Volatility Forecasting with Neural Networks for Portfolio Optimization of cryptocurrencies indexes

Lina Benzemma, Emma Eberle, Sharon Chemmama

January 2025

Abstract

This study explores a portfolio optimization framework that integrates advanced risk management methodologies with volatility forecasting, applied to a diverse cryptocurrency portfolio. We evaluate the predictive performance of neural networks and conduct a rigorous benchmark of various forecasting models. These predictions are then employed to implement a risk-parity portfolio strategy, providing insights into its effectiveness in enhancing risk-adjusted returns

Keywords : Portfolio Optimization, Risk Management, Volatility Forecasting, Neural Networks, Cryptocurrency Investments, Machine Learning, Asset Allocation.

1 Introduction

Cryptocurrencies have emerged as a transformative force in the global financial system, offering decentralized digital alternatives to traditional currencies and assets. These assets are characterized by their high levels of volatility, which makes them attractive for speculative trading but challenging for risk management, portfolio construction, and forecasting. Understanding and forecasting cryptocurrency volatility is crucial for investors and portfolio managers aiming to optimize returns while mitigating risk.

Given the inherent unpredictability and nonlinear dynamics of cryptocurrency markets, traditional methods to forecast volatility often fail. To address these challenges, we employ various neural networks models and conduct a comparative analysis to identify the most accurate and robust estimator for future volatility. Our goal is to produce reliable forecasts that effectively capture the intricate complexities and evolving patterns of cryptocurrency markets.

Our approach involves testing various features as additional predictors of volatility. To this end, we will incorporate different features to evaluate their predictive capabilities. We will also propose exploring a hybrid approach by incorporating GARCH volatility predictions into the input features of neural network models. This integration aims to combine the strengths of traditional statistical methods with those of deep learning techniques, fostering synergy between established approaches and modern computational frameworks.

Building on these volatility forecasts, we implement a portfolio optimization framework that leverages various strategies informed by the predicted volatility. A comparative analysis of the resulting portfolio performances will be conducted, focusing on returns to evaluate the effectiveness of each approach.

2 Forecasting Methodology

2.1 Time Series Creation

In this article, we analyze a multi-asset portfolio composed of N equities, indexed by $i = \{1, \ldots, N\}$, observed at different time steps t (in days).

We utilize the daily closing prices of the three cryptocurrency indices as the price at each time step t. These prices are then transformed into **logarithmic returns**. The log returns of the ith asset are calculated as a time series $r_{i,t}$ by:

$$r_t^i = \ln\left(\frac{P_t^i}{P_{t-1}^i}\right) \tag{1}$$

with P_t^i value of daily closing price evaluated of equity i at time t.

In the context of cryptocurrencies, the use of log returns is particularly relevant due to the high volatility and frequent price swings. It allows for a more robust analysis of returns while mitigating the influence of outliers.

We next computed the **rolling volatility** of the log returns for the i-th asset, defined as the standard deviation of the returns over a moving window of a fixed size. The rolling volatility at time t is given by:

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

with N as the size of the rolling window and $\bar{r}_t$ as the mean of the log returns over the time window. Throughout this article, N is set to 30, corresponding to a 30-day period. This choice aligns with the calculation of standard deviations based on the monthly open market in the cryptocurrency sector.

The rolling volatility provides a dynamic measure of market risk and highlights periods of increased uncertainty. In cryptocurrency markets, where prices are prone to sharp fluctuations, monitoring volatility is essential for understanding the risk-return trade-off.

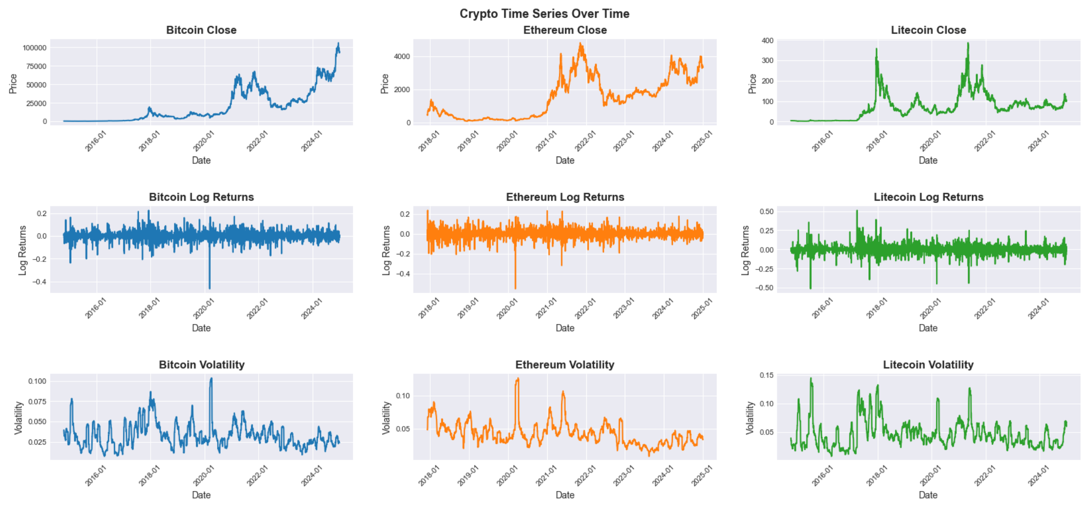


Figure 1: Time Series of the Three selected Cryptocurrencies

2.2 Feature creation

To enhance the accuracy of our volatility prediction models, we will incorporate additional time-series features as input variables. These features will be aligned with historical volatility as the input features to capture relevant patterns and dependencies, ultimately improving the model's forecasting capability.

2.2.1 GARCH(1,1) Predictions

The Generalized Autoregressive Conditional Heteroskedasticity (GARCH) model is widely used to estimate and predict time-varying volatility in financial time series.

The GARCH(1,1) model is specified as:

$$\sigma_t^2 = \omega + \alpha \epsilon_{t-1}^2 + \beta \sigma_{t-1}^2 \tag{3}$$

where σ_t^2 is the conditional variance at time t, ω is a constant term capturing the long-term volatility level, α measures the impact of past squared returns (ARCH effect),

β captures the persistence of past volatility (GARCH effect), and ϵ_{t-1} represents past innovations in returns.

