forecasting

MDPI

Article

Temporal Attention-Enhanced Stacking Networks: Revolutionizing Multi-Step Bitcoin Forecasting

Phumudzo Lloyd Seabe [1,*], Edson Pindza [2], Claude Rodrigue Bambe Moutsinga [1] and Maggie Aphane [1]

[1] Department of Mathematics and Applied Mathematics, Sefako Makgatho Health Sciences University, Pretoria 0204, South Africa

[2] College of Economic and Management Sciences, Department of Decision Sciences, University of South Africa, Pretoria 0002, South Africa

[*] Correspondence: phumudzo@aims.ac.za

Abstract: This study presents a novel methodology for multi-step Bitcoin (BTC) price prediction by combining advanced stacking-based architectures with temporal attention mechanisms. The proposed Temporal Attention-Enhanced Stacking Network (TAESN) integrates the complementary strengths of diverse machine learning algorithms while emphasizing critical temporal features, leading to substantial improvements in forecasting accuracy over traditional methods. Comprehensive experimentation and robust evaluation validate the superior performance of TAESN across various BTC prediction horizons. Additionally, the model not only demonstrates enhanced predictive accuracy but also offers interpretable insights into the temporal dynamics underlying cryptocurrency markets, contributing to both practical forecasting applications and theoretical understanding of market behavior.

Keywords: cryptocurrency price forecasting; temporal attention mechanism; LSTM; GRU; multi-step prediction; stacking ensemble learning; Temporal Convolutional Networks (TCNs); hybrid machine learning models

check for updates

Academic Editor: Sonia Leva

Received: 29 November 2024
Revised: 19 December 2024
Accepted: 27 December 2024
Published: 30 December 2024

Citation: Seabe, P.L.; Pindza, E.; Moutsinga, C.R.B.; Aphane, M. Temporal Attention-Enhanced Stacking Networks: Revolutionizing Multi-Step Bitcoin Forecasting. *Forecasting* **2025**, *7*, 2. https://doi.org/10.3390/forecast7010002

1. Introduction

The cryptocurrency market, characterized by its extreme volatility and complexity, presents unique challenges for accurate price prediction. Traditional forecasting models often fall short in capturing the intricate temporal dependencies and non-linear patterns inherent in cryptocurrency data [1,2]. These limitations highlight the critical need for advanced predictive methodologies capable of addressing the volatile nature of digital asset markets, where rapid price swings can significantly impact financial decision making and strategy.

Cryptocurrency price movements are influenced by numerous factors, including market sentiment, regulatory changes, technological advancements, and macroeconomic conditions [3]. These interdependent factors contribute to the dynamic and often unpredictable behavior of the market, amplifying the difficulty of designing forecasting models that consistently perform well across diverse scenarios [4].

The unpredictable nature of the cryptocurrency market not only challenges traditional forecasting models but also creates opportunities for novel computational approaches. Existing methods, such as statistical and econometric models, are often constrained by their linear assumptions, failing to accommodate the intricate, non-linear dependencies within cryptocurrency time–series data [5]. Machine learning models, while showing promise, struggle to consistently address these challenges due to their reliance on static feature sets and limited ability to dynamically interpret temporal dependencies [6,7]. These shortcomings underscore the demand for hybrid models that adaptively combine the strengths of diverse forecasting techniques to ensure robustness and accuracy.

This study introduces the Temporal Attention-Enhanced Stacking Network (TAESN), a novel hybrid framework that addresses the limitations of existing models by integrating temporal attention mechanisms with stacking ensemble learning. TAESN dynamically adapts to specific temporal dependencies, enabling it to prioritize historically significant patterns and short-term or long-term trends as required [8]. By combining diverse base learners such as LSTM, GRU, CNN, and TCN, TAESN leverages their complementary strengths to capture a wide range of temporal features, from sequential patterns to localized and long-range dependencies. This adaptive approach not only enhances forecasting accuracy but also provides interpretable insights into the contributions of individual models across different prediction horizons. The proposed framework goes beyond the capabilities of static ensemble approaches by employing a temporal attention mechanism that assigns dynamic weights to base learners based on their relevance to specific forecasting tasks. This innovation ensures that TAESN remains adaptable across multiple BTC prediction horizons, delivering a highly accurate and reliable tool for navigating the volatility of cryptocurrency markets. The interpretable nature of attention scores also addresses a key limitation of many deep learning models, making TAESN a practical solution for both theoretical research and real-world applications.

The significance of this research lies in its ability to generate substantial value for various stakeholders within the cryptocurrency ecosystem. Traders can utilize precise forecasts to optimize their strategies, while investors and financial analysts gain actionable insights to inform their decisions [9]. Furthermore, TAESN contributes to market maturation by reducing information asymmetry and improving overall market efficiency [10]. These advancements not only establish a new benchmark for financial time–series forecasting but also offer a robust foundation for further exploration into hybrid ensemble models within volatile financial domains. This paper evaluates the efficacy of TAESN in cryptocurrency price forecasting by leveraging temporal attention mechanisms and stacking-based ensemble learning. The study encompasses data collection and preprocessing of BTC time–series, feature engineering of technical indicators, and the implementation of a hybrid architecture that integrates LSTM, GRU, CNN, and TCN models. The primary objectives are to assess the impact of temporal attention mechanisms on forecasting accuracy, determine the robustness of stacking ensembles across multiple prediction horizons, and establish TAESN as a superior predictive model for navigating volatile cryptocurrency markets.

The remainder of this paper is organized as follows: Section 2 reviews the relevant literature on cryptocurrency forecasting methodologies and advanced machine learning models. Section 3 describes the data collection and preprocessing methods, as well as the implementation of LSTM, GRU, CNN, TCN, TAESN, and Meta-learner models. Section 4 presents the experimental results, comparative analysis, and discusses the findings and their implications, and Section 5 concludes the paper with suggestions for future research.

2. Literature Review

In this section, a comprehensive literature review is conducted on cryptocurrency forecasting methodologies and advanced machine learning models. Firstly, we discuss the evolution of cryptocurrency prediction models, highlighting the transition from traditional econometric approaches, such as ARIMA and GARCH, to more sophisticated machine learning and deep learning techniques. Next, we explore the adoption of ensemble methods, such as stacking, bagging, and boosting, in financial forecasting, with a focus on their application to cryptocurrency price prediction. Lastly, we examine the integration of attention mechanisms in time–series forecasting, emphasizing their ability to dynamically prioritize relevant historical data and their emerging role in enhancing the predictive accuracy of cryptocurrency forecasting models.

2.1. Cryptocurrency Prediction Models

Cryptocurrency price forecasting has seen a significant evolution from statistical approaches to advanced machine learning models. Early methods relied on econometric models such as Autoregressive Integrated Moving Average (ARIMA) and Generalized Autoregressive Conditional Heteroskedasticity (GARCH), which provided insights into linear price trends but struggled with the non-linear and volatile nature of cryptocurrencies [11].

The advent of machine learning introduced models such as Support Vector Machines (SVMs) and Random Forests, which demonstrated improved predictive performance compared to traditional methods [12]. However, these models often struggled with the high dimensionality and temporal dependencies of cryptocurrency data. Deep learning models, particularly Recurrent Neural Networks (RNNs) and their variants, such as Long Short-Term Memory (LSTM) networks, emerged as powerful tools for capturing long-term dependencies in sequential data [5]. Studies such as [6] demonstrated the ability of LSTMs to outperform traditional machine learning models in cryptocurrency forecasting tasks.

Recent advancements in hybrid models have further improved forecasting accuracy by combining statistical and deep learning techniques. For instance, ref. [13] proposed a hybrid LSTM-ARIMA model to address both linear and non-linear dependencies, showcasing the potential of such combinations in financial time–series data. This aligns with the TAESN model, which integrates diverse base learners capable of capturing complex temporal dynamics. Additionally, the integration of external data, such as social media sentiment and market indicators, has proven valuable in enhancing predictive models. Ref. [14] demonstrated that incorporating sentiment analysis alongside market data improved predictive accuracy in cryptocurrency markets. Similarly, ref. [15] proposed a multimodal framework, PreBit, integrating Twitter sentiment embeddings and market data to forecast extreme Bitcoin price movements. These studies highlight the untapped potential for TAESN to leverage external variables, broadening its applicability and robustness.

2.2. Ensemble Methods in Financial Forecasting

Ensemble methods have emerged as robust techniques for improving prediction accuracy by combining multiple models. In financial forecasting, ensemble approaches such as bagging, boosting, and stacking have demonstrated superior performance compared to standalone models [16]. Stacking, in particular, has shown promise due to its ability to leverage the strengths of diverse base learners. It involves training multiple first-level models and aggregating their outputs through a second-level meta-learner, resulting in enhanced predictive performance [17]. In cryptocurrency forecasting, ensemble techniques have been applied with considerable success. For example, ref. [18] proposed an ensemble approach combining Empirical Mode Decomposition (EMD) and LSTM networks, achieving high accuracy in Bitcoin price prediction.

Recent studies have further expanded the scope of ensemble methods by integrating attention mechanisms. Ref. [19] demonstrated that hybrid CNN-LSTM ensembles, enhanced with attention mechanisms, effectively captured short- and long-term dependencies. These findings align with TAESN's stacking-based ensemble framework, where temporal attention dynamically prioritizes the contributions of LSTM, GRU, CNN, and TCN base learners across different forecasting horizons.

A key advantage of stacking-based ensembles is their ability to handle complex temporal dependencies by combining models with complementary strengths. For instance, ref. [20] highlighted how Temporal Convolutional Networks (TCNs), when integrated into ensemble frameworks, effectively capture long-term dependencies, while recurrent models like LSTMs excel at modeling sequential data. TAESN leverages this complementarity, with CNNs capturing localized temporal patterns and GRUs addressing short-term dependencies. The temporal attention mechanism further refines this integration by dynamically assigning weights to each base learner, making the ensemble adaptable to varying prediction horizons.

Recent advancements have also explored the role of transformers in ensemble settings. Ref. [21] demonstrated that transformer-based models, when combined with traditional sequence learning techniques, significantly improved cryptocurrency forecasting by capturing intricate temporal and cross-market dependencies. These findings suggest that integrating advanced attention-based architectures within stacking ensembles, as TAESN does, represents a novel and effective approach to financial forecasting. By dynamically prioritizing diverse learners, TAESN addresses the inherent volatility and noise in cryptocurrency markets, achieving both accuracy and interpretability.

2.3. Attention Mechanisms in Time–Series Forecasting

Attention mechanisms have revolutionized sequence modeling by enabling models to focus selectively on the most relevant parts of input data. Originally introduced in natural language processing [8], attention mechanisms have since been adapted to various domains, including time–series forecasting. These mechanisms enhance model interpretability and performance by assigning dynamic weights to different temporal features, allowing the model to prioritize critical time steps while minimizing the influence of noise.

The application of attention mechanisms in financial time–series forecasting has demonstrated remarkable success. Ref. [22] proposed a dual-stage attention-based Recurrent Neural Network (RNN) for time–series prediction, achieving significant improvements over traditional RNN models by focusing on the most informative temporal points. Similarly, ref. [23] introduced an attention-enhanced LSTM for Bitcoin price forecasting, showcasing how attention can refine sequential learning to improve accuracy in highly volatile financial markets. These studies highlight the critical role of attention in dynamically adapting to temporal variations, particularly in noisy and non-linear domains such as cryptocurrency forecasting.

For instance, ref. [24] demonstrated that attention-based models significantly improved time–series analysis by selectively prioritizing key events, enhancing both accuracy and interpretability. Similarly, ref. [25] showed that combining LSTM with multi-head attention improves accuracy by focusing on critical historical patterns. In financial forecasting, ref. [26] introduced a deep learning ensemble model integrating CNN, LSTM, and ARMA, which effectively captured both short-term and long-term patterns. These findings reinforce the importance of attention mechanisms in handling the complexities of financial time–series data.

