A. Peck, and G. Geoffrey Vining. *Introduction to Linear Regression Analysis*. 5th ed. Wiley, 2012.

[17] Volodymyr Mnih et al. “Playing Atari with Deep Reinforcement Learning”. In: *arXiv preprint arXiv:1312.5602* (2013).

[18] Keiron O’Shea and Ryan Nash. “An introduction to convolutional neural networks”. In: *arXiv preprint arXiv:1511.08458* (2015).

[19] Sepp Hochreiter and Jürgen Schmidhuber. “Long short-term memory”. In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[20] Ashish Vaswani et al. “Attention is All you Need”. In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[21] Michael Ayitey Junior et al. “Forex market forecasting using machine learning: Systematic Literature Review and meta-analysis”. In: *Journal of Big Data* 10.1 (2023), p. 9.

[22] László Vancsura, Tibor Tatay, and Tibor Bareith. “Navigating AI-Driven Financial Forecasting: A Systematic Review of Current Status and Critical Research Gaps”. In: *Forecasting* 7.3 (2025). ISSN: 2571-9394. DOI: 10.3390/forecast7030036. URL: https://www.mdpi.com/2571-9394/7/3/36.

[23] Xi Tang and Yumei Xie. “Exchange Rate Forecasting: A Deep Learning Framework Combining Adaptive Signal Decomposition and Dynamic Weight Optimization”. In: *International Journal of Financial Studies* 13.3 (2025). ISSN: 2227-7072. DOI: 10.3390/ijfs13030151. URL: https://www.mdpi.com/2227-7072/13/3/151.

[24] T Soni Madhulatha and Dr Md Atheeq Sultan Ghori. "Forecasting Currency Exchange Rates Using ARIMA, ETS and RNN: A Machine Learning Perspective". In: *Journal of Theoretical and Applied Information Technology* 103.15 (2025).

[25] Shiyang Li et al. "Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting". In: *Advances in neural information processing systems* 32 (2019).

[26] Zhongbao Zhou et al. "Multi-source data driven cryptocurrency price movement prediction and portfolio optimization". In: *Expert Systems with Applications* 219 (2023), p. 119600. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2023.119600. URL: https://www.sciencedirect.com/science/article/pii/S095741742300101X.

[27] R. Alsini et al. "Forecasting Cryptocurrency's Buy Signal with a Bagged Tree Learning Approach to Enhance Purchase Decisions". In: *Frontiers in Big Data* 7 (May 2024), p. 1369895. DOI: 10.3389/fdata.2024.1369895. URL: https://doi.org/10.3389/fdata.2024.1369895.

[28] Saiful Kiram, Munirul Ula, and Kurniawati. "A Comparative Study of Temporal Convolutional Network and Gated Recurrent Unit for Predicting *Ethereum* Prices". In: *Applied Engineering Innovation and Technology* 2.1 (2025), pp. 1–8. DOI: 10.62777/aeit.v2i1.55. URL: https://doi.org/10.62777/aeit.v2i1.55.

[29] Armin Mazinani, Luca Davoli, and Gianluigi Ferrari. "Deep learning algorithms for cryptocurrency price prediction: A comparative analysis". In: *Distributed Ledger Technologies: Research and Practice* 4.1 (2025), pp. 1–38.

[30] I. Agakishiev, W. K. Härdle, D. Becker, et al. "Regime Switching Forecasting for Cryptocurrencies". In: *Digital Finance* 7 (2025), pp. 107–131. DOI: 10.1007/s42521-024-00123-2. URL: https://doi.org/10.1007/s42521-024-00123-2.

[31] T.O. Kehinde, O.J. Adedokun, and A. et al. Joseph. "Helformer: an attention-based deep learning model for cryptocurrency price forecasting". In: *Big Data* 12 (2025). URL: https://doi.org/10.1186/s40537-025-01135-4.

[32] Tian Zhou et al. "FEDformer: Frequency Enhanced Decomposed Transformer for Long-term Series Forecasting". In: *Proceedings of the 39th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri et al. Vol. 162. Proceedings of Machine Learning Research. PMLR, 17–23 Jul 2022, pp. 27268–27286. URL: https://proceedings.mlr.press/v162/zhou22g.html.

[33] George Zerveas et al. "A transformer-based framework for multivariate time series representation learning". In: *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. ACM. 2021, pp. 2114–2124.

[34] Jacob Devlin et al. "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)*. 2019, pp. 4171–4186.

[35] Alexey Dosovitskiy et al. "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale". In: *International Conference on Learning Representations*. 2021. URL: https://openreview.net/forum?id=YicbFdNTTy.

[36] Haixu Wu et al. "Autoformer: decomposition transformers with auto-correlation for long-term series forecasting". In: *Proceedings of the 35th International Conference on Neural Information Processing Systems*. NIPS '21. Red Hook, NY, USA: Curran Associates Inc., 2021. ISBN: 9781713845393.

[37] Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. "Reformer: The efficient transformer". In: *arXiv preprint arXiv:2001.04451* (2020). URL: https://arxiv.org/abs/2001.04451.

[38] Haoyi Zhou et al. "Informer: Beyond efficient transformer for long sequence time-series forecasting". In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 35. 12. 2021, pp. 11106–11115.

[39] 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. URL: https://openreview.net/forum?id=vSVLM2j9eie.

[40] Yuqi Nie et al. *A Time Series is Worth 64 Words: Long-term Forecasting with Transformers*. 2023. arXiv: 2211.14730 [cs.LG]. URL: https://arxiv.org/abs/2211.14730.

[41] Wanlin Cai et al. "MSGNet: learning multi-scale inter-series correlations for multivariate time series forecasting". In: *AAAI'24/IAAI'24/EAAI'24*. AAAI Press, 2024. ISBN: 978-1-57735-887-9. DOI: 10.1609/aaai.v38i10.28991. URL: https://doi.org/10.1609/aaai.v38i10.28991.

[42] Yuxuan Wang et al. "TimeXer: empowering transformers for time series forecasting with exogenous variables". In: *Proceedings of the 38th International Conference on Neural Information Processing Systems*. NIPS '24. Vancouver, BC, Canada: Curran Associates Inc., 2025. ISBN: 9798331314385.

[43] Ce Chi et al. "InjectTST: Injecting Global Information into Independent Channels for Long Time Series Forecasting". In: *ICASSP 2025 - 2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2025, pp. 1–5. DOI: 10.1109/ICASSP49660.2025.10888704.

[44] Yong Liu et al. "iTransformer: Inverted Transformers Are Effective for Time Series Forecasting". In: *The Twelfth International Conference on Learning Representations*. 2024. URL: https://openreview.net/forum?id=JePfAI8fah.

[45] Yizhi Cao et al. "MSPatch: A multi-scale patch mixing framework for multivariate time series forecasting". In: *Expert Systems with Applications* 273 (2025), p. 126849. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.

eswa.2025.126849. URL: https://www.sciencedirect.com/science/article/pii/S0957417425004713.

[46] David L. John, Sebastian Binnewies, and Bela Stantic. "Cryptocurrency Price Prediction Algorithms: A Survey and Future Directions". In: *Forecasting* 6.3 (2024), pp. 637–671. ISSN: 2571-9394. DOI: 10.3390/forecast6030034. URL: https://www.mdpi.com/2571-9394/6/3/34.

[47] Sravan Karthick T. *Expert System for Bitcoin Forecasting: Integrating Global Liquidity via TimeXer Transformers*. 2026. arXiv: 2512.22326 [cs.LG]. URL: https://arxiv.org/abs/2512.22326.

[48] Wajeeha Badar et al. "Enhanced Interpretable Forecasting of Cryptocurrency Prices Using Autoencoder Features and a Hybrid CNN-LSTM Model". In: *Mathematics* 13.12 (2025). ISSN: 2227-7390. DOI: 10.3390/math13121908. URL: https://www.mdpi.com/2227-7390/13/12/1908.

