TILBURG UNIVERSITY

PREDICTING STOCK MARKET VOLATILITY USING NEURAL NETWORK MODELS

A COMPARATIVE STUDY ON BIDIRECTIONAL TEMPORAL CONVOLUTIONAL NETWORK (BITCN), NEURAL HIERARCHICAL INTERPOLATION FOR TIME SERIES (NHITS), TIME-SERIES DENSE ENCODER (TIDE), AND TEMPORAL FUSION TRANSFORMER(TFT) ACROSS 20 YEARS OF S&P 500 CONSTITUENT STOCKS

JOHN ALEJANDRO MENDOZA ORTIZ

THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE IN DATA SCIENCE & SOCIETY
AT THE SCHOOL OF HUMANITIES AND DIGITAL SCIENCES
OF TILBURG UNIVERSITY

STUDENT NUMBER

2109321

COMMITTEE

ir. Federico Zamberlan
dr. Silvy Collin

LOCATION

Tilburg University
School of Humanities and Digital Sciences
Department of Cognitive Science &
Artificial Intelligence
Tilburg, The Netherlands

DATE

January 10th, 2025

WORD COUNT

8799

ACKNOWLEDGMENTS

I would like to express my heartfelt gratitude to my parents, John Jairo and Leonila, for their unconditional support, guidance, and love throughout this journey. To my family, who always believed in me, wished for my success, and inspired me to deliver my best, thank you for being my foundation. To my friends, both near and far, your unwavering support and encouragement have helped me grow into the person I am today.

PREDICTING STOCK MARKET VOLATILITY USING NEURAL NETWORK MODELS

A COMPARATIVE STUDY ON BIDIRECTIONAL TEMPORAL CONVOLUTIONAL NETWORK (BITCN), NEURAL HIERARCHICAL INTERPOLATION FOR TIME SERIES (NHITS), TIME-SERIES DENSE ENCODER (TIDE), AND TEMPORAL FUSION TRANSFORMER(TFT) ACROSS 20 YEARS OF S&P 500 CONSTITUENT STOCKS

JOHN ALEJANDRO MENDOZA ORTIZ

Abstract

This thesis addresses the challenge of predicting stock market volatility, a critical metric for risk management and investment strategies. The research evaluates four machine learning models: Bidirectional Temporal Convolutional Network (BiTCN), Neural Hierarchical Interpolation for Time Series (NHITS), Time-series Dense Encoder (TiDE), and Temporal Fusion Transformer (TFT). The study focuses on forecasting 30-day historical volatility using data from 385 S&P 500 companies spanning 20 years (2004-2023). Additionally, the models' generalization is tested on the Amsterdam Exchange Index (AEX).

The methodology includes a comprehensive evaluation of model performance across multiple forecast horizons (1, 5, 10, and 20 days). Key metrics such as Quasi-Likelihood Error (QLIKE) and Diebold-Mariano (DM) tests are employed to assess accuracy and statistical robustness. The findings indicate that NHITS excels in minimizing prediction error, while TFT consistently shows strong performance in statistical tests, particularly for generalization. Experimentation with reduced data availability further reveals BiTCN's adaptability to constrained datasets.

This research contributes to understanding the strengths and limitations of advanced machine learning models in financial volatility forecasting, highlighting the benefits of incorporating firm-specific and technical indicators to enhance prediction accuracy and robustness across different market conditions.

1 DATA SOURCE, ETHICS, CODE, AND TECHNOLOGY STATEMENT

The dataset used in this thesis is sourced from Datastream International (2024) and maintained by Refinitiv Workspace (formerly known as Eikon), under the London Stock Exchange Group (LSEG). This data provides financial information for the constituent companies of the S&P 500 and AEX indices. As the dataset focuses exclusively on corporate entities, no human subjects are involved.

The code developed for this thesis is publicly accessible via a GitHub repository [1]. Portions of the code have been adapted from Souto and Moradi (2024), with all reused or adapted fragments explicitly marked within the notebooks. A comprehensive list of libraries and packages used for development and analysis is provided in Appendix A (page 49) for reference.

During this thesis, ChatGPT by OpenAI (2024) was used to refine language, resolve coding errors, and assist in academic writing. All outputs were reviewed and edited by the author, who assumes full responsibility for the final text. Figure 1 was created using Lucidchart [2], references were managed using Zotero [3], and the document was written using Overleaf [4]. No other typesetting tools or services were used.

All figures presented in this thesis are original and created by the author. The computational models were executed using Google Colab's A100 GPUs.

[1] https://github.com/johmmendoza/MSc-DataScience-Thesis-StockVolatilityNN
[2] https://www.lucidchart.com/
[3] https://www.zotero.org/
[4] https://www.overleaf.com/

2 INTRODUCTION

Volatility measures the variation in financial asset prices over time, typically using the standard deviation of returns. It represents uncertainty and risk: high volatility indicates large price swings, presenting both opportunities and risks for investors. Volatility also affects investor confidence, with stable markets encouraging investment and high volatility often triggering selloffs and economic instability (Takahashi et al., 2021).

In financial markets, volatility is critical as both a risk measure and a signal of market behavior. It underpins metrics like Value-at-Risk (VaR) and expected shortfall, essential for risk management strategies (Brownlees et al., 2011; Engle & Manganelli, 2004). Accurate volatility forecasting helps market participants anticipate price movements, optimize portfolios, and balance risk and return (BIS, 2019; Lopez, 2001).

2.1 *Societal relevance*

Volatility forecasting is vital for financial stability, especially during periods of economic uncertainty. For instance, during the 2008 financial crisis, high market volatility destabilized global financial systems (Brownlees et al., 2011). Accurate predictions allow regulators to set adequate capital buffers for banks, as emphasized in the Basel framework (BIS, 2019; The Hong Kong Institute Of Bankers, 2008).

This study's findings on stock volatility may also apply broadly to other financial time series, such as exchange rates and interest rates, which share characteristics like volatility clustering and fat-tailed distributions (Sewell, 2011; Tsay, 2005). Accurate forecasts benefit investors by guiding portfolio management and mitigating risks (Brownlees et al., 2011), as seen during the COVID-19 pandemic when rapid asset price changes impacted investments.

From a macroeconomic perspective, volatility data informs monetary and fiscal policies. Central banks and regulators use it to evaluate market health and adjust interest rates or liquidity provisions to stabilize economies (Díaz et al., 2024). For corporations, reliable volatility forecasts enhance decision-making on capital investments, mergers, and workforce management, supporting economic growth and job stability (Evans et al., 2022)

2.2 *Scientific relevance*

The prediction of realized volatility is crucial in financial economics, impacting portfolio management, risk assessment, and investment strategies

(Atkins et al., 2018; Bašta & Molnár, 2018; Bonato et al., 2022; Bouri et al., 2021; Gobato Souto, 2023). Traditional models like the Generalized Autoregressive Conditional Heteroskedasticity (GARCH) (Bollerslev, 2023) and HAR (Corsi, 2008) have provided foundational insights, while recent ML models such as Temporal Fusion Transformer (TFT) (Lim et al., 2021) and NHITS (Challu et al., 2022) have advanced forecasting accuracy.

The insights from this research may apply to other financial time series, such as exchange rates, which share features like leverage effects and volatility clustering (Sezer et al., 2020). By applying ML models to 20 years of S&P 500 data and testing them on the Amsterdam Exchange Index (AEX), this study assesses model adaptability and the value of combining technical and macroeconomic indicators. This approach benefits both short-term traders and long-term investors (Gradojevic & Tsiakas, 2021; Müller et al., 1997) .

Despite ML success in predicting volatility, combining technical indicators and macroeconomic variables has been under explored. Souropanis and Vivian (2023) and L. Liu and Pan (2020) argue that these indicators provide complementary information, improving predictions. Technical indicators focus on short-term market movements, while macroeconomic variables capture long-term trends, enhancing understanding of market dynamics (Engle et al., 2013; Mittnik et al., 2015). This combination can leverage different frequency components to improve model robustness across market conditions.

This study also relevant to the debate on whether economically motivated models or purely data-driven approaches are better for forecasting (Bianchi et al., 2020; Campbell & Thompson, 2007; Kelly & Pruitt, 2013). While advanced models enhance predictive performance, economic restrictions may improve interpretability and stability. Combining technical indicators and macroeconomic predictors, as discussed by Lin (2018) and Neely et al. (2014), offers a way to understand how market information influences volatility, providing actionable insights for risk management and decision-making.

2.3 *Research strategy and research questions*

Given the context above, this research aims to explore the following main research question:

To what extent can neural network models predict stock volatility?

To guide the main research question, three sub-research questions were formulated:

RQ1 *How do Bidirectional Temporal Convolutional Network (BiTCN), Neural Hierarchical Interpolation for Time Series (NHITS), Time-series Dense Encoder (TiDE), and Temporal Fusion Transformer (TFT) algorithms perform in predicting the volatility of constituent stocks from the S&P 500 over a 20-year span?*

RQ2 *Can these models retain their prediction capabilities when their training period is halved?*

RQ3 *How well do the trained models perform when forecasting beyond the test of the S&P500 as well as applied to the constituent stocks of the Amsterdam Exchange Index (AEX)?*

3 RELATED WORK

Volatility forecasting has been approached using a range of methods, from classical econometric models to modern machine learning architectures. GARCH-type models are often chosen for their theoretical clarity and ease of use. However, the nonlinear nature of financial markets has also prompted investigations into neural networks and other flexible ML algorithms, aided by the increasing availability of high-frequency data.

3.1 *Econometric vs. Machine Learning Approaches*

Classical econometric models—particularly GARCH variants—remain popular for their relatively transparent assumptions and ease of interpretation. For instance, basic GARCH formulations capture volatility clustering (a hallmark of financial time series), while extensions such as Exponential Generalized Autoregressive Conditional Heteroskedasticity (EGARCH) and Fractionally Integrated Generalized Autoregressive Conditional Heteroskedasticity (FIGARCH) aim to model asymmetries and long memory. However, a growing literature points to the Heterogeneous Autoregressive (HAR) model as a strong benchmark. One reason is its intuitive design, splitting realized volatility into short-, medium-, and long-term components.

Using daily realized volatility data from the Oxford-Man Institute's Realized Library[5] (spanning over a decade for ten global equity indices[6]),

[5] The Oxford-Man Institute's Realized Library is no longer available https://oxford-man.ox.ac.uk/research/realized-library/.

[6] Bovespa/BVSP (Brazil), CAC 40 (France), DAX (Germany), Euro Stoxx 50 (Europe), FTSE 100 (UK), Nasdaq 100 (US), Nifty 50 (India), Nikkei 225 (Japan), Shanghai Composite Index/SCI (China), S&P 500 (US)

Branco et al. (2024) find that HAR models perform on par with, or sometimes better than, more complex out-of-sample ML methods. Notably, the study highlights that ML gains become less apparent when longer forecast horizons are considered, suggesting that simpler models such as HAR can be robust for multiday and monthly horizons.

In contrast, Zhang et al. (2024) test various neural network architectures on rolling windows of S&P 500 realized volatility data. Their forecasts are evaluated under short (1-day to 3-day) and moderate (5-day to 10-day) horizons, using metrics such as MSE and QLIKE. The findings reveal strong short-horizon performance for neural networks, but accuracy deteriorates for forecasts extending beyond a week. These results imply that while deep learning can capture short-term nonlinearities effectively, its advantage may wane if market conditions shift or if the horizon is extended, mirroring the resilience of simpler econometric structures in stable market regimes.

It is increasingly recognized that the performance differences hinge on market volatility regimes.Gunnarsson et al. (2024) show that neural networks and other ML models outperform GARCH-based models during turbulent periods, likely because ML can adapt to sudden spikes or drops in market volatility. Conversely, econometric methods remain competitive when conditions normalize, reflecting their strong inductive bias for steady-state patterns. This dichotomy has fueled interest in hybrid systems, where a regime-classification module might switch between (or combine) ML and traditional models according to real-time volatility indicators.

3.2 *Advanced Econometric Extensions and Hierarchical Forecasting*

Although HAR has proven robust, its fixed-parameter form can miss shifts in market volatility. To address this, Luo et al. (2023) propose a time-varying HAR that employs dynamic model selection (DMS) and dynamic model averaging (DMA). Their dataset consists of high-frequency Chinese financial futures spanning multiple years, which are aggregated into daily realized volatilities. Using rolling-window forecasts evaluated by both RMSE and QLIKE, they demonstrate that dynamic parameter updating can yield significant gains over static HAR in out-of-sample predictions, especially during market disruptions.

An alternative stream of research addresses the challenge of forecasting index-level volatility by first modeling individual constituent stocks, then aggregating (or "reconciling") their volatility predictions. Caporin et al. (2024) apply bottom-up and regression-based reconciliation methods to data for the 30 Dow Jones Industrial Average (DJIA) stocks [7]. Their Partial

[7] The data for these evaluations were sourced from https://www.kibot.com/, comparable to the NYSE TAQ dataset.

Variance (PV(3)) shrinkage-based reconciliation (PV(3)shr) breaks intraday returns into three partial variances according to predefined quantiles, ensuring consistency between stock-level and index-level forecasts. Their results, based on QLIKE and MSE over 1-day to 22-day horizons, show that PV(3)shr significantly outperforms direct forecasting, particularly by capturing idiosyncratic signals at the stock level that aggregate into more accurate index-level volatility predictions.

3.3 *Deep Learning Models for Financial Time Series*

Volatility forecasts in highly granular limit order book (LOB) data have also been explored. Gobato Souto (2023) utilize data from LOBSTER: Limit Order Book System [8] to examine stock realized volatility in the S&P 100 index. Their primary evaluation metrics include RMSE, MAE, MAPE, and QLIKE, the latter being particularly relevant for its sensitivity to volatility forecast errors. The study focuses on the NHITS architecture, which integrates traditional time-series decomposition with deep neural sub-networks. The authors report that NHITS achieves a QLIKE score of 3.300% in the main sample, outperforming GARCH (4.992%) and LSTM (3.489%) across all 80 stocks, demonstrating its effectiveness in short-term volatility forecasting.

Building on the same dataset, Souto and Moradi (2024) investigate the application of second-generation Transformer variants (Autoformer and PatchTST) to stock realized volatility. Their findings emphasize the models' ability to capture both local and long-range dependencies, particularly with limited historical data. Notably, PatchTST achieves a QLIKE score of 3.096% for one-day-ahead forecasts (H = 1), surpassing the baseline HAR model's 3.249%, which highlights PatchTST's superior handling of volatility clustering in short-term forecasts.

In parallel, Díaz et al. (2024) evaluate LSTM and GRU models—both well-known for capturing temporal dependencies in sequential data—on high-frequency S&P 500 data. Their rolling-window experiments track short (1 to 5-day) and longer (10 to 20-day) horizons, showing that LSTM and GRU excel in near-term forecasts. Notably, LSTM achieves a QLIKE score of 68% for one-day-ahead forecasts, outperforming the baseline HAR model's 81%, reflecting its strong performance in capturing volatility fluctuations over short horizons. However, the magnitude of improvement over econometric models diminishes for longer horizons, highlighting the trade-off between capturing complex short-run patterns versus maintaining stable long-run forecasts.

[8] The Efficient Reconstructor at Humboldt Universität zu Berlinhttps://lobsterdata.com/index.php

3.4 *Integrating Technical and Macroeconomic Indicators*

Technical indicators (e.g., moving averages, RSI) continue to attract attention for capturing short-run market inefficiencies or behavioral biases. In equity risk premium forecasts, Neely et al. (2014) show that combining these indicators with macroeconomic factors significantly enhances out-of-sample forecasts , especially under increased uncertainty. This finding emphasizes that technical signals often relate to momentum or short-term overbought/oversold conditions, complementing more fundamental economic data.

From a long-horizon perspective, Lin (2018) propose a Partial Least Squares (PLS) approach designed to filter out noisy idiosyncratic fluctuations in commonly used technical indicators. Evaluated on both U.S. and international market data, their PLS-based "technical analysis index" consistently surpasses simpler macroeconomic predictors in both in-sample and out-of-sample tests, capturing persistent trends that conventional technical indicators may miss. Likewise, Engle et al. (2013) demonstrate that combining technical variables with macroeconomic fundamentals in a GARCH-MIDAS framework can significantly improve forecasts over monthly and quarterly horizons. Such integration underscores the layered nature of volatility, with short-run dynamics often driven by sentiment or technical factors, and longer-run fluctuations more closely tied to economic cycles and monetary policy signals.

3.5 *Literature Gap*

Empirical findings on whether machine learning (ML) models surpass traditional econometric approaches for volatility forecasting remain inconclusive. For instance, Branco et al. (2024) and Souropanis and Vivian (2023) demonstrate that linear models, especially HAR, still outperform many nonlinear ML methods in forecasting realized volatility (RV) for global equity indices. Although enriching these models with implied volatility and macroeconomic variables generally improves predictive accuracy, the advantage of advanced ML techniques has not been consistently proven—particularly for longer horizons or more stable markets.

Nevertheless, Transformer-based models have shown the ability to capture temporal dependencies and adapt to high-volatility market conditions, as evidenced by Souto and Moradi (2024), who highlight Autoformer and PatchTST as particularly effective with limited historical data. Alongside this, ML methods (e.g., LSTM, GRU) excel in short-term forecasts, though their dominance often tapers off for medium- to long-term predictions. This underscores the potential benefits of hybrid or regime-aware modeling, where econometric and deep learning approaches complement each other in different market states.

A gap lies in integrating technical and macroeconomic indicators into DL architectures. While Souropanis and Vivian (2023) and L. Liu and Pan (2020) illustrate that combining technical signals and macroeconomic data can enhance forecast accuracy, studies do so often relying on simpler model frameworks. Hence, further research is warranted to explore how advanced deep learning architectures, particularly Transformers, can seamlessly integrate technical indicators and macroeconomic variables for improved volatility forecasting across multiple time horizons and market regimes. This includes assessing whether these augmented models can retain their short-term advantages while maintaining robust long-horizon performance and adapting effectively to evolving market conditions.

4 METHOD

4.1 *Methodology Overview*

The methodology overview is illustrated in Figure 1, which outlines the steps taken in this research. Section 4.2 provides details about the dataset, while Section 4.3 explains the pre processing steps, including data splitting, imputation, and the creation of exogenous variables. Section 4.4 introduces the models used in the study, and Section 4.5 describes the methodology for hyper-parameter optimization. Section 4.4 outlines the experimental setup for model deployment, and Section 4.7 covers the evaluation methodology, including performance metrics and statistical tests employed.

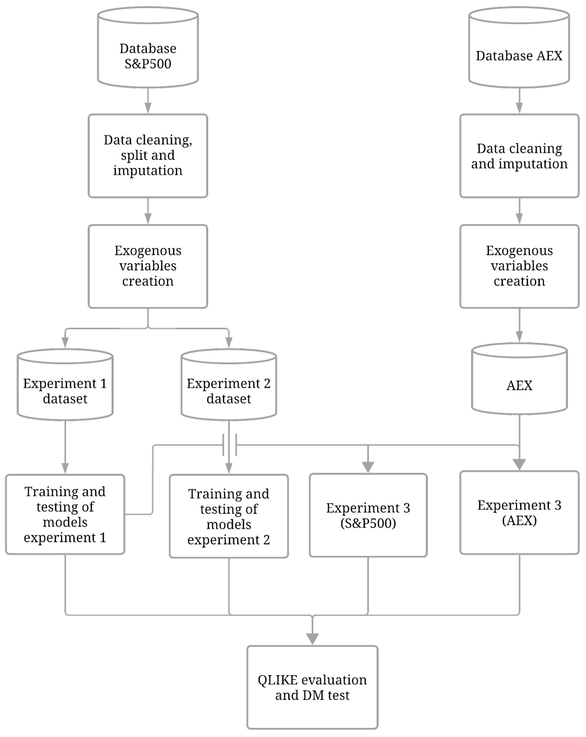
Database S&P500
Database AEX
Data cleaning, split and imputation
Data cleaning and imputation
Exogenous variables creation
Exogenous variables creation
Experiment 1 dataset
Experiment 2 dataset
AEX
Training and testing of models experiment 1
Training and testing of models experiment 2
Experiment 3 (S&P500)
Experiment 3 (AEX)
QLIKE evaluation and DM test

Figure 1: Methodology Overview

4.2 *Dataset Description*

This study analyzes 385 continuously traded S&P 500 companies from January 1, 2004, to December 31, 2023, covering 5,217 business days per stock and totaling 2,008,545 observations. The analysis spans major events such as the 2008 financial crisis, the Covid-19 pandemic, the 2010 flash crash, the European Debt Crisis (2010–2012), the 2018 cryptocurrency crash, and the 2018–2019 U.S.-China Trade War, providing a robust foundation for evaluating models across diverse market conditions.

In comparison, the AEX dataset includes 18 companies from January 1, 2016, to December 31, 2023. Although smaller, this dataset offers valuable insights into a different market, enabling a cross-market evaluation of model performance.

Both datasets include 30-day historical volatility (840E) as the target variable, along with six exogenous technical indicators inspired by Souropanis and Vivian (2023) and L. Liu and Pan (2020): annualized dividend yield (DY), price-to-book ratio (PTBV), closing price (P), opening price (PO), traded volume (VO), and price-to-earnings ratio (PE). Appendix B (page 50) provides descriptive statistics for both datasets.

While L. Y. Liu et al. (2015) recommends realized volatility (RV) calculated over 5-minute intervals, this study uses 840E data sourced from Datastream International (2024) for its accessibility to financial professionals. LSEG calculates volatility using the formula described in Equation 1

$$840E = \sqrt{250} \times \sigma\left(\ln\left(\frac{X_t}{X_{t-1}}\right), 30D\right) \times 1.000 \tag{1}$$

Where:

- X_t: the current day's closing price.
- X_{t-1}: the previous day's closing price.
- $\ln\left(\frac{X_t}{X_{t-1}}\right)$: calculates the natural logarithmic returns.
- σ: the standard deviation calculated over a 30-day period.
- $\sqrt{250}$: the annualizing factor, assuming 250 trading days in a year.

Formula 1 captures how much the price of a financial instrument fluctuates on average over a 30-day period, expressed as an annualized percentage. A volatility value of 0.12 implies that the instrument's price is expected to fluctuate within a 12% range (one standard deviation) over a year.

4.3 *Pre processing steps*

4.3.1 *Data splitting*

The S&P500 dataset includes only companies with complete data from January 1, 2004, to December 31, 2023, ensuring minimal bias from missing target variable data. After filtering, exogenous variables (DY, PTBV, P, PO, VO, and PE) were included, though these variables contained missing data.

To facilitate model training and evaluation, the S&P500 dataset was split into training (2004-01-01 to 2018-01-01), validation (2018-01-01 to 2019-12-31), and test (2020-01-01 to 2023-12-31) sets, following a 70/10/20 split. This approach enables models to train on historical data, validate on recent events, and test on the most current data, including high-volatility periods such as the Covid-19 pandemic. The segmentation by calendar years preserves real-world financial trends and market regimes, ensuring a realistic evaluation of model performance.

