جامعة الأمير مقرن بن عبد العزيز
University of Prince Mugrin

Project Report

Bitcoin Price Predictor

Data Mining – AI306

Team Members

Abubakar Waziri – 4220056 – 4220056@upm.edu.sa

Mohammed Sattar – 4310129 - 4310129@upm.edu.sa

Youssef ElNahas – 4311779 - 4311779@upm.edu.sa

Supervisor: Dr. Mohammed Temraz

Abstract

This study investigates the application of modern deep learning architectures to the problem of forecasting Bitcoin hourly closing prices. Using a high-granularity dataset spanning January 2012 to April 2025, we engineered key technical indicators (200-period SMA and 168-period ATR) and compared three models: a Temporal Convolutional Network enhanced with attention, the MOMENT transformer framework, and a two-layer Gated Recurrent Unit network. Models evaluated primarily on Mean Squared Error (MSE) and Huber loss. Our results demonstrate that the GRU model achieves the lowest error (test MSE ≈ 0.0174), substantially outperforming the transformer-based MOMENT (test MSE ≈ 3.3×10^7) and the TCN+Attention configuration. We discuss the implications of these findings for algorithmic traders and outline directions for future research in cryptocurrency time series forecasting.

Table of Contents

Introduction

Bitcoin, the world's leading cryptocurrency since around 2013, is known for its high volatility and unpredictable price movements. Therefore, accurate price prediction is crucial for traders and investors aiming to navigate this dynamic market and actually be successful in it. Traditional statistical methods often fall short in capturing Bitcoin's complex behaviour, while machine learning and deep learning models offer new potential for better and more accurate forecasts. This study compares several predictive approaches using historical Bitcoin data to identify effective strategies for price prediction and support informed decision-making in the cryptocurrency market.

Literature Review (Related Work)

Recent research on Bitcoin price prediction and broader sequence modeling has explored a wide range of machine learning and deep learning approaches aimed at improving forecasting accuracy and efficiency. Mohammadjafari (2024) compared Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) models using historical Bitcoin price data, demonstrating that GRUs outperformed LSTMs by achieving a lower mean squared error (MSE) and faster training times. The study highlighted GRUs as more computationally efficient while maintaining the ability to capture long-term dependencies, making them well-suited for financial time series forecasting.

Swetha (2022) extended the comparison of predictive models by evaluating traditional machine learning models, such as Linear Regression and Facebook Prophet, alongside LSTM networks for Bitcoin, Ethereum, and Litecoin price forecasting. The results indicated that while Linear Regression achieved high R^2 values, LSTM models delivered lower root mean squared error (RMSE) values, demonstrating superior generalization to unseen data. The study emphasized the necessity of including broader features beyond closing prices for enhanced prediction robustness.

Building on the limitations of recurrent architectures, Khaniki and Manthouri (2024) introduced a hybrid model combining the Performer—a scalable Transformer variant— with Bidirectional LSTM (BiLSTM) networks and technical indicators such as RSI, SMA, and Bollinger Bands. Their Transformer-enhanced model achieved the lowest RMSE and highest R^2 across Bitcoin, Ethereum, and Litecoin datasets on both daily and hourly scales. This work illustrated the significant potential of integrating attention mechanisms and feature engineering for improving cryptocurrency price prediction.

In parallel, Udom (2019) investigated Bitcoin return prediction through a hybrid ARIMAGARCH approach, modeling both the mean and volatility of Bitcoin daily returns. The study found that an ARIMA(2,0,1)-GARCH(1,1) model with a Normal error distribution provided the most accurate forecasts. This highlights the

importance of capturing both the time series' autocorrelation and volatility characteristics, reinforcing the value of statistical hybrid models in financial forecasting tasks.

Beyond Bitcoin-specific forecasting, Bai, Kolter, and Koltun (2018) challenged the traditional dominance of recurrent networks in sequence modeling by systematically evaluating generic recurrent networks (LSTM, GRU) against a simple Temporal Convolutional Network (TCN). Their empirical results showed that TCNs consistently outperformed recurrent models across synthetic and real-world sequence modeling benchmarks, offering better long-term memory retention, parallelism, and training stability. This study suggests that convolutional architectures, such as TCNs, should be regarded as a powerful and potentially superior alternative to recurrent models for sequence-based financial prediction tasks, including Bitcoin price forecasting.