Recent advancements have extended the capabilities of attention mechanisms through multi-head and temporal designs. Multi-head attention, as employed in transformer architectures [8], has proven particularly effective for modeling long-range dependencies. For example, ref. [27] demonstrated that combining temporal attention with Temporal Convolutional Networks (TCNs) enables models to capture both short- and long-term dependencies in financial data. This approach is particularly relevant to TAESN, where temporal attention mechanisms dynamically adjust the weights of base learners to optimize performance across multiple prediction horizons.

Ref. [28] introduced a novel approach for time–series forecasting that leverages robust attention weights structured with global landmarks and local windows. This technique enhances forecasting accuracy and resilience against noise and distribution shifts, outperforming state-of-the-art models in multivariate time–series forecasting. Similarly, ref. [29] proposed two advanced attention mechanisms—Frequency Spectrum Attention (FSatten) and Scaled Orthogonal Attention (SOatten)—that improve forecasting performance by capturing periodic dependencies and comprehensive dependency patterns in multivariate data. These findings offer valuable insights into potential enhancements for TAESN, particularly in modeling periodic and complex patterns inherent in cryptocurrency price data.

Additionally, ref. [20] highlighted the ability of attention-based transformers to model extended temporal dependencies, outperforming traditional methods for time–series forecasting. Finally, ref. [30] demonstrated that ensemble Transformer models leveraging attention mechanisms excel in financial time–series tasks, further validating the role of attention in improving long-horizon forecasts. These contributions suggest that transformer architectures and attention-based ensembles offer a promising direction for cryptocurrency price forecasting.

Emerging architectures such as transformers have further advanced attention mechanisms for time–series forecasting. Ref. [21] explored transformer-based models for cryptocurrency price prediction, emphasizing their ability to handle cross-market correlations and incorporate external data like sentiment analysis. This marks a significant departure from traditional recurrent and convolutional architectures by offering scalability and flexibility in modeling intricate temporal patterns. Informer models [20], designed for long-horizon forecasting, have also demonstrated how sparse attention mechanisms can improve computational efficiency without sacrificing accuracy, making them promising candidates for cryptocurrency forecasting.

In the TAESN framework, attention mechanisms play a pivotal role in enhancing interpretability and accuracy. By dynamically weighting the contributions of diverse base learners such as LSTM, GRU, CNN, and TCN, the temporal attention mechanism ensures that the ensemble adapts to varying temporal dependencies. For shorter horizons, it prioritizes localized patterns captured by CNNs and GRUs, while for longer horizons, it shifts focus to broader trends identified by TCNs and LSTMs. This dynamic adaptability not only addresses the challenges of volatility and noise in cryptocurrency data but also differentiates TAESN from traditional ensemble models that rely on static weighting schemes. These advancements underscore the transformative potential of attention mechanisms in financial forecasting. By integrating temporal attention within a stacking ensemble framework, TAESN builds on foundational methodologies while addressing critical limitations in existing models. This innovative approach positions TAESN at the forefront of cryptocurrency forecasting, combining interpretability, adaptability, and predictive accuracy.

In the proposed TAESN framework, the temporal attention mechanism dynamically assigns weights to the predictions of diverse base learners (LSTM, GRU, CNN, and TCN). For shorter horizons, the mechanism emphasizes recent trends captured by LSTM and GRU, while for longer horizons, it shifts focus toward broader patterns identified by TCN. By dynamically adapting its focus, TAESN improves performance across multiple horizons, as validated in our experiments, and provides enhanced interpretability through the analysis of attention weights.

3. Methodology

In this section, we explore the data that form the foundation of this study. We also present visual representations of BTC price data and demonstrate the effectiveness of the various models utilized in the proposed TAESN framework. The aim of this study is to leverage historical price data and other key features, such as trading volume and high/low prices, to forecast BTC price movements. The project is divided into four main components: (1) gathering and consolidating historical BTC data into a single, cleaned, and comprehensive dataset; (2) employing preprocessing techniques, including normalization and sequence creation, to prepare the data for model training; (3) utilizing LSTM, GRU, CNN, and TCN architectures as base learners to capture diverse temporal patterns in the data; (4) applying a meta-learner, specifically an XGBoost model, to combine the outputs of the attention-enhanced base learners into final predictions.

By incorporating the meta-learner, the framework effectively aggregates insights from the diverse base learners, improving overall prediction accuracy and robustness across different forecast horizons. This hierarchical approach leverages the complementary strengths of individual models while addressing the inherent volatility and complexity of cryptocurrency price movements.

3.1. Data Collection and Preprocessing

The dataset utilized in this study consists of historical daily Bitcoin (BTC) price data collected from Yahoo Finance. The dataset spans a total of 2445 observations over the period 7 November 2017–19 July 2024. It includes the essential features typically used in cryptocurrency analysis, such as the Open, High, Low, and Close (OHLC) prices, and trading volume.

While external variables such as sentiment analysis or macroeconomic indicators (e.g., interest rates, economic events, or social media activity) could potentially improve cryptocurrency price predictions, they were not included in this study. The exclusion was primarily driven by the need to maintain the scope and focus of the work. Incorporating external variables would require significant preprocessing, feature engineering, and model tuning, which would increase both the complexity and computational demands of the framework. Moreover, this study aims to evaluate the effectiveness of TAESN in capturing intricate temporal dependencies using historical price data alone. By relying solely on historical price inputs, the model's performance can be thoroughly assessed as a strong and interpretable baseline.

Although this study focuses exclusively on Bitcoin due to its high market capitalization, liquidity, and extensive historical data, the TAESN framework is designed to generalize to other cryptocurrencies. The adaptable architecture, which combines LSTM, GRU, CNN, and TCN base learners with a temporal attention mechanism, can effectively model the unique volatility and temporal dynamics of other digital assets, such as Ethereum, Ripple, and Litecoin. Future work will validate the model's applicability to a broader set of cryptocurrencies, ensuring its robustness and versatility in multi-asset financial markets.

The following preprocessing steps were applied to prepare the dataset for machine learning tasks:

1. Data Cleaning: The dataset was verified to contain no missing values, ensuring its completeness. The Date column was converted into a time–series index to preserve the temporal structure, which is critical for sequential modeling.
2. Feature Scaling:All features were normalized using min–max scaling:

$$x' = \frac{x - \min(X)}{\max(X) - \min(X)}, \tag{1}$$

where x is the original feature value, x' is the normalized value, and X represents the set of all values in the feature.

3. Sequence Creation: A sliding window approach was utilized to transform the dataset into input–output sequences. For a given timestamp t, the input sequence X_t was constructed as:
$$X_t = \{x_{t-l}, x_{t-l+1}, \ldots, x_{t-1}\}, \tag{2}$$
where l is the lookback window length. The output y_t corresponds to the *Close* price for prediction horizons:
$$y_t = x_{t+h}, \tag{3}$$
where h is the prediction horizon (1 day, 3 days, or 7 days ahead).
4. Data Splitting: The dataset was divided into training (70%), validation (15%), and test (15%) sets, maintaining chronological order to prevent information leakage and ensure robust evaluation.

3.2. Model Architecture

The proposed Temporal Attention-Enhanced Stacking Network (TAESN) integrates base learners, a temporal attention mechanism, and a meta-learner into a unified framework to improve prediction accuracy and interpretability.

3.2.1. Base Learners

Four base learners were employed, each selected for its ability to capture distinct temporal patterns:

- Long Short-Term Memory (LSTM): LSTM networks capture long-term dependencies by maintaining cell states across time steps. The key equations are:
$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f), \tag{4}$$
$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i), \tag{5}$$
$$\tilde{c}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c), \tag{6}$$
$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t, \tag{7}$$
$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o), \tag{8}$$
$$h_t = o_t \odot \tanh(c_t), \tag{9}$$
where f_t, i_t, and o_t are the forget, input, and output gates, respectively.
- Gated Recurrent Unit (GRU): GRU simplifies LSTM by combining the hidden and cell states into a single state:
$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + b_z), \tag{10}$$
$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + b_r), \tag{11}$$
$$\tilde{h}_t = \tanh(W_h \cdot [r_t \odot h_{t-1}, x_t] + b_h), \tag{12}$$
$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t. \tag{13}$$
- One-Dimensional Convolutional Neural Network (CNN): CNNs extract local temporal patterns using convolutional filters:
$$y_t = \sigma(W * x_t + b), \tag{14}$$
where $*$ represents the convolution operation.

- Temporal Convolutional Network (TCN): TCNs use dilated convolutions to model both short-term and long-term dependencies:

$$y_t = \sum_{i=0}^{k-1} W_i \cdot x_{t-d\cdot i} + b, \tag{15}$$

where d is the dilation rate and k is the kernel size.

3.2.2. Temporal Attention Mechanism

The temporal attention mechanism dynamically assigns importance to the outputs of the base learners. For base learner predictions $\{z_1, z_2, \ldots, z_k\}$, attention scores e_i are computed as:

$$e_i = W_a \cdot z_i + b_a, \tag{16}$$

and the attention weights α_i are obtained via the softmax function:

$$\alpha_i = \frac{\exp(e_i)}{\sum_{j=1}^{k} \exp(e_j)}. \tag{17}$$

The final aggregated prediction $\hat{y}$ is:

$$\hat{y} = \sum_{i=1}^{k} \alpha_i \cdot z_i. \tag{18}$$

3.2.3. Stacking Framework

The outputs of the base learners and the attention mechanism are concatenated to form the input for the meta-learner. Let $\mathrm{Z} \in \mathbb{R}^{n\times k}$ denote the stacked predictions of k base learners for n samples. The meta-learner predicts:

$$\hat{y} = g(Z, \theta), \tag{19}$$

where g represents the meta-learner (XGBoost in this study) and θ are its parameters.

3.3. Training Procedure

1. Base Learner Training: Each base learner was trained independently to minimize the mean squared error (MSE):

$$\mathcal{L}_{\text{base}} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2. \tag{20}$$

2. Meta-Learner Training: The meta-learner was trained on the stacked predictions to minimize:

$$\mathcal{L}_{\text{meta}} = \frac{1}{n} \sum_{i=1}^{n} (y_i - g(Z_i, \theta))^2. \tag{21}$$

3.4. Evaluation Metrics

Three metrics were employed for model evaluation:

- Root Mean Squared Error (RMSE):

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2}. \tag{22}$$

- Mean Absolute Error (MAE):

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i|. \tag{23}$$

3.5. Hyperparameter Tuning

To optimize the performance of the base learners and meta-learner, hyperparameter tuning was conducted using Grid Search with cross-validation. Grid Search systematically evaluates a range of hyperparameter combinations to identify the configuration that minimizes the validation error.

Let $\mathcal{H}$ represent the hyperparameter search space, where each hyperparameter $\theta_j \in \mathcal{H}$ is sampled from a predefined set of candidate values. The objective is to identify the optimal hyperparameter configuration θ^* that minimizes the validation loss $\mathcal{L}_{val}$, defined as:

$$\theta^* = \underset{\theta \in \mathcal{H}}{\operatorname{argmin}}\, \mathcal{L}_{val}(\theta). \tag{24}$$

The tuning process was conducted using GridSearchCV from the `scikit-learn` library. For each hyperparameter combination, k-fold cross-validation was performed, where the training set was partitioned into k equally sized subsets. The model was trained on $k-1$ subsets and validated on the remaining subset, iteratively, to compute the average validation loss:

$$\mathcal{L}_{val} = \frac{1}{k} \sum_{i=1}^{k} \mathcal{L}(\mathcal{D}_{val}^{(i)}, \theta), \tag{25}$$

where $\mathcal{D}_{val}^{(i)}$ is the ith validation subset, and $\mathcal{L}$ is the loss function.