4.3.2 *Exploratory Data Analysis*

The main objective of this section is to test for fat-tailed characteristics in both datasets. Five statistical tests were applied to identify features commonly associated with fat-tailed distributions. The Augmented Dickey-Fuller (ADF) Test evaluates stationarity, as non-stationary series can amplify volatility and contribute to heavy tails. The Autoregressive Conditional Heteroskedasticity (ARCH) Test detects volatility clustering, a hallmark of fat-tailed behavior, through heteroskedasticity. Change Point Detection identifies structural shifts, which often signal abrupt changes linked to heavy tails. The Skewness and Kurtosis Test assesses excess kurtosis and skewness, direct indicators of deviation from normality. Finally, the Jarque-Bera Test evaluates overall normality, with failure indicating characteristics typically associated with fat tails. Based on the results summarized in Appendix C (page 51), these tests confirm the presence of fat-tailed behavior in the datasets, justifying the use of a Student-t loss as a robust fit for such distributions.

4.3.3 *Data imputation*

Based on the missing data analysis for both datasets in Appendix D (page 53), some variables exhibit a not at random (MNAR) missing data mechanism, where missingness may depend on data collection or reporting processes. Iterative imputation with Bayesian Ridge Regression was used because it models each variable as a function of others, capturing dependencies in the data. Bayesian Ridge Regression handles multicollinearity and provides probabilistic estimates, ensuring imputations are consistent with the dataset's structure while minimizing bias.

4.3.4 *Technical indicators*

Technical indicators predict stock volatility by capturing market dynamics that fundamental analysis often misses. A key concept in stock markets is the "leverage effect" (Christie, 1982), where negative returns lead to higher future volatility due to increased financial leverage, making stocks riskier. This asymmetric return-volatility relationship enables technical indicators to predict realized volatility (RV) by analyzing these patterns (Veronesi, 1999).

Volatility arises through several channels. Buying signals from technical rules attracts traders, creating noise and increasing volatility (Brown & Cliff, 2004). Herding behavior and cognitive biases among less informed investors disrupt market equilibrium, while negative signals amplify volatility through short-sale constraints (Hong & Stein, 2003) and volatility feedback (Campbell & Hentschel, 1991). High liquidity also contributes to excessive volatility as "noise" trading dominates over news (Frankel & Froot, 1990).

Market sentiment further impacts volatility. During high sentiment periods, optimistic trends reduce risk aversion, lowering volatility (Shu & Chang, 2015). Conversely, pessimistic sentiment or extreme trends can heighten volatility (Lee et al., 2002).

This study incorporates 13 technical indicators based on Moving Average (MA), Momentum (MOM), Relative Strength Index (RSI), and Exponential Moving Average (EMA) strategies, following Souropanis and Vivian (2023). These indicators identify trends and produce daily buy-sell signals ("1" for buy, "0" for sell) under the assumption of consistent market patterns (Menkhoff & Taylor, 2007).

Although initially debated under the Efficient Market Hypothesis, technical indicators are now widely studied for explaining market anomalies. Research shows their effectiveness in forecasting various asset classes, including exchange rates (Allen & Taylor, 1990), equities (Neely et al., 2014), bond premia (Goh et al., 2013), hedge funds (Smith et al., 2014), and oil (Czudaj, 2019).

4.3.4.1 Moving Average Following recent literature such as Panopoulou and Souropanis (2019), Lin (2018), and Marshall et al. (2017), we use MA-based rules as the initial category of technical indicators. These rules trigger buy or sell signals when a short-term MA crosses a long-term MA, thus suggesting a change in trend direction. A buy signal occurs when the short-term moving average surpasses the long-term moving average, whereas a sell signal is generated in the opposite case. The MA calculation is given by:

$$x_{i,j} = \begin{cases} 1 & \text{if } MA_{s,j} \geq MA_{l,j} \\ 0 & \text{if } MA_{s,j} < MA_{l,j} \end{cases}, \quad MA_{j,l} = \left(\frac{1}{j}\right) \sum_{i=0}^{j-1} P_{t-i}, \quad \text{for } j = s, l \qquad (2)$$

where P_t represents the price on day t, and j is either the short-term period (s) or the long-term period (l). We utilize $s \in \{1, 2, 3\}$ days and $l \in \{9, 12\}$ days, denoting these rules as $MA(s, l)$.

4.3.4.2 Momentum Momentum indicators, another group of trend-following rules, are simple yet prevalent in literature (Goh et al., 2013; Jamali & Yamani, 2019; Panopoulou & Souropanis, 2019) these indicators suggest that the movement of asset prices tends to persist. Marshall et al. (2017) notes a close relationship between MA and Momentum rules, though the latter tends to respond more slowly to changes. The Momentum signal is generated when the current price (P_t) is higher or lower than the price k periods prior, represented as:

$$S_{i,t} = \begin{cases} 1 & \text{if } P_t \geq P_{t-k} \\ 0 & \text{if } P_t < P_{t-k} \end{cases} \tag{3}$$

where k is set to 9 or 12 days, denoted as $\text{MOM}(k)$.

4.3.4.3 Relative strength index The third group of indicators includes oscillators such as the Relative Strength Index (RSI). RSI is a contrarian indicator that seeks to determine whether an asset is overbought or oversold, thereby indicating potential market corrections (Levy, 1967; Wilder, 1978). RSI values range between 0 and 100, where higher values indicate overbought conditions and lower values indicate oversold conditions. The RSI formula is defined as:

$$RS_{i,t} = 100 - \frac{100}{1 + RS_t} \tag{4}$$

where RS_t is the relative strength at time t, calculated as the ratio of the n-period moving average of upclose (uc_t) to the moving average of downclose (dc_t):

$$RS_t = \frac{MA_t^n(dc_t)}{MA_t^n(uc_t)} \tag{5}$$

The upclose (uc_t) and downclose (dc_t) measures are defined as:

$$uc_t = \begin{cases} \Delta P_t & \text{if } \Delta P_t \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

$$dc_t = \begin{cases} \Delta P_t & \text{if } \Delta P_t \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

where $\Delta P_t = P_t - P_{t-1}$. The buy or sell signals are generated based on whether the RSI value exceeds or falls below a threshold.

$$S_{i,t} = \begin{cases} 1 & \text{if } RSI_{i,t} < 50 \\ 0 & \text{if } RSI_{i,t} \geq 50 \end{cases} \tag{8}$$

We use two versions of RSI, with n set to 7 and 14 days.

4.3.5 *Exponential moving average*

The fourth and final group of indicators involves Exponential Moving Averages (EMA). Unlike simple MAs, EMAs assign exponentially greater weight to recent price observations, making them more responsive to new trends (Panopoulou & Souropanis, 2019; Zarrabi et al., 2017). Like MA rules, a buy signal is generated when the short-term EMA exceeds the long-term EMA, and a sell signal is generated in the opposite case:

$$S_{i,t} = \begin{cases} 1 & \text{if } EMA_{s,t} \geq EMA_{l,t} \\ 0 & \text{if } EMA_{s,t} < EMA_{l,t} \end{cases} \tag{9}$$

The EMA is calculated using the following formula:

$$EMA_t = (S_t - EMA_{t-1}) \cdot m + EMA_{t-1} \tag{10}$$

where m is the weighting multiplier or accelerator, given by $m = \frac{2}{j+1}$. We consider the following EMA combinations: $EMA(3,9)$, $EMA(5,9)$, and $EMA(5,12)$.

4.3.6 *Firm indicators*

This research incorporates firm-specific financial indicators as predictors inspired by the variables used in return forecasting (Welch & Goyal, 2007). Firm-specific measures like Dividend Yield (DY), Price to Book Ratio (PTBV), and Earnings Price Ratio (PE) offer insights into a company's valuation, market position, and risk exposure. This approach builds on the relationship between economic cycles and equity volatility (Engle & Rangel, 2008; Schwert, 1989). Additionally, trading Volume (VO) connects company-level activity to market volatility, as highlighted by Mittnik et al. (2015).

At company level, the following indicators function as firm-specific predictors, providing insights into individual company performance and valuation. This setup allows us to assess each company's unique market dynamics:

1. **Dividend Yield (DY)**: Company-specific profitability measure, reflecting dividend returns relative to price.

2. **Price to Book Ratio (PTBV)**: Indicates company valuation relative to net assets.

3. **Opening Price (PO)**: Captures early trading sentiment for each company.

4. **Volume (VO)**: Shows liquidity and investor interest on a company-by-company basis.
5. **Earnings Price Ratio (PE)**: Reflects company profitability, indicating market confidence.

These company-specific predictors help model idiosyncratic risk and individual firm trends, enhancing forecasting precision across entities.

4.4 *Models methodology*

4.4.1 *Bidirectional Temporal Convolutional Network (BiTCN)*

The Bidirectional Temporal Convolutional Network (BiTCN), introduced by Sprangers et al. (2023), uses temporal convolutional networks (TCNs) to improve the efficiency and accuracy of time series forecasting. BiTCN's architecture includes two TCNs—one for encoding future covariates and one for past observations. This bidirectional setup allows the model to capture both historical and future information, improving forecasting.

BiTCN enhances efficiency by reducing sequential operations compared to recurrent models like LSTMs. Using dilated convolutions, BiTCN captures long-term dependencies without the slow training times of recurrent networks. The convolutional design also supports parallel processing, which speeds up training.

BiTCN uses a Student's t-distribution as the loss function to improve robustness. This is suited for financial time series, which often show fat-tailed behavior and high volatility. The Student's t-distribution penalizes large errors less severely than a Gaussian distribution, making BiTCN more resilient to outliers.

Overall, BiTCN's bidirectional encoding, temporal convolutions, and robust loss function make it effective for time series forecasting, especially in finance. The model uses future information efficiently, allowing it to manage large datasets and produce reliable forecasts across different time horizons.

4.4.2 *Neural Hierarchical Interpolation for Time Series (NHITS)*

The Neural Hierarchical Interpolation for Time Series (NHITS) is a deep learning model introduced by Challu et al. (2022) for hierarchical time series forecasting. It extends N-BEATS to manage multi-rate signal sampling and multi-scale synthesis, improving accuracy and efficiency, especially for long-horizon forecasting. NHITS divides the input time series into segments with different temporal granularity, allowing each segment to focus on relevant dynamics.

The model consists of multiple stacks, each with several blocks. Each block uses a Multilayer Perceptron (MLP) to generate coefficients for back casting and forecasting. Back casting removes unwanted components from the input, refining the signal for the next block, while forecasting predicts future values. This helps NHITS capture both short-term variations and long-term dependencies, making it effective for financial forecasting.

NHITS uses hierarchical interpolation to estimate values across different temporal granularity. This captures long-range dependencies without the high computational cost of models like Transformers. Each block focuses on various aspects of the time series, such as high- or low-frequency components, contributing to the final forecast through interpolation.

Overall, NHITS is computationally efficient and accurate, making it practical for forecasting short- and long-term trends. Its hierarchical structure effectively manages complex time series data, including volatility and abrupt changes, which is crucial for financial forecasting.

4.4.3 *Temporal Fusion Transformer (TFT)*

The Temporal Fusion Transformer (TFT), introduced by Lim et al. (2021), is a time series forecasting model that combines Transformer attention mechanisms with components for temporal data. TFT manages multivariate time series, modeling long-range dependencies and interactions between components, making it suitable for applications like financial market analysis.

TFT's architecture includes recurrent layers, multi-head attention blocks, and specialized temporal processing layers. Recurrent layers capture short-term dependencies, while attention mechanisms model long-term dependencies and interactions across variables. TFT uses gating mechanisms, like the Gated Residual Network (GRN), to skip irrelevant data, reducing complexity and improving efficiency.

The model's interpretable multi-head attention provides insights into relationships across time steps, which is valuable for financial forecasting where transparency is important. TFT also uses variable selection networks to dynamically select relevant features, improving accuracy by focusing on the most informative variables at each time step.

Overall, TFT's mix of attention, recurrent processing, and feature selection makes it well-suited for complex time series forecasting. Its ability to manage static and time-varying covariates and offer interpretable insights helps model non-linear interactions in financial time series, providing accurate forecasts for both short- and long-term horizons.

4.4.4 *Time-series Dense Encoder (TiDE)*

The Time-series Dense Encoder (TiDE), introduced by Das et al. (2024), is a simple and effective model for long-term time series forecasting. TiDE uses dense layers to encode past values and covariates, making it computationally lightweight compared to transformer-based models. It aims to balance complexity and accuracy, particularly for scenarios involving historical data and covariates.

TiDE's architecture has two main parts: encoding and decoding. In the encoding phase, TiDE uses residual blocks to project input covariates to a lower-dimensional space, followed by a dense encoder that combines past covariates, static features, and historical time series into a compact representation. This captures the underlying structure of the time series for future predictions.

In the decoding phase, TiDE uses dense and temporal decoding. The dense decoder maps the encoded representation to the future horizon, while the temporal decoder generates the final forecast, incorporating covariate information to improve accuracy. This "highway" connection allows TiDE to better capture the impact of covariates on future values.

TiDE also uses residual connections to retain linear prediction capability while benefiting from non-linear modeling. A global residual connection from historical data to the forecast allows TiDE to behave like a linear model if needed, enhancing robustness. This combination makes TiDE versatile for financial forecasting, balancing complexity and performance.

4.5 *Model hyper-parameters*

This study optimizes hyper-parameters for all benchmark models through 30 trials, using approximately 12.5% of the training data (around two years) as a validation set. Hyper-parameter optimization is conducted through the Ray Tune package (Liaw et al., 2018), which provides a scalable framework for managing optimization trials and implements the Tree-structured Parzen Estimator (TPE) a Bayesian algorithm (Bergstra et al., 2011; Dask Development Team, 2016) that efficiently explore promising regions of the hyper-parameter space. Subsection 4.5.1 hast the details for the search space, including parameters and ranges, while the optimal settings for all neural network models are provided in Appendix I (page 69).

The use of a Student-t loss function further strengthens the forecasting of stock volatility. Its robustness against outliers and ability to capture heavy tails in financial data make it ideal for this context. This aligns with financial research emphasizing the need for models that accurately estimate tail risks and market extremes (Jiaona Li, 2024; Tan et al., 2022; Yan et al., 2019). By leveraging the Student-t distribution, the models achieve improved accuracy and resilience when handling both typical and extreme market fluctuations.

4.5.1 *Search space*

The following describes the hyper-parameter search space for each benchmark model. The hyper-parameters are based on the implementations of Olivares et al. (2022) and were selected through 30 trials using the TPE algorithm.

Common hyper-parameters include `learning_rate`, which controls the step size during model optimization, ranging from 0.01 to 0.00001 [9]; `scaler_type`, which specifies the type of normalization applied to temporal inputs (e.g., standard scaler normalizes to zero mean and unit variance, while robust scaler uses median and inter-quartile range); `max_steps`, which sets the maximum number of training steps; `batch_size`, which determines the number of series per batch; and `windows_batch_size`, which defines the number of data windows sampled per batch.

The following hyperparameters were reduced from continuous ranges to discrete sets: `learning_rate`, `hidden_size`, and `dropout`. This reduction ensures that the optimization process focuses on values that are computationally feasible and commonly used in time series forecasting tasks, improving the efficiency of the search and avoiding unnecessary exploration of less practical ranges.

Table 1: Hyper-parameter Space for BiTCN

Hyper-parameters	**Options**
hidden_size	[16, 32]
dropout	Random value in [0.0, 0.99]
learning_rate	[0.01, 0.005, 0.001, 0.0005, 0.0001, 0.00005, 0.00001]
scaler_type	[None, robust, standard]
max_steps	[500, 1000, 2000]
batch_size	[32, 64, 128, 256]
windows_batch_size	[128, 256, 512, 1024]

[9] While defining the learning rate as continuous range for hyper-parameter search could allow the optimization algorithm to explore intermediate values more freely, this approach was constrained by resource limitations, including GPU memory and computational capacity. To avoid model crashes and excessive resource consumption, the search space was limited to predefined values to ensure a stable and efficient search process. Despite these constraints, the search was designed to identify configurations that generalize well to the independent dataset

For BiTCN, as shown in Table 1, the `hidden_size` parameter controls the number of units in the hidden state of TCN (16 or 32), and `dropout` specifies the dropout rate (randomly sampled between 0.0 and 0.99). BiTCN uses these parameters to improve efficiency and robustness while capturing temporal dependencies through its bidirectional encoding of past and future covariates. Other common parameters are included to support the model's convolutional architecture and training process.

Table 2: Hyper-parameter Space for NHTIS

Hyper-parameters	**Options**
n_pool_kernel_size	[[2, 2, 1], 3 * [1], 3 * [2], 3 * [4], [8, 4, 1], [16, 8, 1]]
n_freq_downsample	[[168, 24, 1], [24, 12, 1], [180, 60, 1], [60, 8, 1], [40, 20, 1], [1, 1, 1]]
learning_rate	[0.01, 0.005, 0.001, 0.0005, 0.0001, 0.00005, 0.00001]
scaler_type	[None, robust, standard]
max_steps	Sampled in multiples of 100 from 500 to 1500
batch_size	[32, 64, 128, 256]
windows_batch_size	[128, 256, 512, 1024]

For NHITS, Table 2 outlines key hyper-parameters like `n_pool_kernel_size`, which specifies pooling window sizes (e.g., [2, 2, 1] or [16, 8, 1]), and `n_freq_downsample`, which defines the downsampling rates for hierarchical decomposition (e.g., [168, 24, 1] or [60, 8, 1]). These parameters enable NHITS to focus on patterns across multiple temporal scales, leveraging its hierarchical structure for both short- and long-horizon forecasts. NHITS splits input series into segments of varying temporal granularity, enhancing its ability to manage complex dynamics.

Table 3: Hyper-parameter Space for TFT

Hyper-parameters	**Options**
hidden_size	[64, 128, 256]
n_head	[4, 8]
learning_rate	[0.01, 0.005, 0.001, 0.0005, 0.0001, 0.00005, 0.00001]
scaler_type	[None, robust, standard]
max_steps	[500, 1000, 2000]
batch_size	[32, 64, 128, 256]
windows_batch_size	[128, 256, 512, 1024]

For TFT, as shown in Table 3, the `hidden_size` parameter sets the size of embeddings and encoders (64, 128, or 256), while `n_head` determines the number of attention heads in the temporal fusion decoder (4 or 8). These hyper-parameters allow the model to capture long-range dependencies and multivariate interactions. TFT combines attention mechanisms, recurrent layers, and feature selection to handle static and time-varying covariates effectively, making it suitable for time series with nonlinear interactions.

Table 4: Hyper-parameter Space for TiDE

TiDE Hyper-parameters	**Options**
hidden_size	[256, 512, 1024]
decoder_output_dim	[8, 16, 32]
temporal_decoder_dim	[32, 64, 128]
num_encoder_layers	[1, 2, 3]
num_decoder_layers	[1, 2, 3]
temporal_width	[4, 8, 16]
dropout	[0.0, 0.1, 0.2, 0.3, 0.5]
layernorm	[True, False]
learning_rate	[0.01, 0.005, 0.001, 0.0005, 0.0001, 0.00005, 0.00001]
scaler_type	[None, robust, standard]
max_steps	Sampled in multiples of 100 from 500 to 1500
batch_size	[32, 64, 128, 256]
windows_batch_size	[128, 256, 512, 1024]

For TiDE, Table 4 specifies hyper-parameters like `hidden_size` for dense MLPs (256, 512, or 1024), `decoder_output_dim` for the decoder's output size (8, 16, or 32), and `temporal_decoder_dim` for the temporal decoder's hidden size (32, 64, or 128). The number of encoder and decoder layers (`num_encoder_layers` and `num_decoder_layers`) ranges from 1 to 3, and `temporal_width` controls the lower projected temporal dimension (4, 8, or 16). `dropout` ranges from 0.0 to 0.5, and `layernorm` indicates whether layer normalization is applied. These parameters balance simplicity and accuracy, enabling TiDE to process historical data and covariates efficiently.

Unlike Souto's research, which established a more general hyperparameter configuration based on their main experiment and applied it across subsequent experiments, this study conducts a distinct hyperparameter search for both Experiment 1 (including the exogenous variable setup) and Experiment 2. This approach allows each model to adapt to the specific characteristics of each experiment, providing a more realistic and fair comparison.

4.6 *Experimental setup*

The empirical findings in this study are further validated through a series of experiments designed to evaluate the stability and reliability of the forecasting models under different conditions. These experiments help to ensure that conclusions are not dependent on specific data configurations or modeling assumptions, but instead reflect the underlying patterns of financial time series (Souto & Moradi, 2024).

The first experiment involves modifying the data split for training and testing. While the main analysis uses an 80%/20% split, this experiment examines the impact of using this split ratio, a common variation in forecasting studies (Joseph, 2022). This test helps to determine whether the models perform consistently with a different allocation of training and testing data.

The second experiment involves reducing the training data to half of its original size. This scenario simulates situations where data availability is limited, such as newly listed stocks or markets with sparse historical data (Mizik, 2014). By training the models on a constrained dataset, this experiment evaluates their ability to make predictions when limited information is available.

In the third experiment, the trained models from the first experiment are used to predict the across horizon of the S&P500 and AEX index. This test not only assesses the model's transferability and performance on a different index but also helps to understand its generalization capabilities.

In general, these experiments validate the findings of the main analysis, providing the assurance that the model performance is consistent and not influenced by specific data or methodological choices. This validation is crucial in financial forecasting, where models are often used to support high-stakes decision-making (D'Ecclesia & Clementi, 2021; Li, 2022).

4.7 *Evaluation methodology*

4.7.1 *Forecast horizon and evaluation metric*

Building on research by Souto and Moradi (2024), this study evaluates forecasting models across four forecast horizons: H = 1, H = 5, H = 10, and H = 20. These correspond to one business day, one business week, two business weeks, and four business weeks ahead. Short-term horizons (H=1, H=5) are crucial for day traders and analysts needing quick reactions, while medium-term horizons (H=10, H=20) are more relevant for institutional investors and portfolio managers (Gobato Souto, 2023). Using multiple horizons allows for a comprehensive evaluation of predictive capabilities under different market conditions. Short-term forecasts capture immediate market reactions, while medium-term forecasts help assess trends over weeks.

The models are evaluated using the Quasi-Likelihood Error (QLIKE), a metric commonly used to assess volatility forecasts. QLIKE is suitable for financial time series because it accounts for the heavy tails and changing variance typical of financial returns. Measures how well a model predicts the variance or realized volatility, with an emphasis on penalizing large prediction errors.

$$\text{QLIKE} = \frac{1}{s \cdot n} \sum_{j=1}^{s} \sum_{i=1}^{n} \left(\frac{y_{ij}}{\hat{y}_{ij}} - \log\left(\frac{y_{ij}}{\hat{y}_{ij}}\right) - 1 \right) \tag{11}$$

Where y_{ij} represents the actual observation, $\hat{y}_{ij}$ is the corresponding prediction, n is the total number of observations and s is the number of stocks considered.

The use of QLIKE allows for a focused evaluation of each model's ability to predict realized variance or volatility, providing a robust measure of forecasting performance, especially in the presence of high volatility.

4.7.2 *Statistical tests*

First proposed by Diebold and Mariano (1995) and later refined by Harvey et al. (1997), the Diebold-Mariano (DM) test is used to evaluate the performance of the models for each stock separately. Unlike Souto and Moradi (2024) who evaluated the best performing Transformer model for each forecast horizon, this research applies the DM test across all models to determine how often one model outperforms another. The DM test is conducted for the QLIKE error metric to verify the accuracy and reliability of the forecasting models. This approach provides a detailed assessment of the models' capability in forecasting the volatility of individual stocks, helping to understand their strengths and limitations in predicting financial time series (Chiriac & Voev, 2011; Diebold & Mariano, 1995; Dutta & Das, 2022; Gallo & Otranto, 2015; Gobato Souto, 2023; Zhang et al., 2024).