To forecast future volatility, the model uses the estimated parameters $\hat{\omega}$, $\hat{\alpha}$, and $\hat{\beta}$ along with historical return shocks and past conditional volatility. The one-step-ahead forecast for conditional volatility is given by:

$$\hat{\sigma}_{t+1} = \sqrt{\hat{\omega} + \hat{\alpha}\epsilon_t^2 + \hat{\beta}\sigma_t^2} \tag{4}$$

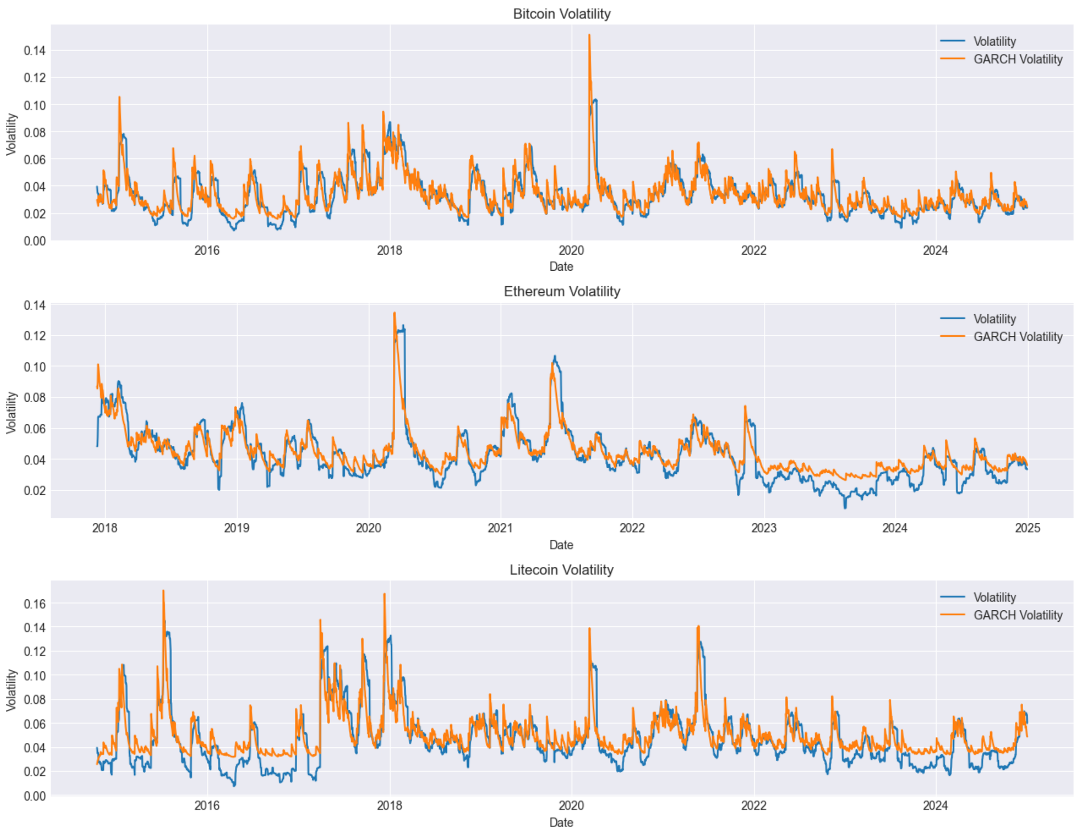


Figure 2: Garch(1,1) Volatility Prediction

As shown in the graph, the forecasts generated by GARCH(1,1) only exhibit low accuracy, despite effectively capturing high variance peaks and corresponding volatility clusters.

2.2.2 Others features

In addition to GARCH-based volatility estimates and historical volatility, several features can be extracted to enhance financial time series analysis and forecasting. These features capture different aspects of market dynamics, such as smoothed volatility trends, price range fluctuations, and trading activity variations.

- To capture longer-term volatility patterns, moving averages of volatility over different time windows are computed. Specifically, weekly and monthly smoothed volatilities are obtained using rolling averages.

- The log trading range is used to measure the relative intraday price fluctuation and is computed as:

$$LogTradingRange_t = \log(High_t) - \log(Low_t), \tag{5}$$

  where $High_t$ and Low_t represent the highest and lowest prices observed on day t. This feature captures the amplitude of daily price movement, which can be indicative of market uncertainty and potential reversals.

- Trading volume plays a crucial role in market dynamics, reflecting liquidity and investor participation. The log volume change is computed as:

$$LogVolumeChange_t = \log(Volume_t) - \log(Volume_{t-1}), \tag{6}$$

  where $Volume_t$ denotes the total traded volume at time t. This feature highlights significant changes in trading activity, which often precede price movements and volatility shifts.

This enriched representation of market conditions enhances the predictive capabilities of models, particularly when used in machine learning and deep learning applications for financial forecasting.

2.3 The forecasting task

The primary objective of this forecasting task is to predict future market volatility over a short-term horizon. In this study, we aim to predict the aggregated volatility over the next 5 days, denoted as $y[t, t+5]$, using historical features extracted from a lookback window of 30 days, represented as $X[t-30, t-1]$. Mathematically, the forecasting model can be expressed as:

$$y[t, t+5] = f(X[t-30, t-1]), \tag{7}$$

where $f(\cdot)$ is a function approximated by the predictive model.

To maintain robustness and adaptability to changing market conditions, we employ a rolling window approach. At each time step t, the model uses the most recent 30 days of historical data to generate a forecast for the 5-day future volatility. This mechanism ensures that the model continuously updates its predictions based on the latest available information, adapting to evolving market dynamics.

The selection of the 30-day lookback window balances the need for capturing relevant past information while avoiding excessive lag that could introduce outdated patterns. A shorter window might not capture meaningful volatility trends, whereas a longer window could incorporate irrelevant or outdated information. The 5-day forecasting horizon is chosen to align with short-term portfolio rebalancing needs, providing actionable insights for risk-adjusted decision-making.

2.4 Neural Network Models

2.4.1 Recurrent Neural Networks (RNNs)

Recurrent Neural Networks (RNNs) are a class of artificial neural networks specifically designed to handle sequential and time-series data. Unlike traditional neural networks, RNNs maintain an internal memory that captures temporal dependencies by processing data sequentially, making them particularly effective for modeling dynamic patterns over time. The inherent ability of RNNs to learn non-linear and complex relationships in sequential datasets makes them well-suited for forecasting the volatility of cryptocurrencies, which exhibit high-frequency, non-linear price fluctuations and temporal dependencies. By leveraging historical data and learning temporal patterns, RNNs can provide robust and accurate predictions of market volatility, offering valuable insights for risk management and trading strategies.

2.4.2 Long Short-Term Memory (LSTM) Networks

Long Short-Term Memory (LSTM) networks are an advanced type of Recurrent Neural Network (RNN) designed to address the vanishing gradient problem in traditional RNNs. The hidden units in an LSTM are composed of a unique structure including cell states and gating mechanisms—input, output, and forget gates—that regulate the flow of information. This architecture enables LSTM hidden units to retain long-term dependencies while selectively forgetting irrelevant information, making them highly effective for learning from sequential data. By maintaining and updating cell states dynamically, LSTM hidden units allow for more accurate modeling of complex temporal patterns, such as the intricate dependencies and high volatility observed in cryptocurrency price movements.

2.4.3 Gated Recurrent Units (GRUs)

Gated Recurrent Units (GRUs) are a simplified variant of Long Short-Term Memory (LSTM) networks, to achieve similar performance with a more streamlined architecture. GRU hidden units combine the functionalities of the input, output, and forget gates found in LSTMs into two primary gates: the reset gate and the update gate. This reduction in complexity allows GRUs to be computationally efficient while maintaining the ability to capture long-term dependencies in sequential data. The reset gate controls how much past information is forgotten, while the update gate determines the extent to which new information is incorporated into the hidden state. GRU hidden units are particularly effective in modeling temporal patterns in time-series data, such as cryptocurrency price movements, where efficiency and the ability to learn dependencies over varying time scales are critical.