[49] Sicheng Fu, Fangfang Zhu, and Xiangdong Liu. "A Predictive Framework Integrating Multi-Scale Volatility Components and Time-Varying Quantile Spillovers: Evidence from the Cryptocurrency Market". In: *arXiv preprint arXiv:2507.22409* (2025).

[50] S. Hansun, A. Wicaksana, and A. Q. M. Khaliq. "Multivariate Cryptocurrency Prediction: Comparative Analysis of Three Recurrent Neural Networks Approaches". In: *Journal of Big Data* 9.1 (2022), p. 50. DOI: 10.1186/s40537-022-00601-7. URL: https://doi.org/10.1186/s40537-022-00601-7.

[51] Pasquale De Rosa, Pascal Felber, and Valerio Schiavoni. "Practical forecasting of cryptocoins timeseries using correlation patterns". In: *Proceedings of the 17th ACM international conference on distributed and event-based systems*. 2023, pp. 80–90.

[52] Sang-Ha Sung et al. "A Study on Cryptocurrency Log-Return Price Prediction Using Multivariate Time-Series Model". In: *Axioms* 11.9 (2022). ISSN:

2075-1680. DOI: 10.3390/axioms11090448. URL: https://www.mdpi.com/2075-1680/11/9/448.

[53] Amit Kumar and Taoran Ji. "CryptoPulse: Short-Term Cryptocurrency Forecasting with Dual-Prediction and Cross-Correlated Market Indicators". In: *2024 IEEE International Conference on Big Data (BigData)*. IEEE, Dec. 2024, pp. 1–8. DOI: 10.1109/bigdata62323.2024.10982029. URL: http://dx.doi.org/10.1109/BigData62323.2024.10982029.

[54] Lu Han, Han-Jia Ye, and De-Chuan Zhan. *The Capacity and Robustness Trade-off: Revisiting the Channel Independent Strategy for Multivariate Time Series Forecasting*. 2023. arXiv: 2304.05206 [cs.LG]. URL: https://arxiv.org/abs/2304.05206.

[55] Tianfeng Wang and Gaojie Cui. "Ratsf: Empowering customer service volume management through retrieval-augmented time-series forecasting". In: *arXiv preprint arXiv:2403.04180* (2024).

[56] Patrick Lewis et al. "Retrieval-augmented generation for knowledge-intensive NLP tasks". In: *Proceedings of the 34th International Conference on Neural Information Processing Systems*. NIPS '20. Vancouver, BC, Canada: Curran Associates Inc., 2020. ISBN: 9781713829546.

[57] Kutay Tire et al. "Retrieval augmented time series forecasting". In: *arXiv preprint arXiv:2411.08249* (2024).

[58] Huanyu Zhang et al. "Timeraf: Retrieval-augmented foundation model for zero-shot time series forecasting". In: *IEEE Transactions on Knowledge and Data Engineering* (2025).

[59] Sungwon Han et al. *Retrieval Augmented Time Series Forecasting*. 2025. arXiv: 2505.04163 [cs.LG]. URL: https://arxiv.org/abs/2505.04163.

[60] Zichao Li. "Retrieval-Augmented Forecasting with Tabular Time Series Data". In: *The 4th Table Representation Learning Workshop at ACL 2025*. 2025. URL: https://openreview.net/forum?id=yCaIbO5JHo.

[61] Silin Yang et al. "TimeRAG: Boosting LLM Time Series Forecasting via Retrieval-Augmented Generation". In: *ICASSP 2025–2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2025, pp. 1–5.

[62] Mengxi Xiao et al. *TimeRAG: It's Time for Retrieval-Augmented Generation in Time-Series Forecasting*. 2024. URL: https://openreview.net/forum?id=GvzL4LuycW.

[63] Chidaksh Ravuru, Sagar Srinivas Sakhinana, and Venkataramana Runkana. "Agentic retrieval-augmented generation for time series analysis". In: *arXiv preprint arXiv:2408.14484* (2024).

[64] Jingwei Liu et al. "Retrieval-augmented diffusion models for time series forecasting". In: *Advances in Neural Information Processing Systems* 37 (2024), pp. 2766–2786.

[65] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. "Sequence to Sequence Learning with Neural Networks". In: *Advances in Neural Information Processing Systems*. Vol. 27. Curran Associates, Inc., 2014.

[66] John N. Kallianiotis. "The Foreign Exchange Market". In: *Exchange Rates and International Financial Economics: History, Theories, and Practices*. New York: Palgrave Macmillan US, 2013, pp. 51–82. ISBN: 978-1-137-31888-6. DOI: 10.1057/9781137318886_2. URL: https://doi.org/10.1057/9781137318886_2.

[67] Olumide Sunday Adesina and Lawrence Ogechukwu Obokoh. "A hybrid framework of deep learning and traditional time series models for exchange rate prediction". In: *Scientific African* 29 (2025), e02818. ISSN: 2468-2276. DOI: https://doi.org/10.1016/j.sciaf.2025.e02818. URL: https://www.sciencedirect.com/science/article/pii/S246822762500287X.

[68] Usman Ullah and Zhensheng Huang. "Forecasting Currency Exchange Rate Through a Hybrid Time-Series Analysis". In: *IEEE Transactions on Con-*

sumer Electronics 71.2 (2025), pp. 6027–6040. DOI: 10.1109/TCE.2025.3574013.

[69] Shicheng Hu et al. "Incorporating improved directional change and regime change detection to formulate trading strategies in foreign exchange markets". In: *Physica A: Statistical Mechanics and its Applications* 622 (2023), p. 128810. ISSN: 0378-4371. DOI: https://doi.org/10.1016/j.physa.2023.128810. URL: https://www.sciencedirect.com/science/article/pii/S0378437123003655.

[70] Oumaima Abouzaid and Faouzi Boussedra. "Artificial intelligence and exchange rate forecasting: assessing predictive accuracy and macroeconomic sensitivity". In: *Frontiers in Applied Mathematics and Statistics* 11 (2025), p. 1654093. DOI: 10.3389/fams.2025.1654093. URL: https://www.frontiersin.org/journals/applied-mathematics-and-statistics/articles/10.3389/fams.2025.1654093.

[71] Ali Muzafar Qureshi. "Machine Learning Prediction of Exchange Rate Using Macro Economic Indicators". In: *Available at SSRN 5287273* (2025).

[72] Yunzhong Yu. "Comprehensive Analysis of Exchange Rate Prediction: Influencing Factors, Methodologies, and Future Prospects". In: *Proceedings of the 1st International Conference on E-commerce and Artificial Intelligence (ECAI 2024)*. SCITEPRESS, 2025, pp. 200–204. ISBN: 978-989-758-726-9. DOI: 10.5220/0013212800004568.

[73] Ziqian Niu. "Comparative Analysis Based on Machine Learning Model Predicting Exchange Rate Fluctuations". In: *Proceedings of the 2024 2nd International Conference on Management Innovation and Economy Development (MIED 2024)*. Atlantis Press, 2024, pp. 471–480. ISBN: 978-94-6463-542-3. DOI: 10.2991/978-94-6463-542-3_55. URL: https://doi.org/10.2991/978-94-6463-542-3_55.

[74] Tantai Somya, Nutthapradh Jomsaeng, and Chotiros Surapholchai. "Foreign Exchange Rate Predictions using Deep Learning". In: *Proceedings of AMC2025* 2 (2025), p. 113003.

[75] Tomas Pečiulis et al. "Forecasting of cryptocurrencies: Mapping trends, influential sources, and research themes". In: *Journal of Forecasting* 43.6 (2024), pp. 1880–1901. DOI: https://doi.org/10.1002/for.3114. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/for.3114. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/for.3114.

[76] Sattarov Otabek and Jaeyoung Choi. "From Prediction to Profit: A Comprehensive Review of Cryptocurrency Trading Strategies and Price Forecasting Techniques". In: *IEEE Access* 12 (2024), pp. 87039–87064. DOI: 10.1109/ACCESS.2024.3417449.