5 RESULTS

5.1 *First experiment*

In this section, the forecasting performance of various models on the main analysis test set is evaluated using the QLIKE metric and the Diebold-Mariano (DM) Test. Table 5 summarizes the aggregated QLIKE values for horizons of 1, 5, 10, and 20 days, while table 6 presents the results of the DM Test counts reflect the frequency with which a given model's forecasts are statistically superior to those of its competitors. A higher count indicates a more consistent performance in individual company forecasts.

Table 5: QLIKE metric values (in %) for various models across forecast horizons (h = 1, 5, 10, 20) from Experiment 1 test set. Lower QLIKE values indicate better performance, and the three best-performing models for each horizon are underlined.

model	**h = 1**	**h = 5**	**h = 10**	**h = 20**
BiTCN	0.175	1.127	2.029	5.841
BiTCN1	0.176	1.112	2.304	6.306
NHITS	0.175	1.113	2.115	5.879
NHITS1	0.175	1.085	2.026	5.645
TFT	0.177	1.132	2.095	6.054
TFT1	0.175	1.115	2.269	5.836
TiDE	0.176	1.117	2.093	6.015
TiDE1	0.176	1.096	2.114	5.926

Table 5 shows QLIKE values are generally similar across all horizons, with no clear dominant model. However, NHITS1 consistently achieves the lowest QLIKE values, demonstrating its robustness in forecasting volatility. QLIKE values increase with longer horizons, indicating a decline in forecast accuracy over time. For instance, all models show a significant rise in QLIKE values from horizon 1 to horizon 20, underscoring the difficulty of long-term forecasting.

Table 6: DM test counts for different models across forecast horizons (h = 1, 5, 10, 20) from Experiment 1. Higher counts indicate better performance. The top three models per horizon are highlighted in green with double underlines, and the bottom three are highlighted in red.

model	**h = 1**	**h = 5**	**h = 10**	**h = 20**
BiTCN	53	869	190	462
BiTCN1	151	566	299	1598
NHITS	146	540	929	514
NHITS1	10	103	111	189
TFT	177	1460	753	1091
TFT1	26	353	1416	424
TiDE	337	638	740	970
TiDE1	248	32	736	562

Table 6 summarizes the DM test results, highlighting each model's relative predictive power. At shorter horizons (h = 1 and h = 5), TiDE and TFT dominate, with TiDE achieving the highest count at h = 1, showcasing its strength in forecasting short-term volatility. At longer horizons (h = 10 and h = 20), TFT and BiTCN1 become more competitive. Despite NHITS1's strong QLIKE performance, its lower DM test counts suggest consistent but less statistically significant outperformance compared to peers.

5.2 *Second experiment*

This section evaluates model performance with reduced training data, simulating limited data scenarios like newly listed stocks or sparse historical data (Mizik, 2014). Table 7 presents forecasting accuracy across 1, 5, 10, and 20-day horizons using the QLIKE metric. Table 8 summarizes DM Test results, showing how often each model statistically outperforms others in forecasting individual companies under these conditions.

Table 7: QLIKE metric values (in %) for various models across forecast horizons (h = 1, 5, 10, 20) from Experiment 2 test set. Lower QLIKE values indicate better performance, and the three best-performing models for each horizon are underlined.

model	**h = 1**	**h = 5**	**h = 10**	**h = 20**
BiTCN	0.180	1.119	2.261	5.988
BiTCN1	0.183	<u>1.106</u>	2.319	6.285
NHITS	0.176	<u>1.111</u>	2.146	<u>5.851</u>
NHITS1	<u>0.174</u>	1.164	<u>2.026</u>	<u>5.830</u>
TFT	0.175	1.137	2.264	6.305
TFT1	<u>0.174</u>	1.261	<u>2.005</u>	6.207
TiDE	0.177	<u>1.112</u>	2.113	5.855
TiDE1	<u>0.175</u>	1.302	<u>2.061</u>	<u>5.700</u>

The Metric Value section for experiment 2 table 7 shows slightly higher QLIKE values than the Original table, with the same trend of increasing QLIKE values from h1 to h20. NHITS models still perform best overall, with NHITS1 achieving the lowest values at most horizons, including h5, where it outperforms NHITS—an improvement over the Original results.

Table 8: DM test counts for different models across forecast horizons (h = 1, 5, 10, 20) from Experiment 2. Higher counts indicate better performance. The top three models per horizon are highlighted in green with double underlines, and the bottom three are highlighted in red.

model	**h = 1**	**h = 5**	**h = 10**	**h = 20**
BiTCN	1209	413	2009	732
BiTCN1	318	126	1253	834
NHITS	183	223	927	480
NHITS1	7	316	189	407
TFT	123	631	764	1729
TFT1	52	1252	96	1295
TiDE	746	270	706	475
TiDE1	45	93	317	235

Table 8 presents the DM Test results for the reduced dataset, comparing model performance across 1, 5, 10, and 20-day horizons. BiTCN performs strongly at h1, h10, and h20, with a top-3 ranking at h5. BiTCN1 shows moderate results, excelling at h10 and h20 but lagging at shorter horizons. NHITS models deliver consistent but weaker results, with NHITS1 slightly improving at h5 but underperforming elsewhere. TFT and TFT1 perform well at h5 and h20, though TFT1 drops sharply at h10. TiDE and TiDE1 perform worse than in the original dataset, with TiDE showing some strength at h1 but falling behind at other horizons.

5.3 *Third experiment*

This section evaluates model forecasting performance in two scenarios to assess generalization. The first scenario uses the S&P500 dataset, where models predict horizons immediately after the test set, testing their ability to generalize beyond training. The second scenario evaluates models on the smaller AEX dataset, covering 18 companies from 2016–2023, with predictions starting from 2023-12-31, similar to the S&P500 setup. The smaller AEX dataset results in lower DM Test values due to fewer observations but provides insights into performance in a distinct, limited-data market. Horizon 1 is excluded due to insufficient observations, and in Experiment 3, horizons strictly represent the number of days specified.

5.3.1 *S&P500 Forecasting.*

Table 9: QLIKE metric values (in %) for various models across forecasting horizons (h = 5, 10, 20) on the S&P500 in Experiment 3. Lower QLIKE values indicate better performance, and the three best-performing models for each horizon are underlined.

model	**h = 5**	**h = 10**	**h = 20**
BiTCN	0.483	0.736	1.471
BiTCN1	0.492	0.875	1.528
NHITS	0.488	0.732	1.372
NHITS1	0.447	0.648	1.770
TFT	0.461	0.733	1.703
TFT1	0.550	0.878	1.780
TiDE	0.488	0.743	1.402
TiDE1	0.398	1.007	1.416

Looking at table 9, it is evident that the models show considerable variability in performance. At the 5-day horizon, the TiDE1 model records the lowest QLIKE value (0.398%), while TFT1 has the highest QLIKE (0.550%). As the forecast horizon increases to 10 days, NHITS1 demonstrates the best performance (0.648%), whereas TiDE1's value rises significantly to 1.007%. For the 20-day horizon, NHITS stands out with the lowest QLIKE (1.372%), whereas TFT1 registers the highest QLIKE (1.780%).

Table 10: DM test counts for different models across forecast horizons (h = 5, 10, 20) on the S&P500 in Experiment 3. Higher counts indicate better performance. The top three models per horizon are highlighted in green with double underlines, and the bottom three are highlighted in red

model	**h = 5**	**h = 10**	**h = 20**
BiTCN	352	409	655
BiTCN1	335	860	648
NHITS	402	437	574
NHITS1	286	433	896
TFT	267	499	971
TFT1	493	856	893
TiDE	361	510	581
TiDE1	333	1056	696

In table 10 is evident that the performance values vary substantially across the different forecast horizons and models. For instance, TiDE1 performs the best at a 10-day horizon with 1,056 counts, while TFT scores the highest for the 20-day horizon with 971 counts. NHITS1 has relatively high performance at the 20-day horizon, with a count of 896, while TFT1 also consistently shows strong performance across horizons 5, 10, and 20 days. Conversely, the models like NHITS and BiTCN have more moderate counts compared to others for most horizons.

5.3.2 *AEX Forecasting.*

Table 11: QLIKE metric values (in %) for various models across forecasting horizons (h = 5, 10, 20) on the AEX in Experiment 3. Lower QLIKE values indicate better performance, and the three best-performing models for each horizon are underlined

model	**h = 5**	**h = 10**	**h = 20**
BiTCN	0.618	0.515	1.447
BiTCN1	0.565	<u>0.461</u>	<u>1.268</u>
NHITS	0.573	0.585	<u>1.274</u>
NHITS1	<u>0.537</u>	<u>0.489</u>	1.283
TFT	<u>0.547</u>	0.773	1.593
TFT1	0.584	0.650	1.764
TiDE	0.593	<u>0.508</u>	1.309
TiDE1	<u>0.557</u>	0.719	<u>1.023</u>

Table 11 presents the QLIKE metric values for the AEX dataset, For the 5-day horizon, NHITS1 achieves the lowest error value, followed closely by TFT and TiDE1. For the 10-day horizon, BiTCN1 performs best, followed by NHITS1 and TiDE, which also have relatively lower values. TiDE1 exhibits the best performance for the 20-day horizon, indicating lower forecast errors compared to the other models.

Table 12: DM test counts for different models across forecast horizons (h = 5, 10, 20) on the AEX in Experiment 3. Higher counts indicate better performance. The top three models per horizon are highlighted in green with double underlines, and the bottom three are highlighted in red

model	**h = 5**	**h = 10**	**h = 20**
BiTCN	23	22	34
BiTCN1	16	13	24
NHITS	16	23	31
NHITS1	17	21	46
TFT	14	31	50
TFT1	33	35	66
TiDE	32	13	20
TiDE1	17	30	28

Table 12 consistently shows TFT1's strong performance, particularly at the 10-day and 20-day horizons, with the highest values of 35 and 66 respectively. TiDE also performs well for the 5-day horizon, with a value of 32, indicating robustness in short-term forecasting. In contrast, models like BiTCN1 generally exhibit lower counts across all horizons, suggesting a lower consistency in outperforming other models.

5.4 *Experiments summary*

Across the three experiments, QLIKE values consistently increase with longer horizons, indicating the difficulty of long-term volatility forecasting. NHITS1 excels in achieving low QLIKE values, even with reduced training data, suggesting effective capture of volatility dynamics. However, NHITS1's modest DM Test counts show that its superior average accuracy does not always translate into consistent outperformance across individual company forecasts. In contrast, TFT1 consistently achieves high DM Test counts, particularly in generalization scenarios involving both S&P500 and AEX datasets, highlighting its adaptability and reliable outperformance in diverse market conditions.

Overall, the error analysis shows that while some models like NHITS1 excel in forecast accuracy (QLIKE), they may lack statistical robustness across individual forecasts. TFT1 emerges as a reliable model, balancing low forecast errors with high DM Test counts, making it a strong candidate for accurate and statistically robust forecasting.

5.5 *Understanding TFT*

In Experiment 1, TFT achieved the highest DM test results across all horizons, whereas TFT1 demonstrated a notable impact only at horizon 10. In Experiment 3, TFT1 consistently outperformed across all horizons, while TFT excelled specifically at horizon 20 for the S&P500 and at horizons 10 and 20 for the AEX. To further investigate, the TFT model at horizon 10 was tested using only technical indicators (TechTFT) and only firm-level indicators (FirmTFT). Table 13 presents unchanged QLIKE values for TFT and TFT1, alongside the newly added QLIKE results for FirmTFT and TechTFT. Table 14 reexamines the DM test results, comparing the four versions of TFT to evaluate their statistical significance against one another. The corresponding comparisons for the BiTCN, NHITS, and TiDE models are provided in Appendix E (page 59).

Table 13: QLIKE metric values (in %) for TFT's configurations across forecasting horizons (h = 1, 5, 10, 20) in Experiment 1. Lower QLIKE values indicate better performance, and the two best-performing models for each horizon are underlined

model	**h = 1**	**h = 5**	**h = 10**	**h = 20**
TFT	0.177	1.132	<u>2.095</u>	6.054
FirmTFT	0.178	<u>1.094</u>	<u>2.169</u>	<u>5.983</u>
TechTFT	<u>0.175</u>	<u>1.084</u>	2.368	6.080
TFT1	<u>0.175</u>	1.115	2.269	<u>5.836</u>

The QLIKE metric values for the four versions of TFT—TFT, FirmTFT, TechTFT, and TFT1—are presented across four forecast horizons: 1, 5, 10, and 20 days. In terms of QLIKE, TechTFT has the lowest error for horizon 1 (0.175%), while FirmTFT outperforms in horizon 5 (1.094%). TFT1 achieves the lowest error for the 20-day horizon with a value of 5.836%. The differences between these models suggest that using specific types of indicators impacts the model performance at different time horizons.

Table 14: DM test counts for TFT's configurations across forecast horizons (h = 1, 5, 10, 20) in Experiment 1. Higher counts indicate better performance. The top two models per horizon are highlighted in green with double underlines, and the bottom two are highlighted in red

model	**h = 1**	**h = 5**	**h = 10**	**h = 20**
TFT	<u>139</u>	<u>890</u>	140	<u>403</u>
FirmTFT	<u>97</u>	186	241	279
TechTFT	58	29	<u>335</u>	<u>402</u>
TFT1	5	<u>226</u>	<u>415</u>	102

TFT achieves the highest count for horizon 5 (890), indicating it significantly outperformed the other models more frequently at this horizon. TechTFT has the highest count for horizon 10 (335), while TFT1 outperforms for horizon 10 as well, but with the highest count (415) among all horizons. The counts suggest that the different versions of TFT excel at different time-frames, depending on the type of indicators used.

5.5.1 *TFT H=10 models' QLIKE for Individual companies across the test set*

This section shows the QLIKE behavior of the TechTFT and TFT1 models, which were chosen based on their top DM scores in Table 14. Instead of aggregating QLIKE values into a single loss metric, this section examines these models across the test set for all companies. This helps identify extreme loss instances that may be missed with aggregated metrics. Appendix F (page62) provides the same analysis for the TFT0 and FirmTFT models.

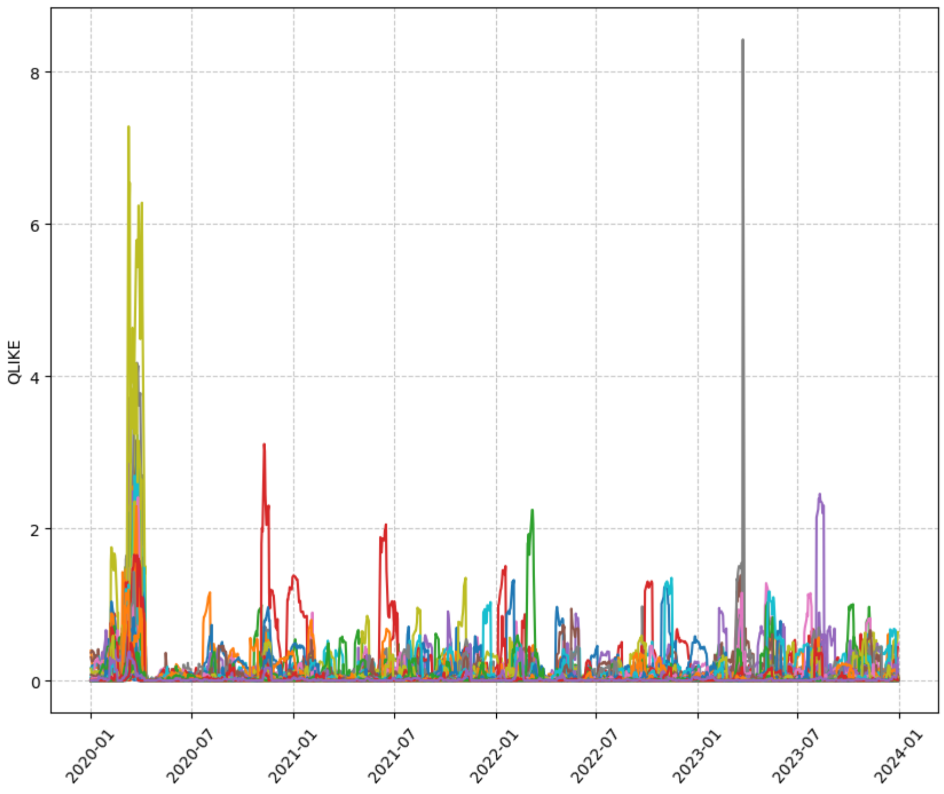

Figure 2: TechTFT's QLIKE over the test set in experiment 1 per company

Figure 2 represents TechTFT QLIKE loss over time. It also exhibits some notable spikes, although fewer than those observed in FirmTFT, suggesting that while technical indicators generally perform well, there are still challenges in predicting certain time periods or companies.

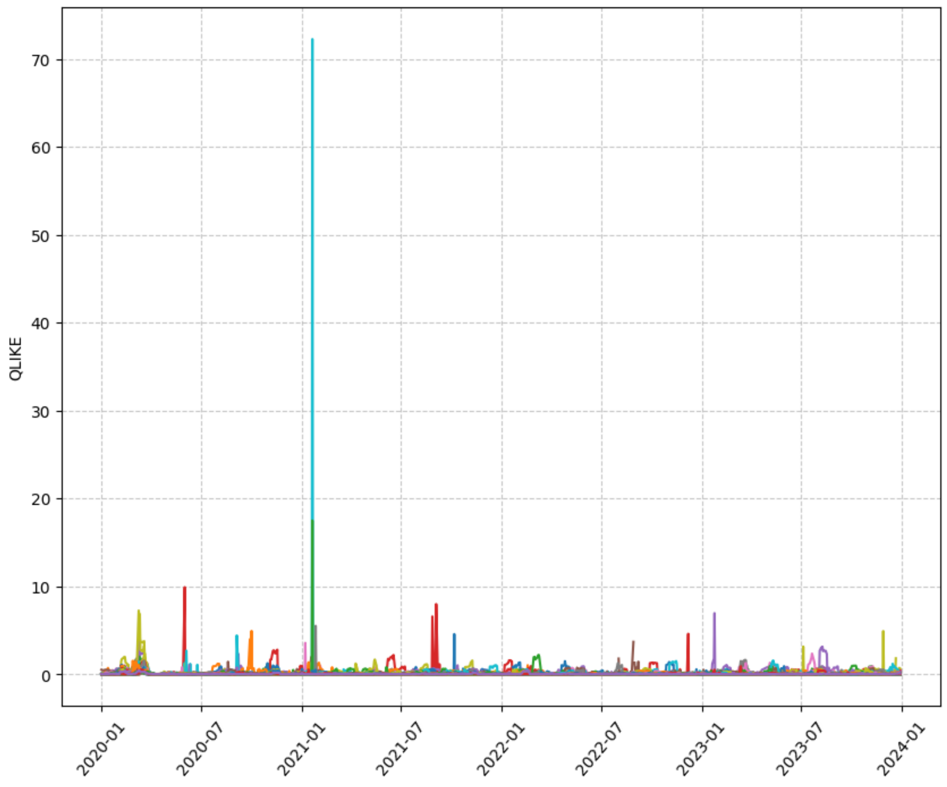

Figure 3: TFT1's QLIKE over the test set in experiment 1 per company

In figure 3, TFT1 shows QLIKE loss over time with one major outlier spike in early 2021. Overall, TFT1 has fewer high QLIKE values compared to FirmTFT, indicating a more consistent performance across the test period.

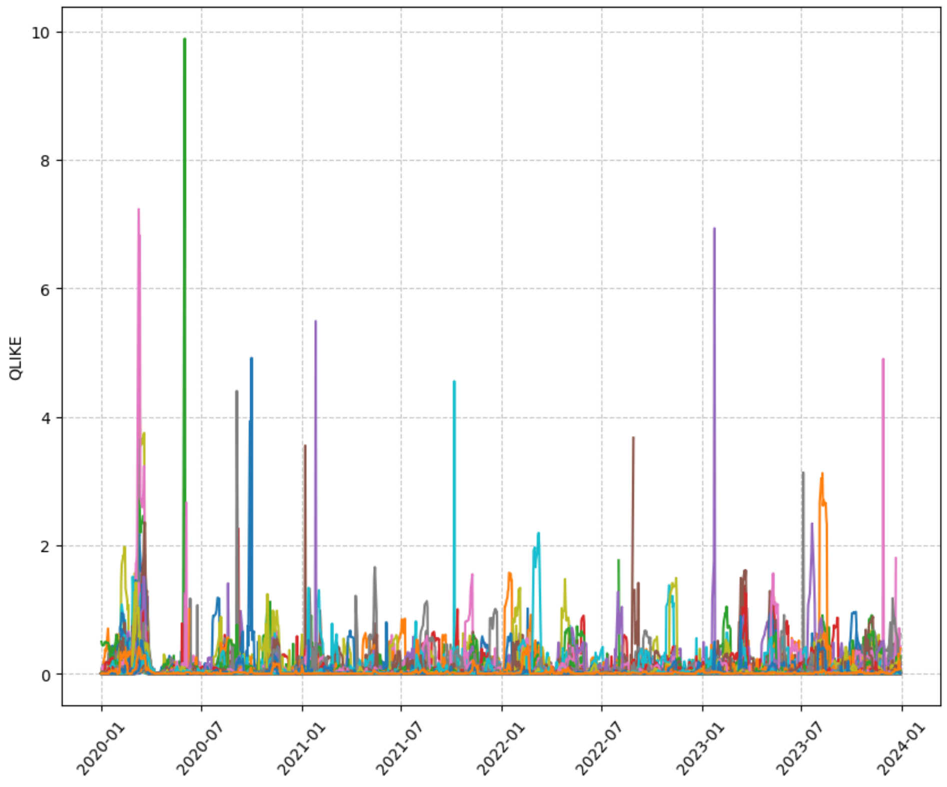

Figure 4: TFT1's QLIKE filtered over the test set in experiment 1 per company

The last figure 4 shows the filtered version of TFT1, presenting a smoother performance across companies by reducing the impact of large outliers. This makes the overall pattern more consistent and highlights the general effectiveness of the model.

While some models exhibit significant spikes in QLIKE loss, suggesting challenges with prediction accuracy during volatile periods, filtering outliers has been effective in highlighting more consistent trends. The filtered versions of FirmTFT and TFT1, in particular, suggest that these models have potential for stable performance, with reduced variability in error across different companies and time periods.

5.5.2 *Companies with Lowest Average QLIKE and Outliers of TFT1 H=10*

The two companies with the lowest average QLIKE are visualized in Figure 5, while the companies responsible for the outliers in Figure 3 are highlighted in Figure 6. The corresponding values are provided in Appendix G (page 65).