2.4.4 Temporal Convolutional Neural Networks (TCNNs)

Temporal Convolutional Neural Networks (TCNNs) are a class of deep learning architectures specifically designed for sequential and time-series data, offering a compelling alternative to Recurrent Neural Networks (RNNs). Unlike RNNs, TCNNs leverage convolutional layers with causal and dilated convolutions to capture temporal dependencies and hierarchical patterns over different time scales. The use of dilation enables TCNNs to expand their receptive field efficiently, allowing them to model long-term dependencies without the vanishing gradient problem inherent in RNN-based architectures. Additionally, TCNNs process data in parallel rather than sequentially, leading to faster training and inference. These features make TCNNs particularly suited for tasks such as forecasting cryptocurrency volatility, where capturing complex, non-linear, and long-range temporal patterns is crucial for accurate predictions.

2.5 Measuring Predictions Accuracy

To assess the forecasting performance on the test dataset, we use four widely recognized error metrics: Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE). These metrics provide a comprehensive evaluation of prediction accuracy by measuring both absolute and relative errors.

- **Mean Squared Error (MSE)**: Measures the average squared difference between the predicted values $\hat{y}_t$ and the actual values y_t. It penalizes larger errors more heavily, making it sensitive to outliers:

$$MSE = \frac{1}{N}\sum_{t=1}^{N}(y_t - \hat{y}_t)^2. \tag{8}$$

- **Root Mean Squared Error (RMSE)**: Defined as the square root of MSE, RMSE restores the error unit to the same scale as the original data, providing an interpretable measure of forecast accuracy:

$$RMSE = \sqrt{\frac{1}{N}\sum_{t=1}^{N}(y_t - \hat{y}_t)^2}. \tag{9}$$

- **Mean Absolute Error (MAE)**: Computes the average absolute difference between predictions and actual values, offering a more robust measure that does not overemphasize large errors:

$$MAE = \frac{1}{N}\sum_{t=1}^{N}|y_t - \hat{y}_t|. \tag{10}$$

- **Mean Absolute Percentage Error (MAPE)**: Expresses the forecasting error as a percentage of actual values, making it particularly useful for relative performance comparisons:

$$MAPE = \frac{100}{N}\sum_{t=1}^{N}\left|\frac{y_t - \hat{y}_t}{y_t}\right|. \tag{11}$$

By minimizing these error metrics on the test dataset, we ensure that the forecasting model provides reliable volatility predictions, which are essential for financial decision-making and risk management.

During the training phase of our neural networks, we use Mean Squared Error (MSE) as the loss function. MSE is a widely adopted loss metric for regression tasks because it penalizes large errors more heavily due to the squared term, encouraging the model to prioritize minimizing significant deviations. Additionally, MSE provides a smooth and differentiable objective, facilitating stable gradient-based optimization, which is crucial for efficient neural network training. By minimizing MSE, the model learns to generate accurate and stable volatility forecasts, aligning with our goal of precise risk estimation.

3 Data analysis and results

In this section, we present the numerical results obtained from our study. The primary goal of this work is to predict the volatility of three major cryptocurrencies: Bitcoin, Ethereum, and Litecoin. We use their daily closing prices as the foundational dataset for our analysis.

The data spans a time interval of 10 years, more precisely:

- **Training Set:** September 2014–December 2024 for the Bitcoin and Litecoin.September 2018–December 2024 for the Ethereum. The training process evaluates model performance across various configurations.
- **Test Set:** January 2024–January 2025, representing the evaluation period and historical simulation window for testing different portfolio management strategies.

3.1 General information about the dataset

In our study, we focus on a portfolio comprising three major cryptocurrencies that are representative of the digital asset market:

1. **Bitcoin (BTC):** Launched in 2009, Bitcoin is the first decentralized cryptocurrency and remains the most well-known and widely used digital asset. It operates on a peer-to-peer network using blockchain technology to ensure security and transparency. Bitcoin's value and adoption have grown significantly, making it a key indicator of the overall cryptocurrency market.
2. **Ethereum (ETH):** Introduced in 2015, Ethereum is a decentralized blockchain platform that enables the creation of smart contracts and decentralized applications (dApps). Its native cryptocurrency, Ether, is the second-largest by market capitalization. Ethereum's innovative features and widespread use cases make it an essential component of the cryptocurrency ecosystem.
3. **Litecoin (LTC):** Created in 2011 as a "lighter" alternative to Bitcoin, Litecoin features faster transaction confirmation times and a different hashing algorithm (Scrypt). It has maintained a steady presence in the market and is often used as a testing ground for new cryptocurrency technologies.

By analyzing the historical volatility of these cryptocurrencies, we aim to develop robust prediction models that capture their unique market dynamics and provide actionable insights for investors and portfolio managers.

3.2 Model Selection

3.2.1 Hyperparameter Optimization Using Grid Search

To improve the predictive accuracy and robustness of our models, we performed a **grid search** to fine-tune their hyperparameters. Since deep learning models are highly sensitive to hyperparameter choices, optimizing these parameters allows us to enhance both training stability and generalization performance.

For each model (LSTM, GRU, LSTM-GRU, and TCNN), we optimized the following hyperparameters:

- **Number of epochs:** Defines the number of training iterations.
- **Batch size:** Controls the number of samples per update.
- **Validation split:** Determines the proportion of data used for validation.
- **Dropout rate:** Prevents overfitting by randomly deactivating neurons.
- **Learning rate:** Adjusts the step size for weight updates.

The best hyperparameters were selected based on the lowest **Root Mean Squared Error (RMSE)** on the validation set. These optimal configurations were then used for final model training and evaluation.

3.2.2 Training and Testing Performance

To compare the models, we evaluated their predictive accuracy using the following key error metrics: **Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE)**.

Since the performance trends were consistent across all three cryptocurrencies (Bitcoin, Ethereum, Litecoin), we present detailed results for Bitcoin as a representative example. Full results for Ethereum and Litecoin are available in the appendix.

Model	**MSE**	**RMSE**	**MAE**	**MAPE (%)**
LSTM	0.0000	0.0058	0.0039	13.7460
GRU	0.0000	0.0054	0.0035	12.0199
LSTM-GRU	0.0000	0.0052	0.0033	11.2199
TCNN	0.0000	0.0048	0.0028	8.7078

Table 1: Training performance metrics for different models on Bitcoin dataset.

Model	**MSE**	**RMSE**	**MAE**	**MAPE (%)**
LSTM	0.0000	0.0034	0.0028	11.0630
GRU	0.0000	0.0033	0.0027	10.7214
LSTM-GRU	0.0000	0.0028	0.0022	8.6480
TCNN	0.0000	0.0024	0.0017	6.4901

Table 2: Testing performance metrics for different models on Bitcoin dataset.