[77] Amin Azari. "Bitcoin price prediction: An ARIMA approach". In: *arXiv preprint arXiv:1904.05315* (2019).

[78] Yang Si. "Using ARIMA model to analyse and predict bitcoin price". In: *BCP Business Management* 34 (Dec. 2022), pp. 1210–1216. DOI: 10.54691/bcpbm.v34i.3161.

[79] Navmeen Latif et al. "Comparative performance of LSTM and ARIMA for the short-term prediction of bitcoin prices". In: *Australasian Accounting, Business and Finance Journal* 17.1 (2023), pp. 256–276.

[80] K Ganesh, M Anbazhagan, and Shreyas Visweshwaran. "An Empirical Analysis on ARIMA and Regression Models for Time Series Forecasting on Bitcoin Dataset". In: *2024 IEEE International Conference on Information Technology, Electronics and Intelligent Communication Systems (ICITEICS)*. IEEE. 2024, pp. 1–9.

[81] Huaigang Long et al. "Seasonality in the Cross-Section of Cryptocurrency Returns". In: *Finance Research Letters* 35 (2020), p. 101566. ISSN: 1544-

6123. DOI: https://doi.org/10.1016/j.frl.2020.101566. URL: https://www.sciencedirect.com/science/article/pii/S154461232030235X.

[82] Shrikant Panigrahi. "Predicting Cryptocurrency Prices Comparing Different Machine Learning Techniques: A Performance Analysis for Pre and Post COVID-19 Pandemic Periods". In: *International Journal of Economics and Financial Issues* 15.3 (2025), pp. 124–138.

[83] Hakan Yildirim and Festus Victor Bekun. "Predicting volatility of bitcoin returns with ARCH, GARCH and EGARCH models". In: *Future Business Journal* 9.1 (2023), p. 75. ISSN: 2314-7210. DOI: 10.1186/s43093-023-00255-8. URL: https://doi.org/10.1186/s43093-023-00255-8.

[84] Junyi Zhu. "Bitcoin Price Prediction: ARIMA & SARIMA vs Linear Regression". In: *Advances in Economics, Management and Political Sciences* 61.1 (2023), pp. 47–54.

[85] Umar Islam et al. "Empowering global ethereum price prediction with EtherVoyant: a state-of-the-art time series forecasting model". In: *Neural Computing and Applications* (2024). Published online: 27 Aug 2024. DOI: 10.1007/s00521-024-10169-3.

[86] Robert F. Engle. "Autoregressive Conditional Heteroskedasticity with Estimates of the Variance of United Kingdom Inflation". In: *Econometrica* 50.4 (1982), pp. 987–1008.

[87] Francisco Joao Matos Costa. "Forecasting volatility using GARCH models". MA thesis. Universidade do Minho (Portugal), 2017.

[88] Abuduwali Aibai, Jiansuer Julaiti, and Shangde Gou. "The asymmetric effects of upside and downside risks in cryptocurrency markets: Insights from the LUNA and FTX crises". In: *Finance Research Letters* 67 (2024), p. 105750. ISSN: 1544-6123. DOI: https://doi.org/10.1016/j.frl.2024.105750. URL: https://www.sciencedirect.com/science/article/pii/S1544612324007803.

[89] Rafik Nafkha and Dorota Żebrowska-Suchodolska. "Modelling and Estimating Volatility Using Conditional Variance-Type Models". In: *Proceedings of the 28th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems (KES 2024)*. Institute of Information Technology, University of Life Sciences – SGGW. 2024.

[90] Rohaifa Khaldi, Abdellatif El Afia, and Raddouane Chiheb. "Forecasting of BTC volatility: comparative study between parametric and nonparametric models". In: *Progress in Artificial Intelligence* 8 (July 2019), pp. 511–523. DOI: https://doi.org/10.1007/s13748-019-00196-w.

[91] Genaro Sucarrat. "The log-GARCH model via ARMA representations". In: *Financial mathematics, volatility and covariance modelling*. Routledge, 2019, pp. 336–359.

[92] Zhuanxin Ding, Clive W.J. Granger, and Robert F. Engle. "A long memory property of stock market returns and a new model". In: *Journal of Empirical Finance* 1.1 (1993), pp. 83–106. ISSN: 0927-5398. DOI: https://doi.org/10.1016/0927-5398(93)90006-D. URL: https://www.sciencedirect.com/science/article/pii/092753989390006D.

[93] Jean-Michel Zakoian. "Threshold heteroskedastic models". In: *Journal of Economic Dynamics and Control* 18.5 (1994), pp. 931–955. ISSN: 0165-1889. DOI: https://doi.org/10.1016/0165-1889(94)90039-6. URL: https://www.sciencedirect.com/science/article/pii/0165188994900396.

[94] LAWRENCE R. GLOSTEN, RAVI JAGANNATHAN, and DAVID E. RUNKLE. "On the Relation between the Expected Value and the Volatility of the Nominal Excess Return on Stocks". In: *The Journal of Finance* 48.5 (1993), pp. 1779–1801. DOI: https://doi.org/10.1111/j.1540-6261.1993.tb05128.x. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1540-6261.1993.tb05128.x. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1540-6261.1993.tb05128.x.

[95] Robert F. Engle and Gary G. J. Lee. "A Long-Run and Short-Run Component Model of Stock Return Volatility". In: *Cointegration, Causality, and Forecasting: A Festschrift in Honour of Clive W. J. Granger* (1999), pp. 475–497.

[96] yang Shen and Haoyuan Wang. "Valuation and Forecasting of Cryptocurrency: Analysis of Bitcoin, Ethereum and Dogecoin". In: *BCP Business Management* 38 (2023), pp. 1067–1074. DOI: 10.54691/bcpbm.v34i.3161.

[97] G. Schwenkler and H. Zheng. "News-driven peer co-movement in crypto markets". In: *Journal of Corporate Finance* 93 (2025), p. 102772. ISSN: 0929-1199. DOI: https://doi.org/10.1016/j.jcorpfin.2025.102772. URL: https://www.sciencedirect.com/science/article/pii/S0929119925000409.

[98] Fikrie Hartanta Sembiring, Regita Putri Permata, and Rifdatun Ni'mah. "Comparative Study of Hybrid ARIMA-LSTM and ARIMAX-LSTM for Bitcoin Forecasting with Data Partitioning". In: *CAUCHY: Jurnal Matematika Murni dan Aplikasi* 10.2 (2025), pp. 722–737.

[99] Mohamad El Abaji and Ramzi A Haraty. "Enhancing Bitcoin Forecast Accuracy by Integrating AI, Sentiment Analysis, and Financial Models". In: *International Journal of Computing and Digital Systems* 17.1 (2024), pp. 1–15.

[100] Tulika Shrivastava et al. "Benchmarking modeling architectures for cryptocurrency price prediction using financial and social media data". In: *Social Network Analysis and Mining* 15.1 (2025), p. 93. ISSN: 1869-5469. DOI: 10.1007/s13278-025-01520-0. URL: https://doi.org/10.1007/s13278-025-01520-0.

[101] Yi-Shuai Ren et al. "Past, present, and future of the application of machine learning in cryptocurrency research". In: *Research in International Business and Finance* 63 (2022), p. 101799. ISSN: 0275-5319. DOI: https://doi.org/

10.1016/j.ribaf.2022.101799. URL: https://www.sciencedirect.com/science/article/pii/S0275531922001854.

[102] Divya Chaudhary and Sushil Kumar Saroj. "Cryptocurrency Price Prediction Using Supervised Machine Learning Algorithms". In: *ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal* 12.1 (2023). Regular Issue, e31490. ISSN: 2255-2863. DOI: 10.14201/adcaij.31490. URL: https://adcaij.usal.es/index.php/ADCAIJ/article/view/31490.