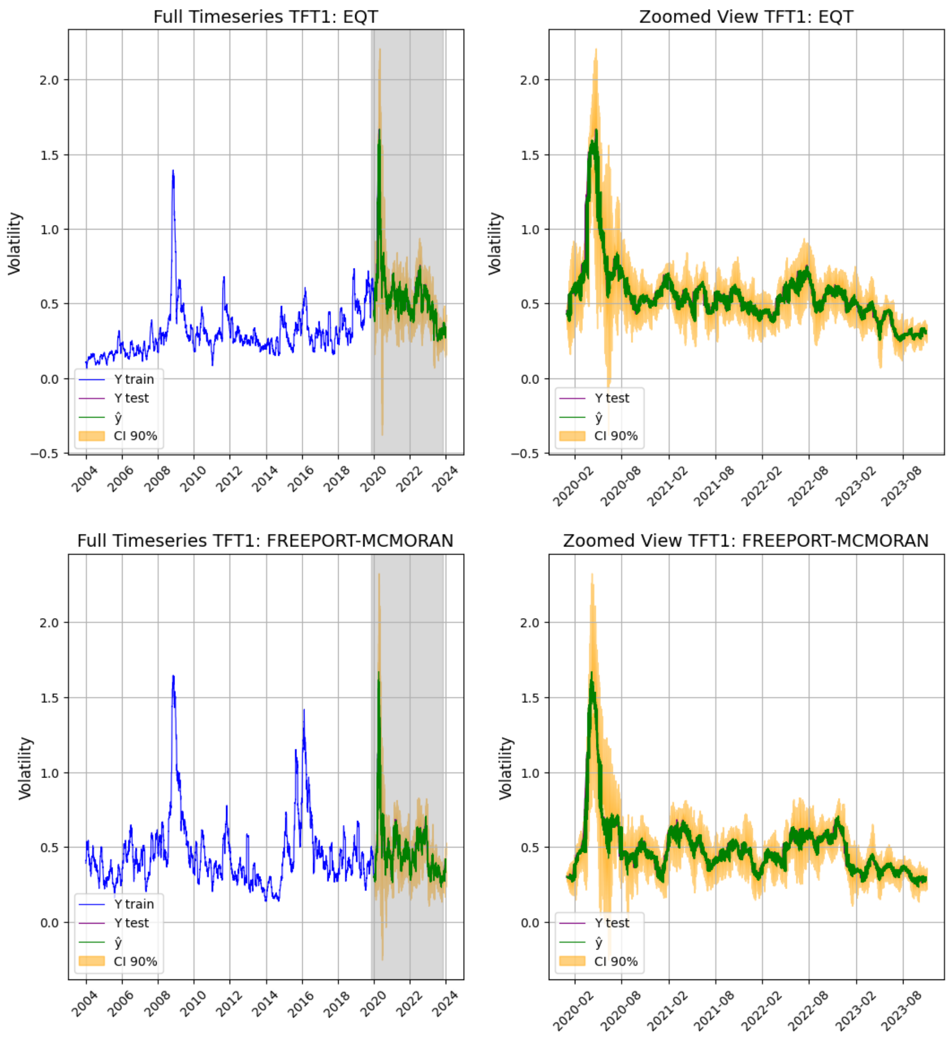

Figure 5: Visualization of the two companies with the lowest average QLIKE values for TFT1 H=10 in Experiment 1. The left panels show the full time series with training data (blue) and the shaded test area. The right panels zoom in on the test period, showing actual test values (y, purple), test predictions ($\hat{y}$, green), and 90% confidence intervals (yellow). The y values are present but not clearly visible as they closely overlap with the actual test values, blending into the plot.

The first set of figures compares actual and predicted volatility for companies with the lowest average QLIKE. The left panel shows the full timeseries from 2004 to 2024, while the right panel zooms in on 2020-2024. For EQT, volatility increased until 2020, then stabilized. Predicted values align well with test data, though there are deviations during high volatility. FREEPORT-MCMORAN also shows good alignment, but uncertainty increases during volatile periods.

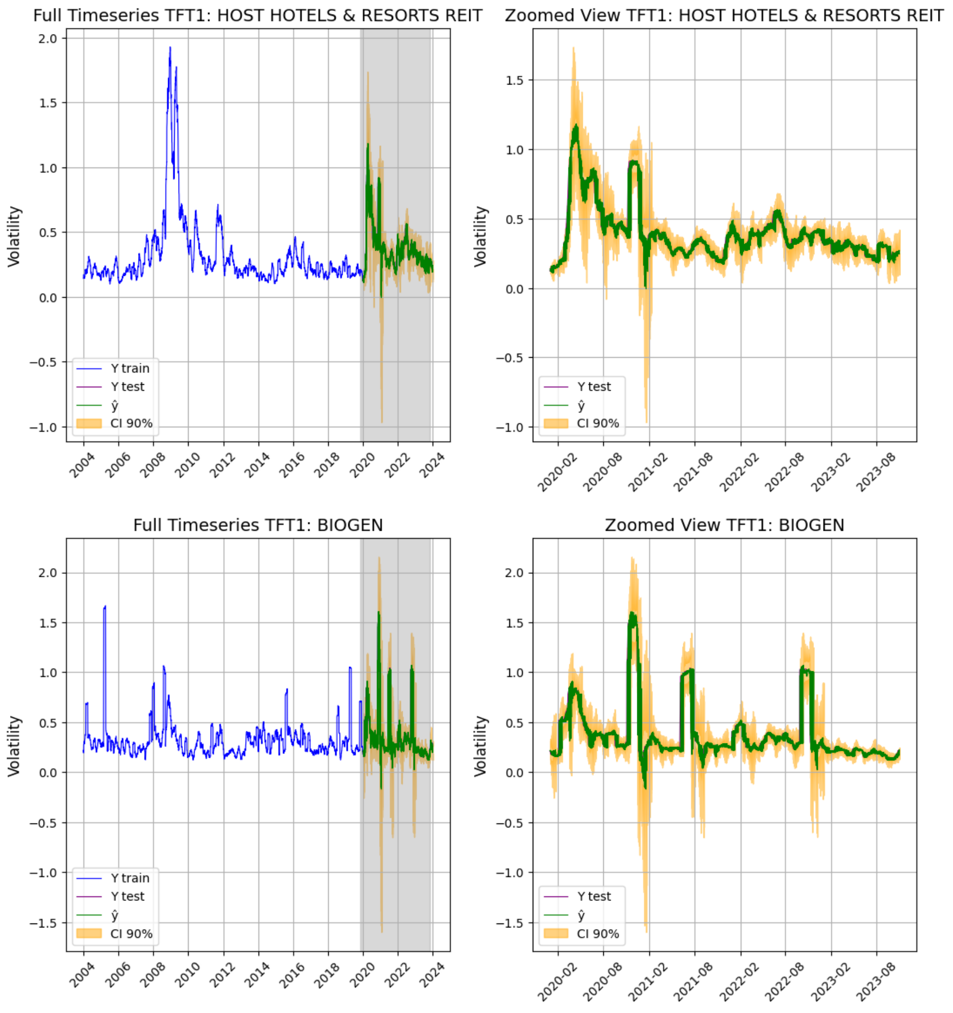

Figure 6: Visualization for two of the outliers companies QLIKE values on TFT1 H=10 in Experiment 1. The left panels show the full time series with training data (blue) and the shaded test area. The right panels zoom in on the test period, showing actual test values (y, purple), test predictions ($\hat{y}$, green), and 90% confidence intervals (yellow). The y values are present but not clearly visible as they closely overlap with the actual test values, blending into the plot.

HOST HOTELS & RESORTS REIT shows a volatility peak before 2020, which then stabilizes. Predictions match the test data but show wider confidence intervals during volatility spikes. Both HOST HOTELS & RESORTS REIT and BIOTECH had significant QLIKE outliers in a previous analysis. Here, outliers, particularly around February 2021, are visible through the widened confidence intervals, indicating challenges in model accuracy during these periods. BIOTECH displays higher volatility with spikes that the model mostly captures, but with gaps during sudden shifts.

5.5.3 *Averaged TFT1 across horizons*

Table 13 shows that TFT1 had among the best QLIKE performance on horizons 1 and 20, figures 7 and 8 will showcase it respectively. Horizon 5 and 10 are available on Appendix H (page 67).

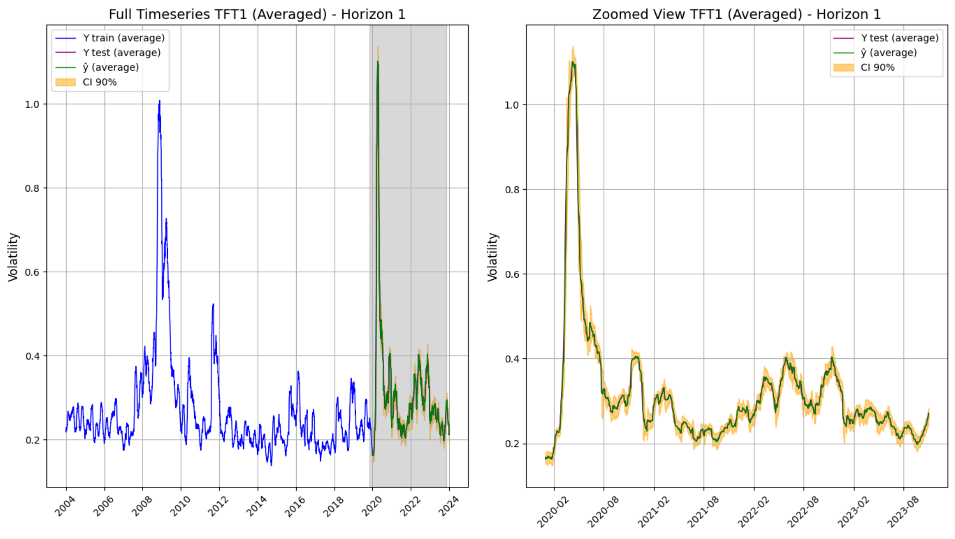

Figure 7: Visualization of the average data across companies for TFT1 H=1 in Experiment 1. The left panels show the full time series with training data (blue) and the shaded test area. The right panels zoom in on the test period, showing actual test values (y, purple), test predictions ($\hat{y}$, green), and 90% confidence intervals (yellow). The y values are present but not clearly visible as they closely overlap with the actual test values, blending into the plot.

Figure 7 shows the TFT1 model's predicted average volatility (green) closely tracking actual values during the test period, particularly during high-volatility events like COVID-19. The 90% confidence interval (orange) remains narrow for most of the test phase, widening slightly during volatile periods. The right plot focuses on 2020-2023, highlighting the model's accuracy, with forecasted values aligning well with actual data and minimal uncertainty in the confidence interval. Overall, TFT1 reliably captures both long-term trends and short-term volatility shifts with high predictive confidence.

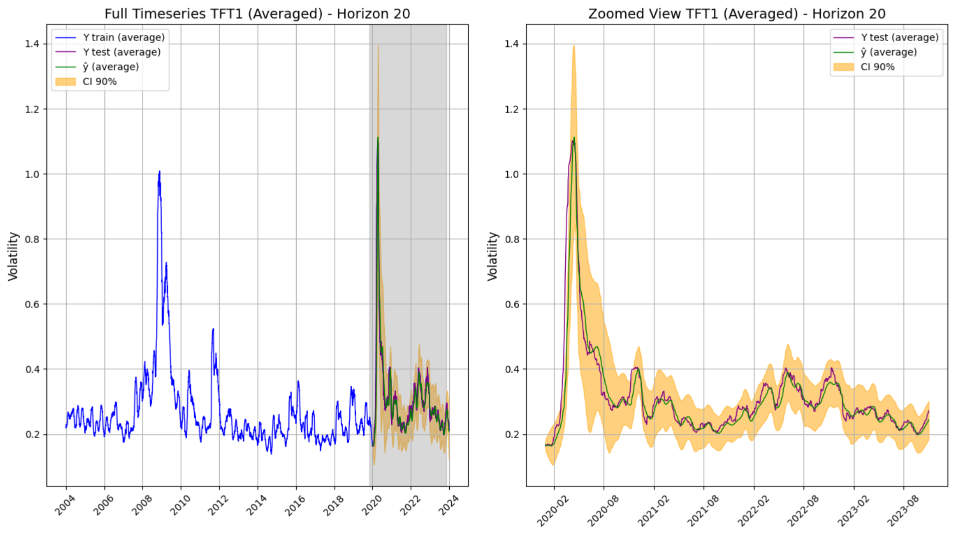

Figure 8: Visualization of the average data across companies for TFT1 H=20 in Experiment 1. The left panels show the full time series with training data (blue) and the shaded test area. The right panels zoom in on the test period, showing actual test values (y, purple), test predictions ($\hat{y}$, green), and 90% confidence intervals (yellow).

Figure 8 shows the 90% confidence interval (CI) widening significantly during the test period, indicating greater uncertainty in long-term forecasts. The right plot focuses on 2020-2023, showing the model's predictions closely tracking actual values, though with wider CIs compared to shorter horizons. This reflects the challenge of accurately forecasting long-term volatility, especially in turbulent markets. The model captures general volatility patterns, but the broader CI highlights increased uncertainty over extended horizons.

It is remarkable that in Figures 5, 6, and 7, the predicted values ($\hat{y}$) closely follow the actual values (y), even during periods of high volatility, such as in 2020. Although Figure8 shows a less precise alignment, it remains evident that the trends in y are replicated by $\hat{y}$, albeit with a slight lag, which highlights the confidence interval's coverage at a longer predictive horizon.

6 DISCUSSION

The aim of this thesis was to assess the predictions capabilities of BiTCN, NHITS, TFT and TiDE over the stock volatility. This research found that the selected model might have a similar aggregated QLIKE value but different performance thanks to DM test implemented.

6.1 *Results discussion Model*

In Experiment 1, NHITS1 had the lowest QLIKE values across all horizons, indicating strong average performance. However, NHITS1's Diebold-Mariano (DM) test results showed inconsistent statistical superiority across individual stocks, unlike TFT, which consistently ranked among the top 3 across all horizons. Suggesting that NHITS1's lower errors are not always consistent across stocks. Gobato Souto (2023) similarly found that NHITS, while promising, did not consistently outperform other models. Our findings confirm NHITS1's effectiveness but also its inconsistency, especially at longer horizons.

In Experiment 2, BiTCN models showed significant improvement under constrained data, excelling at horizons 1, 10, and 20. According to Gobato Souto (2023), NHITS was a robust model for realized volatility at horizon 5, but another model outperformed NHITS as the preferred choice, NHITS here maintained strong QLIKE values but lacked statistical dominance, aside from slight improvement at horizon 5. TFT and TFT1 performed well at horizons 5 and 20, while BiTCN outperformed NHITS across several horizons, highlighting its adaptability with limited data. Unlike Gobato Souto (2023) findings where another model led, BiTCN models took the lead in this data-constrained setting, suggesting they are better suited for newly listed stocks or sparse data.

Experiment 3 aims to forecast beyond the test set of the first experiment and evaluate performance on an unseen index (AEX) to assess performance beyond the test set dates and data. Results show that model performance varies by forecast horizon and dataset. In the S&P500 dataset, TiDE1 performs best for short-term forecasts, while NHITS and NHITS1 excel at longer horizons. TFT1 has higher QLIKE values but shows robustness across all horizons in DM Test results. TiDE1 and TFT are particularly effective at 10-day and 20-day horizons, respectively. In the AEX dataset, TFT1 consistently performs well across all horizons, indicating good generalization. TiDE1 excels at long-term horizons, while NHITS and NHITS1 are stable for shorter horizons. BiTCN models underperform, struggling with the smaller dataset.

The results from the Understanding TFT section provide a nuanced view of the strengths of different TFT models across varying forecast horizons. The QLIKE results indicate that TechTFT performs best for very short-term forecasts (1 day), likely because technical indicators are more suited for immediate movements. FirmTFT shows strength at a slightly longer horizon (5 days), suggesting that firm-specific information contributes effectively to short-term but not immediate forecasts. For longer horizons, TFT1 consistently delivers lower errors, indicating its robustness for predicting long-term volatility trends. These findings align with the conclusions of Souropanis and Vivian (2023), which highlight the benefits of combining technical indicators and macroeconomic variables due to their differing time-frequency impacts on forecasting.

The DM test results further illustrate the specialization of each TFT version. TFT has the highest frequency of outperforming the other models at the 5-day horizon, while TechTFT and TFT1 excel at medium-term forecasts, particularly at the 10-day horizon. This suggests that leveraging a combination of models may be beneficial depending on the specific forecast horizon required. Moreover, visualizations of individual company QLIKE losses reveal that the models perform well during stable periods, but all versions experience challenges during high-volatility spikes. Notably, filtering outliers helps to demonstrate the more consistent aspects of FirmTFT and TFT1, indicating potential areas for improving model stability, particularly under volatile market conditions.

6.2 *Limitations*

The main limitation of this research is the use of Datastream International (2024) 840E measure instead of 5-minute realized volatility (RV), as suggested by L. Y. Liu et al. (2015). High-frequency measures like 5-minute RV better capture the true quadratic variation of financial assets, enabling more accurate estimation of short-term volatility patterns. By relying on daily data, this study may miss intraday price nuances, reducing the precision and responsiveness of volatility estimates to rapid market changes.

Some hyperparameters, such as the learning rate, were restricted to predefined sets instead of continuous ranges to reduce experiment runtime and prevent model crashes caused by memory limitations. This approach ensured stable execution within the available computational resources, particularly for models with high GPU memory requirements. While predefined values helped control the search complexity, continuous ranges might have allowed the optimization algorithm to explore intermediate values and potentially identify better-performing configurations.

Another limitation is the exclusion of macroeconomic indicators, as highlighted by Souropanis and Vivian (2023). Firm-specific and technical indicators focus on short-term trends but fail to account for broader economic factors like inflation, interest rates, and liquidity. This omission may overlook key drivers of lower-frequency volatility components, potentially limiting the robustness and effectiveness of the models.

6.3 *Relevance*

This study extends current volatility forecasting literature by incorporating exogenous variables often overlooked in traditional models. Gobato Souto (2023) and Souto and Moradi (2024) showed the potential of recent models in predicting realized volatility (RV), but this study adds firm-specific and macroeconomic variables for a broader view of market dynamics. Souropanis and Vivian (2023) highlighted the complementary role of these indicators, which can enhance forecast robustness, an aspect not fully explored by Olivares et al. (2022). While NHITS1 had great average QLIKE value performance across the dataset, the Temporal Fusion Transformer (TFT) consistently outperformed it in Diebold-Mariano (DM) tests and showed strong out-of-sample performance, highlighting its effectiveness in capturing market patterns and adaptability across conditions. The high DM test significance underlines TFT's predictive strength, making it a promising tool for practical use.

This research uses a larger dataset, covering 385 S&P 500 companies over 20 years compared to 80 companies over 12 years in Gobato Souto (2023) and Souto and Moradi (2024), allowing a more thorough evaluation of model performance under different conditions. By combining technical indicators and macroeconomic variables, the study provides insights into the consistency and robustness of machine learning approaches for predicting stock volatility. Applying these models to the Amsterdam Exchange Index (AEX) offers a cross-market perspective, demonstrating how well-advanced ML models generalize across financial environments. The cross-market evaluation highlights the versatility of models like TFT and TiDE1, making them valuable for local and international investors.

6.4 *Future Work*

Apart from addressing the aforementioned limitations, a detailed analysis of the contribution of each exogenous variable to the prediction of volatility could provide significant insights. While the Diebold-Mariano (DM) test indicates how many companies a model outperformed its peers, it sometimes masks instances where the QLIKE value for certain companies is exceptionally high. This limitation could be mitigated by employing Quaedvlieg (2021) Multi-Horizon comparison, which offers a more comprehensive assessment across horizons, thereby providing a robust metric for model performance across different forecast setups.

For future research, it's important to note that TFT, despite being older and more resource-intensive, remains relevant due to its ability to handle exogenous variables. However, newer transformer models lack this capability. On the other hand, models like BiTCN, NHITS, and TiDE are lighter, handle exogenous variables, and achieve comparable results, making them worthy of deeper exploration.

7 CONCLUSION

In this study, we investigated the forecasting capabilities of four machine learning models: Bidirectional Temporal Convolutional Network (BiTCN), Neural Hierarchical Interpolation for Time Series (NHITS), Time-series Dense Encoder (TiDE), and Temporal Fusion Transformer (TFT) in predicting the volatility of constituent stocks from the S&P 500 over a 20-year period. This analysis was performed across three experiments, each exploring different data conditions to answer our research questions.

7.1 *RQ1: How do Bidirectional Temporal Convolutional Network (BiTCN), Neural Hierarchical Interpolation for Time Series (NHITS), Time-series Dense Encoder (TiDE), and Temporal Fusion Transformer (TFT) algorithms perform in predicting the volatility of constituent stocks from the S&P 500 over a 20-year span?*

Overall, all models performed well, with NHITS1 achieving the lowest QLIKE values in Experiment 1, suggesting high average accuracy. However, NHITS1 exhibited inconsistent statistical dominance across individual stocks, unlike TFT, which consistently ranked among the top three models across different horizons. TFT's consistent performance was evidenced by strong results in both QLIKE and the Diebold-Mariano (DM) test, making it a reliable choice for volatility forecasting over various horizons.

7.2 *RQ2: Can these models retain their prediction capabilities when their training period is halved?*

When the training data was reduced by half, all models exhibited slightly higher QLIKE values compared to the original results, indicating a modest decline in performance. BiTCN models showed significant improvement under constrained data, particularly at horizons 1, 10, and 20. NHITS models maintained strong QLIKE values, but their statistical dominance was limited. TFT and TFT1 adapted well to reduced data conditions, especially at horizons 5 and 20, highlighting their adaptability in settings with limited data availability, such as newly listed stocks.

7.3 *RQ3: How well do the trained models perform when forecasting beyond the test of the S&P500 as well as applied to the constituent stocks of the Amsterdam Exchange Index (AEX)?*

The models showed considerable variability in performance depending on the forecast horizon and dataset. In the S&P500 dataset, TiDE1 was the best performer for short-term forecasts, while NHITS and NHITS1 excelled at longer horizons. TFT1 exhibited robustness across all horizons in DM test results, demonstrating its adaptability for generalization beyond the test set. In the AEX dataset, TFT1 consistently performed well across all horizons, indicating its capability to generalize across different datasets, while TiDE1 stood out for long-term forecasting.

7.4 *Main RQ: To what extent can neural network models predict stock volatility?*

The results indicate that machine learning algorithms can effectively predict volatility for stocks, particularly when models are well-tuned for different forecast horizons. NHITS1 consistently achieved the lowest QLIKE values, reflecting high accuracy, while TFT1 demonstrated superior generalization capabilities, outperforming peers in DM tests across various scenarios. However, challenges remain in ensuring consistency across individual forecasts, particularly during periods of high market volatility. Filtering extreme outliers proved useful in demonstrating the more stable aspects of model performance, particularly for FirmTFT and TFT1.

REFERENCES

Allen, H., & Taylor, M. (1990). Charts, noise and fundamentals in the london foreign exchange market. *Economic Journal*, *100*(400), 49–59. https://EconPapers.repec.org/RePEc:ecj:econjl:v:100:y:1990:i:400:p:49-59

Atkins, A., Niranjan, M., & Gerding, E. (2018). Financial news predicts stock market volatility better than close price. *The Journal of Finance and Data Science*, *4*(2), 120–137. https://doi.org/10.1016/j.jfds.2018.02.002

Bašta, M., & Molnár, P. (2018). Oil market volatility and stock market volatility. *Finance Research Letters*, *26*, 204–214. https://doi.org/10.1016/j.frl.2018.02.001

Bergstra, J., Bardenet, R., Bengio, Y., & Kégl, B. (2011). Algorithms for hyper-parameter optimization. *Proceedings of the 24th International Conference on Neural Information Processing Systems*, 2546–2554.