3.2.3 Model Selection and Performance Analysis

Model selection was based on **forecast accuracy at the longest prediction horizon** (day t+5). The results show clear performance differences between the models:

- **TCNN achieved the lowest RMSE, MAE, and MAPE** in both training and testing, demonstrating its ability to capture volatility patterns while maintaining strong generalization.
- **LSTM-GRU outperforms standalone LSTM and GRU**, showing that hybrid architectures improve predictive accuracy.
- **TCNN exhibits a smaller train-test performance gap**, indicating better generalization and reduced overfitting.
- **Lower MAPE values confirm TCNN's reliability**, which is critical in volatile financial markets where percentage-based errors are significant.

Since similar trends were observed for Ethereum and Litecoin, we selected **TCNN as the final model for volatility forecasting**. The next section presents a detailed analysis of its predictive performance, including visual comparisons between predicted and actual values.

3.2.4 TCNN Performance Analysis on Bitcoin

To assess the effectiveness of TCNN in forecasting cryptocurrency volatility, we evaluated its performance over different forecast horizons using the Bitcoin dataset. The model was trained with a lookback period of 30 days and a forecast horizon of 5 days. Below, we present its accuracy on both training and test sets.

Performance Across Forecast Horizons TCNN demonstrates consistently low error metrics, maintaining strong predictive accuracy over multiple time steps. Tables 3 and 4 summarize its performance during training and testing, respectively.

Time Step	MSE	RMSE	MAE	MAPE (%)
1	0.0000	0.0034	0.0023	6.89
2	0.0000	0.0036	0.0022	6.78
3	0.0000	0.0039	0.0023	7.14
4	0.0000	0.0047	0.0028	8.42
5	0.0000	0.0048	0.0028	8.71

Table 3: Training performance of TCNN across forecast horizons for Bitcoin.

Time Step	MSE	RMSE	MAE	MAPE (%)
1	0.0000	0.0018	0.0013	4.88
2	0.0000	0.0019	0.0013	4.88
3	0.0000	0.0019	0.0014	5.28
4	0.0000	0.0023	0.0016	5.99
5	0.0000	0.0024	0.0017	6.49

Table 4: Testing performance of TCNN across forecast horizons for Bitcoin.

Across both training and testing phases, we observe an expected increase in RMSE as the forecast horizon extends. However, the error progression remains gradual, suggesting that TCNN maintains predictive stability even over longer time steps. Notably, the MAPE remains below 7% on the test set, reinforcing the model's suitability for financial forecasting, where relative accuracy is crucial.

RMSE Evolution: Training vs. Testing To better visualize the error progression over time, Figure 4 and Figure 3 compares RMSE trends during training and testing.

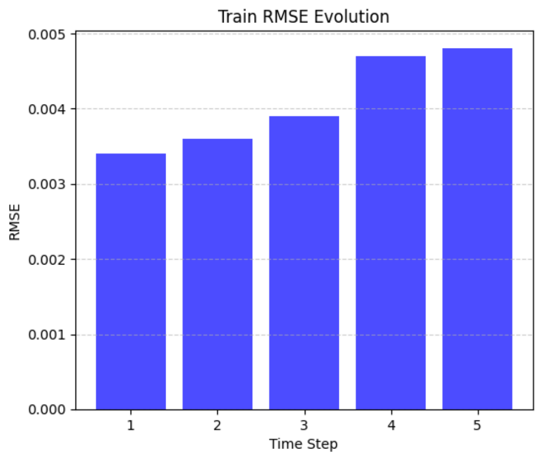


Figure 3: RMSE evolution during training-Bitcoin.

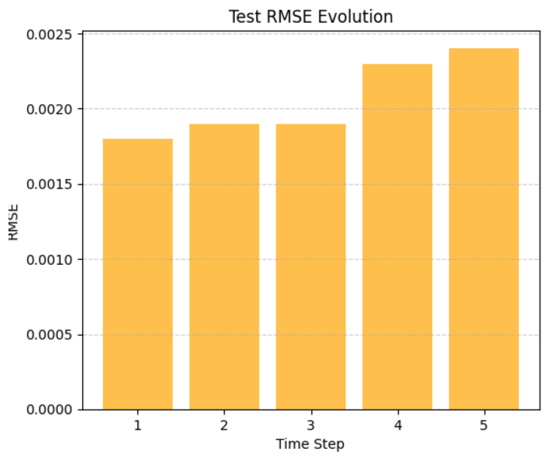


Figure 4: RMSE evolution during testing-Bitcoin.

The RMSE increase observed in both training and testing follows a predictable trend, as uncertainty naturally rises with longer forecast horizons. However, a key observation is that **test RMSE remains consistently lower than train RMSE**, suggesting that TCNN does not overfit the training data but instead generalizes effectively. This robustness is essential for real-world applications, where models must maintain reliability on unseen market conditions.

Conclusion Overall, TCNN effectively captures Bitcoin's volatility patterns while maintaining stable predictive performance. The model exhibits a controlled error increase over time without significant divergence, reinforcing its utility for medium-term forecasting. Its ability to avoid overfitting while maintaining low MAPE values makes it a strong candidate for cryptocurrency volatility prediction. The next section presents a visual comparison between predicted and actual volatility trends, further validating its forecasting capabilities.

3.2.5 TCNN-Based Volatility Prediction for Cryptocurrencies

To evaluate the effectiveness of TCNN in forecasting cryptocurrency volatility, we applied the model to Bitcoin, Ethereum, and Litecoin datasets. Figure 5 to Figure 7 presents a comparison between the predicted and actual volatility values for each cryptocurrency.

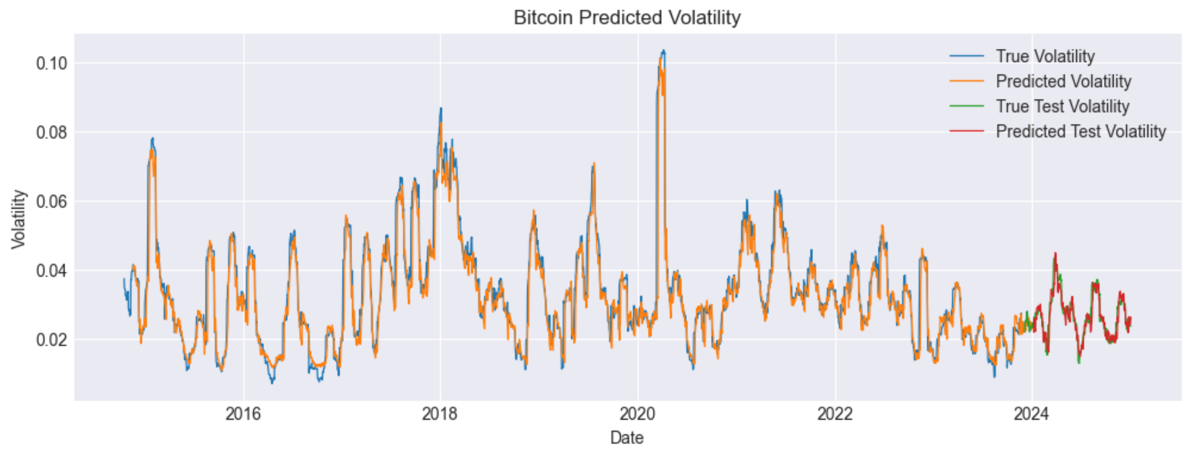


Figure 5: Predicted vs. actual volatility for Bitcoin.