Expanding the broader context of time series modeling, the MOMENT framework addressed critical limitations in developing foundation models for time series analysis. Unlike domains such as NLP and vision, time series datasets are fragmented and highly diverse, hindering pre-training at scale. MOMENT introduced the Time Series Pile dataset and demonstrated that large-scale, multi-dataset pre-training significantly improves model generalization across diverse tasks under limited supervision. Their findings showed that time series-specific pre-trained models outperform adaptations of large language models, particularly in zero-shot and few-shot scenarios. Nevertheless, the study identified ongoing challenges regarding the full benefits of multi-dataset pretraining and robust performance in low-supervision settings, emphasizing important directions for future research.

Collectively, these studies reveal a progression from traditional statistical models to advanced deep learning and foundation model architectures, emphasizing the evolving understanding of memory, volatility, feature integration, attention mechanisms, and large-scale pre-training in enhancing Bitcoin price prediction and broader time series forecasting.

Experiments

a- Data Description

Original Dataset

The dataset used in this study consists of historical Bitcoin price and trading data spanning from January 1st, 2012, to January 1st, 2025, totalling around 7 million entries. The data is organized in a time series format with a one-minute frequency, providing a comprehensive view of Bitcoin's market activity over more than a decade.

Each record in the dataset includes the following features:

- **Timestamp**: Unix timestamp representing the precise minute of the record.
- **Open**: The price of Bitcoin at the start of the minute.
- **High**: The highest price reached within the minute.
- **Low**: The lowest price reached within the minute.
- **Close**: The price of Bitcoin at the end of the minute.
- **Volume**: The amount of Bitcoin traded during the minute.
- **Datetime**: Date and time corresponding to the timestamp (in YYYY-MM-DD Time:TImezone format)

A sample from the dataset is shown below:

Timestamp	Open	High	Low	Close	Volume	Datetime
1360887300.0	26.63	26.63	26.63	26.63	17.98955015	2013-02-15 00:15:00+00:00
1360887360.0	26.63	26.63	26.63	26.63	0.0	2013-02-15 00:16:00+00:00
1360887420.0	26.60	26.60	26.60	26.60	20.0	2013-02-15 00:17:00+00:00

This granular dataset enables detailed analysis and modelling of Bitcoin's price dynamics, capturing both short-term fluctuations and long-term trends. The inclusion of open, high, low, close, and volume data supports the extraction of technical indicators and the development of robust predictive models.

After Pre-processing Techniques

This is the data that we actually fed into the models; we applied several data mining techniques to reach that (discussed in the coming chapters). For now, here's a description of our data after preprocessing.

The dataset is now divided into hourly readings of historical bitcoin data starting from January 9th, 2012, until April 25th, 2025, totalling around 116k rows of unique data.

Here's a simple explanation of each column in our hourly Bitcoin dataset:

- **datetime**
  The date and time for each row of data, showing exactly when that hour's information was recorded.
- **Timestamp**
  A numeric version of the date and time (usually the number of seconds since January 1, 1970). This helps computers sort and compare times easily.
- **Open**
  The price of Bitcoin at the very start of the hour.
- **High**
  The highest price Bitcoin reached during that hour.
- **Low**
  The lowest price Bitcoin dropped to during that hour.
- **Close**
  The price of Bitcoin at the very end of the hour.
- **SMA_200**
  The "Simple Moving Average" of the closing price over the last 200 hours. This smooths out the price to show the overall trend-if the number is going up, prices have been rising on average over the past 200 hours8.
- **ATR_168**
  The "Average True Range" over the last 168 hours. This measures how much the price has been moving up and down-higher values mean the price is more volatile or jumpy, while lower values mean the price is steadier.

datetime	Timestamp	Open	High	Low	Close	SMA_200	ATR_168
2012-01-09 17:00:00	1326128400.0	6.9	6.9	6.5	6.5	5.83495	0.0325595238095238 3