Bianchi, D., Büchner, M., & Tamoni, A. (2020). Bond risk premiums with machine learning. *The Review of Financial Studies*, *34*(2), 1046–1089. https://doi.org/10.1093/rfs/hhaa062

BIS. (2019). The Basel Framework (III). https://www.bis.org/baselframework/BaselFramework.pdf

Bollerslev, T. (2023). Reprint of: Generalized Autoregressive Conditional Heteroskedasticity. *Journal of Econometrics*, *234*, 25–37. https://doi.org/10.1016/j.jeconom.2023.02.001

Bonato, M., Cepni, O., Gupta, R., & Pierdzioch, C. (2022). Forecasting realized volatility of international REITs: The role of realized skewness and realized kurtosis. *Journal of Forecasting*, *41*(2), 303–315. https://doi.org/10.1002/for.2813

Bouri, E., Gkillas, K., Gupta, R., & Pierdzioch, C. (2021). Forecasting Realized Volatility of Bitcoin: The Role of the Trade War. *Computational Economics*, *57*(1), 29–53. https://doi.org/10.1007/s10614-020-10022-4

Branco, R. R., Rubesam, A., & Zevallos, M. (2024). Forecasting realized volatility: Does anything beat linear models? *Journal of Empirical Finance*, *78*, 101524. https://doi.org/10.1016/j.jempfin.2024.101524

Brown, G. W., & Cliff, M. T. (2004). Investor sentiment and the near-term stock market. *Journal of Empirical Finance*, *11*(1), 1–27. https://doi.org/10.1016/j.jempfin.2002.12.001

Brownlees, C. T., Engle, R. F., & Kelly, B. T. (2011, August). A practical guide to volatility forecasting through calm and storm. https://doi.org/10.2139/ssrn.1502915

Campbell, J. Y., & Hentschel, L. (1991, June). *No News is Good News: An Asymmetric Model of Changing Volatility in Stock Returns* (NBER Working Papers No. 3742). National Bureau of Economic Research, Inc. https://ideas.repec.org/p/nbr/nberwo/3742.html

Campbell, J. Y., & Thompson, S. B. (2007). Predicting excess stock returns out of sample: Can anything beat the historical average? *The Review of Financial Studies*, *21*(4), 1509–1531. https://doi.org/10.1093/rfs/hhm055

Caporin, M., Di Fonzo, T., & Girolimetto, D. (2024). Exploiting Intraday Decompositions in Realized Volatility Forecasting: A Forecast Reconciliation Approach. *Journal of Financial Econometrics*, nbae014. https://doi.org/10.1093/jjfinec/nbae014

Challu, C., Olivares, K. G., Oreshkin, B. N., Garza, F., Mergenthaler-Canseco, M., & Dubrawski, A. (2022). N-hits: Neural hierarchical interpolation for time series forecasting. https://arxiv.org/abs/2201.12886

Chiriac, R., & Voev, V. (2011). Modelling and forecasting multivariate realized volatility. *Journal of Applied Econometrics*, *26*(6), 922–947. https://doi.org/10.1002/jae.1152

Christie, A. A. (1982). The stochastic behavior of common stock variances: Value, leverage and interest rate effects. *Journal of Financial Economics*, *10*(4), 407–432. https://doi.org/https://doi.org/10.1016/0304-405X(82)90018-6

Corsi, F. (2008). A Simple Approximate Long-Memory Model of Realized Volatility. *Journal of Financial Econometrics*, *7*(2), 174–196. https://doi.org/10.1093/jjfinec/nbp001

Czudaj, R. L. (2019). Crude oil futures trading and uncertainty. *Energy Economics*, *80*, 793–811. https://doi.org/10.1016/j.eneco.2019.01.002

Das, A., Kong, W., Leach, A., Mathur, S., Sen, R., & Yu, R. (2024). Long-term forecasting with tide: Time-series dense encoder. https://arxiv.org/abs/2304.08424

Dask Development Team. (2016). *Dask: Library for dynamic task scheduling*. http://dask.pydata.org

Datastream International. (2024, August). Available: Lseg workspace.

D'Ecclesia, R. L., & Clementi, D. (2021). Volatility in the stock market: ANN versus parametric models. *Annals of Operations Research*, *299*(1-2), 1101–1127. https://doi.org/10.1007/s10479-019-03374-0

Díaz, J. D., Hansen, E., & Cabrera, G. (2024). Machine-learning stock market volatility: Predictability, drivers, and economic value. *International Review of Financial Analysis*, *94*, 103286. https://doi.org/10.1016/j.irfa.2024.103286

Diebold, F., & Mariano, R. (1995). Comparing predictive accuracy. *Journal of Business & Economic Statistics*, *13*(3), 253–63. https://EconPapers.repec.org/RePEc:bes:jnlbes:v:13:y:1995:i:3:p:253-63

Dutta, A., & Das, D. (2022). Forecasting realized volatility: New evidence from time-varying jumps in vix. *Journal of Futures Markets*, *42*(12), 2165–2189. https://doi.org/https://doi.org/10.1002/fut.22372

Engle, R. F., Ghysels, E., & Sohn, B. (2013). Stock market volatility and macroeconomic fundamentals. *The Review of Economics and Statistics*, *95*(3), 776–797. https://doi.org/10.1162/REST_a_00300

Engle, R. F., & Manganelli, S. (2004). CAViaR: Conditional Autoregressive Value at Risk by Regression Quantiles. *Journal of Business & Economic Statistics*, *22*(4), 367–381. https://doi.org/10.1198/073500104000000370

Engle, R. F., & Rangel, J. G. (2008). The spline-garch model for low frequency volatility and its global macroeconomic causes. *Review of Financial Studies*, *21*. https://doi.org/10.2139/ssrn.939447

Evans, G. W., Hommes, C., McGough, B., & Salle, I. (2022). Are long-horizon expectations (de-)stabilizing? Theory and experiments. *Journal of Monetary Economics*, *132*, 44–63. https://doi.org/10.1016/j.jmoneco.2022.08.002

Frankel, J. A., & Froot, K. A. (1990). Chartists, Fundamentalists, and Trading in the Foreign Exchange Market. *The American Economic Review*, *80*(2,), 181–185. http://www.jstor.org/stable/2006566

Gallo, G. M., & Otranto, E. (2015). Forecasting realized volatility with changing average levels. *International Journal of Forecasting*, *31*(3), 620–634. https://doi.org/10.1016/j.ijforecast.2014.09.005

Gobato Souto, H. (2023). NHITS for Forecasting Stock Realized Volatility. *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.4650761

Goh, J., Jiang, F., Tu, J., & Zhou, G. Forecasting government bond risk premia using technical indicators [25th Australasian Finance and Banking Conference 2012, Asian Finance Association (AsFA) 2013 Conference]. In: 25th Australasian Finance and Banking Conference 2012, Asian Finance Association (AsFA) 2013 Conference. 2013, July. https://doi.org/10.2139/ssrn.1914227

Gradojevic, N., & Tsiakas, I. (2021). Volatility cascades in cryptocurrency trading. *Journal of Empirical Finance*, *62*, 252–265. https://doi.org/10.1016/j.jempfin.2021.04.005

Gunnarsson, E. S., Isern, H. R., Kaloudis, A., Risstad, M., Vigdel, B., & Westgaard, S. (2024). Prediction of realized volatility and implied volatility indices using AI and machine learning: A review. *International Review of Financial Analysis*, *93*, 103221. https://doi.org/10.1016/j.irfa.2024.103221

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., . . . Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2

Harvey, D., Leybourne, S., & Newbold, P. (1997). Testing the equality of prediction mean squared errors. *International Journal of Forecasting*, *13*(2), 281–291. https://doi.org/10.1016/S0169-2070(96)00719-4

Hong, H., & Stein, J. C. (2003). Differences of Opinion, Short-Sales Constraints, and Market Crashes. *Review of Financial Studies*, *16*(2), 487–525. https://doi.org/10.1093/rfs/hhg006

Hunter, J. D. (2007). Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, *9*(3), 90–95. https://doi.org/10.1109/MCSE.2007.55

Jamali, I., & Yamani, E. (2019). Out-of-sample exchange rate predictability in emerging markets: Fundamentals versus technical analysis. *Journal of International Financial Markets, Institutions and Money*, *61*, 241–263. https://doi.org/10.1016/j.intfin.2019.04.002

Jiaona Li. (2024). The Tail Characteristics of Several Asymmetric Distributions and Their Applications in the Field of Financial Data Modelling and Machine Learning. *Journal of Electrical Systems*, *20*(3), 3093–3101. https://doi.org/10.52783/jes.4737

Joseph, V. R. (2022). Optimal ratio for data splitting. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, *15*(4), 531–538. https://doi.org/10.1002/sam.11583

Kelly, B., & Pruitt, S. (2013). Market expectations in the cross-section of present values. *The Journal of Finance*, *68*(5), 1721–1756. https://doi.org/https://doi.org/10.1111/jofi.12060

Lee, W. Y., Jiang, C. X., & Indro, D. C. (2002). Stock market volatility, excess returns, and the role of investor sentiment.

Levy, R. A. (1967). Relative strength as a criterion for investment selection. *The Journal of Finance*, *22*(4), 595–610. https://doi.org/10.1111/j.1540-6261.1967.tb00295.x

Li, J. (2022). The Comparison of LSTM, LGBM, and CNN in Stock Volatility Prediction: https://doi.org/10.2991/aebmr.k.220307.147

Liaw, R., Liang, E., Nishihara, R., Moritz, P., Gonzalez, J. E., & Stoica, I. (2018). Tune: A research platform for distributed model selection and training. *arXiv preprint arXiv:1807.05118*.

Lim, B., Arık, S. Ö., Loeff, N., & Pfister, T. (2021). Temporal Fusion Transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, *37*(4), 1748–1764. https://doi.org/10.1016/j.ijforecast.2021.03.012

Lin, Q. (2018). Technical analysis and stock return predictability: An aligned approach. *Journal of Financial Markets*, *38*, 103–123. https://doi.org/10.1016/j.finmar.2017.09.003

Liu, L., & Pan, Z. (2020). Forecasting stock market volatility: The role of technical variables. *Economic Modelling*, *84*, 55–65. https://doi.org/10.1016/j.econmod.2019.03.007

Liu, L. Y., Patton, A. J., & Sheppard, K. (2015). Does anything beat 5-minute RV? A comparison of realized measures across multiple asset classes. *Journal of Econometrics*, *187*(1), 293–311. https://doi.org/10.1016/j.jeconom.2015.02.008

Lopez, J. A. (2001). Evaluating the predictive accuracy of volatility models. *Journal of Forecasting*, *20*(2), 87–109. https://doi.org/10.1002/1099-131X(200103)20:2<87::AID-FOR782>3.0.CO;2-7

Luo, J., Chen, Z., & Wang, S. (2023). Realized volatility forecast of financial futures using time-varying HAR latent factor models. *Journal of Management Science and Engineering*, *8*(2), 214–243. https://doi.org/10.1016/j.jmse.2022.10.005

Marshall, B. R., Nguyen, N. H., & Visaltanachoti, N. (2017). Time series momentum and moving average trading rules. *Quantitative Finance*, *17*(3), 405–421. https://doi.org/10.1080/14697688.2016.1205209

McKinney, W. (2010). Data Structures for Statistical Computing in Python. In S. van der Walt & J. Millman (Eds.), *Proceedings of the 9th Python in Science Conference* (pp. 56–61). https://doi.org/10.25080/Majora-92bf1922-00a

Menkhoff, L., & Taylor, M. P. (2007). The obstinate passion of foreign exchange professionals: Technical analysis. *Journal of Economic Literature*, *45*(4), 936–972. https://doi.org/10.1257/jel.45.4.936

Mittnik, S., Robinzonov, N., & Spindler, M. (2015). Stock market volatility: Identifying major drivers and the nature of their impact. *Journal of Banking & Finance*, *58*, 1–14. https://doi.org/10.1016/j.jbankfin.2015.04.003

Mizik, N. (2014). Assessing the Total Financial Performance Impact of Brand Equity with Limited Time-Series Data. *Journal of Marketing Research*, *51*(6), 691–706. https://doi.org/10.1509/jmr.13.0431

Müller, U. A., Dacorogna, M. M., Davé, R. D., Olsen, R. B., Pictet, O. V., & von Weizsäcker, J. E. (1997). Volatilities of different time resolutions — analyzing the dynamics of market components [High Frequency Data in Finance, Part 1]. *Journal of Empirical Finance*, *4*(2), 213–239. https://doi.org/https://doi.org/10.1016/S0927-5398(97)00007-8

Neely, C. J., Rapach, D. E., Tu, J., & Zhou, G. (2014). Forecasting the equity risk premium: The role of technical indicators. *Management Science*, *60*(7), 1772–1791. https://doi.org/10.1287/mnsc.2013.1838

Olivares, K. G., Challú, C., Garza, F., Canseco, M. M., & Dubrawski, A. (2022). NeuralForecast: User friendly state-of-the-art neural forecasting models. https://github.com/Nixtla/neuralforecast

OpenAI. (2024). Chatgpt (nov 22 version) [large language model] [Accessed: 2024-12-01]. https://chat.openai.com/chat

Panopoulou, E., & Souropanis, I. (2019). The role of technical indicators in exchange rate forecasting. *Journal of Empirical Finance*, *53*, 197–221. https://doi.org/10.1016/j.jempfin.2019.07.004

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Quaedvlieg, R. (2021). Multi-horizon forecast comparison. *Journal of Business & Economic Statistics*, *39*(1), 40–53. https://doi.org/10.1080/07350015.2019.1620074

Schwert, G. W. (1989). Why Does Stock Market Volatility Change Over Time? *The Journal of Finance*, *44*(5), 1115–1153. https://doi.org/10.1111/j.1540-6261.1989.tb02647.x

Seabold, S., & Perktold, J. (2010). Statsmodels: Econometric and statistical modeling with python. *9th Python in Science Conference*.

Sewell, M. (2011, January). *Characterization of financial time series* (Research Note No. RN/11/01). UCL Department of Computer Science. http://www.cs.ucl.ac.uk/fileadmin/UCL-CS/research/Research_Notes/RN_11_01.pdf

Sezer, O. B., Gudelek, M. U., & Ozbayoglu, A. M. (2020). Financial time series forecasting with deep learning : A systematic literature review: 2005–2019. *Applied Soft Computing*, *90*, 106181. https://doi.org/10.1016/j.asoc.2020.106181

Shu, H.-C., & Chang, J.-H. (2015). Investor Sentiment and Financial Market Volatility. *Journal of Behavioral Finance*, *16*(3), 206–219. https://doi.org/10.1080/15427560.2015.1064930

Smith, D. M., Wang, N., Wang, Y., & Zychowicz, E. J. (2014). Sentiment and the Effectiveness of Technical Analysis: Evidence from the Hedge Fund Industry. *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.2457289

Souropanis, I., & Vivian, A. (2023). Forecasting realized volatility with wavelet decomposition. *Journal of Empirical Finance*, *74*, 101432. https://doi.org/10.1016/j.jempfin.2023.101432

Souto, H. G., & Moradi, A. (2024). Can transformers transform financial forecasting? *China Finance Review International*. https://doi.org/10.1108/CFRI-01-2024-0032

Sprangers, O., Schelter, S., & De Rijke, M. (2023). Parameter-efficient deep probabilistic forecasting. *International Journal of Forecasting*, *39*(1), 332–345. https://doi.org/10.1016/j.ijforecast.2021.11.011

Takahashi, M., Watanabe, T., & Omori, Y. (2021). Forecasting Daily Volatility of Stock Price Index Using Daily Returns and Realized Volatility. *Econometrics and Statistics*, S2452306221000964. https://doi.org/10.1016/j.ecosta.2021.08.002

Tan, S. K., Chan, J. S. K., & Ng, K. H. (2022). Modelling and forecasting stock volatility and return: A new approach based on quantile Rogers–Satchell volatility measure with asymmetric bilinear CARR model. *Studies in Nonlinear Dynamics & Econometrics*, *26*(3), 437–474. https://doi.org/10.1515/snde-2019-0101

The Hong Kong Institute Of Bankers. (2008, September). *Bank Asset and Liability Management* (1st ed.). Wiley. https://doi.org/10.1002/9781119444497

Truong, C., Oudre, L., & Vayatis, N. (2020). Selective review of offline change point detection methods. *Signal Processing*, *167*, 107299. https://doi.org/https://doi.org/10.1016/j.sigpro.2019.107299

Tsay, R. S. (2005, August). *Analysis of Financial Time Series* (1st ed.). Wiley. https://doi.org/10.1002/0471746193

Veronesi, P. (1999). Stock Market Overreaction to Bad News in Good Times: A Rational Expectations Equilibrium Model. *The Review of Financial Studies*, *12*(5).

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., . . . SciPy 1.0 Contributors. (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, *17*, 261–272. https://doi.org/10.1038/s41592-019-0686-2

Waskom, M. L. (2021). Seaborn: Statistical data visualization. *Journal of Open Source Software*, *6*(60), 3021. https://doi.org/10.21105/joss.03021

Welch, I., & Goyal, A. (2007). A comprehensive look at the empirical performance of equity premium prediction. *The Review of Financial Studies*, *21*(4), 1455–1508. https://doi.org/10.1093/rfs/hhm014

Wilder, J. (1978). *New Concepts in Technical Trading Systems*. Trend Research. https://books.google.nl/books?id=WesJAQAAMAAJ

Yan, X., Wu, Q., & Zhang, W. (2019, October). Cross-sectional Learning of Extremal Dependence among Financial Assets [arXiv:1905.13425 [q-fin]]. Retrieved November 12, 2024, from http://arxiv.org/abs/1905.13425

Zarrabi, N., Snaith, S., & Coakley, J. (2017). Fx technical trading rules can be profitable sometimes! *International Review of Financial Analysis*, *49*. https://doi.org/10.1016/j.irfa.2016.12.010

Zhang, C., Pu, X., Cucuringu, M., & Dong, X. (2024). Forecasting realized volatility with spillover effects: Perspectives from graph neural networks. *International Journal of Forecasting*, S0169207024000967. https://doi.org/10.1016/j.ijforecast.2024.09.002

APPENDIX A

The data cleaning, pre processing, and modeling was performed in Python Version 3.10 using the following libraries and packages:

- **Nixtla Neural Forecast** (Olivares et al., 2022)
- **Pandas** (McKinney, 2010)
- **Numpy** (Harris et al., 2020)
- **Matplotlib** (Hunter, 2007)
- **Seaborn** (Waskom, 2021)
- **Ray Tune** (Liaw et al., 2018)
- **HyperOptSearch** (Bergstra et al., 2011)
- **Ruptures** (Truong et al., 2020)
- **dask** (Dask Development Team, 2016)
- **Scikit-learn** (Pedregosa et al., 2011)
  - `IterativeImputer`
  - `BayesianRidge`
- **Statsmodels** (Seabold & Perktold, 2010)
  - `adfuller`
  - `het, arch`
- **SciPy** (Virtanen et al., 2020)
  - `skew`
  - `adfuller`
  - `jarque-bera`
  - `kstest`
  - `t`

APPENDIX B

Appendix B showcases the summary statistics for both datasets with its exogenous variables

Table 15: S&P500 Summary statistics

Variable	Count	Mean	Std	Min	25%	50%	75%	Max
840E	2008545	0.277	0.179	0.042	0.171	0.231	0.322	4.226
MA(2,9)	2008545	0.545	0.498	0.000	0.000	1.000	1.000	1.000
MA(3,9)	2008545	0.545	0.498	0.000	0.000	1.000	1.000	1.000
MA(1,12)	2008545	0.550	0.497	0.000	0.000	1.000	1.000	1.000
MA(2,12)	2008545	0.550	0.497	0.000	0.000	1.000	1.000	1.000
MA(3,12)	2008545	0.550	0.497	0.000	0.000	1.000	1.000	1.000
MOM(9)	2008545	0.551	0.497	0.000	0.000	1.000	1.000	1.000
MOM(12)	2008545	0.555	0.497	0.000	0.000	1.000	1.000	1.000
RSI(7)	2008545	0.451	0.498	0.000	0.000	0.000	1.000	1.000
RSI(14)	2008545	0.434	0.496	0.000	0.000	0.000	1.000	1.000
EMA(3,9)	2008545	0.558	0.497	0.000	0.000	1.000	1.000	1.000
EMA(5,9)	2008545	0.562	0.496	0.000	0.000	1.000	1.000	1.000
EMA(5,12)	2008545	0.565	0.496	0.000	0.000	1.000	1.000	1.000
DY	2008545	1.830	1.629	0.000	0.410	1.640	2.810	65.190
PTBV	2008545	3.389	47.620	-2303.010	1.730	2.760	4.650	1495.570
P	2008545	82.122	174.468	0.000	26.676	47.890	85.930	7024.820
PO	2008545	82.106	174.398	0.000	26.670	47.890	85.924	7028.500
VO	2008545	8875.863	45744.304	0.000	1017.800	2307.200	5602.500	5092086.000
PE	2008545	41.242	679.933	-3131.161	14.700	20.700	30.800	103350.000

Table 16: AEX Summary statistics

Variable	Count	Mean	Std	Min	25%	50%	75%	Max
840E	42246	0.273	0.141	0.046	0.180	0.238	0.328	1.355
MA(2,9)	42246	0.549	0.498	0.000	0.000	1.000	1.000	1.000
MA(3,9)	42246	0.550	0.497	0.000	0.000	1.000	1.000	1.000
MA(1,12)	42246	0.553	0.497	0.000	0.000	1.000	1.000	1.000
MA(2,12)	42246	0.555	0.497	0.000	0.000	1.000	1.000	1.000
MA(3,12)	42246	0.554	0.497	0.000	0.000	1.000	1.000	1.000
MOM(9)	42246	0.557	0.497	0.000	0.000	1.000	1.000	1.000
MOM(12)	42246	0.557	0.497	0.000	0.000	1.000	1.000	1.000
RSI(7)	42246	0.441	0.497	0.000	0.000	0.000	1.000	1.000
RSI(14)	42246	0.422	0.494	0.000	0.000	0.000	1.000	1.000
EMA(3,9)	42246	0.564	0.496	0.000	0.000	1.000	1.000	1.000
EMA(5,9)	42246	0.568	0.495	0.000	0.000	1.000	1.000	1.000
EMA(5,12)	42246	0.569	0.495	0.000	0.000	1.000	1.000	1.000
DY	42246	3.301	2.498	0.000	1.270	2.655	4.990	23.960
PTBV	42246	3.106	3.381	-3.372	0.920	2.120	4.100	26.190
P	42246	61.239	91.747	1.670	18.460	35.321	70.828	770.500
PO	42246	61.222	91.770	1.332	18.450	35.310	70.707	770.500
VO	42246	4163.235	6230.771	0.000	470.900	1204.050	5941.441	133015.800
PE	42246	29.758	112.996	-22.429	10.300	17.500	28.800	1783.300

APPENDIX C

This appendix presents the results of statistical tests conducted on the S&P 500 and AEX datasets to evaluate the presence of fat-tailed distributions in the time series. Fat-tailed distributions, often characterized by volatility clustering, structural shifts, skewness, and heavy kurtosis, pose significant challenges for traditional forecasting methods but can be effectively addressed using a Student-t loss. By applying tests such as the Augmented Dickey-Fuller, ARCH, Change Point Detection, Skewness/Kurtosis, and Jarque-Bera, the analysis assesses the statistical properties of the data. The results demonstrate widespread non-stationarity, volatility clustering, structural instability, and deviations from normality, reinforcing the suitability of the Student-t loss for robust modeling and forecasting in these datasets.