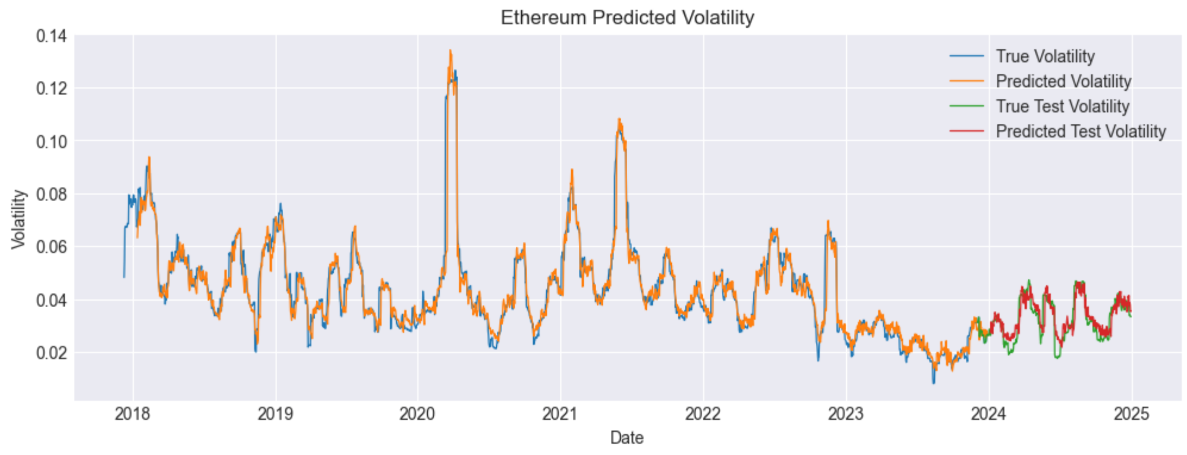


Figure 6: Predicted vs. actual volatility for Ethereum.

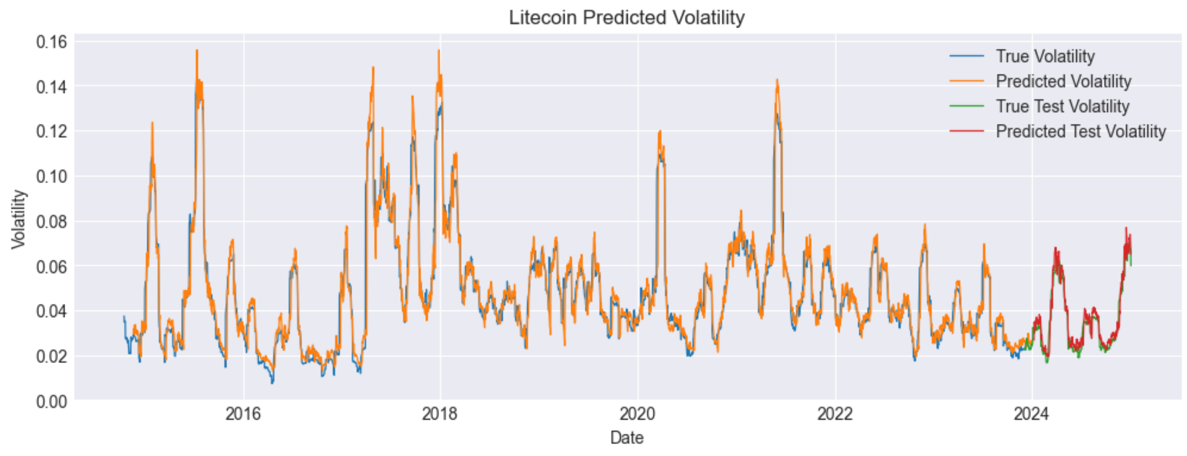


Figure 7: Predicted vs. actual volatility for Litecoin.

Across all three assets, the TCNN model successfully captures the temporal evolution of volatility, closely following the actual fluctuations in the market. The model exhibits strong alignment with observed volatility patterns, even in periods of heightened price instability.

Why TCNN Outperforms LSTM and GRU LSTM and GRU are commonly used for time series forecasting due to their ability to model sequential dependencies. However, TCNN offers several key advantages for financial volatility prediction:

- **Efficient Feature Extraction:** TCNN uses convolutional layers to capture relevant temporal patterns without relying on sequential memory mechanisms, reducing issues like vanishing gradients.
- **Lower Computational Cost:** Unlike LSTM and GRU, which process data sequentially, TCNN applies convolutional filters in parallel, leading to faster training while maintaining accuracy.
- **Better Generalization:** Convolutional layers effectively capture localized dependencies, improving robustness to varying market conditions.
- **Stable Predictions Across Horizons:** TCNN consistently maintains predictive accuracy over multiple forecast steps, while LSTM can exhibit drift in long-term forecasts.
- **Reduced Overfitting:** Dropout layers in TCNN enhance regularization, providing reliable predictions without extensive hyperparameter tuning.

TCNN demonstrates strong predictive performance in cryptocurrency volatility forecasting. Its ability to efficiently capture patterns, reduce computational cost, and maintain stability over multiple horizons makes it a reliable alternative to recurrent architectures.

4 Portfolio Optimization

Cryptocurrency portfolio management presents specific challenges due to their high volatility and unstable correlations between different assets. To optimize resource allocation, we have implemented a method based on volatility forecasting using deep learning models such as GRU, LSTM, and TCNN. The results of these forecasts are integrated into a portfolio construction algorithm that relies on risk parity and volatility targeting strategies.

4.1 Portfolio Construction

We predicted the volatility of three cryptocurrencies: Bitcoin, Ethereum, and Litecoin. The TCNN provided the most accurate volatility forecasts, making it the chosen model for all three cryptocurrencies to optimize risk parity and portfolio management.

4.1.1 Risk Parity Theory

A risk parity portfolio aims to equalize risk allocation across different asset classes by overweighting safer assets and underweighting riskier ones. Unlike traditional approaches based on mean-variance optimization, risk parity does not require estimating expected returns, which is a significant advantage.

4.1.2 Risk Parity Calculation

1. **Volatility Measurement:** We use volatility forecasts generated by the TCNN model. This model effectively captures the complex and non-linear dynamics of

cryptocurrency time series.

2. **Full vs Simplified Approach:** In theory, the marginal contribution of each asset to the total portfolio risk can be calculated by accounting for correlations between assets:

$$MC_{i,t} = w_{i,t}\sigma_{i,t}\beta_{i,t} \tag{12}$$

where

$$\beta_{i,t} = \frac{Cov(\sigma_{i,t}, \sigma_{p,t})}{(\sigma_{p,t})^2} \tag{13}$$

However, this approach requires the estimation of the covariance matrix, a complex and resource-intensive process, especially for a large number of assets. Moreover, in the context of cryptocurrencies, correlations are often unstable, further complicating precise estimation of these relationships.