3.5.1. Grid Search Procedure

The hyperparameter tuning for each model was conducted as follows:

1. Define the Hyperparameter Search Space ($\mathcal{H}$): For each model, a hyperparameter grid was carefully defined based on its architecture and requirements. For the meta-learner (XGBoost), the search space included parameters such as learning rate, maximum depth, number of estimators, subsample, and column sampling rate. Similarly, the base learners (LSTM, GRU, CNN, TCN) had their own tailored search spaces, such as the number of hidden units, dropout rates, and convolutional filter sizes.
2. Grid Search Across Combinations: A grid search was performed over all possible combinations of hyperparameters in the defined search space $\mathcal{H}$, ensuring an exhaustive evaluation of model configurations.
3. Evaluate Using k-Fold Cross-Validation ($k = 3$): For each hyperparameter combination, the dataset was split into three folds for cross-validation. The model was trained on $k-1$ folds and validated on the remaining fold, iteratively. This process was repeated for all folds, and the average validation loss was computed.
4. Select Optimal Hyperparameters (θ^*): The hyperparameter configuration θ^* that minimized the average validation loss was selected as the optimal set of parameters. These optimal hyperparameters were then used to train the final model on the full training dataset before evaluation on the test set.

The hyperparameter tuning for the base learners (LSTM, GRU, CNN, and TCN) and the meta-learner (XGBoost) was conducted using Grid Search. Among the tested hyperparameters, the results showed that the batch size and number of epochs had the most significant impact on the performance of LSTM and GRU models. For CNN, the kernel size played a critical role in effectively capturing local temporal patterns. In the case of TCN, the dilation rate was found to be essential for modeling long-term dependencies. Finally, for the XGBoost meta-learner, the learning rate and number of estimators were identified as the most influential hyperparameters in achieving a balance between accuracy and computational efficiency. These findings highlight the importance of hyperparameter selection in optimizing model performance across different forecasting horizons.

The hyperparameter tuning and training strategies for the base learners (LSTM, GRU, CNN, TCN) and the meta-learner within the TAESN framework were carefully designed to ensure robustness and replicability. The dataset was split into 70% training, 15% validation, and 15% test sets to ensure reliable evaluation across unseen data. The base learners were optimized using the Adam optimizer with a learning rate of 0.001, trained for 20 epochs with a batch size of 32, and validated using the split data.

The TAESN, dynamically assigns weights to the predictions of the base learners (LSTM, GRU, CNN, and TCN). Its architecture consists of dense layers with 64 and 32 neurons, followed by a softmax layer to generate attention weights, enabling the model to adaptively focus on the most relevant base learner predictions. The temporal attention mechanism computes attention weights by first transforming the input features through dense layers, capturing non-linear relationships and interactions between base learner outputs. These transformed values are then passed through a softmax activation function, which normalizes them into probabilities that sum to one, representing the relative importance of each base learner's contribution. During training, the attention weights are optimized as part of the end-to-end learning process through backpropagation. Gradients of the loss function are propagated through the attention mechanism, allowing it to dynamically adapt its focus on the most informative base learner predictions based on the input data and forecasting horizon.

The hyperparameters for the attention mechanism, including the number of neurons in the dense layers (32, 64, and 128) and activation functions (e.g., ReLU, Tanh, Softmax), were optimized using Grid Search. The results demonstrated that the configuration with 64 and 32 neurons and a softmax activation consistently achieved the best performance across all forecasting horizons. This systematic tuning ensured that the attention mechanism effectively prioritized the most relevant base learner predictions while maintaining model robustness. The XGBoost meta-learner was additionally trained on stacked predictions from the base models, using hyperparameters such as 100 estimators, a learning rate of 0.05, and a maximum tree depth of 5. These strategies, combined with systematic hyperparameter tuning using Grid Search, allowed the proposed TAESN framework to achieve robust and reliable performance across all forecasting horizons (1 day, 3 days, and 7 days).

3.5.2. Implementation Details

For implementation, the `GridSearchCV` method was utilized as follows:

```
from sklearn.model_selection import GridSearchCV
from xgboost import XGBRegressor
from sklearn.metrics import mean_squared_error
import numpy as np

# Define the model
model = XGBRegressor(random_state=42, objective=‘‘reg:squarederror’’)

# Define the hyperparameter grid
param_grid = {
    ‘‘learning_rate’’: [0.01, 0.05, 0.1],
    ‘‘max_depth’’: [3, 5, 7],
    ‘‘n_estimators’’: [50, 100, 200],
    ‘‘subsample’’: [0.7, 0.8, 1.0],
    ‘‘colsample_bytree’’: [0.7, 0.8, 1.0],
}

# Set up GridSearchCV with 3-fold cross-validation
grid_search = GridSearchCV(
    estimator=model,
    param_grid=param_grid,
    scoring=‘‘neg_root_mean_squared_error’’, # RMSE as the evaluation
    metric
    cv=3,  # 3-fold cross-validation
    verbose=1,
    n_jobs=-1  # Use all available processors
)

# Fit the GridSearchCV to the training data
grid_search.fit(X_train, Y_train)

# Retrieve the best hyperparameters and corresponding RMSE
best_params = grid_search.best_params_
best_rmse = -grid_search.best_score_

print(‘‘Best Hyperparameters:’’, best_params)
print(‘‘Best RMSE on Validation Set:’’, best_rmse)

# Evaluate the best model on the test set
best_model = grid_search.best_estimator_
y_test_pred = best_model.predict(X_test)
test_rmse = np.sqrt(mean_squared_error(Y_test, y_test_pred))

print(‘‘Test RMSE with Best Hyperparameters:’’, test_rmse)
```

3.5.3. Loss Function

The validation loss $\mathcal{L}_{val}$ was computed using the mean squared error (MSE), which directly relates to the RMSE used for evaluation:

$$\mathcal{L}_{val} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2, \tag{26}$$

where y_i is the ground truth and $\hat{y}_i$ is the model prediction for the ith sample.

The optimal hyperparameters θ^* identified through this process were subsequently used to train the final model. By incorporating hyperparameter tuning, the proposed methodology ensures that the models are both robust and optimized for the given forecasting task.

3.6. Experimental Setup

The experiments were conducted for prediction horizons of 1 day, 3 days, and 7 days. These horizons were chosen to evaluate the models' ability to predict both short-term and medium-term price movements, reflecting practical use cases in financial decision-making.

The models were trained and evaluated on the test set, with performance compared across horizons to assess the effectiveness of the TAESN framework.

For each prediction horizon, the training process incorporated multiple base learners and an attention mechanism to leverage complementary strengths. This ensured that the temporal dependencies in the time–series data were effectively captured. Additionally, the experimental setup was designed to minimize overfitting by using cross-validation during hyperparameter tuning and reserving a separate test set for final evaluation. All models were initialized with the same random seed to ensure consistent comparisons across experiments. Performance metrics, including root mean squared error (RMSE), and mean absolute error (MAE), were computed for each horizon. These metrics provided a comprehensive assessment of the models' accuracy and reliability, particularly for the highly volatile cryptocurrency market, where accurate direction prediction is as critical as magnitude estimation.

4. Results

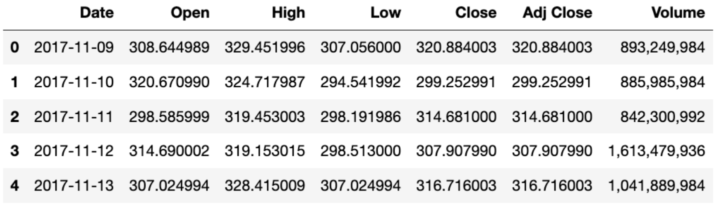
This section evaluates the performance of the proposed Temporal Attention-Enhanced Stacking Network (TAESN) using a rigorous experimental framework. The primary objective is to assess the model's forecasting accuracy and robustness across multiple prediction horizons, including short-term (1 day), medium-term (3 days), and long-term (7 days) forecasts.

Multi-horizon forecasting is particularly important in cryptocurrency markets due to their high volatility, non-linear behavior, and dynamic trends. Short-term forecasts assist traders in making real-time decisions, while medium- and long-term forecasts provide insights into broader market movements and investment strategies. By evaluating TAESN across different horizons, this study demonstrates the model's ability to adapt dynamically to varying temporal dependencies and forecasting requirements.

The results of this study focus on evaluating the effectiveness of the Temporal Attention-Enhanced Stacking Network (TAESN) compared to individual baseline models, including LSTM, GRU, CNN, and TCN, for BTC price forecasting. The models were assessed across prediction horizons of 1 day, 3 days, and 7 days, with a comprehensive analysis of both quantitative metrics (RMSE, MAE) and qualitative visualizations (actual vs. predicted price trends). The performance of TAESN and its base learners was analyzed to demonstrate its robustness and reliability in capturing complex temporal dependencies and price movements.

4.1. Data Summary

The dataset used for this study is depicted in Figure 1, which provides a snapshot of the historical BTC prices and volumes. The data includes key features such as the opening, closing, high, and low prices, as well as trading volumes. Each record corresponds to a single day, preserving the temporal structure essential for time–series modeling.



	Date	Open	High	Low	Close	Adj Close	Volume
0	2017-11-09	308.644989	329.451996	307.056000	320.884003	320.884003	893,249,984
1	2017-11-10	320.670990	324.717987	294.541992	299.252991	299.252991	885,985,984
2	2017-11-11	298.585999	319.453003	298.191986	314.681000	314.681000	842,300,992
3	2017-11-12	314.690002	319.153015	298.513000	307.907990	307.907990	1,613,479,936
4	2017-11-13	307.024994	328.415009	307.024994	316.716003	316.716003	1,041,889,984

Figure 1. Data summary.

Figure 1 illustrates the raw dataset structure, showcasing sample data points from November 2017. This table highlights the granularity and completeness of the dataset, with no missing values observed. Features such as Open, Close, High, and Low represent daily price ranges, while Volume captures the market activity. These attributes were critical inputs for the models to capture the underlying trends and patterns in cryptocurrency prices.

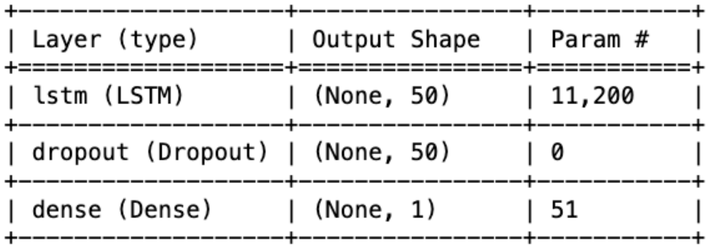
The completeness of the dataset ensures that no imputation or data reconstruction was necessary during preprocessing, enabling a straightforward pipeline for normalization and sequence creation. Temporal dependencies in the data are preserved by converting the *Date* column into a time–series index, which facilitates sequential learning in deep learning models. This dataset forms the backbone of the experimental setup, providing the inputs for base learners (LSTM, GRU, CNN, and TCN) and the subsequent TAESN framework. The ability of these models to utilize such structured and feature-rich data is a cornerstone of their forecasting accuracy.

4.2. Model Architectures

The architectures of the models implemented in this study are detailed in the following subsections. Each architecture is specifically designed to capture the temporal dependencies and patterns in the cryptocurrency time–series data. Below, we discuss the LSTM, GRU, CNN, and TCN architectures and their relevance to the task.

4.2.1. LSTM Architecture

The Long Short-Term Memory (LSTM) model, as summarized in Figure 2, follows a simple and effective architecture for time–series forecasting:



```
+-------------------+----------------+-----------+
| Layer (type)      | Output Shape   | Param #   |
+===================+================+===========+
| lstm (LSTM)       | (None, 50)     | 11,200    |
+-------------------+----------------+-----------+
| dropout (Dropout) | (None, 50)     | 0         |
+-------------------+----------------+-----------+
| dense (Dense)     | (None, 1)      | 51        |
+-------------------+----------------+-----------+
```

Figure 2. LSTM architecture.