Table 17: Fat Tailed Distribution Tests on the S&P 500 dataset

Test	**Failed count**
Augmented Dickey-Fuller	383
Autoregressive Conditional Heteroskedasticity	385
Change Point	385
Skewness/Kurtosis	385
Jarque-Bera	385

For the S&P 500 dataset the results reveal that out of 385 companies, 383 failed the ADF test, indicating that most series are not stationary, which can amplify volatility and contribute to heavy tails. All 385 companies failed the ARCH test, confirming the presence of volatility clustering, a hallmark of fat-tailed distributions. Similarly, all 385 companies failed the change point detection test, highlighting structural shifts in every series, which are often associated with heavy tails. The skewness and kurtosis test also failed for all 385 companies, demonstrating significant skewness or excess kurtosis—both indicative of fat-tailed behavior. Lastly, the Jarque-Bera test failed across all companies, confirming that the series deviate significantly from normality. These findings strongly support the presence of fat-tailed distributions in the data, justifying the use of the Student-t loss, which is robust to such distributions and well-suited for modeling and forecasting under these conditions.

Table 18: Fat Tailed Distribution Tests on the AEX dataset

Test	**Failed count**
Augmented Dickey-Fuller	18
Autoregressive Conditional Heteroskedasticity	18
Change Point	18
Skewness/Kurtosis	18
Jarque-Bera	18

The results for the AEX dataset reveal that all 18 companies failed the ADF test, indicating that none of the time series are stationary, a condition that can amplify volatility and contribute to heavy tails. Additionally, all 18 companies failed the ARCH test, confirming the presence of volatility clustering, a key characteristic of fat-tailed distributions. The change point detection test also failed for all 18 companies, highlighting structural shifts in every series, which are often associated with heavy tails. Similarly, the skewness and kurtosis test failed for all 18 companies, showing significant skewness or excess kurtosis, both indicative of fat-tailed behavior. Lastly, the Jarque-Bera test failed across all companies, confirming that the series deviate significantly from normality. These consistent failures strongly support the presence of fat-tailed distributions in the data, justifying the use of the Student-t loss, which is robust to such distributions and well-suited for forecasting under these conditions.

APPENDIX D

Appendix D analyzes missing data in the S&P500 and AEX datasets, identifying key variables and time periods with significant gaps. Some variables show Missing Not at Random (MNAR) patterns linked to data collection or reporting processes. To address this, iterative imputation with Bayesian Ridge Regression was used. This method models each variable as a function of the others, handles multicollinearity, and ensures consistent and unbiased imputations.

Table 19: Percentage of Missing Data by Variable S&P500

Variable	**Missing Percentage (%)**
`unique_id`	0.000
`ds`	0.000
`840E`	0.000
`VO`	5.518
`P`	0.000
`PE`	4.128
`PO`	1.830
`PTBV`	5.556
`DY`	0.000

For the S&P500 dataset core variables like unique_id, ds, and 840E are complete, with no missing values. Variables DY, PTBV, and P have minimal missingness (less than 1%), while PO and VO show moderate levels (around 4%). The variable PE has the highest missing percentage (7.714%), requiring particular attention during imputation to ensure data integrity and analysis reliability.

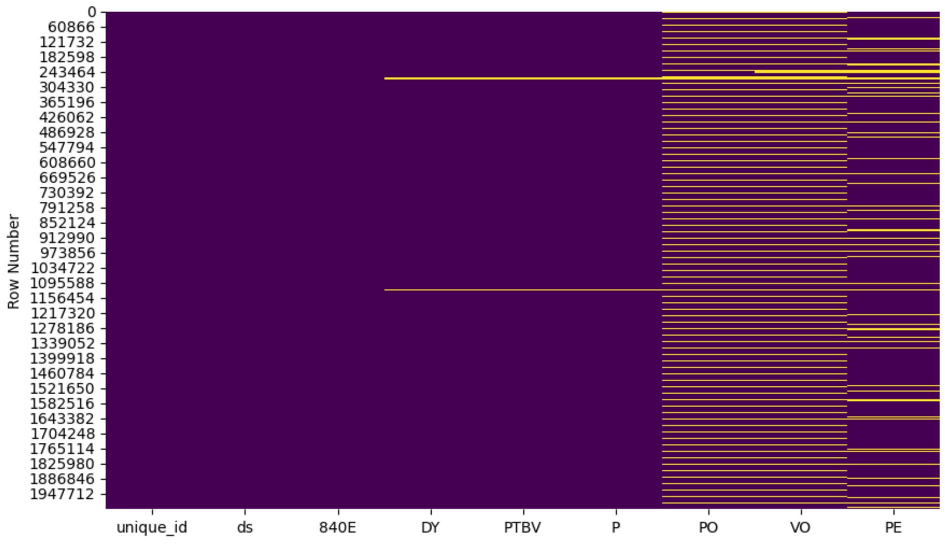

Figure 9: Missing Data Heatmap S&P500

The heatmap visualizes the missing data patterns across all variables in the S&P500 dataset. Variables like unique_id, ds, and 840E are fully observed, as indicated by the absence of yellow (missing values). However, variables such as DY, PTBV, and P show minimal missingness, with sporadic yellow lines. In contrast, PO, VO, and PE exhibit more substantial missingness, represented by more frequent and clustered yellow segments, particularly for PE. This pattern suggests that missing data is not uniformly distributed and may be influenced by systematic factors like reporting or data collection practices. Such patterns highlight the importance of robust imputation strategies to address missing data while preserving underlying relationships.

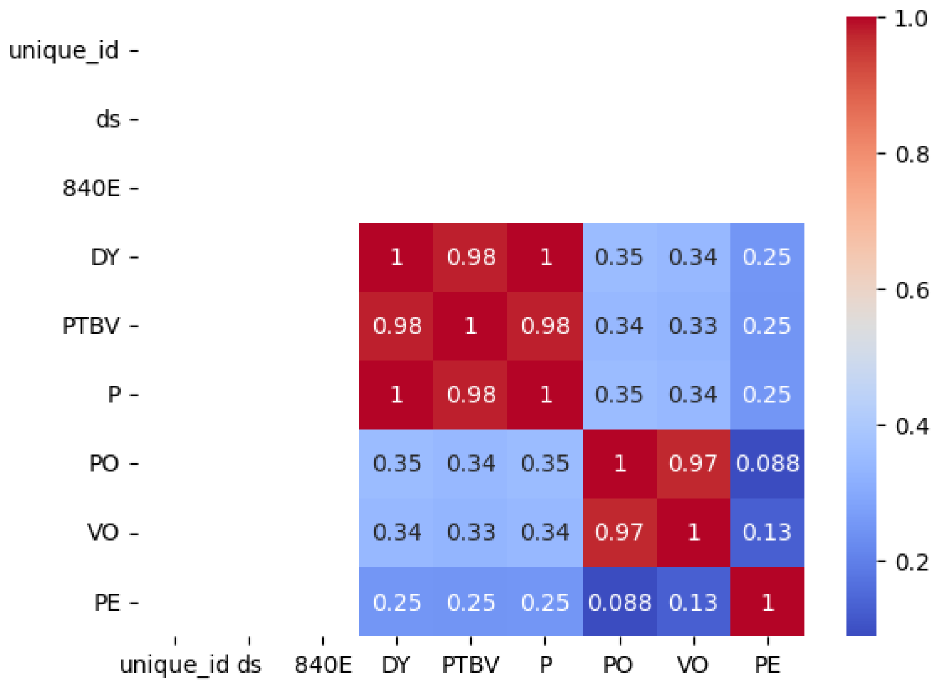

Figure 10: Correlation of Missingness between Variables S&P500

The correlation heat map for the S%P500 dataset illustrates the relationships between the missingness of variables in the dataset. High correlations, such as those between DY, PTBV, and P (close to 1), indicate that these variables tend to have missing values simultaneously, suggesting a pattern of dependency in their missingness. Conversely, low correlations, such as between PO and PE (0.088), reflect minimal or no relationship in their missingness patterns. This information is useful for identifying groups of variables with shared missing data patterns, which can inform strategies for data imputation or further analysis of missing data mechanisms.

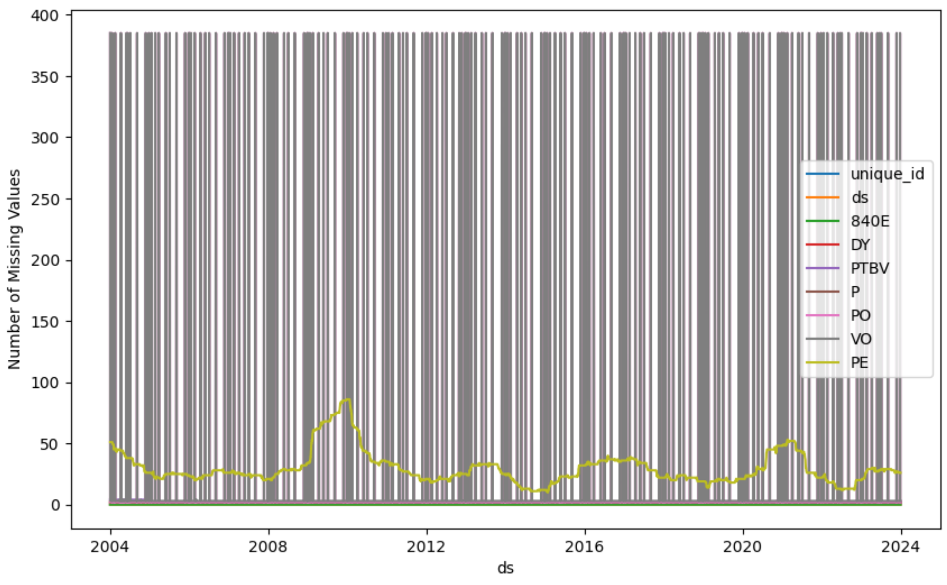

Figure 11: Missing Data Across Time between Variables AEX

The line plot visualizes the number of missing values for each variable over a longer time span (2004–2024). While most variables, such as 840E, exhibit no missing values throughout the period, some variables, such as DY and PTBV, show fluctuating patterns of missingness. Notably, there are spikes in missing data around 2009–2010 and again near 2020, potentially corresponding to specific events or disruptions during these times. The plot indicates that missingness in the dataset is both variable- and time-dependent, with some periods experiencing higher levels of data gaps, which may require targeted imputation or adjustments to maintain data integrity.

Table 20: Percentage of Missing Data by Variable AEX

Variable	**Missing Percentage (%)**
`unique_id`	0.000
`ds`	0.000
`840E`	0.000
`VO`	5.518
`P`	0.000
`PE`	4.128
`PO`	1.830
`PTBV`	5.556
`DY`	0.000

For the AEX dataset most variables, such as unique_id, ds, 840E, P, and DY, are complete, with no missing values. However, VO (5.518%), PTBV (5.556%), PE (4.128%), and PO (1.830%) exhibit varying degrees of missingness. These percentages highlight the need for careful imputation methods, particularly for VO and PTBV, which have the highest missing rates.

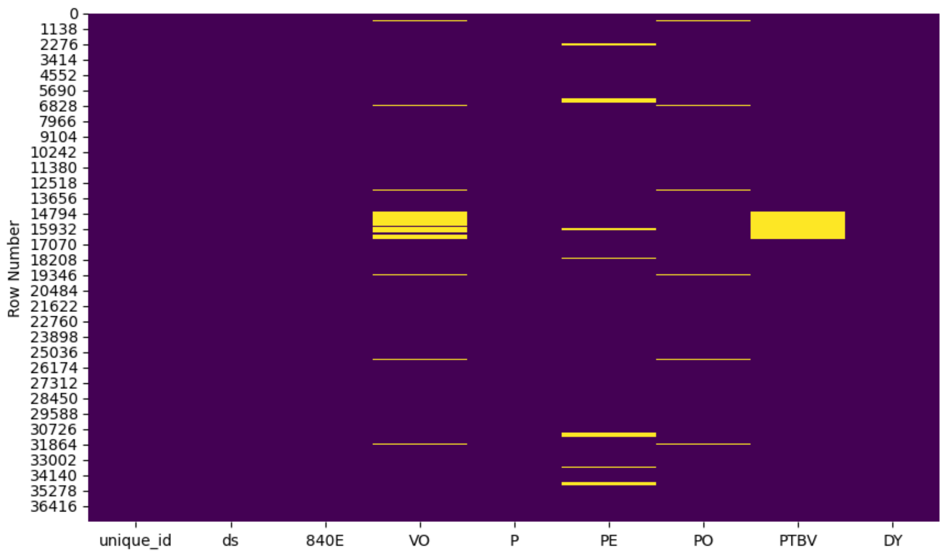

Figure 12: Missing Data Heatmap AEX

The heatmap reveals the distribution of missing values across the variables in the AEX dataset. unique_id, ds, and 840E show no missing values, as indicated by the absence of yellow lines. However, variables like VO, P, and PE have sporadic missingness, with small gaps scattered throughout. In contrast, PTBV and PO show more significant and clustered missingness, particularly for PTBV, which has a dense block of missing data in certain sections. This pattern suggests that the missing data is not uniformly random and may stem from systematic factors, such as reporting periods or data collection inconsistencies. Proper imputation methods will be necessary to handle these missing values effectively while preserving the dataset's integrity.

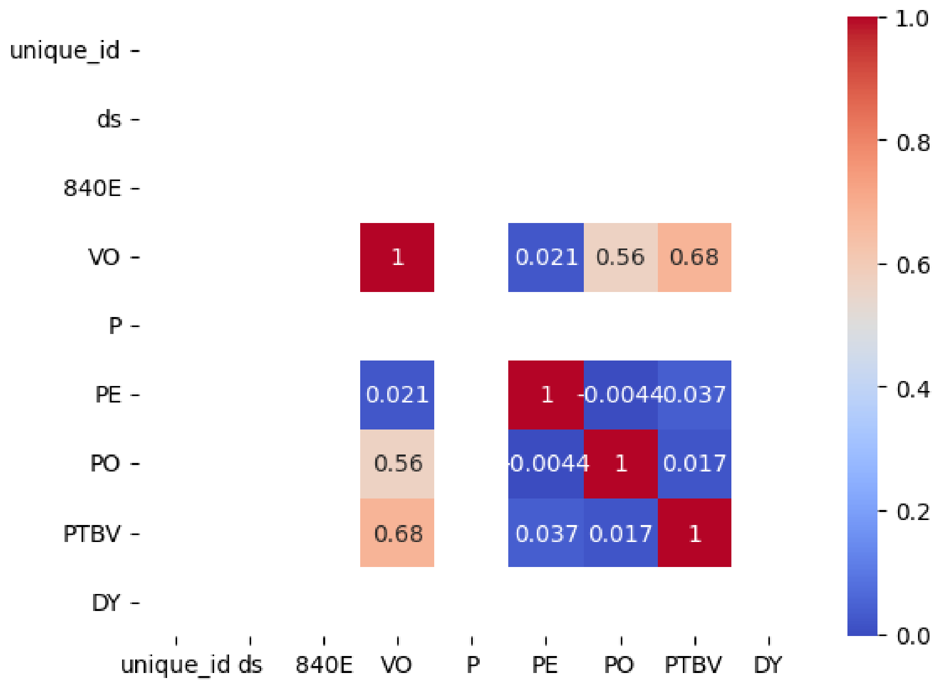

Figure 13: Correlation of Missingness between Variables AEX

The correlation heat map for the AEX dataset highlights the relationships between the values of variables in the dataset. Notable positive correlations include VO and PTBV (0.68), and VO and PO (0.56), indicating a tendency for these variables to increase or decrease together. In contrast, variables like PE and PO (0.0044) or PE and PTBV (0.037) show negligible correlations, suggesting little to no linear association between them. The overall pattern suggests stronger relationships between certain pairs of variables, while others appear independent, which can guide feature selection or further analysis.

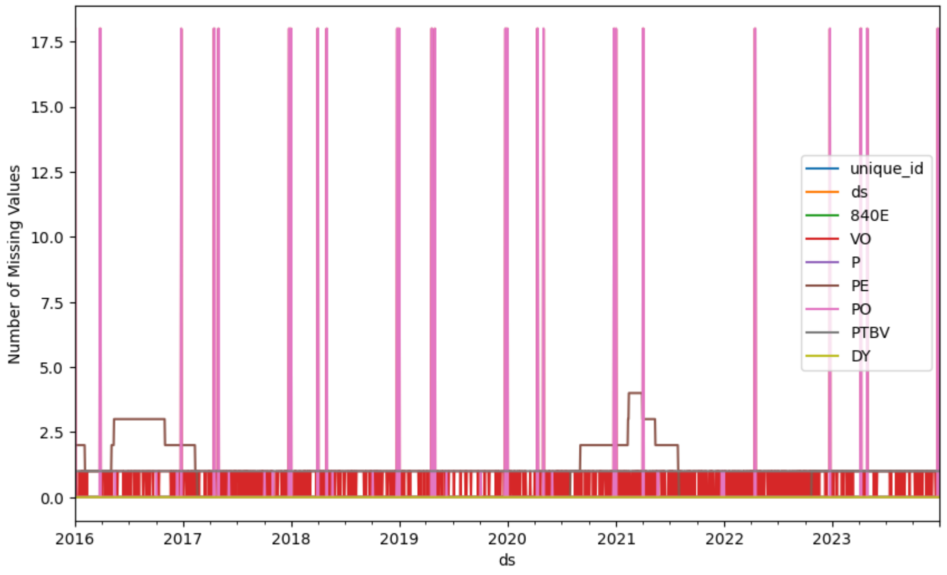

Figure 14: Missing Data Across Time between Variables AEX

The line plot visualizes the number of missing values for each variable over time (ds). It reveals patterns of missingness across the dataset, with some variables, such as PO, showing significant spikes in missing values at certain points in time, particularly between 2016 and 2023. Other variables, like VO and DY, appear to have relatively consistent missingness or minimal occurrences. This indicates that the missingness of some variables is time-dependent, with specific periods exhibiting higher levels of missing data, which may correspond to data collection issues or external events affecting data availability during those times.

Appendix D shows that some variables, like VO, PTBV, and PO, have systematic missingness (MNAR). Iterative imputation with Bayesian Ridge Regression effectively addressed these gaps by modeling dependencies between variables and minimizing bias. This approach preserved the dataset's structure and ensured accurate imputations for further analysis.

APPENDIX E

Appendix E showcases and tests each model configuration, including those without exogenous variables (0), with only Firm indicators (FIRM), only Technical indicators (TECH), and with both sets of indicators (1).

Table 21: QLIKE metric values (in %) for BiTCN's configurations across forecasting horizons (h = 1, 5, 10, 20) in Experiment 1. Lower QLIKE values indicate better performance, and the two best-performing models for each horizon are underlined

model	**h = 1**	**h = 5**	**h = 10**	**h = 20**
BiTCN	<u>0.175</u>	1.127	<u>2.029</u>	<u>5.841</u>
FirmBiTCN	0.178	<u>1.111</u>	<u>1.995</u>	6.331
TechBiTCN	0.179	<u>1.147</u>	2.155	<u>5.986</u>
BiTCN1	<u>0.176</u>	1.112	2.304	6.306

he QLIKE metric table shows that BiTCN configurations generally perform well across horizons, with BiTCN and FirmBiTCN achieving the best results at shorter horizons (h = 1, 5) and BiTCN and TechBiTCN performing better at longer horizons (h = 10, 20). Lower QLIKE values indicate that these models are more effective at minimizing forecast errors.

Table 22: DM test counts for BiTCN's configurations across forecast horizons (h = 1, 5, 10, 20) in Experiment 1. Higher counts indicate better performance. The top two models per horizon are highlighted in green with double underlines, and the bottom two are highlighted in red

model	**h = 1**	**h = 5**	**h = 10**	**h = 20**
BiTCN	2	<u>197</u>	174	46
FirmBiTCN	<u>317</u>	98	65	<u>775</u>
TechBiTCN	<u>452</u>	<u>404</u>	<u>841</u>	207
BiTCN1	41	31	<u>221</u>	<u>550</u>

The DM test table highlights the relative performance of BiTCN configurations, with TechBiTCN excelling in most horizons (h = 1, 5, 10) and FirmBiTCN showing strong performance at horizon h = 20. The counts reveal that some configurations, like BiTCN1, under perform at shorter horizons, emphasizing the variability in model effectiveness across horizons.

Table 23: QLIKE metric values (in %) for NHITS' configurations across forecasting horizons (h = 1, 5, 10, 20) in Experiment 1. Lower QLIKE values indicate better performance, and the two best-performing models for each horizon are underlined

model	**h = 1**	**h = 5**	**h = 10**	**h = 20**
NHITS	<u>0.175</u>	<u>1.113</u>	2.115	5.879
FirmNHITS	0.176	1.170	<u>2.075</u>	6.688
TechNHITS	0.176	1.172	2.120	<u>5.814</u>
NHITS1	<u>0.175</u>	<u>1.085</u>	<u>2.026</u>	<u>5.645</u>

The QLIKE table indicates that NHITS1 consistently ranks among the top two configurations across all horizons, demonstrating strong performance. TechNHITS performs well for h = 20, while FirmNHITS achieves competitive results at h = 10, but it falls short for other horizons.

Table 24: DM test counts for NHITS' configurations across forecast horizons (h = 1, 5, 10, 20) in Experiment 1. Higher counts indicate better performance. The top two models per horizon are highlighted in green with double underlines, and the bottom two are highlighted in red

model	**h = 1**	**h = 5**	**h = 10**	**h = 20**
NHITS	122	426	322	304
FirmNHITS	37	134	184	878
TechNHITS	77	178	370	246
NHITS1	0	17	18	82

The DM test results show that NHITS and NHITS1 dominate performance across most horizons, particularly at h = 1 and h = 5, while FirmNHITS achieves the highest count at h = 20. However, NHITS1 struggles at longer horizons, highlighting trade-offs in its effectiveness.

Table 25: QLIKE metric values (in %) for TiDE's configurations across forecasting horizons (h = 1, 5, 10, 20) in Experiment 1. Lower QLIKE values indicate better performance, and the two best-performing models for each horizon are underlined

model	**h = 1**	**h = 5**	**h = 10**	**h = 20**
TiDE	0.176	1.117	2.093	6.015
FirmTiDE	0.175	1.093	2.212	6.203
TechTiDE	0.176	1.108	2.164	5.654
TiDE1	0.176	1.096	2.114	5.926

The QLIKE table shows that FirmTiDE performs best at shorter horizons (h = 1, 5), while TechTiDE and TiDE1 excel at longer horizons (h = 10, 20). Overall, TiDE configurations demonstrate stable performance with competitive QLIKE values.

Table 26: DM test counts for TiDE's configurations across forecast horizons (h = 1, 5, 10, 20) in Experiment 1. Higher counts indicate better performance. The top two models per horizon are highlighted in green with double underlines, and the bottom two are highlighted in red

model	**h = 1**	**h = 5**	**h = 10**	**h = 20**
TiDE	64	587	98	425
FirmTiDE	2	204	455	772
TechTiDE	37	248	315	26
TiDE1	46	14	135	263

The DM test results indicate that FirmTiDE achieves the highest counts at h = 10 and h = 20, while TiDE performs well at h = 1 and h = 5. TechTiDE and TiDE1 show mixed performance, with notable under performance at certain horizons. This suggests variability in how configurations adapt to different forecasting horizons.

APPENDIX F

Appendix F shows the QLIKE behavior of the TFT0 and FirmTFT models across the test set for all the companies trained.