To simplify implementation, we chose not to use the marginal contribution method. Instead, we assume that assets are uncorrelated, allowing us to calculate the weights based solely on the volatility predicted by the TCNN. Thus, the weight of each asset in the portfolio is determined as follows:

$$w_{i,t} = \frac{1/\hat{\sigma}_{i,t}}{\sum_{l=1}^{N} 1/\hat{\sigma}_{l,t}} \tag{14}$$

where $\hat{\sigma}_{i,t}$ is the predicted volatility of asset i at time t.

This simplification reduces computational complexity while retaining the benefits of risk-based diversification, leveraging more accurate volatility forecasts provided by the TCNN.

4.1.3 Advantages of Risk Parity

- **Optimized Diversification:** By equalizing risks, the portfolio is naturally diversified.
- **Reduced Overall Volatility:** This approach tends to reduce the overall volatility of the portfolio.
- **Independence from Expected Returns:** No need to estimate future returns.
- **Computational Simplicity:** The simplified version used in this project allows for quicker and more efficient implementation while maintaining effective risk management.

4.2 Results of the Risk Parity Method

In this section, we present the results of the simplified risk parity method using predicted volatilities at **T+5**. This approach allocates portfolio weights based on volatility forecasts generated by the TCNN model for *Bitcoin*, *Ethereum*, and *Litecoin*.

While the same methodology was applied for **T+2** predictions, the focus here will be on highlighting the differences between the results obtained at **T+2** and **T+5**. This comparison will provide insights into how varying the prediction horizon influences the portfolio's risk allocation and responsiveness to anticipated market volatility.

4.2.1 Analysis of Risk Parity Results for T+5

The results show that, over the year 2024, *Bitcoin* and *Ethereum* maintained relatively stable weights in the portfolio, reflecting consistent predicted volatilities. *Bitcoin* experienced a significant increase in weight towards the end of the year, suggesting periods of lower predicted volatility, while *Ethereum* remained steady with moderate fluctuations.

In contrast, *Litecoin* exhibited more pronounced fluctuations, with significant weight increases in **May** and **July** and notable decreases in **March** and **December**, indicating periods of both lower and higher predicted volatility or forecast uncertainty.

The adjustment of weights across all assets remained gradual, without extreme variations, demonstrating good responsiveness to volatility predictions. However, the model may underreact to sudden market shocks and shows instability in *Litecoin*'s predictions, highlighting potential areas for refinement in volatility forecasting and risk allocation.

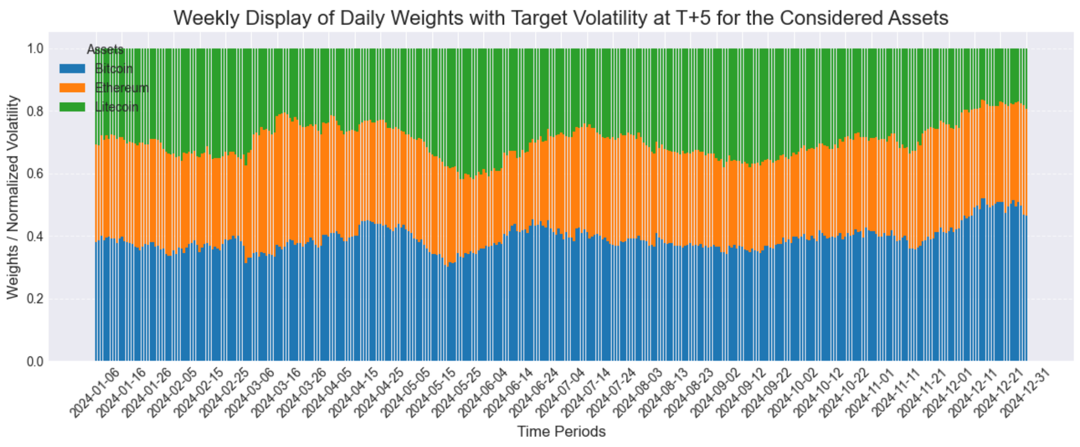


Figure 8: Weekly Display of Daily Weights with Target Volatility at $T + 5$ for Bitcoin, Ethereum, and Litecoin

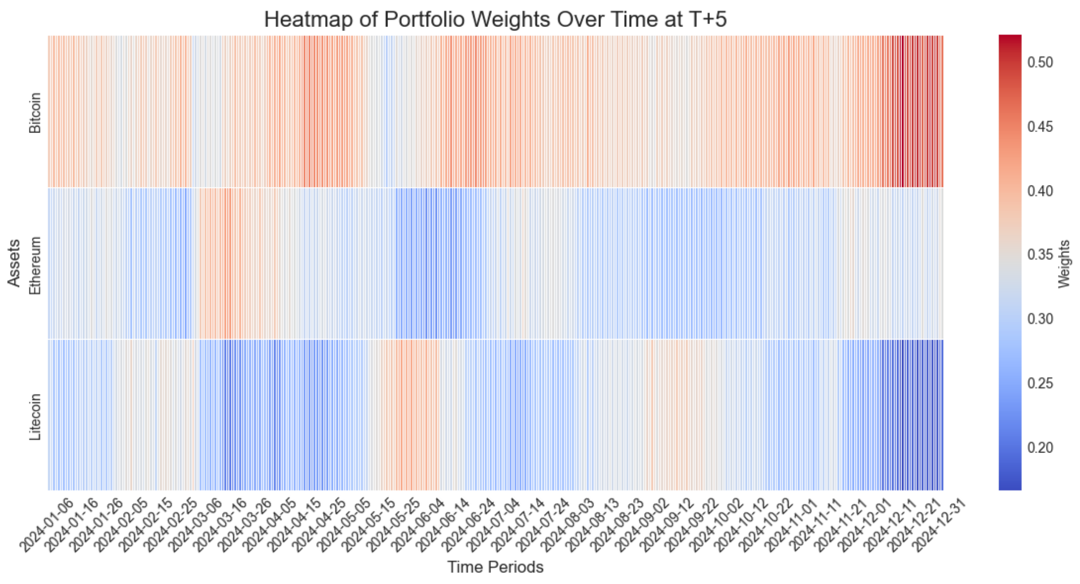


Figure 9: Heatmap of Portfolio Weights Over Time at $T + 5$ for Bitcoin, Ethereum, and Litecoin

The **Dynamic Portfolio (T+5)** recorded a slightly higher cumulative return of **46.77%** compared to **45.18%** for the **Balanced Portfolio** with equal weights. While the dynamic strategy based on predicted volatility provided a marginal improvement in performance and better downside protection during periods of volatility, this small difference suggests limited added value compared to a simple balanced approach. However, these results could be influenced by the specific market conditions in **2024**, a year marked by particular market movements.