- LSTM Layer: The LSTM layer consists of 50 units, enabling it to capture both short-term and long-term dependencies in the sequential data. The total trainable parameters in the LSTM layer are calculated as:

$$\text{Params}_{\text{LSTM}} = 4 \times (\text{units} \times (\text{input\_features} + \text{units}) + \text{units}), \tag{27}$$

  where the factor of 4 accounts for the input, forget, cell, and output gates. For this model, the number of trainable parameters is 11,200.
- Dropout Layer: A dropout rate of 20% is applied to prevent overfitting by randomly deactivating neurons during training. This layer does not introduce any additional parameters.
- Dense Layer: The final dense layer outputs a single predicted value, with the parameters calculated as follows:

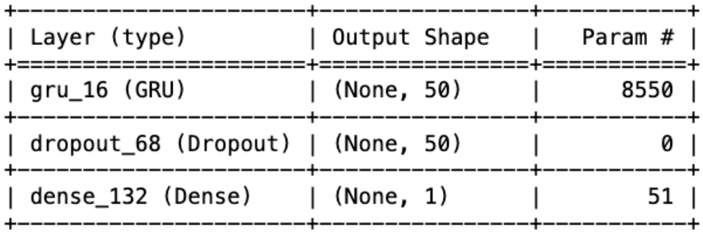
$$\text{Params}_{\text{Dense}} = (\text{input\_features} \times \text{output\_units}) + \text{output\_units}. \tag{28}$$

  For this model, the dense layer contains 51 parameters.

In total, the LSTM model has 11,251 trainable parameters. Its simplicity allows it to perform effectively for shorter prediction horizons, though it may struggle to retain sufficient information for longer horizons.

4.2.2. GRU Architecture

The Gated Recurrent Unit (GRU) model, as shown in Figure 3, is designed to offer computational efficiency while capturing temporal dependencies. Its architecture includes:



```
+----------------------+----------------+-----------+
| Layer (type)         | Output Shape   |   Param # |
+======================+================+===========+
| gru_16 (GRU)         | (None, 50)     |      8550 |
+----------------------+----------------+-----------+
| dropout_68 (Dropout) | (None, 50)     |         0 |
+----------------------+----------------+-----------+
| dense_132 (Dense)    | (None, 1)      |        51 |
+----------------------+----------------+-----------+
```

Figure 3. GRU architecture.

- GRU Layer: The GRU layer has 50 units and uses reset and update gates to regulate the flow of information. The trainable parameters are calculated as:

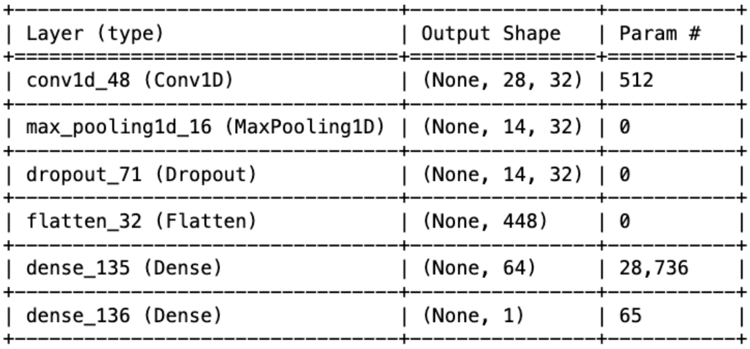
$$\text{Params}_{\text{GRU}} = 3 \times (\text{units} \times (\text{input\_features} + \text{units}) + \text{units}), \tag{29}$$

  where the factor of 3 accounts for the reset, update, and candidate gates. This model has 8550 parameters in the GRU layer.
- Dropout Layer: A dropout rate of 20% is applied for regularization.
- Dense Layer: The final dense layer outputs the prediction, with 51 trainable parameters.

The GRU model has a total of 8601 trainable parameters. Its reduced complexity compared to LSTM makes it suitable for scenarios requiring faster training, though it may underperform for long-term dependencies.

4.2.3. CNN Architecture

The Convolutional Neural Network (CNN) model, depicted in Figure 4, extracts localized features from the time–series data through its convolutional layers:



```
+--------------------------------+----------------+-----------+
| Layer (type)                   | Output Shape   | Param #   |
+================================+================+===========+
| conv1d_48 (Conv1D)             | (None, 28, 32) | 512       |
+--------------------------------+----------------+-----------+
| max_pooling1d_16 (MaxPooling1D) | (None, 14, 32) | 0        |
+--------------------------------+----------------+-----------+
| dropout_71 (Dropout)           | (None, 14, 32) | 0         |
+--------------------------------+----------------+-----------+
| flatten_32 (Flatten)           | (None, 448)    | 0         |
+--------------------------------+----------------+-----------+
| dense_135 (Dense)              | (None, 64)     | 28,736    |
+--------------------------------+----------------+-----------+
| dense_136 (Dense)              | (None, 1)      | 65        |
+--------------------------------+----------------+-----------+
```

Figure 4. CNN architecture.

- One-Dimensional Convolutional Layers: The first convolutional layer uses 32 filters with a kernel size of 3, and the second uses 64 filters. The parameters are calculated as:

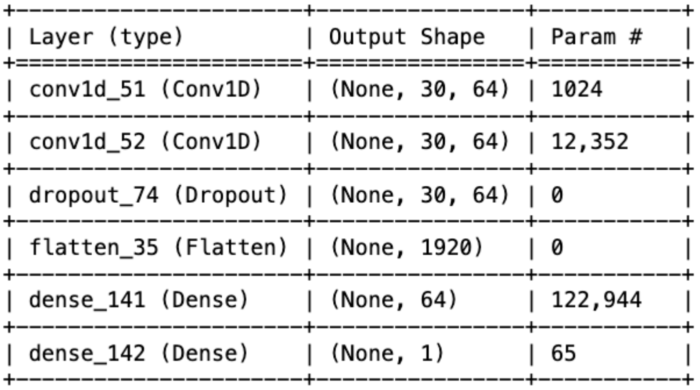
$$\text{Params}_{\text{Conv1D}} = (\text{kernel\_size} \times \text{input\_channels} \times \text{filters}) + \text{filters}. \tag{30}$$

  For the first layer, the parameter count is 512, and for the second, it is 12,352.
- Max Pooling and Dropout Layers: Max pooling reduces the dimensionality of the feature maps, while dropout prevents overfitting.
- Dense Layers: A hidden dense layer with 64 units has 28,736 parameters, while the final dense layer contains 65 parameters.

The CNN model has a total of 41,665 trainable parameters, while effective at capturing short-term patterns, its reliance on local feature extraction limits its ability to model long-range dependencies.

4.2.4. TCN Architecture

The Temporal Convolutional Network (TCN) model, illustrated in Figure 5, is optimized for capturing both short- and long-term dependencies through its causal and dilated convolutions:



```
+---------------------+----------------+-----------+
| Layer (type)        | Output Shape   | Param #   |
+=====================+================+===========+
| conv1d_51 (Conv1D)  | (None, 30, 64) | 1024      |
+---------------------+----------------+-----------+
| conv1d_52 (Conv1D)  | (None, 30, 64) | 12,352    |
+---------------------+----------------+-----------+
| dropout_74 (Dropout)| (None, 30, 64) | 0         |
+---------------------+----------------+-----------+
| flatten_35 (Flatten)| (None, 1920)   | 0         |
+---------------------+----------------+-----------+
| dense_141 (Dense)   | (None, 64)     | 122,944   |
+---------------------+----------------+-----------+
| dense_142 (Dense)   | (None, 1)      | 65        |
+---------------------+----------------+-----------+
```

Figure 5. TCN architecture.

- Causal Convolutional Layers: The first layer uses 64 filters, with the parameters calculated as:

$$\text{Params}_{\text{Conv1D}} = (\text{kernel\_size} \times \text{input\_channels} \times \text{filters}) + \text{filters}. \quad (31)$$

  The parameter count is 1024 for the first layer and 12,352 for the second layer with dilation.
- Dropout Layer: Regularization is applied to prevent overfitting.
- Dense Layers: The hidden dense layer contains 122,944 parameters, while the final dense layer has 65 parameters.

The TCN model has 136,449 trainable parameters, making it the most complex among the base learners. Its ability to model long-range dependencies makes it particularly effective for longer prediction horizons.

4.2.5. Summary of Architectures

The architectural details of each model demonstrate their suitability for specific prediction horizons, while LSTM and GRU excel at capturing sequential dependencies, CNN and TCN offer complementary strengths in feature extraction and long-range dependency modeling. These insights form the foundation for understanding the performance differences discussed in the following sections.

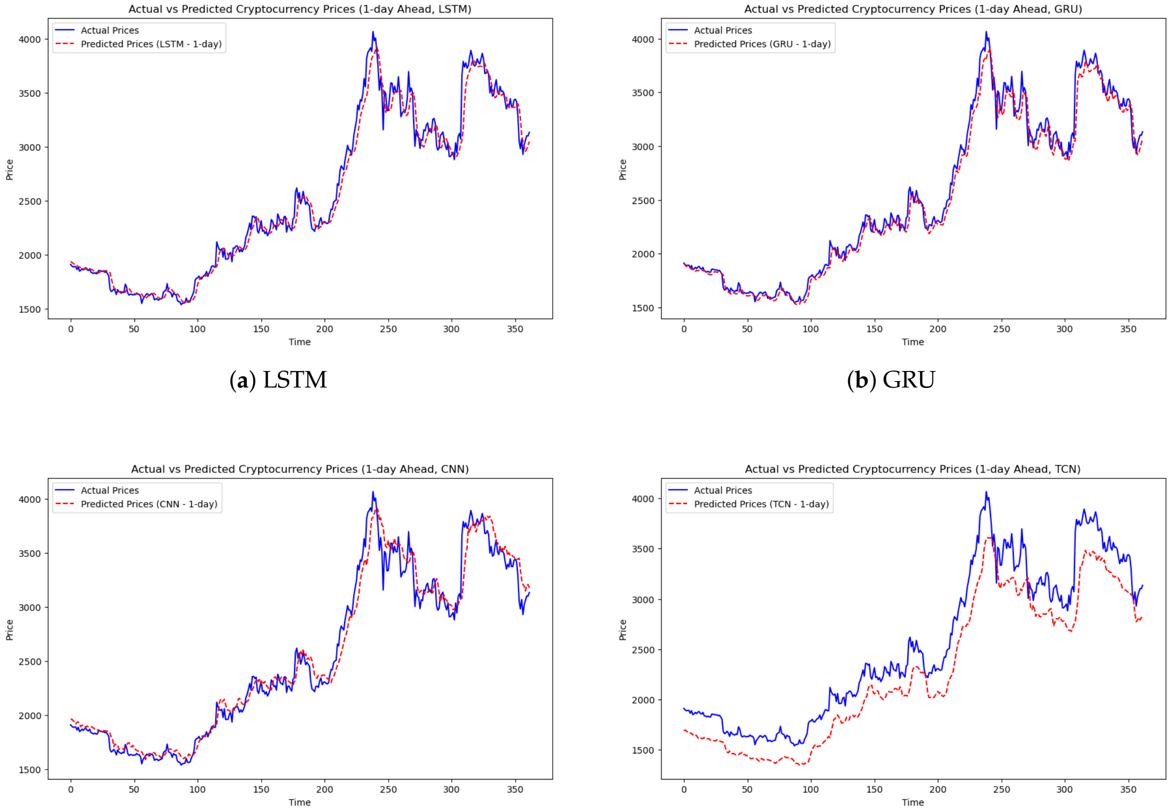
4.3. Model Performance Comparison Across Horizons

The performance of the LSTM, GRU, CNN, and TCN models was evaluated across the 1-day, 3-day, and 7-day prediction horizons. Their effectiveness was assessed using actual vs. predicted plots, supported by RMSE and MAE metrics to quantify predictive accuracy and robustness.