[103] Adrian Ţăran-Moroşan. "The relative strength index revisited". In: *African Journal of Business Management* 5.14 (2011), pp. 5855–5862.

[104] Gabriel Dan I Anghel. "Stock market efficiency and the MACD. Evidence from countries around the world". In: *Procedia economics and finance* 32 (2015), pp. 1414–1431.

[105] Yicong Li. "Optimizing Random Forest Models for Stock Market Prediction with Hyperparameter Analysis". In: *Proceedings of the 2nd Guangdong-Hong Kong-Macao Greater Bay Area International Conference on Digital Economy and Artificial Intelligence*. 2025, pp. 57–61.

[106] Hayette Gatfaoui. "Estimating Fundamental Sharpe Ratios: A Kalman Filter Approach". In: (Apr. 2015). Available at SSRN: https://ssrn.com/abstract=2838935. DOI: 10.2139/ssrn.2838935.

[107] Oliver Steinki and Ziad Mohammad. *Common Metrics for Performance Evaluation: Overview of Popular Performance Measurement Ratios*. Tech. rep. Social Science Research Network, 2015. DOI: 10.2139/ssrn.2662054. URL: https://ssrn.com/abstract=2662054.

[108] Ahmed Bouteska et al. "Cryptocurrency price forecasting–a comparative analysis of ensemble learning and deep learning methods". In: *International Review of Financial Analysis* 92 (2024), p. 103055.

[109] Hani Alnami et al. “An Integrated Framework for Cryptocurrency Price Forecasting and Anomaly Detection Using Machine Learning”. In: *Applied Sciences* 15.4 (2025). ISSN: 2076-3417. DOI: 10.3390/app15041864. URL: https://www.mdpi.com/2076-3417/15/4/1864.

[110] Nezir Köse, Yunus Emre Gür, and Emre Ünal. “Deep Learning and Machine Learning Insights Into the Global Economic Drivers of the Bitcoin Price”. In: *Journal of Forecasting* 44.5 (2025), pp. 1666–1698. DOI: https://doi.org/10.1002/for.3258. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/for.3258. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/for.3258.

[111] H. M. Tanrikulu and H. Pabuccu. “The Effect of Data Types’ on the Performance of Machine Learning Algorithms for Cryptocurrency Prediction”. In: *Computational Economics* (2025). DOI: 10.1007/s10614-025-10919-y. URL: https://doi.org/10.1007/s10614-025-10919-y.

[112] Puguh Hiskiawan, Jovan William, and Louis Feliepe Tio Jansel. “A Hybrid Data Science Framework for Forecasting Bitcoin Prices Using Traditional and AI Models”. In: *Journal of Applied Informatics and Computing* 9.5 (Oct. 2025), pp. 2089–2101. ISSN: 2548-6861.

[113] Phumudzo Lloyd Seabe, Claude Rodrigue Bambe Moutsinga, and Edson Pindza. “Forecasting Cryptocurrency Prices Using LSTM, GRU, and Bi-Directional LSTM: A Deep Learning Approach”. In: *Fractal and Fractional* 7.2 (2023). ISSN: 2504-3110. DOI: 10.3390/fractalfract7020203. URL: https://www.mdpi.com/2504-3110/7/2/203.

[114] Dušan Bohovic. “Comparison of LSTM and GRU Methods for Predicting Gold Exchange Rate against US Dollar”. In: *International Journal Artificial Intelligent and Informatics* 3.1 (2025), pp. 24–29.

[115] Xueli Zhang et al. “Robust recurrent neural networks for time series forecasting”. In: *Neurocomputing* 526 (2023), pp. 143–157. ISSN: 0925-2312. DOI:

https://doi.org/10.1016/j.neucom.2023.01.037. URL: https://www.sciencedirect.com/science/article/pii/S0925231223000462.

[116] Michael Nair1, Mohamed I Marie, and Laila A. Abd-Elmegid. "Prediction of Cryptocurrency Price using Time Series Data and Deep Learning Algorithms". In: *International Journal of Advanced Computer Science and Applications* 14.8 (2023). DOI: 10.14569/IJACSA.2023.0140837. URL: www.ijacsa.thesai.org.

[117] Oluwadamilare Omole and David Enke. "Deep learning for Bitcoin price direction prediction: models and trading strategies empirically compared". In: *Financial Innovation* 10.1 (2024), p. 117.

[118] Nishant Jagannath et al. "An on-chain analysis-based approach to predict ethereum prices". In: *IEEE Access* 9 (2021), pp. 167972–167989.

[119] Guokun Lai et al. "Modeling long-and short-term temporal patterns with deep neural networks". In: *The 41st international ACM SIGIR conference on research & development in information retrieval*. 2018, pp. 95–104.

[120] R Bourday et al. "Cryptocurrency Forecasting Using Deep Learning Models: A Comparative Analysis". In: *HIGHTECH AND INNOVATION JOURNAL : Ital Publication* 5.4 (2024), pp. 1055–1067.

[121] Hongzhi Zhang and M. Omair Shafiq. *Survey of transformers and towards ensemble learning using transformers for natural language processing*. 2024. URL: https://doi.org/10.1186/s40537-023-00842-0.

[122] Wendong Zhao et al. *Enhanced human motion recognition via PPG signals*. 2025. URL: https://doi.org/10.1007/s11760-025-04353-8.

[123] Shubham Singh and Mayur Bhat. *Transformer-based approach for Ethereum Price Prediction Using Crosscurrency correlation and Sentiment Analysis*. 2024. arXiv: 2401.08077 [cs.LG]. URL: https://arxiv.org/abs/2401.08077.

[124] Berkay Yildirim and Murat Taskiran. "Optuna Based Optimized Transformer Model Approach in Bitcoin Time Series Analysis". In: *2024 26th International Conference on Digital Signal Processing and its Applications (DSPA)*. 2024, pp. 1–6. DOI: 10.1109/DSPA60853.2024.10510091.

[125] Takuya Akiba et al. "Optuna: A Next-generation Hyperparameter Optimization Framework". In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. KDD '19. Anchorage, AK, USA: Association for Computing Machinery, 2019, pp. 2623–2631. ISBN: 9781450362016. DOI: 10.1145/3292500.3330701. URL: https://doi.org/10.1145/3292500.3330701.

[126] Esam Mahdi, Carlos Martin-Barreiro, and Xavier Cabezas. "A Novel Hybrid Approach Using an Attention-Based Transformer + GRU Model for Predicting Cryptocurrency Prices". In: *Mathematics* 13.9 (2025). ISSN: 2227-7390. DOI: 10.3390/math13091484. URL: https://www.mdpi.com/2227-7390/13/9/1484.

[127] 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". In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 14.1 (2024), e1519.

[128] Jongseon Kim et al. "A comprehensive survey of deep learning for time series forecasting: architectural diversity and open challenges". In: *Artificial Intelligence Review* 58.7 (2025), pp. 1–95.

[129] Ailing Zeng et al. "Are transformers effective for time series forecasting?" In: *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence*. AAAI'23/IAAI'23/EAAI'23. AAAI Press, 2023. ISBN: 978-

1-57735-880-0. DOI: 10.1609/aaai.v37i9.26317. URL: https://doi.org/10.1609/aaai.v37i9.26317.

[130] Ji-Won Kang, Daihyun Kwon, and Sun-Yong Choi. "A Hybrid Frequency Decomposition–CNN–Transformer Model for Predicting Dynamic Cryptocurrency Correlations". In: *Electronics* 14.21 (2025). ISSN: 2079-9292. DOI: 10.3390/electronics14214136. URL: https://www.mdpi.com/2079-9292/14/21/4136.

[131] Rabia Younas, H. M. Raza Ur Rehman, and G. S. Choi. "Crypto foretell: a novel hybrid attention-correlation based forecasting approach for cryptocurrency". In: *Journal of Big Data* 12 (2025), p. 229. DOI: 10.1186/s40537-025-01291-7. URL: https://doi.org/10.1186/s40537-025-01291-7.