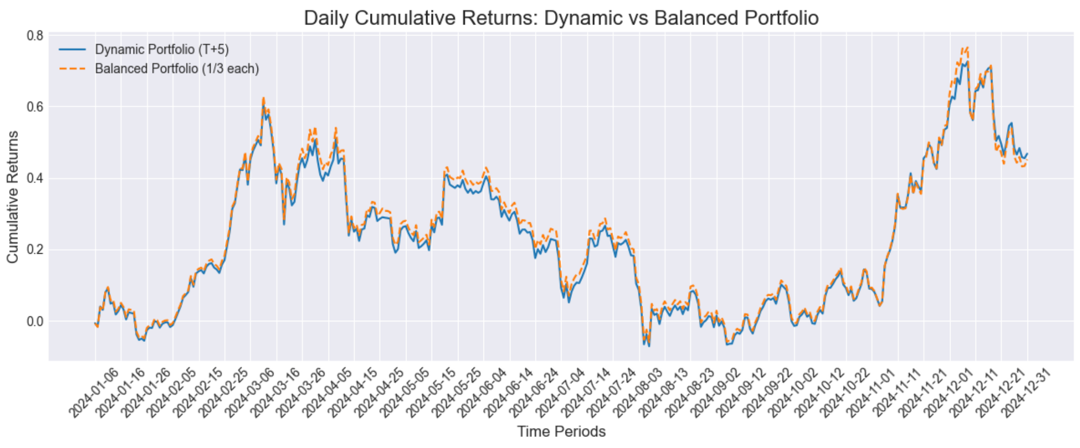


Figure 10: Daily Cumulative Returns: Dynamic vs Balanced Portfolio

4.2.2 Comparison Between $T + 5$ and $T + 2$

The comparison between **T+5** and **T+2** prediction horizons reveals notable differences in portfolio weight allocations across assets. *Bitcoin* shows a slight increase in weight under the T+5 horizon, reflecting stable volatility predictions across both timeframes. In contrast, *Ethereum* experiences a reduction in weight at T+5, suggesting higher volatility forecasts over the longer horizon, leading to more conservative allocations. *Litecoin* exhibits the most significant increase in weight under T+5, indicating that its short-term volatility (T+2) is more pronounced, while longer-term forecasts are more stable. Although these differences are relatively small, they highlight how varying prediction horizons can influence portfolio composition, particularly for assets with more volatile behavior like *Ethereum* and *Litecoin*.

In terms of performance, the **Dynamic Portfolio** based on T+5 predictions achieved a cumulative return of **46.77%**, compared to **45.18%** for the **Balanced Portfolio** with equal weights. However, when using T+2 predictions, the Dynamic Portfolio outperformed both, with a cumulative return of **47.96%**. This suggests that shorter prediction horizons may offer better responsiveness to market conditions, leading to improved returns, while longer horizons provide more stability but slightly lower performance.

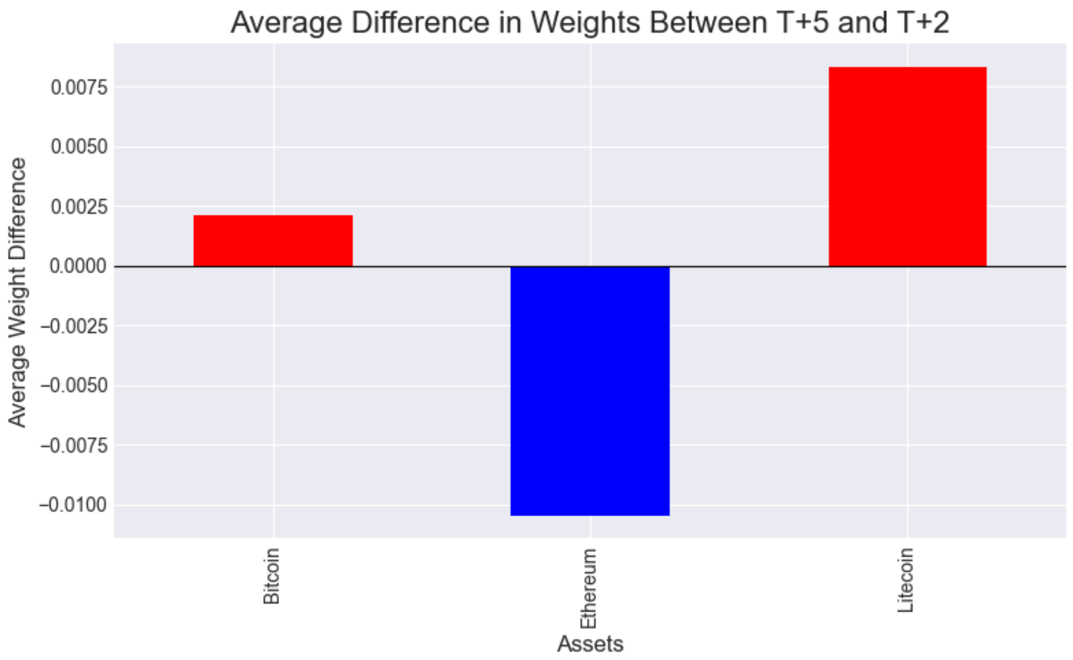


Figure 11: Average Difference in Weights Between $T + 5$ and $T + 2$

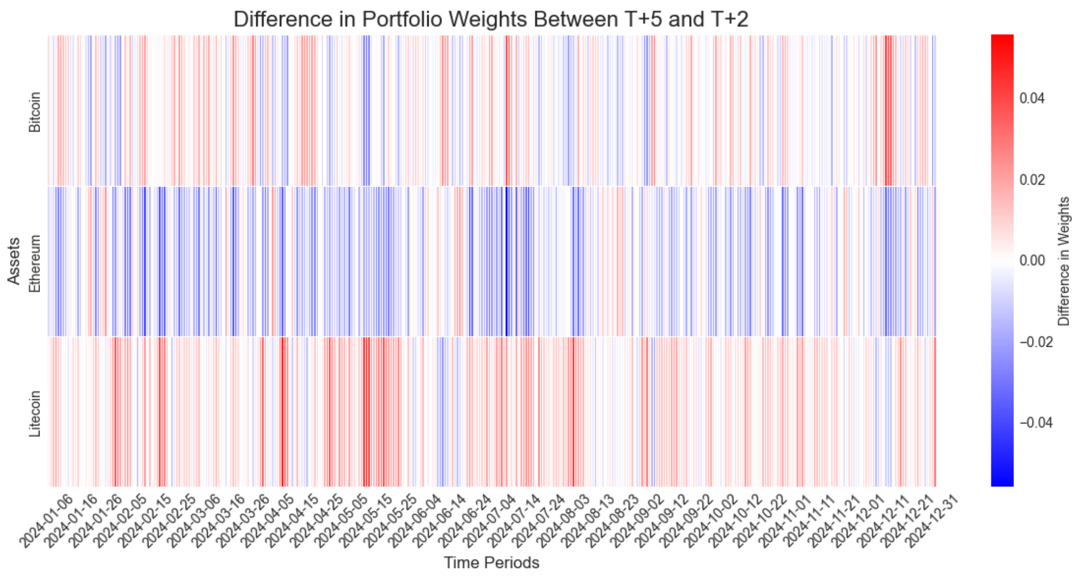


Figure 12: Difference in Portfolio Weights Over Time Between $T + 5$ and $T + 2$

4.3 Volatility Targeting as an Alternative Approach

While our study primarily utilized the **simplified risk parity** method for portfolio construction, an alternative approach worth considering is **volatility targeting**. Unlike risk parity, which seeks to equalize the risk contribution of each asset without explicitly setting a total risk level, volatility targeting aims to maintain a consistent overall portfolio volatility, adjusting asset weights accordingly.