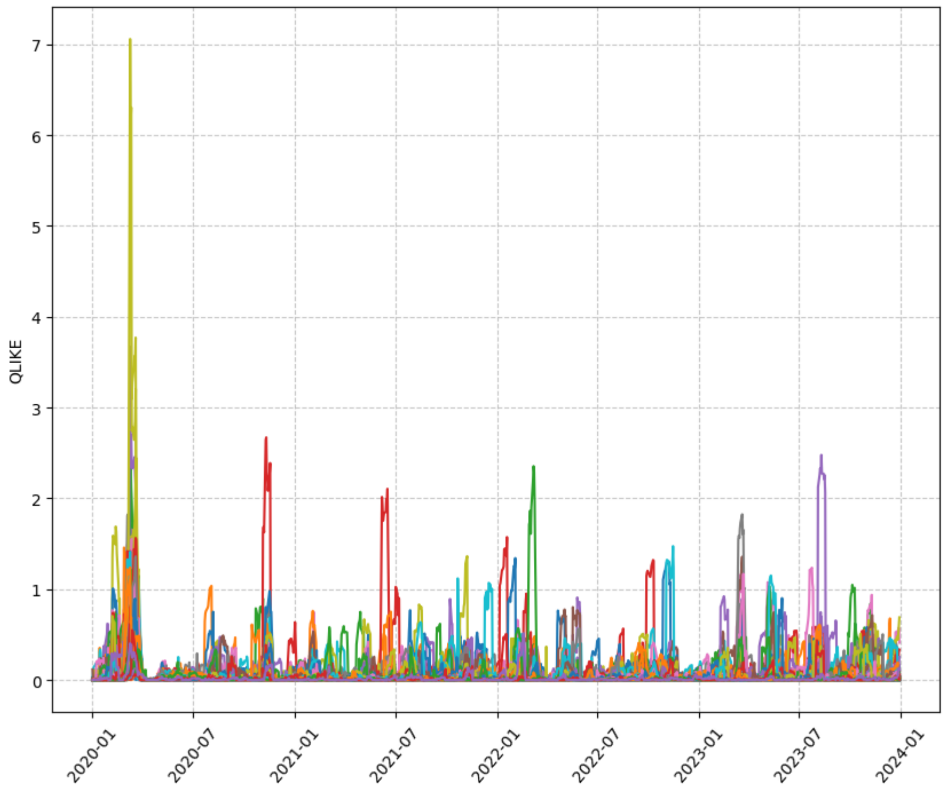

Figure 15: TFT0 QLIKE over the test set in experiment 1 per company

Figure 15 shows the QLIKE loss for TFT0 over time. There is notable variability in QLIKE values across individual companies, with some significant peaks indicating periods of higher prediction errors. These spikes are especially apparent in the early period, highlighting challenges in prediction accuracy during these times.

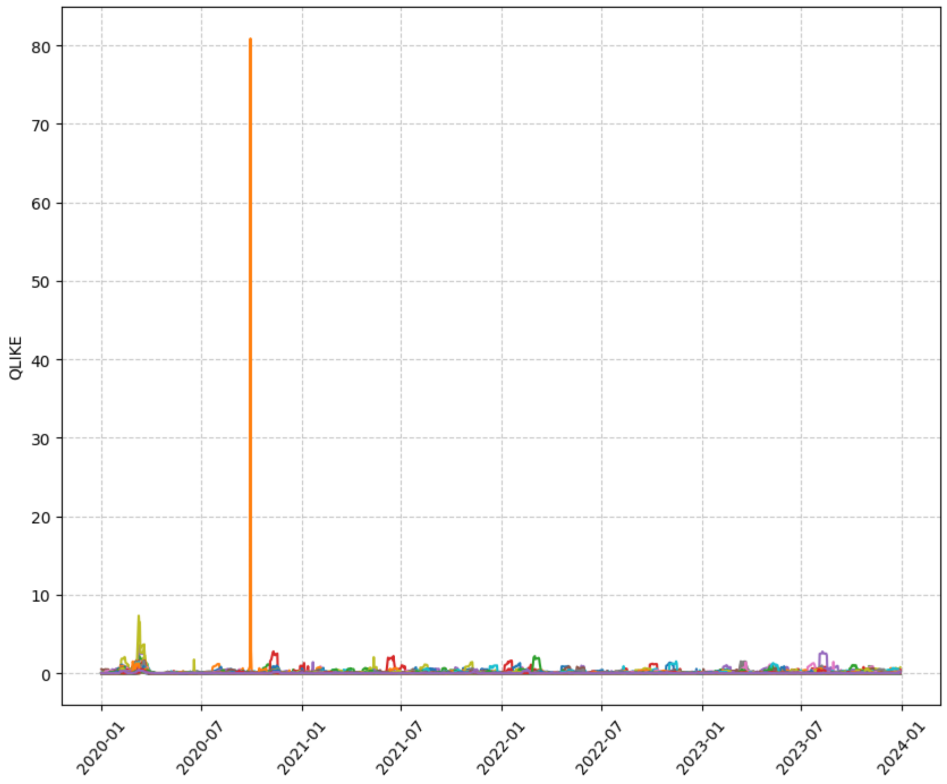

Figure 16: FirmTFT's QLIKE over the test set in experiment 1 per company

In figure 16, FirmTFT's QLIKE loss over time displays an extremely high spike in mid-2020, suggesting a significant deviation in forecast accuracy during that period. This indicates potential issues in the model's ability to handle certain conditions, particularly during volatile periods.

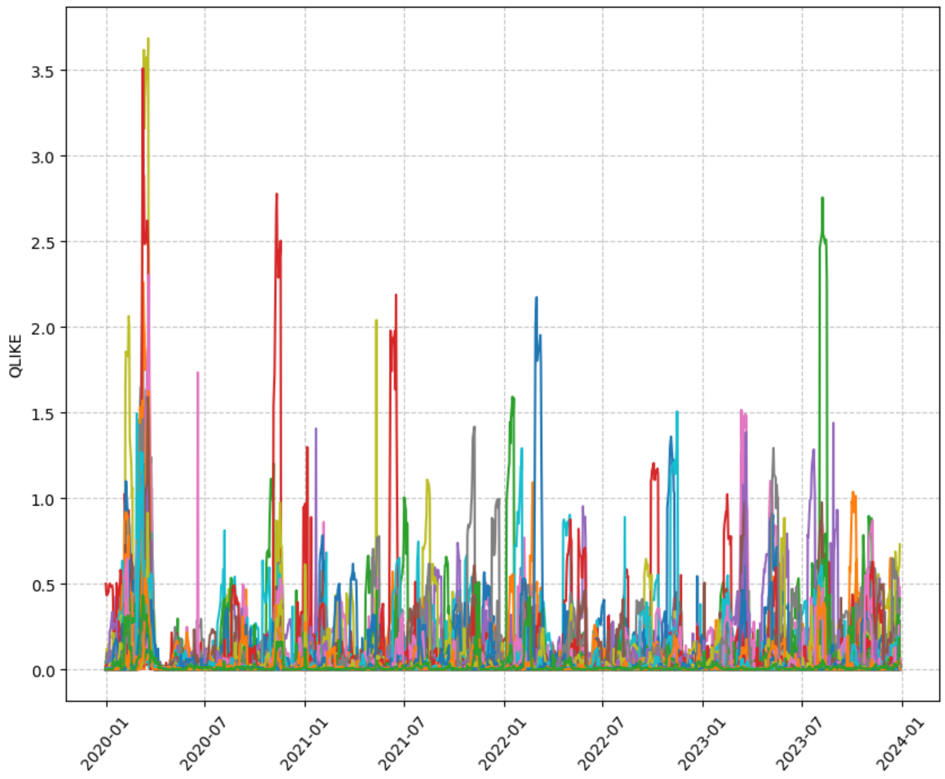

Figure 17: FirmTFT's QLIKE filtered over the test set in experiment 1 per company

Figure 17 shows the filtered FirmTFT's QLIKE over time, where extreme outliers have been removed. This results in a more stable trend with fewer pronounced spikes, which makes it easier to observe the underlying pattern of prediction accuracy across different companies.

APPENDIX G

Appendix G analyzes the QLIKE metric for the TFT1 model at a 10-day forecast horizon. It includes tables showing the companies with the lowest average QLIKE percentages and the highest outlier QLIKE values, highlighting where the model performs best and where it struggles.

Table 27: Top 5 Companies with the Lowest Average QLIKE (%) in TFT1 H = 10

Company	QLIKE (%)
EQT	0.852
FREEPORT-MCMORAN	1.094
EXXON MOBIL	1.175
COTERRA ENERGY	1.210
MICROSOFT	1.237

Table 27 highlights the top 5 companies with the lowest average QLIKE percentages for the TFT1 model at a forecast horizon (H) of 10 days. EQT has the smallest QLIKE value of 0.852%, indicating a high degree of accuracy in the model's forecasts for this company. Similarly, MICROSOFT, with a QLIKE of 1.237%, also shows relatively low forecasting errors. These low QLIKE values reflect the model's effectiveness in predicting volatility for these companies, potentially due to stable market conditions or patterns that align well with the model's assumptions. The inclusion of both energy and technology firms indicates the model's versatility across industries.

Table 28: Top 5 Outliers QLIKE Values in TFT1 H = 10

Date	Company	QLIKE Value
2021-01-20	HOST HOTELS & RESORTS REIT	72.244
2021-01-11	BIOGEN	31.799
2021-01-20	REGIONS FINL.NEW	17.472
2020-06-01	EQUINIX REIT	9.890
2021-09-03	BIOGEN	7.970

The highest outlier in table 28, 72.244, is linked to HOST HOTELS & RESORTS REIT on January 20, 2021. This suggests an instance of significant forecasting error, potentially due to unusual market conditions or specific events affecting the company. BIOGEN appears twice in the table, with substantial QLIKE values on January 11, 2021, and September 3, 2021, highlighting periods where the model struggled to accurately forecast volatility for this company. Other companies such as REGIONS FINL.NEW and EQUINIX REIT also exhibit notable deviations, which could indicate volatility or periods of unexpected market behavior affecting the forecast accuracy.

Table 29: Top 5 Outliers QLIKE Values in TFT Firm H = 10

Date	Company	QLIKE Value
2020-09-29	FIRSTENERGY	80.878
2020-03-09	ONEOK	7.328
2020-03-11	ONEOK	6.470
2020-03-10	ONEOK	6.153
2020-03-19	APA	3.686

The largest outlier on table 29 is associated with FIRSTENERGY on September 29, 2020, indicating a significantly high QLIKE value of 80.878. This may suggest a period of high volatility or model misfit for this firm on that date. The other values, though smaller, also represent notable deviations from expected forecasting accuracy, particularly for ONEOK in March 2020 and APA on March 19, 2020, potentially coinciding with market disruptions or firm-specific events.

APPENDIX H

Appendix H showcase the average data and predictions for the entire studied period for the TFT1 H=20 in Experiment 1 over horizons 5 and 10.

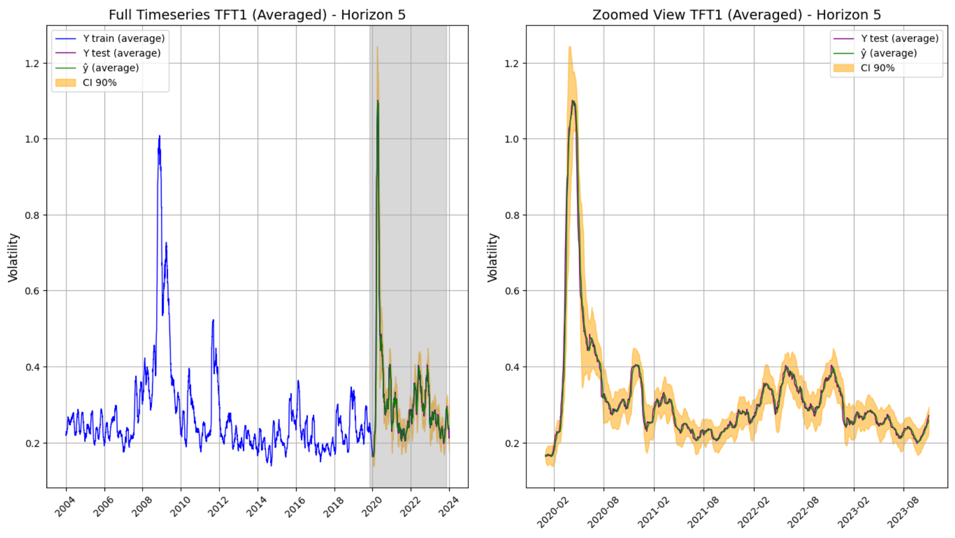

Figure 18: Visualization of the average data across companies for TFT1 H=20 in Experiment 1. The left panels show the full time series with training data (blue) and the shaded test area. The right panels zoom in on the test period, showing actual test values (y, purple), test predictions ($\hat{y}$, green), and 90% confidence intervals (yellow).

Figure 18 displays the performance of the Temporal Fusion Transformer (TFT) model for Horizon 5, predicting volatility five days ahead. In the left plot, the model's predictions ($\hat{y}$) closely track the observed average volatility in both the training and testing periods, with noticeable alignment during major volatility spikes, such as the 2008 financial crisis and the COVID-19 pandemic in 2020. The orange-shaded 90% confidence interval remains relatively narrow throughout most of the forecast period, indicating that the model's uncertainty is well-contained even during periods of heightened market turbulence. The right plot provides a zoomed-in view of the test period from 2020 to 2023, showing that the model effectively captures the short-term volatility patterns with minimal divergence from the observed values. However, the confidence interval slightly widens during volatility spikes, reflecting increased uncertainty during more volatile periods. Overall, the model demonstrates strong predictive performance for the five-day horizon, with a good balance between accuracy and confidence in its forecasts.

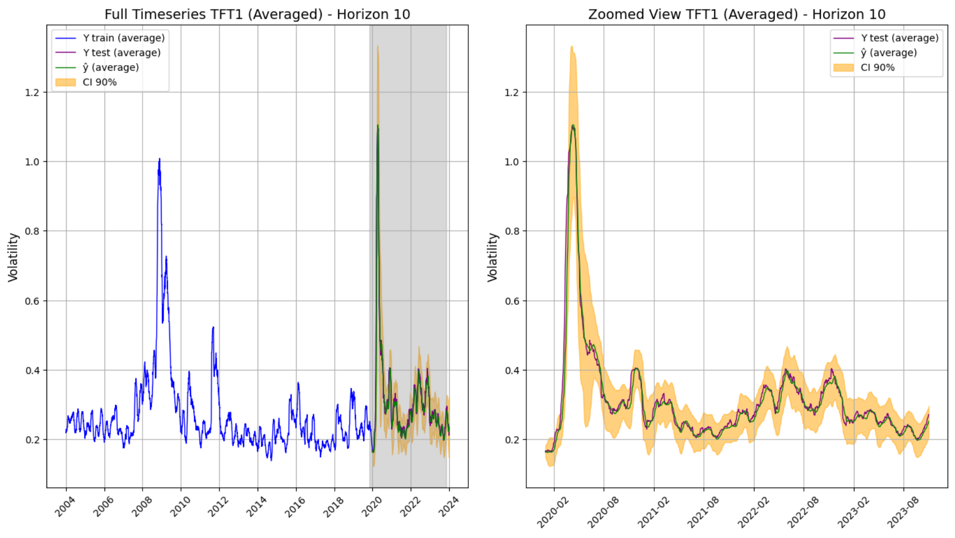

Figure 19: Visualization of the average data across companies for TFT1 H=20 in Experiment 1. The left panels show the full time series with training data (blue) and the shaded test area. The right panels zoom in on the test period, showing actual test values (y, purple), test predictions ($\hat{y}$, green), and 90% confidence intervals (yellow).

Meanwhile, figure 19 illustrates the performance of the Temporal Fusion Transformer (TFT) model for Horizon 10, forecasting volatility ten days ahead. Similar to the Horizon 5 figure, the left plot shows that the model effectively captures the general volatility patterns over the entire time series, with the predicted values ($\hat{y}$) closely following the observed average volatility. The sharp volatility spikes during the financial crisis and the pandemic are well-represented in the predictions, indicating the model's robustness in capturing extreme events. The confidence interval remains relatively narrow in the training period but begins to widen in the test period, reflecting increased uncertainty as the forecast horizon extends. The right plot zooms into the test period from 2020 to 2023, where the model continues to align well with observed values, though the confidence bands are wider compared to the five-day horizon. This indicates that while the model maintains good predictive accuracy for a ten-day forecast, the longer horizon introduces more uncertainty, particularly during volatile market conditions.

APPENDIX I

This appendix holds the tables for the hyper-parameters found for each model's experiment (1 and 2), horizon (1,5,10,20), and configuration (0, Firm, Tech, 1)

- Experiment 1

Table 30: Hyperparameters for BiTCN 0 in experiment 1 and horizon 1

Hyperparameters	**Options**
hidden_size	16
dropout	0.0015855562021411362
learning_rate	0.001
scaler_type	`robust`
max_steps	500
batch_size	128
windows_batch_size	1024

Table 31: Hyperparameters for BiTCN 1 in experiment 1 and horizon 1

Hyperparameters	**Options**
hidden_size	32
dropout	0.2931686241980131
learning_rate	0.01
scaler_type	`standard`
max_steps	1000
batch_size	128
windows_batch_size	1024

Table 32: Hyperparameters for NHITS 0 in experiment 1 and horizon 1

Hyperparameters	**Options**
n_pool_kernel_size	(1, 1, 1)
n_freq_downsample	(180, 60, 1)
learning_rate	0.0005
scaler_type	`robust`
max_steps	1500.0
batch_size	64
windows_batch_size	1024

Table 33: Hyperparameters for NHITS 1 in experiment 1 and horizon 1

Hyperparameters	**Options**
n_pool_kernel_size	(1, 1, 1)
n_freq_downsample	(24, 12, 1)
learning_rate	0.0005
scaler_type	`standard`
max_steps	800.0
batch_size	128
windows_batch_size	1024

Table 34: Hyperparameters for TFT 0 in experiment 1 and horizon 1

Hyperparameters	**Options**
hidden_size	256
n_head	4
learning_rate	5e-05
scaler_type	`standard`
max_steps	500
batch_size	64
windows_batch_size	512

Table 35: Hyperparameters for TFT 1 in experiment 1 and horizon 1

Hyperparameters	**Options**
hidden_size	128
n_head	4
learning_rate	0.0001
scaler_type	`standard`
max_steps	2000
batch_size	64
windows_batch_size	1024

Table 36: Hyperparameters for TiDE 0 in experiment 1 and horizon 1

Hyperparameters	**Options**
hidden_size	512
decoder_output_dim	16
temporal_decoder_dim	32
num_encoder_layers	3
num_decoder_layers	2
temporal_width	8
dropout	0.2
layernorm	False
learning_rate	5e-05
scaler_type	`standard`
max_steps	1200.0
batch_size	256
windows_batch_size	1024

Table 37: Hyperparameters for TiDE 1 in experiment 1 and horizon 1

Hyperparameters	**Options**
hidden_size	1024
decoder_output_dim	16
temporal_decoder_dim	64
num_encoder_layers	3
num_decoder_layers	3
temporal_width	8
dropout	0.1
layernorm	False
learning_rate	0.0005
scaler_type	`standard`
max_steps	1200.0
batch_size	128
windows_batch_size	512

Table 38: Hyperparameters for BiTCN 0 in experiment 1 and horizon 5

Hyperparameters	**Options**
hidden_size	32
dropout	0.0030832791003590487
learning_rate	0.005
scaler_type	`standard`
max_steps	1000
batch_size	256
windows_batch_size	1024

Table 39: Hyperparameters for BiTCN 1 in experiment 1 and horizon 5

Hyperparameters	**Options**
hidden_size	16
dropout	0.045104628910584615
learning_rate	0.0005
scaler_type	`standard`
max_steps	2000
batch_size	128
windows_batch_size	1024

Table 40: Hyperparameters for NHITS 0 in experiment 1 and horizon 5

Hyperparameters	**Options**
n_pool_kernel_size	(1, 1, 1)
n_freq_downsample	(180, 60, 1)
learning_rate	0.0005
scaler_type	`standard`
max_steps	1100.0
batch_size	128
windows_batch_size	1024

Table 41: Hyperparameters for NHITS 1 in experiment 1 and horizon 5

Hyperparameters	**Options**
n_pool_kernel_size	(4, 4, 4)
n_freq_downsample	(60, 8, 1)
learning_rate	0.001
scaler_type	`standard`
max_steps	1400.0
batch_size	128
windows_batch_size	256

Table 42: Hyperparameters for TFT 0 in experiment 1 and horizon 5

Hyperparameters	**Options**
hidden_size	256
n_head	8
learning_rate	0.0005
scaler_type	`standard`
max_steps	1000
batch_size	128
windows_batch_size	256

Table 43: Hyperparameters for TFT 1 in experiment 1 and horizon 5

Hyperparameters	**Options**
hidden_size	128
n_head	4
learning_rate	0.005
scaler_type	`robust`
max_steps	2000
batch_size	128
windows_batch_size	1024

Table 44: Hyperparameters for TiDE 0 in experiment 1 and horizon 5

Hyperparameters	**Options**
hidden_size	512
decoder_output_dim	8
temporal_decoder_dim	32
num_encoder_layers	3
num_decoder_layers	2
temporal_width	8
dropout	0.0
layernorm	False
learning_rate	0.0001
scaler_type	`robust`
max_steps	1000.0
batch_size	128
windows_batch_size	512

Table 45: Hyperparameters for TiDE 1 in experiment 1 and horizon 5

Hyperparameters	**Options**
hidden_size	1024
decoder_output_dim	32
temporal_decoder_dim	64
num_encoder_layers	2
num_decoder_layers	2
temporal_width	8
dropout	0.1
layernorm	False
learning_rate	0.0005
scaler_type	`standard`
max_steps	1100.0
batch_size	128
windows_batch_size	1024

Table 46: Hyperparameters for BiTCN 0 in experiment 1 and horizon 10

Hyperparameters	**Options**
hidden_size	32
dropout	0.006805117506399672
learning_rate	0.001
scaler_type	`standard`
max_steps	2000
batch_size	64
windows_batch_size	1024

Table 47: Hyperparameters for BiTCN 1 in experiment 1 and horizon 10

Hyperparameters	**Options**
hidden_size	16
dropout	0.028560683052457517
learning_rate	0.005
scaler_type	`robust`
max_steps	2000
batch_size	128
windows_batch_size	512

Table 48: Hyperparameters for NHITS 0 in experiment 1 and horizon 10

Hyperparameters	**Options**
n_pool_kernel_size	(2, 2, 2)
n_freq_downsample	(1, 1, 1)
learning_rate	0.0005
scaler_type	`robust`
max_steps	1400.0
batch_size	128
windows_batch_size	256

Table 49: Hyperparameters for NHITS 1 in experiment 1 and horizon 10

Hyperparameters	**Options**
n_pool_kernel_size	(2, 2, 1)
n_freq_downsample	(60, 8, 1)
learning_rate	0.001
scaler_type	`standard`
max_steps	1400.0
batch_size	32
windows_batch_size	512

Table 50: Hyperparameters for TFT 0 in experiment 1 and horizon 10

Hyperparameters	**Options**
hidden_size	256
n_head	8
learning_rate	0.0005
scaler_type	`standard`
max_steps	2000
batch_size	128
windows_batch_size	1024

Table 51: Hyperparameters for TFT 1 in experiment 1 and horizon 10

Hyperparameters	Options
hidden_size	256
n_head	8
learning_rate	5e-05
scaler_type	`standard`
max_steps	2000
batch_size	32
windows_batch_size	256

Table 52: Hyperparameters for TiDE 0 in experiment 1 and horizon 10

Hyperparameters	Options
hidden_size	256
decoder_output_dim	16
temporal_decoder_dim	128
num_encoder_layers	2
num_decoder_layers	3
temporal_width	16
dropout	0.3
layernorm	False
learning_rate	0.0005
scaler_type	`robust`
max_steps	1300.0
batch_size	32
windows_batch_size	1024