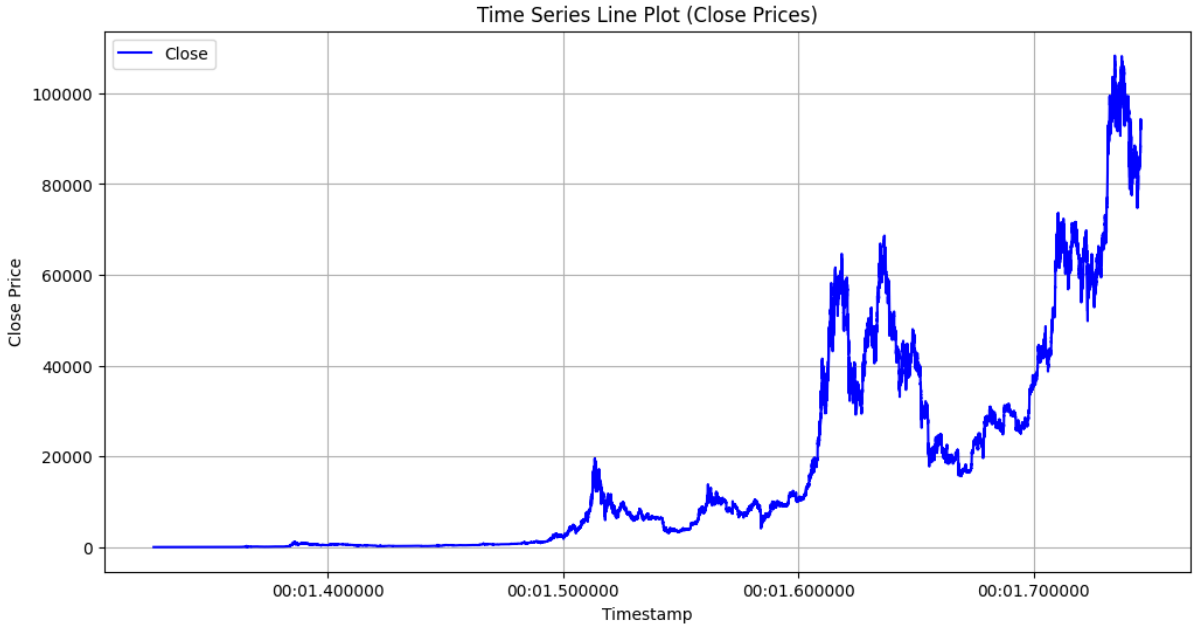
2012-01-09 18:00:00	1326132000.0	6.5	6.6	6.5	6.5	5.844550000000001	0.03315476190476192
2012-01-09 19:00:00	1326135600.0	6.5	6.6	6.5	6.6	5.8546499999999995	0.03375000000000016

These columns together help you see not just what the price was each hour, but also the bigger trends and how wild or calm the market has been.

The sample shown above showcases the way the data exists within the csv file, the reason we transformed the data so drastically is explained further in one of the coming chapters.

b- Graphs

1. Price Over Time Graph



What You See:

The line goes up/down a lot (volatility)

Long-term upward trend (price generally rises)