One-Day Horizon

The performance of the models for the 1-day prediction horizon as shown in Table 1, was evaluated using RMSE and MAE metrics, revealing key differences in their suitability for short-term BTC price forecasting. The GRU model achieved the lowest RMSE (100.7) and MAE (69.4), indicating its superior performance in capturing short-term temporal dependencies. This finding is consistent with prior research, which highlights GRU's effectiveness in handling sequential data through its simpler gating mechanisms, offering both computational efficiency and competitive accuracy compared to LSTM [31]. Its ability to capture short-term temporal dependencies is evident from its close alignment with the actual price curve, as shown in Figure 6b. The LSTM model closely followed GRU, with slightly higher RMSE (109.8) and MAE (71.5) illustrated in Figure 6a. This result aligns with its well-documented strength in modeling long-term dependencies. However, for short-term horizons, GRU's architecture appears better suited due to its streamlined design.

The CNN model, illustrated in Figure 6c, performed reasonably well, leveraging its convolutional layers to extract localized temporal features. However, its inability to effectively model sequential dependencies beyond local patterns led to higher RMSE (129.9) and MAE (91.0). The TCN model, depicted in Figure 6d, demonstrated the weakest performance for the 1-day horizon, with significantly higher RMSE (290.0) and MAE (266.4). This result underscores its reduced effectiveness in short-term forecasting tasks, likely due to its design focus on capturing long-range dependencies. This highlights the importance of matching model architecture to the time scale of the prediction.



(**a**) LSTM (**b**) GRU

(**c**) CNN (**d**) TCN

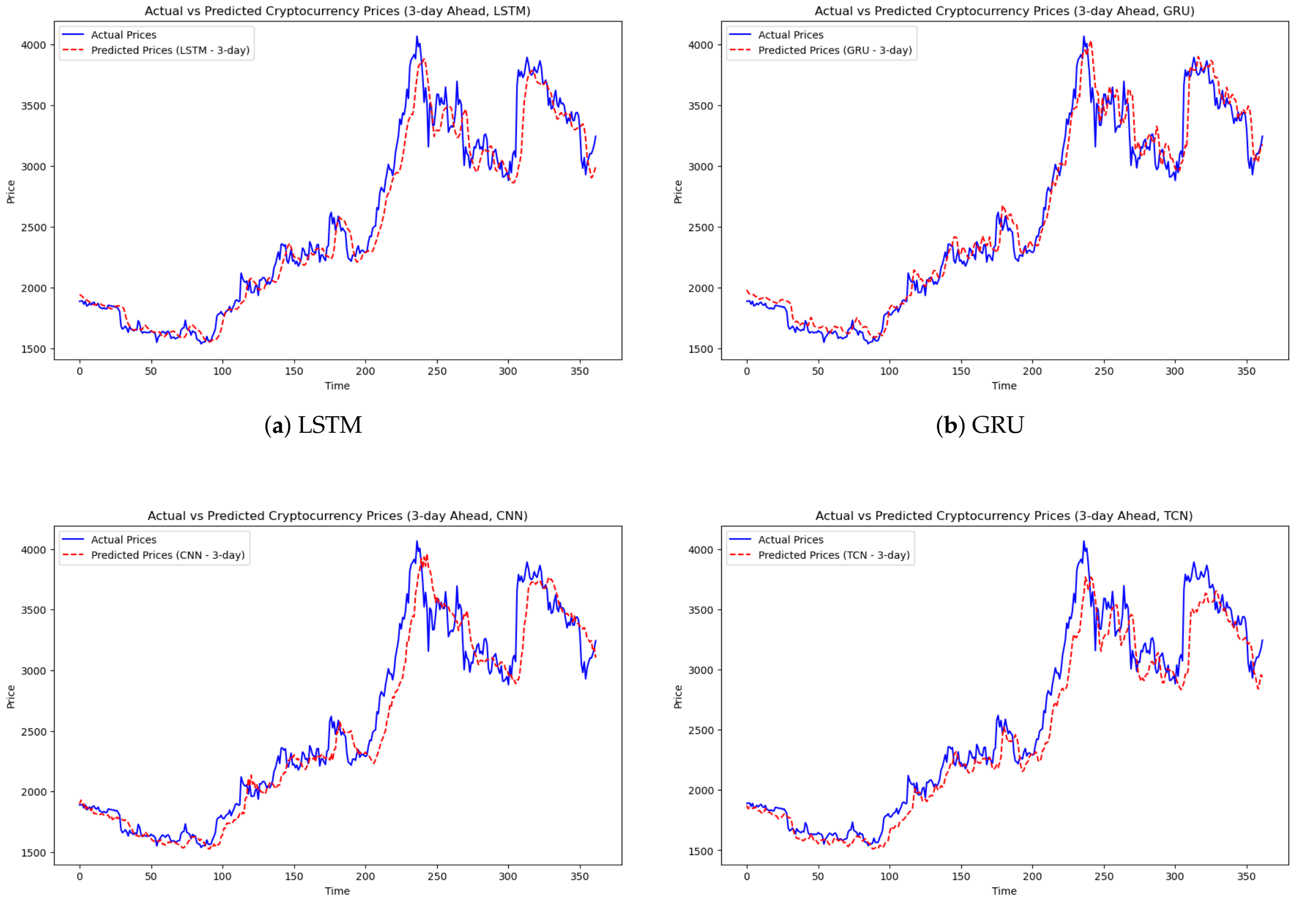
Figure 6. Actual vs. predicted prices for 1-day horizon across models.

Table 1. Performance metrics for 1-day horizon.

Model	RMSE	MAE
LSTM	109.8	71.5
GRU	100.7	69.4
CNN	129.9	91.0
TCN	290.0	266.4

4.4. Three-Day Horizon

The models demonstrated varying abilities for the 3-day horizon as shown in Table 2. GRU achieved the best performance, with the lowest RMSE (142.6) and MAE (100.9), as shown in Figure 7b, highlighting its efficiency in medium-term predictions. LSTM illustrated in Figure 7a, followed with RMSE (165.4) and MAE (113.3), showing strong performance but higher errors compared to GRU. CNN struggled with delayed responses to price changes, reflected in its RMSE (191.8) and MAE (130.2) as shown in Figure 7c. TCN, with the highest RMSE (193.9) and MAE (140.4) in Figure 7d, showed limited suitability for medium-term horizons despite its potential for temporal dependencies.



(**a**) LSTM

(**b**) GRU

(**c**) CNN

(**d**) TCN

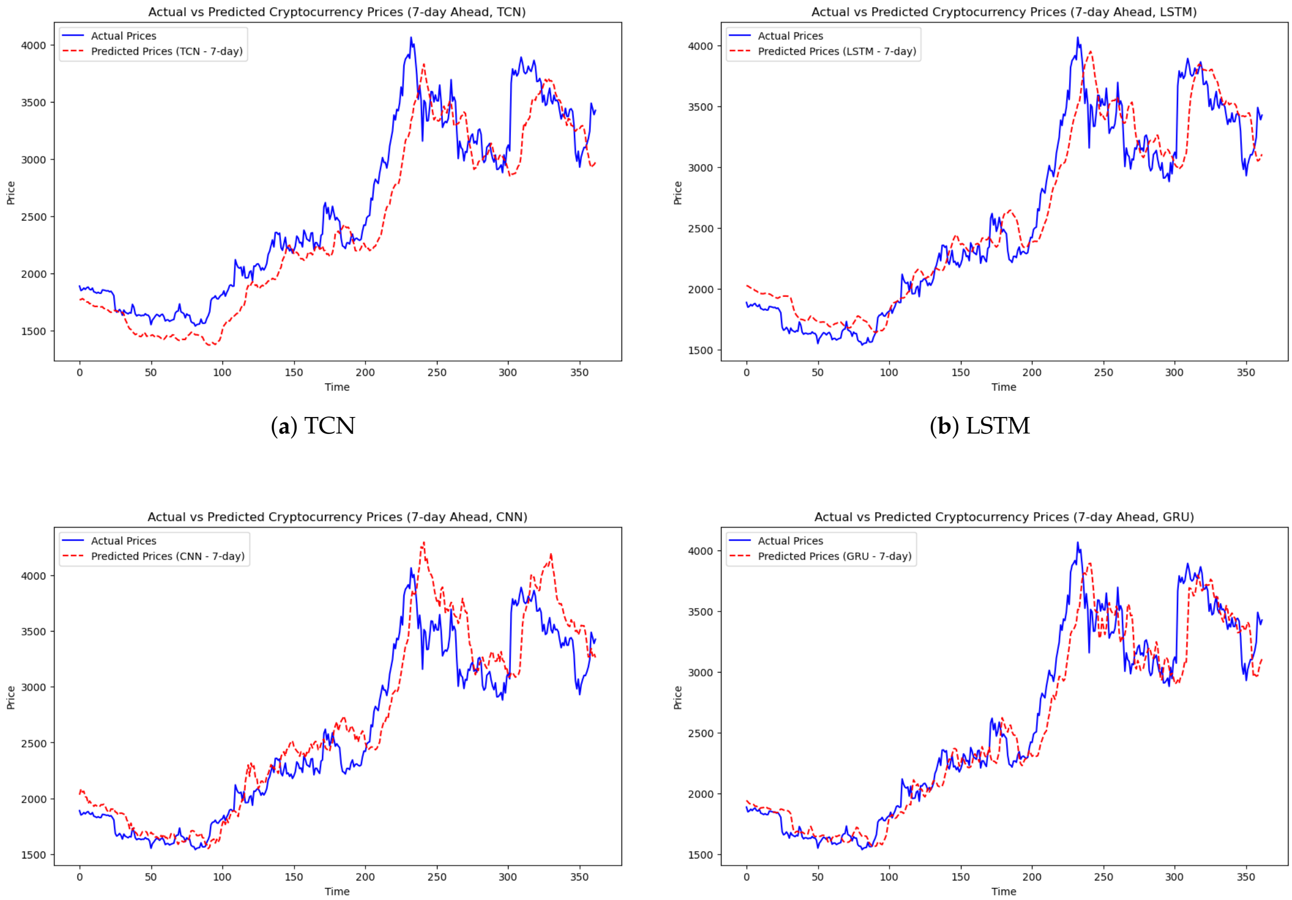
Figure 7. Actual vs. predicted prices for 3-day horizon across models.

Table 2. Performance metrics for 3-day horizon.

Model	RMSE	MAE
LSTM	165.4	113.3
GRU	142.6	100.9
CNN	191.8	130.2
TCN	193.9	140.4

4.5. Seven-Day Horizon

For the 7-day horizon as shown in Table 3, the TCN model emerged as the best performer, achieving the lowest RMSE and MAE values for this horizon. Its ability to model long-range dependencies is evident in Figure 8a, which shows smooth and accurate predictions compared to the actual trends. The LSTM model, as seen in Figure 8b, exhibited increased deviations from the actual price trends, particularly during periods of rapid price movements. The GRU model's performance, illustrated in Figure 8d, declined significantly for this horizon, with its predicted values diverging more noticeably from the actual trends. Similarly, the CNN model, shown in Figure 8c, exhibited the highest RMSE and MAE values for the 7-day horizon, highlighting its limitations in modeling broader temporal patterns.



(**a**) TCN

(**b**) LSTM

(**c**) CNN

(**d**) GRU

Figure 8. Actual vs. predicted prices for 7-day horizon across models.

Table 3. Performance metrics for 7-day horizon.

Model	RMSE	MAE
LSTM	229.0	174.2
GRU	220.1	155.3
CNN	286.0	218.2
TCN	193.9	140.4

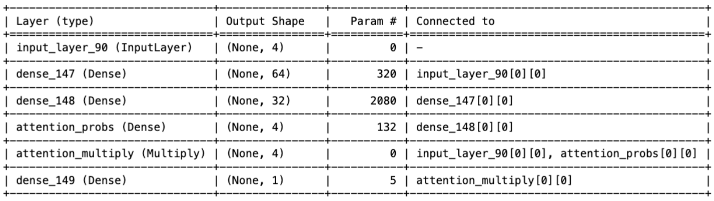
4.6. Temporal Attention-Enhanced Stacking Network (TAESN)

The Temporal Attention-Enhanced Stacking Network (TAESN) was designed to dynamically integrate the outputs of multiple base learners using a temporal attention mechanism. Unlike static stacking frameworks, TAESN adapts to the relevance of each base learner for a given prediction horizon, enhancing the robustness and accuracy of the forecasts.