Table 53: Hyperparameters for TiDE 1 in experiment 1 and horizon 10

Hyperparameters	**Options**
hidden_size	1024
decoder_output_dim	32
temporal_decoder_dim	128
num_encoder_layers	3
num_decoder_layers	3
temporal_width	4
dropout	0.3
layernorm	False
learning_rate	5e-05
scaler_type	`standard`
max_steps	1000.0
batch_size	32
windows_batch_size	512

Table 54: Hyperparameters for BiTCN 0 in experiment 1 and horizon 20

Hyperparameters	**Options**
hidden_size	16
dropout	0.004944255443348664
learning_rate	0.01
scaler_type	`standard`
max_steps	2000
batch_size	64
windows_batch_size	128

Table 55: Hyperparameters for BiTCN 1 in experiment 1 and horizon 20

Hyperparameters	**Options**
hidden_size	32
dropout	0.09502678547677132
learning_rate	0.001
scaler_type	`robust`
max_steps	1000
batch_size	64
windows_batch_size	256

Table 56: Hyperparameters for NHITS 0 in experiment 1 and horizon 20

Hyperparameters	**Options**
n_pool_kernel_size	(8, 4, 1)
n_freq_downsample	(168, 24, 1)
learning_rate	0.005
scaler_type	`standard`
max_steps	1400.0
batch_size	64
windows_batch_size	512

Table 57: Hyperparameters for NHITS 1 in experiment 1 and horizon 20

Hyperparameters	**Options**
n_pool_kernel_size	(16, 8, 1)
n_freq_downsample	(168, 24, 1)
learning_rate	0.001
scaler_type	`standard`
max_steps	900.0
batch_size	64
windows_batch_size	1024

Table 58: Hyperparameters for TFT 0 in experiment 1 and horizon 20

Hyperparameters	**Options**
hidden_size	256
n_head	8
learning_rate	0.0001
scaler_type	`standard`
max_steps	2000
batch_size	128
windows_batch_size	256

Table 59: Hyperparameters for TFT 1 in experiment 1 and horizon 20

Hyperparameters	Options
hidden_size	64
n_head	8
learning_rate	0.0005
scaler_type	`standard`
max_steps	2000
batch_size	64
windows_batch_size	512

Table 60: Hyperparameters for TiDE 0 in experiment 1 and horizon 20

Hyperparameters	Options
hidden_size	512
decoder_output_dim	32
temporal_decoder_dim	64
num_encoder_layers	1
num_decoder_layers	2
temporal_width	16
dropout	0.2
layernorm	False
learning_rate	0.0001
scaler_type	`standard`
max_steps	1500.0
batch_size	64
windows_batch_size	1024

Table 61: Hyperparameters for TiDE 1 in experiment 1 and horizon 20

Hyperparameters	Options
hidden_size	256
decoder_output_dim	8
temporal_decoder_dim	128
num_encoder_layers	1
num_decoder_layers	1
temporal_width	16
dropout	0.3
layernorm	True
learning_rate	0.01
scaler_type	`robust`
max_steps	1400.0
batch_size	32
windows_batch_size	1024

- Experiment 2

Table 62: Hyperparameters for BiTCN 0 in experiment 2 and horizon 1

Hyperparameters	Options
hidden_size	16
dropout	0.1982596820620405
learning_rate	0.005
scaler_type	`robust`
max_steps	2000
batch_size	64
windows_batch_size	128

Table 63: Hyperparameters for BiTCN 1 in experiment 2 and horizon 1

Hyperparameters	Options
hidden_size	32
dropout	0.16663122629381397
learning_rate	0.001
scaler_type	`robust`
max_steps	500
batch_size	32
windows_batch_size	512

Table 64: Hyperparameters for NHITS 0 in experiment 2 and horizon 1

Hyperparameters	Options
n_pool_kernel_size	(2, 2, 1)
n_freq_downsample	(24, 12, 1)
learning_rate	0.0005
scaler_type	`robust`
max_steps	900.0
batch_size	128
windows_batch_size	1024

Table 65: Hyperparameters for NHITS 1 in experiment 2 and horizon 1

Hyperparameters	Options
n_pool_kernel_size	(1, 1, 1)
n_freq_downsample	(60, 8, 1)
learning_rate	0.001
scaler_type	`standard`
max_steps	1500.0
batch_size	128
windows_batch_size	512

Table 66: Hyperparameters for TFT 0 in experiment 2 and horizon 1

Hyperparameters	Options
hidden_size	256
n_head	8
learning_rate	5e-05
scaler_type	`standard`
max_steps	1000
batch_size	64
windows_batch_size	1024

Table 67: Hyperparameters for TFT 1 in experiment 2 and horizon 1

Hyperparameters	Options
hidden_size	256
n_head	4
learning_rate	0.0005
scaler_type	standard
max_steps	1000
batch_size	128
windows_batch_size	1024

Table 68: Hyperparameters for TiDE 0 in experiment 2 and horizon 1

Hyperparameters	Options
hidden_size	1024
decoder_output_dim	32
temporal_decoder_dim	64
num_encoder_layers	1
num_decoder_layers	2
temporal_width	16
dropout	0.5
layernorm	False
learning_rate	0.0001
scaler_type	robust
max_steps	600.0
batch_size	64
windows_batch_size	1024

Table 69: Hyperparameters for TiDE 1 in experiment 2 and horizon 1

Hyperparameters	**Options**
hidden_size	512
decoder_output_dim	8
temporal_decoder_dim	64
num_encoder_layers	1
num_decoder_layers	1
temporal_width	4
dropout	0.0
layernorm	False
learning_rate	0.0001
scaler_type	`standard`
max_steps	1400.0
batch_size	64
windows_batch_size	512

Table 70: Hyperparameters for BiTCN 0 in experiment 2 and horizon 5

Hyperparameters	**Options**
hidden_size	16
dropout	0.00020318441183 98924
learning_rate	0.005
scaler_type	`standard`
max_steps	1000
batch_size	32
windows_batch_size	512

Table 71: Hyperparameters for BiTCN 1 in experiment 2 and horizon 5

Hyperparameters	**Options**
hidden_size	32
dropout	0.07229450701828932
learning_rate	0.001
scaler_type	`standard`
max_steps	1000
batch_size	32
windows_batch_size	1024

Table 72: Hyperparameters for NHITS 0 in experiment 2 and horizon 5

Hyperparameters	**Options**
n_pool_kernel_size	(16, 8, 1)
n_freq_downsample	(180, 60, 1)
learning_rate	0.001
scaler_type	`robust`
max_steps	1300.0
batch_size	64
windows_batch_size	1024

Table 73: Hyperparameters for NHITS 1 in experiment 2 and horizon 5

Hyperparameters	**Options**
n_pool_kernel_size	(2, 2, 1)
n_freq_downsample	(1, 1, 1)
learning_rate	0.0005
scaler_type	`standard`
max_steps	700.0
batch_size	32
windows_batch_size	256

Table 74: Hyperparameters for TFT 0 in experiment 2 and horizon 5

Hyperparameters	**Options**
hidden_size	64
n_head	4
learning_rate	0.005
scaler_type	`robust`
max_steps	2000
batch_size	128
windows_batch_size	1024

Table 75: Hyperparameters for TFT 1 in experiment 2 and horizon 5

Hyperparameters	**Options**
hidden_size	128
n_head	4
learning_rate	0.0005
scaler_type	`robust`
max_steps	2000
batch_size	128
windows_batch_size	1024

Table 76: Hyperparameters for TiDE 0 in experiment 2 and horizon 5

Hyperparameters	**Options**
hidden_size	512
decoder_output_dim	32
temporal_decoder_dim	32
num_encoder_layers	1
num_decoder_layers	3
temporal_width	4
dropout	0.1
layernorm	False
learning_rate	0.0005
scaler_type	`standard`
max_steps	1500.0
batch_size	128
windows_batch_size	1024

Table 77: Hyperparameters for TiDE 1 in experiment 2 and horizon 5

Hyperparameters	**Options**
hidden_size	1024
decoder_output_dim	16
temporal_decoder_dim	32
num_encoder_layers	1
num_decoder_layers	2
temporal_width	8
dropout	0.2
layernorm	False
learning_rate	0.0005
scaler_type	`standard`
max_steps	1500.0
batch_size	128
windows_batch_size	1024

Table 78: Hyperparameters for BiTCN 0 in experiment 2 and horizon 10

Hyperparameters	**Options**
hidden_size	32
dropout	0.004140835965901486
learning_rate	0.005
scaler_type	`standard`
max_steps	500
batch_size	128
windows_batch_size	128

Table 79: Hyperparameters for BiTCN 1 in experiment 2 and horizon 10

Hyperparameters	**Options**
hidden_size	16
dropout	0.0021307631125278004
learning_rate	0.005
scaler_type	`robust`
max_steps	500
batch_size	128
windows_batch_size	512

Table 80: Hyperparameters for NHITS 0 in experiment 2 and horizon 10

Hyperparameters	Options
n_pool_kernel_size	(8, 4, 1)
n_freq_downsample	(24, 12, 1)
learning_rate	0.001
scaler_type	`standard`
max_steps	1100.0
batch_size	128
windows_batch_size	1024

Table 81: Hyperparameters for NHITS 1 in experiment 2 and horizon 10

Hyperparameters	Options
n_pool_kernel_size	(16, 8, 1)
n_freq_downsample	(180, 60, 1)
learning_rate	0.0005
scaler_type	`standard`
max_steps	1500.0
batch_size	32
windows_batch_size	1024

Table 82: Hyperparameters for TFT 0 in experiment 2 and horizon 10

Hyperparameters	Options
hidden_size	128
n_head	8
learning_rate	0.001
scaler_type	`robust`
max_steps	2000
batch_size	64
windows_batch_size	1024

Table 83: Hyperparameters for TFT 1 in experiment 2 and horizon 10

Hyperparameters	**Options**
hidden_size	128
n_head	8
learning_rate	0.005
scaler_type	`standard`
max_steps	2000
batch_size	32
windows_batch_size	512

Table 84: Hyperparameters for TiDE 0 in experiment 2 and horizon 10

Hyperparameters	**Options**
hidden_size	256
decoder_output_dim	16
temporal_decoder_dim	128
num_encoder_layers	3
num_decoder_layers	3
temporal_width	8
dropout	0.3
layernorm	False
learning_rate	0.0005
scaler_type	`standard`
max_steps	1200.0
batch_size	32
windows_batch_size	1024

Table 85: Hyperparameters for TiDE 1 in experiment 2 and horizon 10

Hyperparameters	**Options**
hidden_size	1024
decoder_output_dim	32
temporal_decoder_dim	64
num_encoder_layers	1
num_decoder_layers	3
temporal_width	8
dropout	0.2
layernorm	False
learning_rate	0.0005
scaler_type	`standard`
max_steps	1500.0
batch_size	64
windows_batch_size	1024

Table 86: Hyperparameters for BiTCN 0 in experiment 2 and horizon 20

Hyperparameters	**Options**
hidden_size	32
dropout	0.0615805554892364345
learning_rate	0.005
scaler_type	`standard`
max_steps	2000
batch_size	128
windows_batch_size	256

Table 87: Hyperparameters for BiTCN 1 in experiment 2 and horizon 20

Hyperparameters	**Options**
hidden_size	32
dropout	0.01159221611565845
learning_rate	0.001
scaler_type	`robust`
max_steps	2000
batch_size	128
windows_batch_size	512

Table 88: Hyperparameters for NHITS 0 in experiment 2 and horizon 20

Hyperparameters	**Options**
n_pool_kernel_size	(16, 8, 1)
n_freq_downsample	(60, 8, 1)
learning_rate	0.0005
scaler_type	`standard`
max_steps	800.0
batch_size	64
windows_batch_size	256

Table 89: Hyperparameters for NHITS 1 in experiment 2 and horizon 20

Hyperparameters	**Options**
n_pool_kernel_size	(2, 2, 2)
n_freq_downsample	(168, 24, 1)
learning_rate	0.0005
scaler_type	`standard`
max_steps	1500.0
batch_size	32
windows_batch_size	512

Table 90: Hyperparameters for TFT 0 in experiment 2 and horizon 20

Hyperparameters	**Options**
hidden_size	64
n_head	8
learning_rate	0.005
scaler_type	`standard`
max_steps	1000
batch_size	128
windows_batch_size	1024

Table 91: Hyperparameters for TFT 1 in experiment 2 and horizon 20

Hyperparameters	Options
hidden_size	256
n_head	8
learning_rate	0.01
scaler_type	`standard`
max_steps	2000
batch_size	128
windows_batch_size	1024

Table 92: Hyperparameters for TiDE 0 in experiment 2 and horizon 20

Hyperparameters	Options
hidden_size	1024
decoder_output_dim	16
temporal_decoder_dim	64
num_encoder_layers	1
num_decoder_layers	1
temporal_width	4
dropout	0.5
layernorm	False
learning_rate	0.001
scaler_type	`robust`
max_steps	1100.0
batch_size	64
windows_batch_size	1024

Table 93: Hyperparameters for TiDE 1 in experiment 2 and horizon 20

Hyperparameters	Options
hidden_size	1024
decoder_output_dim	32
temporal_decoder_dim	32
num_encoder_layers	2
num_decoder_layers	1
temporal_width	16
dropout	0.0
layernorm	False
learning_rate	0.0005
scaler_type	standard
max_steps	1100.0
batch_size	128
windows_batch_size	128

- Firm Indicators

Table 94: Hyperparameters for BiTCN with firm indicators only and horizon 1

Hyperparameters	Options
hidden_size	16
dropout	0.2862123404202475
learning_rate	0.001
scaler_type	robust
max_steps	2000
batch_size	128
windows_batch_size	1024

Table 95: Hyperparameters for NHITS with firm indicators only and horizon 1

Hyperparameters	Options
n_pool_kernel_size	(1, 1, 1)
n_freq_downsample	(40, 20, 1)
learning_rate	0.0005
scaler_type	standard
max_steps	1500.0
batch_size	128
windows_batch_size	512

Table 96: Hyperparameters for TFT with firm indicators only and horizon 1

Hyperparameters	Options
hidden_size	128
n_head	8
learning_rate	0.005
scaler_type	`standard`
max_steps	2000
batch_size	128
windows_batch_size	1024

Table 97: Hyperparameters for TiDE with firm indicators only and horizon 1

Hyperparameters	Options
hidden_size	512
decoder_output_dim	32
temporal_decoder_dim	32
num_encoder_layers	1
num_decoder_layers	1
temporal_width	16
dropout	0.1
layernorm	False
learning_rate	0.001
scaler_type	`standard`
max_steps	700.0
batch_size	128
windows_batch_size	1024

Table 98: Hyperparameters for BiTCN with firm indicators only and horizon 5

Hyperparameters	Options
hidden_size	16
dropout	0.052394792456435774
learning_rate	0.01
scaler_type	`standard`
max_steps	500
batch_size	64
windows_batch_size	1024

Table 99: Hyperparameters for NHITS with firm indicators only and horizon 5

Hyperparameters	Options
n_pool_kernel_size	(16, 8, 1)
n_freq_downsample	(180, 60, 1)
learning_rate	0.0005
scaler_type	`standard`
max_steps	1500.0
batch_size	128
windows_batch_size	512

Table 100: Hyperparameters for TFT with firm indicators only and horizon 5

Hyperparameters	Options
hidden_size	64
n_head	8
learning_rate	0.01
scaler_type	`standard`
max_steps	1000
batch_size	64
windows_batch_size	1024

Table 101: Hyperparameters for TiDE with firm indicators only and horizon 5

Hyperparameters	Options
hidden_size	512
decoder_output_dim	16
temporal_decoder_dim	32
num_encoder_layers	2
num_decoder_layers	1
temporal_width	4
dropout	0.5
layernorm	False
learning_rate	0.0005
scaler_type	`standard`
max_steps	1200.0
batch_size	128
windows_batch_size	1024

Table 102: Hyperparameters for BiTCN with firm indicators only and horizon 10

Hyperparameters	**Options**
hidden_size	32
dropout	0.01253507603673587
learning_rate	0.01
scaler_type	`standard`
max_steps	500
batch_size	256
windows_batch_size	512

Table 103: Hyperparameters for NHITS with firm indicators only and horizon 10

Hyperparameters	**Options**
n_pool_kernel_size	(2, 2, 1)
n_freq_downsample	(40, 20, 1)
learning_rate	0.001
scaler_type	`standard`
max_steps	1100.0
batch_size	128
windows_batch_size	512

Table 104: Hyperparameters for TFT with firm indicators only and horizon 10

Hyperparameters	**Options**
hidden_size	64
n_head	8
learning_rate	0.001
scaler_type	`standard`
max_steps	2000
batch_size	128
windows_batch_size	512

Table 105: Hyperparameters for TiDE with firm indicators only and horizon 10

Hyperparameters	**Options**
hidden_size	256
decoder_output_dim	32
temporal_decoder_dim	32
num_encoder_layers	2
num_decoder_layers	3
temporal_width	8
dropout	0.1
layernorm	False
learning_rate	0.005
scaler_type	`standard`
max_steps	900.0
batch_size	128
windows_batch_size	1024

Table 106: Hyperparameters for BiTCN with firm indicators only and horizon 20

Hyperparameters	**Options**
hidden_size	16
dropout	0.20918739692977562
learning_rate	0.01
scaler_type	`standard`
max_steps	500
batch_size	128
windows_batch_size	1024

Table 107: Hyperparameters for NHITS with firm indicators only and horizon 20

Hyperparameters	**Options**
n_pool_kernel_size	(4, 4, 4)
n_freq_downsample	(180, 60, 1)
learning_rate	0.01
scaler_type	`standard`
max_steps	1200.0
batch_size	64
windows_batch_size	256

Table 108: Hyperparameters for TFT with firm indicators only and horizon 20

Hyperparameters	Options
hidden_size	256
n_head	4
learning_rate	0.0005
scaler_type	`standard`
max_steps	2000
batch_size	128
windows_batch_size	1024

Table 109: Hyperparameters for TiDE with firm indicators only and horizon 20

Hyperparameters	Options
hidden_size	512
decoder_output_dim	16
temporal_decoder_dim	128
num_encoder_layers	1
num_decoder_layers	1
temporal_width	4
dropout	0.5
layernorm	True
learning_rate	0.01
scaler_type	`standard`
max_steps	1100.0
batch_size	32
windows_batch_size	256

- Technical Indicators

Table 110: Hyperparameters for BiTCN with technical indicators only and horizon 1

Hyperparameters	Options
hidden_size	16
dropout	0.4954990474438764
learning_rate	0.005
scaler_type	`standard`
max_steps	500
batch_size	64
windows_batch_size	128

Table 111: Hyperparameters for NHITS with technical indicators only and horizon 1

Hyperparameters	Options
n_pool_kernel_size	(2, 2, 1)
n_freq_downsample	(24, 12, 1)
learning_rate	0.001
scaler_type	`standard`
max_steps	1400.0
batch_size	32
windows_batch_size	1024

Table 112: Hyperparameters for TFT with technical indicators only and horizon 1

Hyperparameters	Options
hidden_size	256
n_head	8
learning_rate	0.0001
scaler_type	`standard`
max_steps	2000
batch_size	64
windows_batch_size	1024

Table 113: Hyperparameters for TiDE with technical indicators only and horizon 1

Hyperparameters	Options
hidden_size	1024
decoder_output_dim	16
temporal_decoder_dim	128
num_encoder_layers	2
num_decoder_layers	3
temporal_width	8
dropout	0.1
layernorm	False
learning_rate	0.0005
scaler_type	`standard`
max_steps	1300.0
batch_size	64
windows_batch_size	1024

Table 114: Hyperparameters for BiTCN with technical indicators only and horizon 5

Hyperparameters	Options
hidden_size	32
dropout	0.1602496393228111
learning_rate	0.005
scaler_type	`standard`
max_steps	2000
batch_size	128
windows_batch_size	128

Table 115: Hyperparameters for NHITS with technical indicators only and horizon 5

Hyperparameters	Options
n_pool_kernel_size	(2, 2, 1)
n_freq_downsample	(40, 20, 1)
learning_rate	0.0005
scaler_type	`robust`
max_steps	1000.0
batch_size	128
windows_batch_size	1024

Table 116: Hyperparameters for TFT with technical indicators only and horizon 5

Hyperparameters	Options
hidden_size	256
n_head	8
learning_rate	0.001
scaler_type	`standard`
max_steps	1000
batch_size	128
windows_batch_size	1024

Table 117: Hyperparameters for TiDE with technical indicators only and horizon 5

Hyperparameters	Options
hidden_size	256
decoder_output_dim	8
temporal_decoder_dim	64
num_encoder_layers	3
num_decoder_layers	3
temporal_width	4
dropout	0.1
layernorm	False
learning_rate	0.0001
scaler_type	`standard`
max_steps	1400.0
batch_size	128
windows_batch_size	512

Table 118: Hyperparameters for BiTCN with technical indicators only and horizon 10

Hyperparameters	Options
hidden_size	16
dropout	0.045713963345932 56
learning_rate	0.005
scaler_type	`robust`
max_steps	1000
batch_size	64
windows_batch_size	256

Table 119: Hyperparameters for NHITS with technical indicators only and horizon 10

Hyperparameters	Options
n_pool_kernel_size	(4, 4, 4)
n_freq_downsample	(40, 20, 1)
learning_rate	0.005
scaler_type	`standard`
max_steps	1400.0
batch_size	128
windows_batch_size	1024

Table 120: Hyperparameters for TFT with technical indicators only and horizon 10

Hyperparameters	**Options**
hidden_size	128
n_head	8
learning_rate	0.0005
scaler_type	`robust`
max_steps	2000
batch_size	128
windows_batch_size	512

Table 121: Hyperparameters for TiDE with technical indicators only and horizon 10

Hyperparameters	**Options**
hidden_size	1024
decoder_output_dim	32
temporal_decoder_dim	64
num_encoder_layers	2
num_decoder_layers	2
temporal_width	8
dropout	0.5
layernorm	False
learning_rate	0.001
scaler_type	`robust`
max_steps	1100.0
batch_size	128
windows_batch_size	512

Table 122: Hyperparameters for BiTCN with technical indicators only and horizon 20

Hyperparameters	**Options**
hidden_size	32
dropout	0.027992879831349834
learning_rate	0.0005
scaler_type	`standard`
max_steps	2000
batch_size	128
windows_batch_size	512

Table 123: Hyperparameters for NHITS with technical indicators only and horizon 20

Hyperparameters	**Options**
n_pool_kernel_size	(2, 2, 2)
n_freq_downsample	(180, 60, 1)
learning_rate	0.001
scaler_type	`robust`
max_steps	1400.0
batch_size	64
windows_batch_size	1024

Table 124: Hyperparameters for TFT with technical indicators only and horizon 20

Hyperparameters	**Options**
hidden_size	128
n_head	4
learning_rate	0.005
scaler_type	`robust`
max_steps	2000
batch_size	256
windows_batch_size	512

Table 125: Hyperparameters for TiDE with technical indicators only and horizon 20

Hyperparameters	**Options**
hidden_size	512
decoder_output_dim	8
temporal_decoder_dim	32
num_encoder_layers	1
num_decoder_layers	3
temporal_width	4
dropout	0.3
layernorm	False
learning_rate	0.0001
scaler_type	`standard`
max_steps	700.0
batch_size	128
windows_batch_size	512