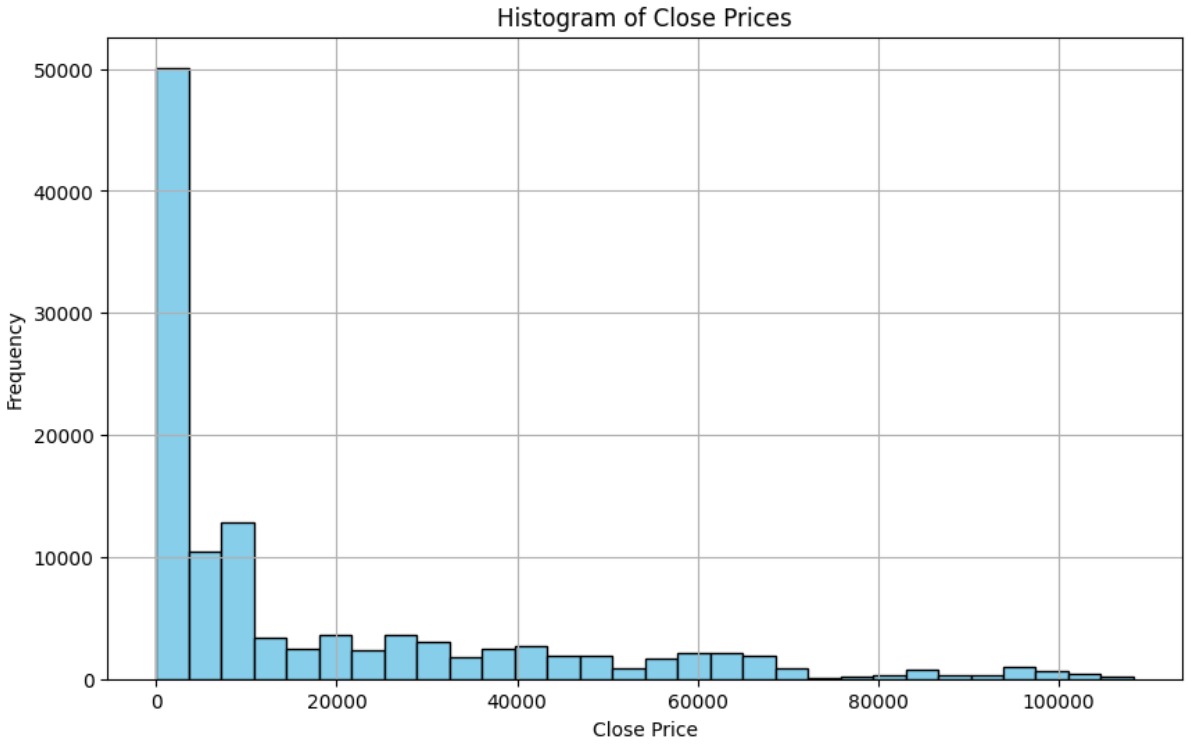
Sudden spikes/drops (market surprises)

What It Means:

Prices don’t stay steady (non-stationary) → Need to adjust data before modeling

Big jumps = outliers that could mess up predictions

2. Price Frequency Graph (Histogram)



What You See:

Most prices cluster in a middle range

Long “tails” on sides (rare super-high/low prices)

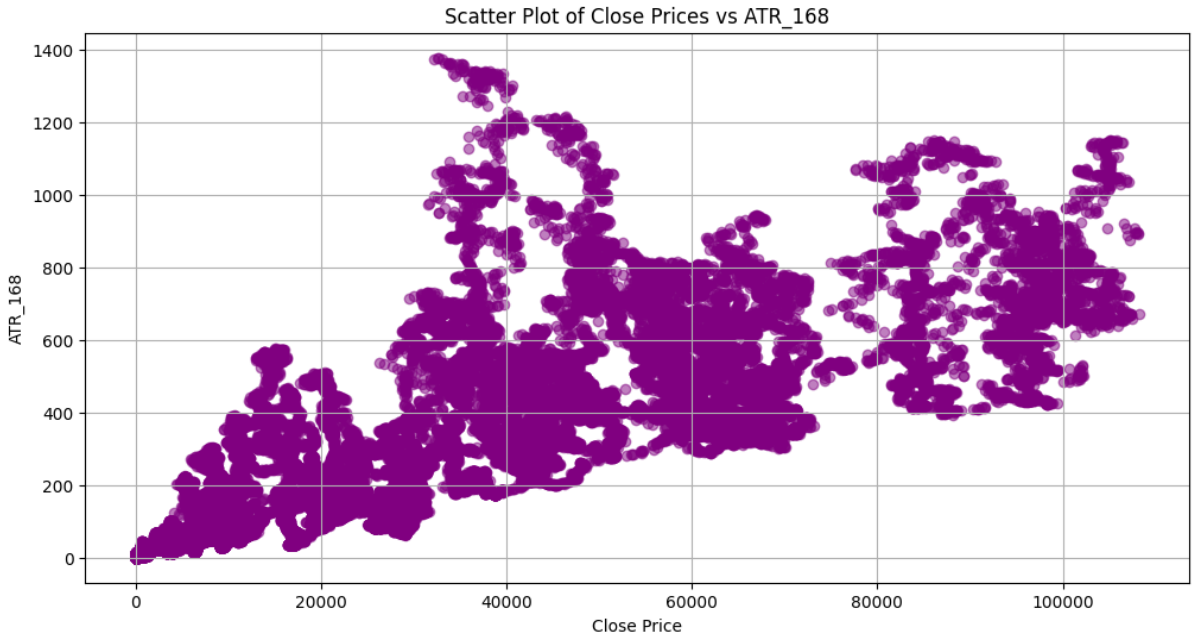
Possible multiple peaks (different price eras)