[132] Xiangjie Kong et al. "Deep learning for time series forecasting: a survey". In: *International Journal of Machine Learning and Cybernetics* (2025), pp. 1–34.

[133] Mohammed Mudassir et al. "Time-Series Forecasting of Bitcoin Prices Using High-Dimensional Features: A Machine Learning Approach". In: *Neural Computing and Applications* 37.28 (Oct. 2025), pp. 22979–22993. ISSN: 1433-3058. DOI: 10.1007/s00521-020-05129-6. URL: https://doi.org/10.1007/s00521-020-05129-6.

[134] Celalettin Kişmiroğlu and Omer Isik. "Temperature Prediction Using Transformer–LSTM Deep Learning Models and Sarimax from a Signal Processing Perspective". In: *Applied Sciences* (2025). URL: https://api.semanticscholar.org/CorpusID:280938387.

[135] Shizhan Liu et al. "Pyraformer: Low-Complexity Pyramidal Attention for Long-Range Time Series Modeling and Forecasting". In: *International Conference on Learning Representations*. 2022. URL: https://openreview.net/forum?id=0EXmFzUn5I.

[136] Shuhan Zhong et al. "A Multi-Scale Decomposition MLP-Mixer for Time Series Analysis". In: *Proc. VLDB Endow.* 17.7 (Mar. 2024), pp. 1723–1736.

ISSN: 2150-8097. DOI: 10.14778/3654621.3654637. URL: https://doi.org/10.14778/3654621.3654637.

[137] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. "The M4 Competition: Results, findings, conclusion and way forward". In: *International Journal of Forecasting* 34.4 (2018), pp. 802–808. ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2018.06.001. URL: https://www.sciencedirect.com/science/article/pii/S0169207018300785.

[138] Minhao Liu et al. "SCINet: Time Series Modeling and Forecasting with Sample Convolution and Interaction". In: *Advances in Neural Information Processing Systems*. Ed. by S. Koyejo et al. Vol. 35. Curran Associates, Inc., 2022, pp. 5816–5828. URL: https://proceedings.neurips.cc/paper_files/paper/2022/file/266983d0949aed78a16fa4782237dea7-Paper-Conference.pdf.

[139] Wenyong Han et al. "Mcformer: multivariate time series forecasting with mixed-channels transformer". In: *IEEE Internet of Things Journal* (2024).

[140] Kurt Shuster et al. "Retrieval augmentation reduces hallucination in conversation". In: *arXiv preprint arXiv:2104.07567* (2021).

[141] Panupong Pasupat, Yuan Zhang, and Kelvin Guu. "Controllable semantic parsing via retrieval augmentation". In: *arXiv preprint arXiv:2110.08458* (2021).

[142] Baoyu Jing et al. "Retrieval based time series forecasting". In: *CIKM'22 workshop on Applied Machine Learning Methods for Time Series Forecasting (AMLTS)* (2022).

[143] Laura C Slivinski et al. "Towards a more reliable historical reanalysis: Improvements for version 3 of the Twentieth Century Reanalysis system". In: *Quarterly Journal of the Royal Meteorological Society* 145.724 (2019), pp. 2876–2908.

[144] Meinard Müller. *Information retrieval for music and motion*. Springer, 2007.

[145] Liang Zhao, Olga Gkountouna, and Dieter Pfoser. "Spatial Auto-regressive Dependency Interpretable Learning Based on Spatial Topological Constraints". In: *ACM Trans. Spatial Algorithms Syst.* 5.3 (Aug. 2019). ISSN: 2374-0353. DOI: 10.1145/3339823. URL: https://doi.org/10.1145/3339823.

[146] Yong Liu et al. "Non-stationary Transformers: Exploring the Stationarity in Time Series Forecasting". In: *Advances in Neural Information Processing Systems (NeurIPS) 35*. 2022. URL: https://arxiv.org/abs/2205.14415.

[147] Abdul Fatir Ansari et al. "Chronos: Learning the Language of Time Series". In: *Transactions on Machine Learning Research* (2024). ISSN: 2835-8856. URL: https://openreview.net/forum?id=gerNCVqqtR.

[148] Vijay Ekambaram et al. "Tiny time mixers (ttms): Fast pre-trained models for enhanced zero/few-shot forecasting of multivariate time series". In: *Advances in Neural Information Processing Systems* 37 (2024), pp. 74147–74181.

[149] Ming Jin et al. "Time-llm: Time series forecasting by reprogramming large language models". In: *arXiv preprint arXiv:2310.01728* (2023).

[150] Gerald Woo et al. "Unified training of universal time series forecasting transformers". In: *Proceedings of the 41st International Conference on Machine Learning (ICML 2024)* 235 (2024), pp. 53140–53164.

[151] Yong Liu et al. "Timer: Generative pre-trained transformers are large time series models". In: *arXiv preprint arXiv:2402.02368* (2024).

[152] Yan Li et al. "Generative time series forecasting with diffusion, denoise, and disentanglement". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 23009–23022.

[153] Alistair EW Johnson et al. "MIMIC-IV, a freely accessible electronic health record dataset". In: *Scientific data* 10.1 (2023), p. 1.

[154] Google DeepMind Gemma Team. "Gemma: Open Models Based on Gemini Research and Technology". In: *arXiv preprint arXiv:2403.08295* (2024).

Available at https://arxiv.org/abs/2403.08295. DOI: 10.48550/arXiv.2403.08295.

[155] AI@Meta. *Llama 3 Model Card*. https://www.llama.com/models/llama-3/. Accessed: 2025-11-10. 2024.

[156] Chao Chen et al. "Freeway Performance Measurement System: Mining Loop Detector Data". In: *Transportation Research Record: Journal of the Transportation Research Board* 1748.1 (2001). Original work published 2001, pp. 96–102. DOI: 10.3141/1748-12. URL: https://doi.org/10.3141/1748-12.

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

[158] Yejun Soun et al. "Accurate stock movement prediction with self-supervised learning from sparse noisy tweets". In: *2022 IEEE International Conference on Big Data (Big Data)*. IEEE. 2022, pp. 1691–1700.

[159] Huizhe Wu et al. "Hybrid deep sequential modeling for social text-driven stock prediction". In: *Proceedings of the 27th ACM international conference on information and knowledge management*. 2018, pp. 1627–1630.

[160] B.W. Matthews. "Comparison of the predicted and observed secondary structure of T4 phage lysozyme". In: *Biochimica et Biophysica Acta (BBA) - Protein Structure* 405.2 (1975), pp. 442–451. ISSN: 0005-2795. DOI: https://doi.org/10.1016/0005-2795(75)90109-9. URL: https://www.sciencedirect.com/science/article/pii/0005279575901099.

[161] A. Trindade. *ElectricityLoadDiagrams20112014*. UCI Machine Learning Repository. 2015. DOI: 10.24432/C58C86. URL: https://doi.org/10.24432/C58C86.

[162] Ailing Zeng et al. "Are Transformers Effective for Time Series Forecasting?" In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 37. 9. 2023, pp. 11121–11128.

[163] Shiyu Wang et al. "TimeMixer: Decomposable Multiscale Mixing for Time Series Forecasting". In: *arXiv preprint arXiv:2405.14616* (2024). URL: https://arxiv.org/abs/2405.14616.

[164] Vijay Ekambaram et al. "TSMixer: Lightweight MLP-Mixer Model for Multivariate Time Series Forecasting". In: *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. KDD '23. Long Beach, CA, USA: Association for Computing Machinery, 2023, pp. 459–469. ISBN: 9798400701030. DOI: 10.1145/3580305.3599533. URL: https://doi.org/10.1145/3580305.3599533.

[165] Pengcheng Yin et al. "TaBERT: Pretraining for joint understanding of textual and tabular data". In: *arXiv preprint arXiv:2005.08314* (2020).