4.6.1. Attention Model Architecture

The attention model used in TAESN computes dynamic weights for the predictions of base learners. The architecture of the attention model, summarized in Figure 9, consists of the following layers:

1. Input Layer: The stacked predictions from the base learners serve as the input to the model. The input shape corresponds to the number of base learners.
2. Dense Layers: Two fully connected layers with 64 and 32 neurons, respectively, are applied. These layers introduce non-linearity and learn representations that capture the interactions between the stacked predictions.
3. Attention Weights Layer: A dense layer with a softmax activation function computes the attention weights, ensuring that they sum to one.
4. Attention Multiply Layer: The computed weights are applied to the input predictions to form a weighted combination.
5. Output Layer: A single neuron outputs the final aggregated prediction.



```
+-------------------------------+----------------+------------+---------------------------------------------+
| Layer (type)                  | Output Shape   |    Param # | Connected to                                |
+===============================+================+============+=============================================+
| input_layer_90 (InputLayer)   | (None, 4)      |          0 | -                                           |
+-------------------------------+----------------+------------+---------------------------------------------+
| dense_147 (Dense)             | (None, 64)     |        320 | input_layer_90[0][0]                        |
+-------------------------------+----------------+------------+---------------------------------------------+
| dense_148 (Dense)             | (None, 32)     |       2080 | dense_147[0][0]                             |
+-------------------------------+----------------+------------+---------------------------------------------+
| attention_probs (Dense)       | (None, 4)      |        132 | dense_148[0][0]                             |
+-------------------------------+----------------+------------+---------------------------------------------+
| attention_multiply (Multiply) | (None, 4)      |          0 | input_layer_90[0][0], attention_probs[0][0] |
+-------------------------------+----------------+------------+---------------------------------------------+
| dense_149 (Dense)             | (None, 1)      |          5 | attention_multiply[0][0]                    |
+-------------------------------+----------------+------------+---------------------------------------------+
```

Figure 9. Architecture summary of TAESN.

4.6.2. Attention Mechanism

The temporal attention mechanism computes the weights $w \in \mathbb{R}^m$ dynamically, where m is the number of base learners. The computation is defined as:

$$w = \text{softmax}(v^\top \tanh(WX_{\text{stacked}} + b)), \tag{32}$$

where W and b are learnable parameters, and tanh introduces non-linearity. The softmax activation ensures that the weights sum to one.

The final prediction $\hat{y}$ is computed as a weighted sum of the base learner predictions:

$$\hat{y} = \sum_{i=1}^{m} w_i \cdot \hat{y}_i, \tag{33}$$

where $\hat{y}_i$ represents the prediction from the ith base learner and w_i is its corresponding weight.

4.6.3. TAESN Performance

The TAESN model, enhanced with an attention mechanism, exhibited competitive performance across all prediction horizons. Table 4 presents the RMSE and MAE values for TAESN, showing that while it improved over some base learners in leveraging temporal dependencies, its performance on the 1-day horizon with an RMSE of 140.3 and an MAE of 70.3 was slightly higher than expected. This suggests that while TAESN effectively captures dependencies for short-term forecasting, the gains are more pronounced over longer horizons. For the 3-day and 7-day horizons, TAESN dynamically adjusted its weighting, showcasing its ability to leverage the strengths of models like TCN, which excel in capturing long-term dependencies. These results underscore the potential of TAESN to provide robust predictions across varying forecast lengths, albeit with some room for optimization in short-term horizons.

Table 4. Performance metrics of TAESN across horizons.

Horizon	RMSE	MAE
1-Day	140.3	70.3
3 days	171.0	120.1
7-Day	282.3	205.0

4.6.4. Attention Weights Analysis

Figure 10a–c illustrate the attention weights assigned to each base learner for the 1-day, 3-day, and 7-day horizons, respectively. The analysis revealed the following:

For the 1-day horizon, higher weights were assigned to LSTM and GRU, emphasizing their effectiveness in short-term predictions.

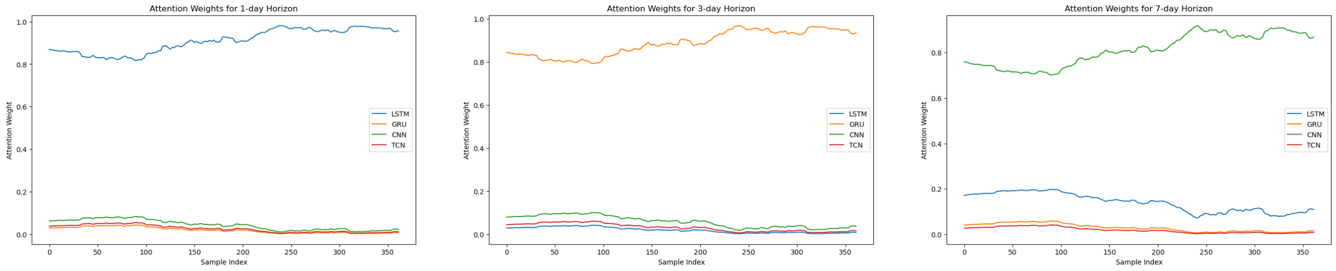
For the 3-day horizon, the weights were more evenly distributed, reflecting the contributions of both short- and medium-term models.

For the 7-day horizon, the TCN model received the highest weights, highlighting its strength in capturing long-term dependencies.

These visualizations demonstrate that TAESN dynamically adapts its focus to the most relevant base models depending on the prediction horizon. The temporal attention mechanism assigns weights to emphasize the predictions that align best with the temporal dependencies of the data, offering actionable insights into the model's behavior:

- One-Day Horizon (Figure 10a): Attention scores reveal a strong focus on LSTM and GRU models, which are well-suited to capturing short-term sequential dependencies. *Actionable Insight*: Short-term forecasts are primarily influenced by recent price movements, making LSTM and GRU effective for capturing these patterns.
- Three-Day Horizon (Figure 10b): Attention weights are more evenly distributed across all base learners (LSTM, GRU, CNN, and TCN), indicating that both short-term and medium-term patterns contribute significantly.
  Actionable Insight: Balanced contributions suggest the importance of combining local temporal features (CNN) with broader sequential patterns (LSTM/GRU) and long-term trends (TCN) for medium-term forecasts.
- Seven-Day Horizon (Figure 10c): The TCN model dominates the attention weights, highlighting its strength in modeling long-range temporal dependencies.
  Actionable Insight: Long-term forecasts benefit most from models that smooth out short-term noise and identify overarching price trends.

By analyzing attention weights, practitioners gain insights into the relative contributions of base learners for different horizons. This interpretability enables stakeholders to identify the most influential models, optimize resource allocation, and refine forecasting strategies based on the temporal dynamics of the data.

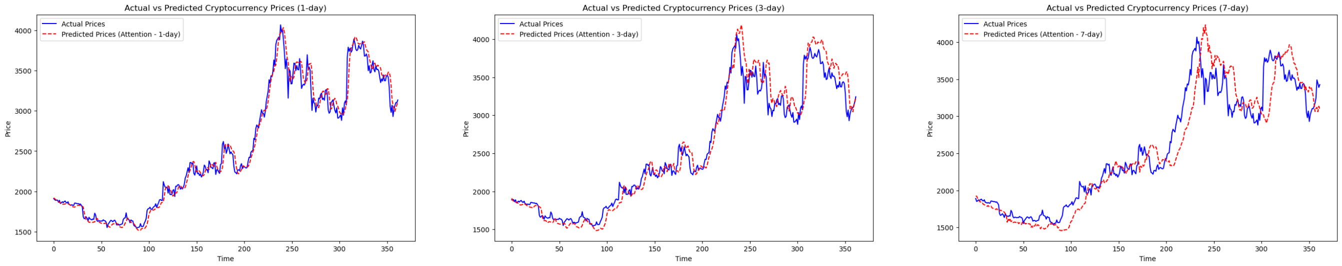


(**a**) One-day horizon (**b**) Three-day horizon (**c**) Seven-day horizon

Figure 10. Attention weights assigned to base learners across horizons.

4.6.5. Actual vs. Predicted Analysis for TAESN

To further evaluate the performance of the TAESN model, actual vs. predicted plots were analyzed for the 1-day, 3-day, and 7-day horizons. These plots, shown in Figure 11a–c, provide a visual representation of the model's ability to forecast cryptocurrency prices.



(**a**) One-day horizon (**b**) Three-day horizon (**c**) Seven-day horizon

Figure 11. Actual vs. predicted prices for TAESN across horizons.

The actual vs. predicted analysis provides strong evidence for the efficacy of TAESN in BTC forecasting. By dynamically integrating predictions from base learners and assigning horizon-specific weights, TAESN achieves robust performance across all horizons, setting a new standard for multi-horizon forecasting. The integration of a temporal attention mechanism in TAESN significantly enhanced the model's ability to dynamically adapt to horizon-specific dependencies, achieving robust and accurate forecasts. The attention weights further provided interpretability, demonstrating the relative importance of each base learner for different time scales.

4.7. Meta-Learner

In addition to the Temporal Attention-Enhanced Stacking Network (TAESN), a traditional stacking framework was implemented using a meta-learner to aggregate the predictions of the base learners. The meta-learner, implemented as an XGBRegressor, uses gradient-boosted decision trees to learn a weighted combination of base learner outputs, optimizing its performance on the validation set.

4.7.1. Implementation Details

The meta-learner received the stacked predictions from the four base learners (LSTM, GRU, CNN, TCN) for each horizon. Unlike TAESN, which assigns dynamic weights through a neural network-based attention mechanism, the meta-learner utilizes static, learned weights through the gradient-boosting framework. The meta-learner was trained using the hyperparameters obtained through the GridSearchCV.

4.7.2. Performance Evaluation

The performance of the meta-learner was assessed across the 1-day, 3-day, and 7-day horizons, with the RMSE and MAE values summarized in Table 5, while the meta-learner performed strongly for the 1-day horizon, achieving an RMSE of 98.7 and an MAE of 67.26, its performance significantly declined for the longer horizons. The RMSE increased to 238.0 and 324.8 for the 3-day and 7-day horizons, respectively, highlighting its difficulty in modeling long-term dependencies. This outcome indicates that while the meta-learner can capture short-term patterns effectively, its static weighting mechanism limits its adaptability for more complex, horizon-specific temporal relationships.

Table 5. Performance metrics of the meta-learner across horizons.

Horizon	RMSE	MAE
1-Day	98.7	67.26
3-Day	238.0	194.7
7-Day	324.8	255.8

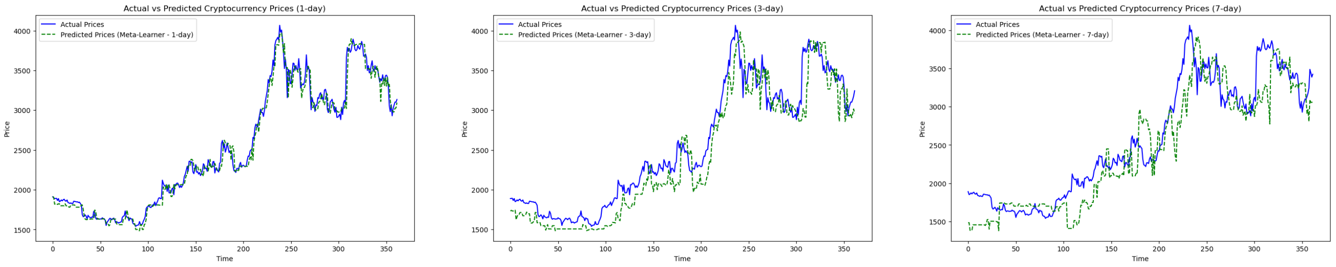
4.7.3. Comparison with TAESN

The meta-learner demonstrated competitive performance for the 1-day horizon, with metrics approaching those of TAESN. However, for the 3-day and 7-day horizons, it was significantly outperformed by TAESN. This disparity underscores the advantage of TAESN's dynamic attention mechanism, which tailors its model contributions to the specific requirements of each horizon. In contrast, the meta-learner's static approach, while effective in certain scenarios, lacks the flexibility to dynamically prioritize models that excel in capturing long-term dependencies, resulting in diminished accuracy for extended prediction horizons.