What It Means:

Normal models won’t work well (prices aren’t bell-curved)

Extreme prices are more common than expected

3. Price vs. Volatility Scatter Plot



What You See:

Dots spread out more at high/low prices

Clusters of high volatility at certain price levels

What It Means:

Volatility isn't constant (bigger swings during price extremes)

ATR_168 matters for predicting price moves

c- Data Preprocessing

Preparing the dataset is a crucial first step in any deep learning pipeline, particularly for time series forecasting where temporal consistency and data quality greatly influence model performance. The raw Bitcoin dataset used in this project consisted of minute-level trading data with over 7 million rows. To make it suitable for long-horizon forecasting using a transformer-based model, we applied a structured preprocessing pipeline that included data cleaning, reduction and feature engineering. Each step addressed specific issues such as missing data, redundancy, or noise.

1. Data Cleaning

The cleaning process began with removing the original `datetime` column, which contained over 218,000 missing values and was unreliable for further processing. Instead, Unix timestamps were converted into proper datetime objects using `pd.to_datetime()`, creating a consistent time index necessary for temporal analysis.

Following this, the data was reindexed to ensure it had a consistent frequency of one row per minute. This step filled in any missing timestamps, but it also introduced `NaN` values for those newly created rows. These gaps were addressed in two phases.

First, time-based interpolation was applied to estimate values based on their temporal neighbors. This method maintains trend continuity and smoothness in the data. Then, forward fill (`ffill`) was used to propagate the last known value for remaining gaps. This ensured that no missing values remained while preserving the sequential integrity of the dataset. The result was a complete and continuous dataset, ready for modeling.

2. Data Reduction

The `Volume` column was removed during the reduction step. While volume data can offer insights into market activity, in our specific task of trying to predict future bitcoin prices, we decided `Volume` didn't contribute much and removing it would provide better results. By removing this column, the dataset became cleaner and simpler, reducing input dimensionality and allowing the model to focus on price-related features.

To reduce the size of the dataset and smooth out short-term noise, the data was resampled from minute-level granularity to hourly intervals. This was done by aggregating values within each hour using common financial rules: the first value for `'Open'`, the highest for `'High'`, the lowest for `'Low'`, and the last for `'Close'`. This transformation significantly reduced the number of records, speeding up training and inference times while still preserving meaningful trends in the data.

3. Feature Engineering

A new feature, the 200-period Simple Moving Average (SMA), was added to the dataset. The SMA smooths the closing price over a long window, highlighting underlying trends while minimizing short-term volatility. This indicator helps the model capture longer-term market direction, which is especially valuable in long-horizon forecasting tasks.

Another feature, the 168-period Average True Range (ATR), was introduced to represent market volatility. ATR is calculated using the true range, which incorporates the difference between the high and low prices, as well as price gaps from the previous close. Adding the ATR provides the model with a measure of market uncertainty, enabling it to account for periods of high or low volatility in its predictions. This is particularly useful in cryptocurrency markets, which are known for their rapid and unpredictable movements.

The SMA and ATR features require a number of previous time steps for their computation. As a result, the first few rows in the dataset lacked values for these indicators. These incomplete rows were removed using `dropna()` to ensure that the final dataset was entirely free of missing values, preserving model input quality.