[166] Eric Jang, Shixiang Gu, and Ben Poole. *Categorical Reparameterization with Gumbel-Softmax*. 2017. arXiv: 1611.01144 [stat.ML]. URL: https://arxiv.org/abs/1611.01144.

[167] Michael W. McCracken and Serena Ng. "FRED-MD: A Monthly Database for Macroeconomic Research". In: *Journal of Business & Economic Statistics* 34.4 (2016), pp. 574–589.

[168] David Salinas, Valentin Flunkert, and Jan Gasthaus. "DeepAR: Probabilistic Forecasting with Autoregressive Recurrent Networks". In: *International Journal of Forecasting* 36.3 (2020), pp. 1181–1191. DOI: 10.1016/j.ijforecast.2019.07.001. URL: https://arxiv.org/abs/1704.04110.

[169] Boris N. Oreshkin et al. "N-BEATS: Neural Basis Expansion Analysis for Interpretable Time Series Forecasting". In: *International Conference on Learning Representations (ICLR)*. Preprint arXiv:1905.10437. 2020.

[170] Bryan Lim et al. "Temporal Fusion Transformers for Interpretable Multi-Horizon Time Series Forecasting". In: *International Journal of Forecasting* 37.4 (2021), pp. 1748–1764.

[171] Jonathan Herzig et al. "TaPas: Weakly Supervised Table Parsing via Pre-Training". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 2020, pp. 4320–4333.

[172] Defu Cao et al. "Multivariate Time Series Forecasting with Dynamic Graph Neural Networks". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2020.

[173] Yoav Levine et al. "Standing on the shoulders of giant frozen language models". In: *arXiv preprint arXiv:2204.10019* (2022).

[174] Ming Jin et al. *Time-LLM: Time Series Forecasting by Reprogramming Large Language Models*. 2024. arXiv: 2310.01728 [cs.LG]. URL: https://arxiv.org/abs/2310.01728.

[175] Huiqiang Wang et al. "MICN: Multi-scale Local and Global Context Modeling for Long-term Series Forecasting". In: *The Eleventh International Conference on Learning Representations*. 2023. URL: https://openreview.net/forum?id=zt53IDUR1U.

[176] Tian Zhou et al. "FiLM: Frequency Improved Legendre Memory Model for Long-Term Time Series Forecasting". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 12677–12690.

[177] David Campos et al. "LightTS: Lightweight Time Series Classification with Adaptive Ensemble Distillation". In: *Proceedings of the ACM on Management of Data* 1.2 (2023), pp. 1–27.

[178] Suleiman Dahir Mohamed, Mohd Tahir Ismail, and Majid Khan Bin Majahar Ali. "Cryptocurrency Returns Over a Decade: Breaks, Trend Breaks and Outliers". In: *Scientific Annals of Economics and Business (continues Analele Stiintifice)* 71.1 (Mar. 2023), pp. 1–20. DOI: 10.47743/saeb-2024-

0003. URL: https://ideas.repec.org/a/aic/saebjn/v71y2023i1p1-20n7.html.

[179] Barış Korkusuz. "Coin impact on cross-crypto realized volatility and dynamic cryptocurrency volatility connectedness". In: *Financial Innovation* 11 (2025). DOI: 10.1186/s40854-025-00881-x.

[180] Tulika Shrivastava et al. "Benchmarking Modeling Architectures for Cryptocurrency Price Prediction Using Financial and Social Media Data". In: *Social Network Analysis and Mining* 15 (2025), p. 93. DOI: 10.1007/s13278-025-01520-0. URL: https://doi.org/10.1007/s13278-025-01520-0.

[181] Sanjeev Kumar et al. "Interconnectivity and investment strategies among commodity prices, cryptocurrencies, and G-20 capital markets: A comparative analysis during COVID-19 and Russian-Ukraine war". In: *International Review of Economics Finance* 88 (2023), pp. 547–593. ISSN: 1059-0560. DOI: https://doi.org/10.1016/j.iref.2023.06.039. URL: https://www.sciencedirect.com/science/article/pii/S1059056023002046.

[182] Pratyush Nath Upreti and María Vásquez Callo-Müller. "Phase One US-China Trade Deal: What Does It Mean for Intellectual Property?" In: *GRUR International* 69.4 (Mar. 2020), pp. 389–393. ISSN: 2632-8623. DOI: 10.1093/grurint/ikaa033. eprint: https://academic.oup.com/grurint/article-pdf/69/4/389/33045771/ikaa033.pdf. URL: https://doi.org/10.1093/grurint/ikaa033.

[183] Karen M. Sutter. *U.S. Export Controls and China: Advanced Semiconductors*. CRS Report R48642. Congressional Research Service, Aug. 2025. URL: https://www.congress.gov/crs_external_products/R/PDF/R48642/R48642.2.pdf.

[184] Antonio Briola et al. "Anatomy of a Stablecoin's failure: The Terra-Luna case". In: *Finance Research Letters* 51 (2023), p. 103358. ISSN: 1544-6123.

DOI: https://doi.org/10.1016/j.frl.2022.103358. URL: https://www.sciencedirect.com/science/article/pii/S1544612322005359.

[185] Michal Banka et al. "Journal of Open Innovation: Technology, Market, and Complexity". In: (2024).

[186] David Vidal-Tomás, Antonio Briola, and Tomaso Aste. "FTX's downfall and Binance's consolidation: The fragility of centralised digital finance". In: *Physica A: Statistical Mechanics and Its Applications* 625 (2023), p. 129044.

[187] Jiageng Liu, Igor Makarov, and Antoinette Schoar. *Anatomy of a run: The terra luna crash*. Tech. rep. National Bureau of Economic Research, 2023.

[188] Haritha Gb and Sahana NB. "Cryptocurrency Price Prediction using Twitter Sentiment Analysis". In: *Computer Science Information Technology (CS IT)* (2023).

[189] Krzysztof PODGÓRSKI and Mateusz MUSZYŃSKI. "CYCLICALITY OF THE CRYPTOCURRENCY MARKET." In: *Scientific Papers of Silesian University of Technology. Organization & Management/Zeszyty Naukowe Politechniki Slaskiej. Seria Organizacji i Zarzadzanie* 222 (2025).

[190] Lennart Ante, Ingo Fiedler, and Elias Strehle. "The influence of stablecoin issuances on cryptocurrency markets". In: *Finance Research Letters* 41 (2021), p. 101867. ISSN: 1544-6123. DOI: https://doi.org/10.1016/j.frl.2020.101867. URL: https://www.sciencedirect.com/science/article/pii/S1544612320316810.

[191] Jan Hauke and Tomasz Kossowski. "Comparison of Values of Pearson's and Spearman's Correlation Coefficients on the Same Sets of Data". In: *Quaestiones Geographicae* 30.2 (2011), pp. 87–93. DOI: 10.2478/v10117-011-0021-1. URL: https://doi.org/10.2478/v10117-011-0021-1.

[192] Kuang-Chieh Yen et al. "Cryptocurrency return dependency and economic policy uncertainty". In: *Finance Research Letters* 56 (2023), p. 104182. ISSN:

1544-6123. DOI: https://doi.org/10.1016/j.frl.2023.104182. URL: https://www.sciencedirect.com/science/article/pii/S1544612323005548.

[193] Lai T Hoang. "Cryptocurrency price-based comovement". In: *International Review of Economics & Finance* (2025), p. 104635.

[194] Di Wu. "Institutional Adoption and Correlation Dynamics: Bitcoin's Evolving Role in Financial Markets". In: *arXiv preprint arXiv:2501.09911* (2025).

[195] Geumil Bae and Jang Ho Kim. "Observing Cryptocurrencies through Robust Anomaly Scores". In: *Entropy* 24.11 (2022). ISSN: 1099-4300. DOI: 10.3390/e24111643. URL: https://www.mdpi.com/1099-4300/24/11/1643.