Methodology:

- A predefined **target volatility** (e.g., 15% annualized) is established, representing the desired level of risk for the portfolio.
- The predicted volatilities of individual assets are obtained using models like TCNN.
- Asset weights are then calculated by dividing the target volatility by each asset's predicted volatility. This results in **lower weights for more volatile assets** and **higher weights for less volatile assets**.
- The weights are finally **normalized** to ensure the total allocation sums to 100% for each time period.

Potential Benefits and Applications: Volatility targeting offers more direct control over the portfolio's risk level compared to risk parity. This method is particularly advantageous in highly volatile markets like cryptocurrencies, where risk levels can fluctuate dramatically. By maintaining a consistent volatility threshold, this approach could lead to **more stable returns** and improved risk management.

Future Exploration: Although we focused on the simplified risk parity method, implementing and testing the volatility targeting strategy could provide valuable insights. Comparing both methods across different prediction horizons (e.g., **T+2** and **T+5**) would help evaluate whether targeting a fixed level of volatility enhances portfolio performance and stability in the dynamic cryptocurrency market.

5 Conclusion

This article explored the application of neural networks for cryptocurrency volatility forecasting and portfolio optimization. By comparing the performance of GRU, LSTM, LSTM-GRU, and TCNN models, we demonstrated that the TCNN stands out due to its ability to capture the complex dynamics of cryptocurrency markets while maintaining high accuracy and robust generalization.

The integration of volatility forecasts into portfolio optimization strategies, particularly the simplified risk parity approach, improved risk-adjusted returns. Although the dynamic approach based on 5-day forecasts slightly outperformed the balanced portfolio, the difference remains modest, highlighting the importance of market conditions in the effectiveness of predictive models.

Our results show that shorter forecasting horizons, such as T+2, can offer better responsiveness to market conditions, while longer horizons provide greater stability. The comparison between risk parity and volatility targeting approaches suggests interesting avenues for future research, particularly in refining risk management in highly volatile environments like cryptocurrencies.

In conclusion, this study highlights the potential of neural networks for volatility forecasting and portfolio optimization while emphasizing the challenges posed by the unpredictable nature of cryptocurrency markets. Future improvements include integrating new features, exploring hybrid architectures, and adapting models to ever-evolving market conditions.

6 Appendix

Table 5: Training performance metrics for different models on Ethereum dataset.

Model	MSE	RMSE	MAE	MAPE (%)
LSTM	0.0000	0.0066	0.0047	12.9362
GRU	0.0000	0.0059	0.0036	8.4059
LSTM-GRU	0.0001	0.0072	0.0056	16.0437
TCNN	0.0000	0.0050	0.0029	7.2701

Table 6: Testing performance metrics for different models on Ethereum dataset.

Model	MSE	RMSE	MAE	MAPE (%)
LSTM	0.0000	0.0059	0.0049	16.9334
GRU	0.0000	0.0043	0.0030	9.5433
LSTM-GRU	0.0000	0.0068	0.0060	20.8574
TCNN	0.0000	0.0041	0.0032	11.0125

Table 7: Training performance metrics for different models on Litecoin dataset.

Model	MSE	RMSE	MAE	MAPE (%)
LSTM	0.0001	0.0077	0.0045	9.5179
GRU	0.0001	0.0084	0.0050	10.6674
LSTM-GRU	0.0001	0.0086	0.0053	12.0694
TCNN	0.0000	0.0069	0.0043	9.7504

Table 8: Testing performance metrics for different models on Litecoin dataset.

Model	MSE	RMSE	MAE	MAPE (%)
LSTM	0.0000	0.0040	0.0028	7.9508
GRU	0.0000	0.0041	0.0030	8.8661
LSTM-GRU	0.0000	0.0040	0.0029	8.3711
TCNN	0.0000	0.0039	0.0031	9.4532

Table 9: Training performance metrics for Ethereum dataset.

Time Step	MSE	RMSE	MAE	MAPE (%)
1	0.0000	0.0031	0.0020	4.8588
2	0.0000	0.0036	0.0021	5.0414
3	0.0000	0.0041	0.0024	5.7927
4	0.0000	0.0046	0.0027	6.6175
5	0.0000	0.0050	0.0029	7.2701

Table 10: Testing performance metrics for Ethereum dataset.

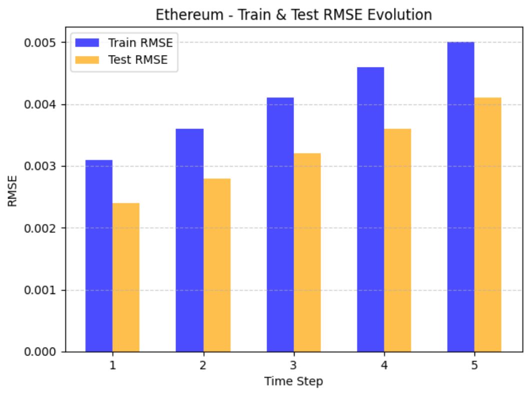
Time Step	MSE	RMSE	MAE	MAPE (%)
1	0.0000	0.0024	0.0019	5.9163
2	0.0000	0.0028	0.0020	6.5243
3	0.0000	0.0032	0.0024	7.8593
4	0.0000	0.0036	0.0028	9.4745
5	0.0000	0.0041	0.0032	11.0125

Table 11: Training performance metrics for Litecoin dataset.

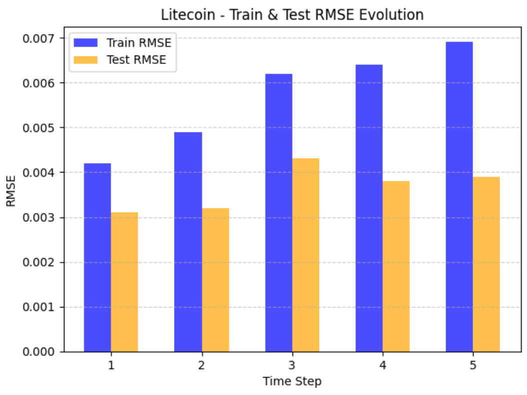
Time Step	MSE	RMSE	MAE	MAPE (%)
1	0.0000	0.0042	0.0027	5.9275
2	0.0000	0.0049	0.0030	6.8610
3	0.0000	0.0062	0.0042	9.3362
4	0.0000	0.0064	0.0040	9.1110
5	0.0000	0.0069	0.0043	9.7504

Table 12: Testing performance metrics for Litecoin dataset.

Time Step	MSE	RMSE	MAE	MAPE (%)
1	0.0000	0.0031	0.0023	6.2495
2	0.0000	0.0032	0.0025	7.0864
3	0.0000	0.0043	0.0035	10.2692
4	0.0000	0.0038	0.0031	9.1126
5	0.0000	0.0039	0.0031	9.4532



(a)



(b)

Figure 13: Comparison of Train and Test RMSE for Ethereum and Litecoin.