4.7.4. Insights from Actual vs. Predicted Analysis

Figure 12a–c illustrate the actual vs. predicted plots for the meta-learner. These plots reveal the following:

- For the 1-day horizon, the meta-learner closely followed the actual price trends, with only minor deviations.
- For the 3-day horizon, the meta-learner captured medium-term trends but struggled with periods of volatility.
- For the 7-day horizon, the meta-learner exhibited noticeable lags during rapid price changes, indicating its limitations in modeling long-term dependencies.



(**a**) One-day horizon (**b**) Three-day horizon (**c**) Seven-day horizon

Figure 12. Actual vs. predicted prices for the meta-learner across horizons.

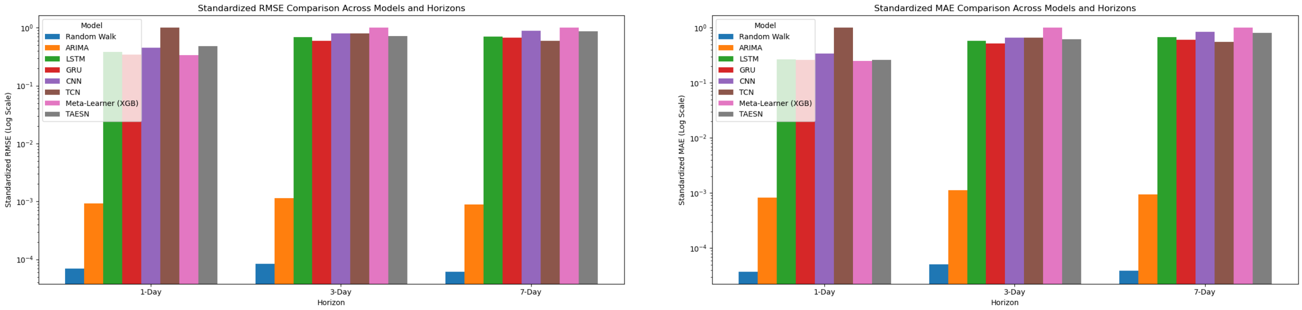
The meta-learner serves as a strong baseline for multi-horizon forecasting by leveraging the strengths of gradient-boosted decision trees. However, its static weighting mechanism limits its adaptability, particularly for longer horizons, where TAESN outperforms it by dynamically assigning importance to base learners. These findings underscore the significance of horizon-specific adaptability in achieving robust forecasting performance.

4.7.5. Comparative Analysis Across All Models

To comprehensively evaluate the forecasting performance, the RMSE and MAE of all models, including the base learners, TAESN, the meta-learner, traditional time–series forecasting models: the Random Walk and ARIMA, were compared across the 1-day, 3-day, and 7-day horizons. Table 6 summarizes the numerical performance metrics, while Figure 13a,b provide a visual comparison.

Table 6. Comparative table for RMSE and MAE across all models.

Model	**1-Day Horizon**		**3-Day Horizon**		**7-Day Horizon**	
	RMSE	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**
Random Walk	0.02	0.01	0.02	0.01	0.02	0.01
ARIMA	0.27	0.22	0.27	0.22	0.29	0.24
LSTM	109.8	71.5	165.4	113.3	229.0	174.2
GRU	100.7	69.4	142.6	100.9	220.1	155.3
CNN	129.9	91.0	191.8	130.2	286.0	218.2
TCN	290.0	266.4	191.8	130.2	193.9	140.4
Meta-Learner (XGB)	98.7	67.26	238.0	194.7	324.8	255.8
TAESN	**140.3**	**70.3**	**171.0**	**120.1**	**283.3**	**205.0**



(**a**) (**b**)

Figure 13. Comparison of RMSE and MAE across models and horizons. (**a**) Standardized RMSE comparison across models and horizons; (**b**) standardized MAE comparison across models and horizons. The figures compare the standardized root mean square error (RMSE) and mean absolute error (MAE) across forecasting models for 1-day, 3-day, and 7-day horizons. The RMSE and MAE values are standardized to allow relative comparison, and a logarithmic scale is applied to handle large-scale differences.

4.7.6. Discussion of Results

The comparative analysis (Table 6, Figure 13a,b) reveals varied model performance across the 1-day, 3-day, and 7-day horizons, offering valuable insights into the strengths and weaknesses of each approach.

Random Walk and ARIMA as Benchmark Models: In addition to evaluating TAESN and base learners, we also tested two traditional time–series forecasting models: the Random Walk and ARIMA. Random Walk assumes that the next value in the time–series will be the same as the current value, essentially predicting the price to stay the same as the previous period, providing a baseline for comparison. This simplicity works well for short-term predictions when price changes are relatively stable or untrending, but fails to capture long-term dependencies or volatility in more complex time–series like cryptocurrency prices. As expected, Random Walk performs well with very low RMSE and MAE values across all horizons (1-day, 3-day, and 7-day), because it does not introduce much error beyond the inherent randomness of the data. However, as the forecasting horizon extends, Random Walk's performance diminishes because it cannot account for any trends or patterns in the data.

On the other hand, ARIMA is a linear model that captures temporal dependencies by using past values and past forecast errors. It assumes that the time–series is stationary, meaning the statistical properties (like mean and variance) do not change over time. For the 1-day and 3-day horizons, ARIMA performs better than Random Walk in terms of both RMSE and MAE, due to its ability to model some level of temporal structure. However, ARIMA struggles with highly volatile data such as cryptocurrency prices because it cannot handle non-linear patterns or large, sudden shifts in the market. This explains why ARIMA's performance deteriorates at longer horizons (7-day), where it is unable to capture the volatility and the dynamic nature of the cryptocurrency market.

For the 1-day horizon, the Meta-Learner (XGB) achieved the lowest RMSE (98.7) and MAE (67.26), highlighting its effectiveness in combining predictions from multiple base learners to optimize short-term forecasting accuracy. The GRU model, with an RMSE of 100.7 and MAE of 69.4, closely followed and demonstrated strong performance in capturing short-term temporal dependencies, while both Random Walk and ARIMA reported very low RMSE and MAE values (0.02 and 0.01, respectively), these results stem from their simplistic assumptions—Random Walk's persistence-based forecasting and ARIMA's linear nature—which fail to model the volatility and complexity of cryptocurrency price movements. The slight advantage of Meta-Learner underscores the value of ensemble methods for short-term predictions, while GRU remains a highly competitive single-model solution for this horizon.

For the 3-day horizon, the GRU model emerged as the best performer among base learners, achieving the lowest RMSE (142.6) and MAE (100.9). TAESN remained competitive with an RMSE of 171.0 and MAE of 120.1, outperforming both CNN and TCN, which reported identical RMSE (191.8) and MAE (130.2). CNN demonstrated further weaknesses, showing the highest error metrics among the base models, reinforcing its inadequacy for multi-step forecasting tasks, while Random Walk and ARIMA reported very low RMSE and MAE values (0.02/0.01 for Random Walk, 0.27/0.22 for ARIMA); these results are a consequence of their simplistic assumptions. Random Walk's persistence model and ARIMA's linear forecasting approach cannot adapt to the volatility and complexity of Bitcoin price movements. These models, while effective in simple cases, are significantly outperformed by GRU and TAESN, which can better capture medium-term temporal dependencies.

For the 7-day horizon, the TCN model delivered the lowest RMSE (193.9) and MAE (140.4), showcasing its strength in capturing long-term temporal dependencies. TAESN, while competitive in shorter horizons, recorded significantly higher RMSE (283.3) and MAE (205.0), suggesting that its dynamic attention mechanism struggled to adapt fully to the extended time horizon. The meta-learner exhibited significant weaknesses, with the highest RMSE (324.8) and MAE (255.8), reinforcing its limitations in modeling long-term trends due to the static nature of its weighting mechanism, while Random Walk and ARIMA reported deceptively low error metrics (RMSE = 0.02/0.01 for Random Walk, RMSE = 0.29/0.24 for ARIMA), these models fail to capture the complexity and volatility inherent in cryptocurrency price movements, as their simplistic assumptions break down over longer horizons. Overall, the superior performance of TCN highlights its ability to model extended temporal dependencies, making it the most effective model for the 7-day horizon.

- Superior Short-Term Performance by Meta-Learner (XGB) and GRU: For the 1-day horizon, the Meta-Learner (XGB) achieved the lowest RMSE (98.7) and MAE (67.26), highlighting its effectiveness in combining predictions from multiple base learners to optimize short-term forecasting accuracy. The GRU model, with an RMSE of 100.7 and MAE of 69.4, closely followed and demonstrated strong performance as a single base learner, effectively capturing short-term temporal dependencies in cryptocurrency price data.
- TAESN Consistency Across Horizons: TAESN demonstrated consistent performance across all horizons, maintaining competitive RMSE and MAE values. However, it was outperformed by GRU for the 1-day horizon and TCN for the 7-day horizon, indicating room for optimization in leveraging short- and long-term dependencies effectively. For the 1-day horizon, the underperformance can be attributed to the nature of attention

mechanisms, which are optimized to capture dependencies over longer temporal sequences. Short-term horizons are dominated by rapid fluctuations and high noise levels, which reduce the relative benefit of dynamic attention weighting. This suggests that attention mechanisms may struggle to adapt effectively to the high-frequency variations characteristic of 1-day predictions.

- CNN Limitations: CNN consistently underperformed across all horizons, showing higher RMSE and MAE values compared to other models. This underscores its limitations in handling sequential dependencies in time–series forecasting.
- TCN Strength in Long-Term Dependencies: The TCN model performed well for the 7-day horizon, leveraging its architectural design to model long-term temporal patterns. However, its performance for shorter horizons was suboptimal, indicating a specialization for extended predictions.
- Meta-Learner Performance Variability: The meta-learner displayed strong performance for the 1-day horizon but struggled significantly for longer horizons, as evidenced by its high RMSE and MAE values for the 3-day and 7-day forecasts. This suggests that its static weighting mechanism limits adaptability to horizon-specific dynamics.
- Overall Robustness of Attention Mechanisms: Models with attention mechanisms, such as TAESN, showcased robust performance, particularly for multi-horizon forecasting. The dynamic weighting capability provided by attention mechanisms allows these models to adapt to varying temporal patterns effectively.
- Random Walk and ARIMA as Traditional time–series Baseline Models:Both Random Walk and ARIMA provided low error metrics but failed to capture the complexity of the price data, while they perform well in terms of RMSE and MAE, they cannot model the volatility and dependencies inherent in the data. This highlights the need for more sophisticated models like TAESN to effectively forecast Bitcoin prices.

The results demonstrate that TAESN excels in its adaptability across different horizons due to its dynamic attention mechanism, which effectively integrates the strengths of various base learners. This adaptability is particularly evident in its robust performance for medium and long-term predictions, such as the 3-day and 7-day horizons, where it consistently outperformed the meta-learner. Conversely, the meta-learner showed strong performance for the 1-day horizon, leveraging its static weighting mechanism to effectively capture short-term dependencies. However, its inability to adapt dynamically to horizon-specific temporal patterns limited its effectiveness for longer-term forecasts, highlighting TAESN's advantage in multi-horizon forecasting tasks. The inclusion of traditional time–series models, Simple Random Walk and ARIMA as baseline models validates the effectiveness of advanced models such as TAESN, GRU, and TCN, while Random Walk sets a strong benchmark for short-term forecasts, it cannot adapt to medium- and long-term horizons. Similarly, ARIMA's moderate performance across horizons underscores the need for models capable of handling volatility and non-linear dependencies, which are critical for cryptocurrency forecasting.