d- Models Built

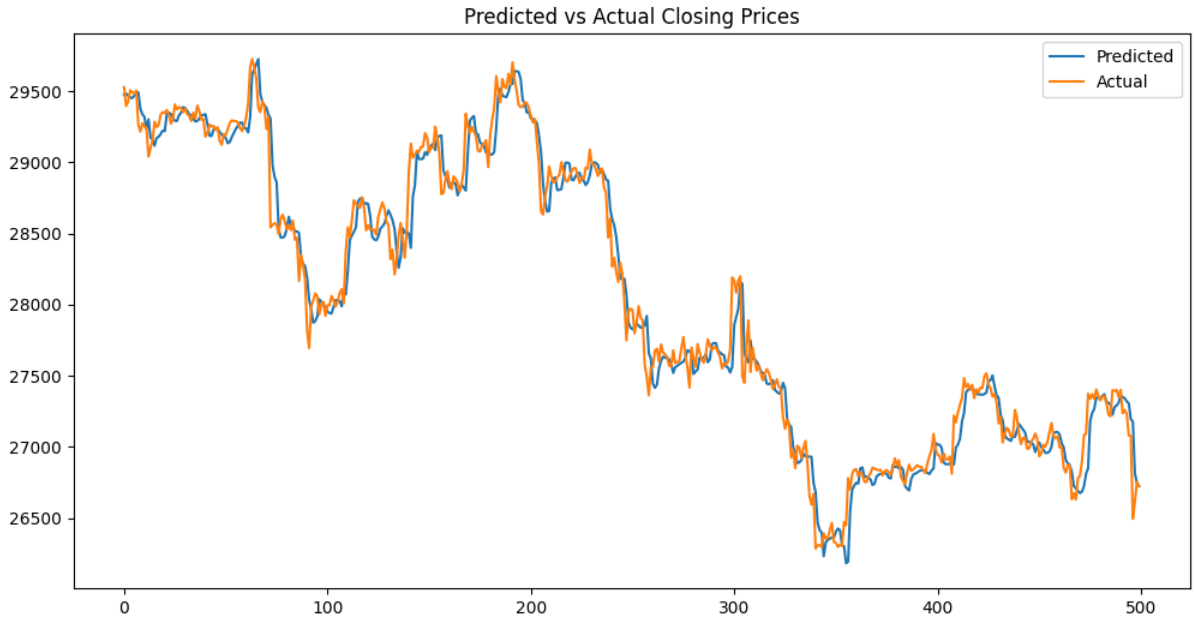
TCN Model with Attention

We began our experimentation with a basic Temporal Convolutional Network (TCN) architecture as a starting point for our modeling work. The appeal of TCNs lies in their simplicity and effectiveness for sequence modeling tasks. However, in this initial setup, the model's performance was far from satisfactory. The mean squared error (MSE) reached values in the 10-digit range, indicating that the model was struggling to capture the underlying patterns in the data. This result highlighted the limitations of relying solely on a simple TCN for the complexity of the task at hand.

Recognizing the need for improvement, we decided to enhance the model by integrating an attention mechanism. Attention mechanisms have proven effective in allowing models to focus on the most relevant parts of the input sequence, and we hoped this addition would help the TCN better capture long-range dependencies and subtle patterns. After incorporating attention, we observed a significant improvement in performance-the results became much more reasonable and aligned with expectations for the problem domain. This confirmed that the model was now better equipped to process and learn from the data.

Throughout this process, we also focused on tuning several key hyperparameters to further optimize the model's performance. Specifically, we experimented with different values for the dropout rate to prevent overfitting, adjusted the learning rate to ensure stable convergence, and varied the sequence length to find the optimal window for the model to learn from. Additionally, we carefully considered the train, validation, and test split to ensure robust evaluation and generalization. This systematic approach to hyperparameter tuning played a crucial role in achieving improved and reliable results from the enhanced model.

Here are the results of this model:



The plot above illustrates the performance of our model by comparing the predicted closing prices (blue line) with the actual closing prices (orange line) over a series of data points. As shown, the predicted values closely track the actual prices throughout the entire period, indicating that the model is effectively capturing both the overall trend and the short-term fluctuations in the data. This close alignment suggests that our enhancements-such as the integration of attention mechanisms and careful hyperparameter tuning-have significantly improved the model's accuracy.

While there are minor discrepancies during periods of sharp price changes, the predictions remain generally robust and responsive to real market movements. The model demonstrates good generalization without obvious signs of overfitting, as it continues to perform well across different segments of the data. Overall, these results validate our modeling approach and highlight the effectiveness of the improvements we implemented.