[196] Simon Rudkin, Wanling Rudkin, and Paweł Dłotko. "On the topology of cryptocurrency markets". In: *International Review of Financial Analysis* 89 (2023), p. 102759. ISSN: 1057-5219. DOI: https://doi.org/10.1016/j.irfa.2023.102759. URL: https://www.sciencedirect.com/science/article/pii/S1057521923002752.

[197] Paraskevi Katsiampa, Larisa Yarovaya, and Damian Zieba. "High-frequency connectedness between Bitcoin and other top-traded crypto assets during the COVID-19 crisis". In: *Journal of International Financial Markets, Institutions and Money* 79 (2022), p. 101578. ISSN: 1042-4431. DOI: https://doi.org/10.1016/j.intfin.2022.101578. URL: https://www.sciencedirect.com/science/article/pii/S1042443122000610.

[198] Alexander Copestake, Davide Furceri, and Tammaro Terracciano. "The Crypto Cycle and Institutional Investors". In: *Available at SSRN 4956413* (2024).

[199] Muhammad Akbar. "Bitcoin Price Forecasting using Seasonal Log-Differenced XGBoost with 2014–2025 Data". In: *Sistem Informasi* 14 (2025). ISSN: 2302-8149. URL: https://sistemasi.ftik.unisi.ac.id/index.php/stmsi/article/view/5547/0.

[200] Yang-Seon Kim et al. “Investigating the impact of data normalization methods on predicting electricity consumption in a building using different artificial neural network models”. In: *Sustainable Cities and Society* 118 (2025), p. 105570. ISSN: 2210-6707. DOI: https://doi.org/10.1016/j.scs.2024.105570. URL: https://www.sciencedirect.com/science/article/pii/S2210670724003962.

[201] Zeqiye Zhan and Song-Kyoo Kim. “Versatile time-window sliding machine learning techniques for stock market forecasting”. In: *Artificial Intelligence Review* 57.8 (2024), p. 209. DOI: 10.1007/s10462-024-10851-x. URL: https://doi.org/10.1007/s10462-024-10851-x.

[202] Chongyang He et al. “A mini-batch algorithm for large-scale learning problems with adaptive step size”. In: *Digital Signal Processing* 143 (2023), p. 104230. ISSN: 1051-2004. DOI: https://doi.org/10.1016/j.dsp.2023.104230. URL: https://www.sciencedirect.com/science/article/pii/S1051200423003251.

[203] Philipp Probst, Anne-Laure Boulesteix, and Bernd Bischl. “Tunability: Importance of hyperparameters of machine learning algorithms”. In: *Journal of Machine Learning Research* 20.53 (2019), pp. 1–32.

[204] Mohaimenul Azam Khan Raiaan et al. “A systematic review of hyperparameter optimization techniques in Convolutional Neural Networks”. In: *Decision Analytics Journal* 11 (2024), p. 100470. ISSN: 2772-6622. DOI: https://doi.org/10.1016/j.dajour.2024.100470. URL: https://www.sciencedirect.com/science/article/pii/S2772662224000742.

[205] IBM. *What are model parameters?* Accessed: 2026-02-18. n.d. URL: https://www.ibm.com/think/topics/model-parameters.

[206] Giorgia Dellaferrera and Gabriel Kreiman. “Error-driven input modulation: solving the credit assignment problem without a backward pass”. In: *International Conference on Machine Learning*. PMLR. 2022, pp. 4937–4955.

[207] Diederik P. Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization". In: *Proceedings of the International Conference on Learning Representations (ICLR)*. 2015. URL: https://arxiv.org/pdf/1412.6980.

[208] Nan Cui. "Applying gradient descent in convolutional neural networks". In: *Journal of Physics: Conference Series*. Vol. 1004. 1. IOP Publishing. 2018, p. 012027.

[209] Robert Rosenbaum. "On the relationship between predictive coding and backpropagation". In: *PLOS ONE* 17.3 (2022), e0266102. DOI: 10.1371/journal.pone.0266102. URL: https://doi.org/10.1371/journal.pone.0266102.

[210] Ruslan Abdulkadirov, Pavel Lyakhov, and Nikolay Nagornov. "Survey of Optimization Algorithms in Modern Neural Networks". In: *Mathematics* 11.11 (2023). ISSN: 2227-7390. DOI: 10.3390/math11112466. URL: https://www.mdpi.com/2227-7390/11/11/2466.

[211] Ahmad Makinde. "Optimizing time series forecasting: A comparative study of adam and nesterov accelerated gradient on lstm and gru networks using stock market data". In: *arXiv preprint arXiv:2410.01843* (2024).

[212] Donghao Luo and Xue Wang. "DeformableTST: Transformer for time series forecasting without over-reliance on patching". In: *Advances in Neural Information Processing Systems* 37 (2024), pp. 88003–88044.

[213] Constantin Aliferis and Gyorgy Simon. "Overfitting, Underfitting and General Model Overconfidence and Under-Performance Pitfalls and Best Practices in Machine Learning and AI". In: *Artificial Intelligence and Machine Learning in Health Care and Medical Sciences: Best Practices and Pitfalls*. Ed. by Gyorgy J. Simon and Constantin Aliferis. Cham: Springer International Publishing, 2024, pp. 477–524. ISBN: 978-3-031-39355-6. DOI: 10.1007/978-3-031-39355-6_10. URL: https://doi.org/10.1007/978-3-031-39355-6_10.

[214] Ratnadip Adhikari and R. K. Agrawal. *An Introductory Study on Time Series Modeling and Forecasting*. 2013. arXiv: 1302.6613 [cs.LG]. URL: https://arxiv.org/abs/1302.6613.

[215] Rob J Hyndman and Anne B Koehler. "Another look at measures of forecast accuracy". In: *International journal of forecasting* 22.4 (2006), pp. 679–688.

[216] Alipujiang Jierula et al. "Study on Accuracy Metrics for Evaluating the Predictions of Damage Locations in Deep Piles Using Artificial Neural Networks with Acoustic Emission Data". In: *Applied Sciences* 11.5 (2021). ISSN: 2076-3417. DOI: 10.3390/app11052314. URL: https://www.mdpi.com/2076-3417/11/5/2314.

[217] Hoda Khoshvaght et al. "A critical review on selecting performance evaluation metrics for supervised machine learning models in wastewater quality prediction". In: *Journal of Environmental Chemical Engineering* 13.6 (2025), p. 119675. ISSN: 2213-3437. DOI: https://doi.org/10.1016/j.jece.2025.119675. URL: https://www.sciencedirect.com/science/article/pii/S2213343725043714.

[218] J. Kaliappan et al. "Performance Evaluation of Regression Models for the Prediction of the COVID-19 Reproduction Rate". In: *Frontiers in Public Health* 9 (2021), p. 695648. DOI: 10.3389/fpubh.2021.695648. URL: https://doi.org/10.3389/fpubh.2021.729795.

[219] Marta Alvir et al. "Reconstruction and analysis of negatively buoyant jets with interpretable machine learning". In: *Marine Pollution Bulletin* 190 (2023), p. 114881. ISSN: 0025-326X. DOI: https://doi.org/10.1016/j.marpolbul.2023.114881. URL: https://www.sciencedirect.com/science/article/pii/S0025326X23003120.

[220] Anna Pölz et al. "Transformer Versus LSTM: A Comparison of Deep Learning Models for Karst Spring Discharge Forecasting". In: *Water Resources Research* 60.4 (2024). e2022WR032602 2022WR032602, e2022WR032602. DOI:

https://doi.org/10.1029/2022WR032602. eprint: https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2022WR032602. URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2022WR032602.

[221] Giacomo Minotti. "Cryptocurrencies price prediction using LSTM neural network model". In: (2023).

[222] Souhaib Ben Taieb et al. "A review and comparison of strategies for multi-step ahead time series forecasting based on the NN5 forecasting competition". In: *Expert Systems with Applications* 39.8 (2012), pp. 7067–7083. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2012.01.039. URL: https://www.sciencedirect.com/science/article/pii/S0957417412000528.