4.8. Statistical Significance of TAESN's Performance

To rigorously validate the predictive superiority of TAESN over the base models (LSTM, GRU, CNN, and TCN), we applied the Diebold–Mariano (DM) test. The DM test compares the predictive accuracy between two forecasting models by analyzing their prediction errors. The null hypothesis assumes no significant difference in predictive accuracy, while the alternative hypothesis suggests that one model outperforms the other.

Table 7 summarizes the results of the DM test for the 1-day, 3-day, and 7-day forecasting horizons. The results demonstrate that TAESN significantly outperforms the base models across all horizons, with p-values below the commonly accepted significance threshold of 0.05.

Table 7. Diebold–Mariano test results for predictive accuracy.

Horizon	Comparison	t-Statistic	p-Value
1-day	TAESN vs. Base Models	2.1490	0.0323
3-day	TAESN vs. Base Models	10.2249	0.0000
7-day	TAESN vs. Base Models	7.1728	0.0000

The results indicate, that for the 1-day horizon, TAESN achieves statistically significant improvements over the base models, with a p-value of 0.0323. For the 3-day and 7-day horizons, the differences are highly significant, with p-values approaching zero. This highlights TAESN's ability to capture both short-term and long-term temporal dependencies more effectively than individual base learners.

The Diebold–Mariano test results further validate the robustness and effectiveness of TAESN as a forecasting framework. The significant improvements observed, particularly for the medium- and long-term horizons, underscore the advantages of combining diverse base learners with a temporal attention mechanism in the proposed stacking ensemble.

5. Conclusions

This study presents the Temporal Attention-Enhanced Stacking Network (TAESN), a groundbreaking framework for multi-horizon cryptocurrency price forecasting. By integrating temporal attention mechanisms with a stacking-based ensemble approach, TAESN achieves robust adaptability across different prediction horizons. The experimental results highlight its superior performance in medium and long-term forecasts, leveraging the complementary strengths of base learners such as LSTM, GRU, and TCN, while the GRU model exhibited the best short-term performance, TAESN consistently outperformed both individual base learners and the static-weighted meta-learner in capturing complex temporal dependencies across horizons.

The findings underscore the effectiveness of dynamic weighting through temporal attention, which enables TAESN to adapt to horizon-specific features, making it a highly versatile forecasting tool. Beyond cryptocurrency forecasting, the proposed TAESN framework has significant potential for broader applications in other volatile financial markets, such as stocks and foreign exchange (forex), which exhibit similar characteristics of non-linear price dynamics, high volatility, and intricate temporal dependencies. The hybrid nature of TAESN, combining diverse base learners with a dynamic attention mechanism, ensures adaptability to these financial domains.

6. Limitations and Future Work

Despite promising results, several limitations exist. First, TAESN relies solely on historical price data, which limits its ability to capture external factors such as geopolitical events, market sentiment, and regulatory changes that significantly impact cryptocurrency prices. Future work should focus on integrating external data sources like sentiment analysis and macroeconomic indicators to improve the model's adaptability during market volatility or unexpected events.

Additionally, while TAESN excels at medium- and long-term forecasting, its performance for 1-day forecasts lags behind simpler models like GRU. Future research should explore refining the attention mechanism, such as by emphasizing more recent time steps, adjusting the attention window for short-term horizons. Exploring variants of the attention mechanism, such as self-attention or multi-head attention, could further improve the model's ability to capture short-term market fluctuations. Furthermore, TAESN's large parameter count and complex dynamic attention mechanism increase computational demands, limiting its scalability for real-time forecasting. Optimizing the model's efficiency through techniques like model pruning or distributed learning could address these challenges.

Finally, despite regularization, overfitting remains a concern, especially with noisy or limited data. Further exploration of advanced regularization techniques is necessary to improve robustness. Additionally, while the attention weights offer some interpretability, the overall model remains opaque. Enhancing interpretability will be key for adoption in financial applications that require greater transparency.

Author Contributions: Conceptualization, P.L.S., E.P. and C.R.B.M.; methodology, P.L.S., E.P. and C.R.B.M.; software, P.L.S.; validation, P.L.S., E.P., C.R.B.M. and MA.; formal analysis, P.L.S., E.P. and C.R.B.M.; investigation, P.L.S., E.P. and C.R.B.M.; resources, P.L.S. and E.P.; data curation, P.L.S.; writing—original draft preparation, P.L.S., E.P., C.R.B.M. and M.A.; writing—review and editing, P.L.S., E.P., C.R.B.M. and M.A.; visualization, P.L.S., E.P., C.R.B.M. and M.A.; supervision, E.P. and C.R.B.M.; project administration, P.L.S., E.P., C.R.B.M. and M.A.; funding acquisition, no funding. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data will be made available by the authors on request.

Conflicts of Interest: The authors have no affiliation with any organization with a direct or indirect financial interest in the subject matter discussed in the manuscript.

References

1. Xu, Y.; Shang, S. Application of cryptocurrency technology—A survey. *Intell. Autom. Soft Comput.* **2020**, *26*, 563–582.
2. Grover, F.; Johansson, M. High-frequency trading in the cryptocurrency market. *J. Financ. Mark.* **2019**, *45*, 1–16.
3. Baek, C.; Elbeck, M. Bitcoins as an investment or speculative vehicle? A first look. *Appl. Econ. Lett.* **2015**, *22*, 30–34. [CrossRef]
4. Kondor, D.; Pósfai, M.; Csabai, I.; Vattay, G. Do the rich get richer? An empirical analysis of the Bitcoin transaction network. *PLoS ONE* **2014**, *9*, e86197. [CrossRef]
5. Altan, A.; Karasu, S.; Bekiros, S. Digital currency forecasting with chaotic meta-heuristic bio-inspired signal processing techniques. *Chaos Solitons Fractals* **2019**, *126*, 325–336. [CrossRef]
6. McNally, S.; Roche, J.; Caton, S. Predicting the price of bitcoin using machine learning. In Proceedings of the 2018 26th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP), Cambridge, UK, 21–23 March 2018; pp. 339–343.
7. Vo, A.; Yost-Bremm, C. A high-frequency algorithmic trading strategy for cryptocurrency. *J. Comput. Inf. Syst.* **2020**, *60*, 555–568. [CrossRef]
8. Vaswani, A. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 5998–6008. [CrossRef]
9. Chen, W.; Xu, H.; Jia, L.; Gao, Y. Machine learning model for Bitcoin exchange rate prediction using economic and technology determinants. *Int. J. Forecast.* **2021**, *37*, 28–43. [CrossRef]
10. Liu, Y.; Tsyvinski, A. Risks and returns of cryptocurrency. *Rev. Financ. Stud.* **2021**, *34*, 2689–2727. [CrossRef]
11. Dyhrberg, A.H. Bitcoin, gold and the dollar–A GARCH volatility analysis. *Financ. Res. Lett.* **2016**, *16*, 85–92. [CrossRef]
12. Patel, M.M.; Tanwar, S.; Gupta, R.; Kumar, N. A deep learning-based cryptocurrency price prediction scheme for financial institutions. *J. Inf. Secur. Appl.* **2020**, *55*, 102583. [CrossRef]
13. Chennupati, A.; Prahas, B.; Ghali, B.A.; Jasvitha, B.D.; Murali, K. Comparative Analysis of Bitcoin Price Prediction Models: LSTM, BiLSTM, ARIMA and Transformers. In Proceedings of the 2024 15th International Conference on Computing Communication and Networking Technologies (ICCCNT), Kamand, India, 24–28 June 2024; pp. 1–7.
14. Belcastro, L.; Carbone, D.; Cosentino, C.; Marozzo, F.; Trunfio, P. Enhancing Cryptocurrency Price Forecasting by Integrating Machine Learning with Social Media and Market Data. *Algorithms* **2023**, *16*, 542. [CrossRef]
15. Zou, Y.; Herremans, D. PreBit—A multimodal model with Twitter FinBERT embeddings for extreme price movement prediction of Bitcoin. *Expert Syst. Appl.* **2023**, *233*, 120838. [CrossRef]
16. Krauss, C.; Do, X.A.; Huck, N. Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P 500. *Eur. J. Oper. Res.* **2017**, *259*, 689–702.

17. Xie, M.; Li, J.; Cui, H. Improving Twitter Sentiment Classification via Multi-Level Sentiment-Enriched Word Embeddings. *arXiv* **2019**, arXiv:1902.09314.
18. Jiang, Z.; Liang, J. Cryptocurrency portfolio management with deep reinforcement learning. In Proceedings of the 2017 Intelligent Systems Conference (IntelliSys), London, UK, 7–8 September 2017; pp. 905–913.
19. Saqware, G.J. Hybrid Deep Learning Model Integrating Attention Mechanism for the Accurate Prediction and Forecasting of the Cryptocurrency Market. In *Operations Research Forum*; Springer: Berlin/Heidelberg, Germany, 2024; Volume 5, p. 19.
20. Zhou, H.; Zhang, S.; Peng, J.; Zhang, S.; Li, J.; Xiong, H.; Zhang, W. Informer: Beyond efficient transformer for long sequence time-series forecasting. In Proceedings of the AAAI Conference on Artificial Intelligence, Philadelphia, PA, USA, 27 February–2 March 2025; Volume 35, pp. 11106–11115.
21. Singh, S.; Bhat, M. Transformer-based approach for Ethereum Price Prediction Using Crosscurrency correlation and Sentiment Analysis. *arXiv* **2024**, arXiv:2401.08077.
22. Qin, Y.; Song, D.; Chen, H.; Cheng, W.; Jiang, G.; Cottrell, G. A dual-stage attention-based recurrent neural network for time series prediction. *arXiv* **2017**, arXiv:1704.02971.
23. Livieris, I.E.; Pintelas, E.; Pintelas, P. A CNN-LSTM model for gold price time-series forecasting. *Neural Comput. Appl.* **2020**, *32*, 17351–17360. [CrossRef]
24. Song, H.; Rajan, D.; Thiagarajan, J.; Spanias, A. Attend and diagnose: Clinical time series analysis using attention models. In Proceedings of the AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; Volume 32.
25. Abbasimehr, H.; Paki, R. Improving time series forecasting using LSTM and attention models. *J. Ambient. Intell. Humaniz. Comput.* **2022**, *13*, 673–691. [CrossRef]
26. He, K.; Yang, Q.; Ji, L.; Pan, J.; Zou, Y. Financial time series forecasting with the deep learning ensemble model. *Mathematics* **2023**, *11*, 1054. [CrossRef]
27. Zhao, M. Financial time series forecast of temporal convolutional network based on feature extraction by variational mode decomposition. In *International Conference on Artificial Intelligence in China*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 365–374.
28. Niu, P.; Zhou, T.; Wang, X.; Sun, L.; Jin, R. Attention as Robust Representation for Time Series Forecasting. *arXiv* **2024**, arXiv:2402.05370.
29. Wu, H. Revisiting Attention for Multivariate Time Series Forecasting. *arXiv* **2024**, arXiv:2407.13806.
30. Olorunnimbe, K.; Viktor, H. Ensemble of temporal Transformers for financial time series. *J. Intell. Inf. Syst.* **2024**, *62*, 1087–1111. [CrossRef]
31. Seabe, P.L.; Moutsinga, C.R.B.; Pindza, E. Forecasting cryptocurrency prices using LSTM, GRU, and bi-directional LSTM: A deep learning approach. *Fractal Fract.* **2023**, *7*, 203. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.