[223] Bidesh Biswas Biki et al. "Analysis and Forecasting of Cryptocurrency Markets Using Bayesian and LSTM-Based Deep Learning Models". In: *Informatics* 12.3 (2025). ISSN: 2227-9709. DOI: 10.3390/informatics12030087. URL: https://www.mdpi.com/2227-9709/12/3/87.

[224] Ben Breitinger. "Transformers for Time-Series Forecasting". Master's thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, June 2024. URL: https://www.ins.uni-bonn.de/media/public/publication-media/Breitinger.pdf?pk=1837.

[225] Zhipeng Liu et al. "Disms-ts: Eliminating redundant multi-scale features for time series classification". In: *Proceedings of the 33rd ACM International Conference on Multimedia*. 2025, pp. 10817–10826.

[226] Guokun Lai et al. "Modeling Long- and Short-Term Temporal Patterns with Deep Neural Networks". In: *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. SIGIR '18. Ann Arbor, MI, USA: Association for Computing Machinery, 2018, pp. 95–104. ISBN: 9781450356572. DOI: 10.1145/3209978.3210006. URL: https://doi.org/10.1145/3209978.3210006.

[227] Ibram Abdelmalak et al. “Channel Dependence, Limited Lookback Windows, and the Simplicity of Datasets: How Biased is Time Series Forecasting?” In: *arXiv preprint arXiv:2502.09683* (2025).

[228] Marcin Watorek et al. “Filtering amplitude dependence of correlation dynamics in complex systems: application to the cryptocurrency market”. In: *arXiv preprint arXiv:2509.18820* (2025).

[229] Huiliang Zhang et al. “Nearest Neighbor Multivariate Time Series Forecasting”. In: *IEEE Transactions on Neural Networks and Learning Systems* 36.7 (July 2025), pp. 12606–12618. ISSN: 2162-2388. DOI: 10.1109/tnnls.2024.3490603. URL: http://dx.doi.org/10.1109/TNNLS.2024.3490603.

[230] Yunyao Cheng et al. “A Memory Guided Transformer for Time Series Forecasting”. In: *Proceedings of the VLDB Endowment* 18.2 (2024), pp. 239–252.

[231] Haolei Xi et al. “Cross-attention mechanism-based spectrum sensing in generalized Gaussian noise”. In: *Scientific Reports* 14 (2024). URL: https://doi.org/10.1038/s41598-024-74341-4.

[232] Mohamed Shaker Ahmed et al. “Cryptocurrency volatility: A review, synthesis, and research agenda”. In: *Research in International Business and Finance* 71 (2024), p. 102472. ISSN: 0275-5319. DOI: https://doi.org/10.1016/j.ribaf.2024.102472. URL: https://www.sciencedirect.com/science/article/pii/S0275531924002654.

[233] Wafa Masmoudi Kammoun. “Return and Volatility Spillover Drivers Among Conventional Cryptocurrencies”. In: *Digital Finance* 8.1 (2025), p. 2. ISSN: 2524-6186. DOI: 10.1007/s42521-025-00167-y. URL: https://doi.org/10.1007/s42521-025-00167-y.

[234] Huaigang Long et al. “Seasonality in the Cross-Section of Cryptocurrency Returns”. In: *Finance Research Letters* 35 (2020), p. 101566. ISSN: 1544-6123. DOI: https://doi.org/10.1016/j.frl.2020.101566. URL: https://www.sciencedirect.com/science/article/pii/S154461232030235X.

[235] Evangelos Pintelas et al. "Investigating the Problem of Cryptocurrency Price Prediction: A Deep Learning Approach". In: *Artificial Intelligence Applications and Innovations*. Vol. 584. Springer, 2020, pp. 99–110. DOI: 10.1007/978-3-030-49186-4_9.

[236] Liyilei Su et al. "A systematic review for transformer-based long-term series forecasting". In: *Artificial Intelligence Review* 58 (2025). DOI: https://doi.org/10.1007/s10462-024-11044-2.

[237] Deborah Adedigba, David Agbolade, and Raza Hasan. "Machine Learning Approaches to Cryptocurrency Trading Optimization: A Comparative Analysis of Predictive Models". In: *Discover Artificial Intelligence* 5.1 (2025), p. 310. ISSN: 2731-0809. DOI: 10.1007/s44163-025-00519-y. URL: https://doi.org/10.1007/s44163-025-00519-y.

[238] Mabsur Fatin Bin Hossain et al. *FinBERT-BiLSTM: A Deep Learning Model for Predicting Volatile Cryptocurrency Market Prices Using Market Sentiment Dynamics*. Papers 2411.12748. arXiv.org, Nov. 2024. URL: https://ideas.repec.org/p/arx/papers/2411.12748.html.

[239] Ming Che Lee. "Temporal Fusion Transformer-Based Trading Strategy for Multi-Crypto Assets Using On-Chain and Technical Indicators". In: *Systems* 13.6 (2025). ISSN: 2079-8954. DOI: 10.3390/systems13060474. URL: https://www.mdpi.com/2079-8954/13/6/474.

[240] Siru Zhong et al. "Time-VLM: Exploring Multimodal Vision-Language Models for Augmented Time Series Forecasting". In: *Forty-second International Conference on Machine Learning*. 2025. URL: https://openreview.net/forum?id=b5h60xQnzM.

[241] Georgios Petmezas et al. "Recent advancements and applications of deep learning in heart failure: systematic review". In: *Computers in Biology and Medicine* 176 (2024), p. 108557. ISSN: 0010-4825. DOI: https://doi.org/10.

1016/j.compbiomed.2024.108557. URL: https://www.sciencedirect.com/science/article/pii/S0010482524006413.

[242] Bahareh Amirshahi and Salim Lahmiri. "Investigating the effectiveness of Twitter sentiment in cryptocurrency close price prediction by using deep learning". In: *Expert Systems* 42.1 (2023), e13428. DOI: https://doi.org/10.1111/exsy.13428. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/exsy.13428. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/exsy.13428.

[243] DeLesley Hutchins et al. "Block-recurrent transformers". In: *Advances in neural information processing systems* 35 (2022), pp. 33248–33261.

[244] Zhiwei Wang et al. "R-transformer: Recurrent neural network enhanced transformer". In: *arXiv preprint arXiv:1907.05572* (2019).

[245] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. "Neural Machine Translation by Jointly Learning to Align and Translate". In: *CoRR* (2016). arXiv: 1409.0473 [cs.CL]. URL: https://arxiv.org/abs/1409.0473.

[246] Thang Luong, Hieu Pham, and Christopher D. Manning. "Effective Approaches to Attention-based Neural Machine Translation". In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal: Association for Computational Linguistics, Sept. 2015, pp. 1412–1421. DOI: 10.18653/v1/D15-1166. URL: https://aclanthology.org/D15-1166/.

[247] Jianpeng Cheng, Li Dong, and Mirella Lapata. "Long Short-Term Memory-Networks for Machine Reading". In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas: Association for Computational Linguistics, Nov. 2016, pp. 551–561. DOI: 10.18653/v1/D16-1053. URL: https://aclanthology.org/D16-1053/.

[248] Gianni Brauwers and Flavius Frasincar. "A general survey on attention mechanisms in deep learning". In: *IEEE transactions on knowledge and data engineering* 35.4 (2021), pp. 3279–3298.

[249] Guoxin Wang et al. "A simple yet effective learnable positional encoding method for improving document transformer model". In: *Findings of the Association for Computational Linguistics: AACL-IJCNLP 2022*. 2022, pp. 453–463.

[250] Ali Reza Sajun, Imran Zualkernan, and Donthi Sankalpa. "A Historical Survey of Advances in Transformer Architectures". In: *Applied Sciences* 14.10 (2024). ISSN: 2076-3417. DOI: 10.3390/app14104316. URL: https://www.mdpi.com/2076-3417/14/10/4316.