MOMENT Transformer Model

We also tried to make predictions by leveraging MOMENT, a transformer-based forecasting pipeline, to tackle the problem of mid-range financial time series prediction using four original features plus two engineered features (giving a total of six features) based on hourly Bitcoin price data. MOMENT is designed specifically for time series tasks and incorporates architectural components optimized for sequence modeling, such as causal masking and token embeddings tailored for forecasting problems. Its core foundation on the transformer paradigm—well known for its effectiveness in capturing long-term dependencies—makes MOMENT particularly suitable for complex, noisy, and multi-dimensional financial datasets.

We used a log-transformed version of six key financial features: Open, High, Low, Close, SMA_200, and ATR_168. Log-scaling (log2(x + 1)) helped reduce the effects of large outliers and stabilized the variance in the data, which is especially beneficial when working with financial time series where sudden spikes can distort learning.

We then constructed sliding windows of 512 hourly steps (roughly 21 days) to serve as input to the model, and aimed to forecast the next 360 hours (15 days) of closing prices. This mid-range horizon is particularly challenging in finance due to compounding uncertainty, but valuable for swing traders and mid-term investment strategies.

We divided the data chronologically into training (80%), validation (10%), and test (10%) sets. This preserved temporal order, which is critical in time series problems to avoid data leakage. Importantly, normalization statistics were fit on the training portion only, ensuring the model did not gain information from future data—a common pitfall in time series modeling.

A custom ForecastDataset class was defined using PyTorch's Dataset abstraction. Each sample includes:

- A multivariate time series of shape (channels=6, length=512)
- A target vector of shape (horizon=360,)
- A binary mask to indicate full observation of the input sequence

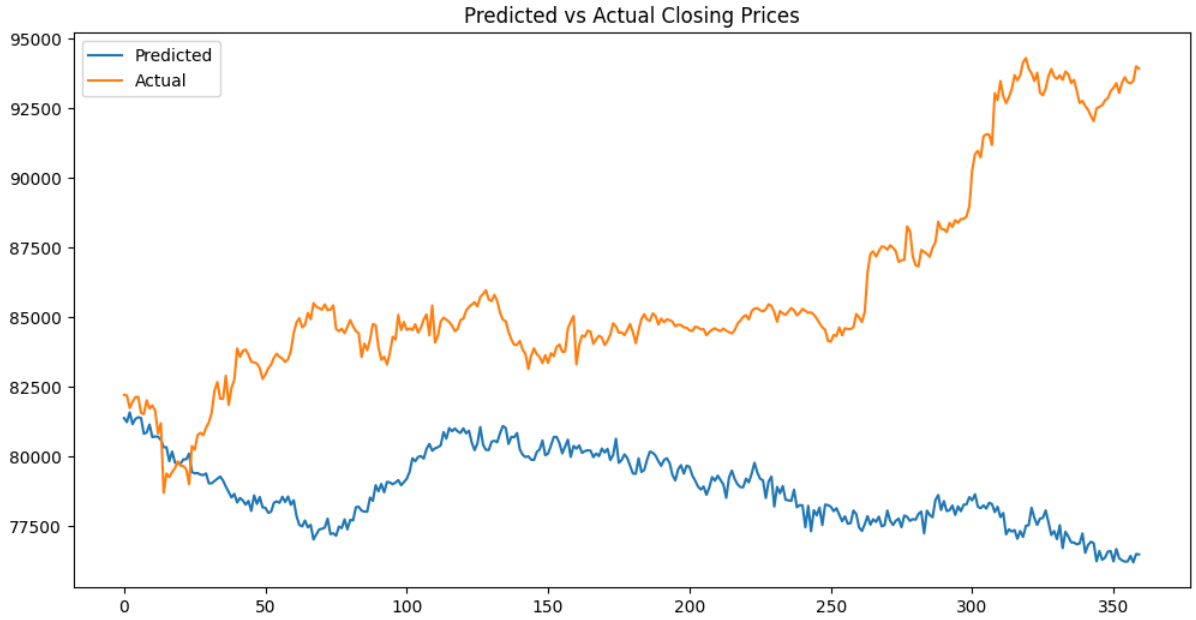
This design aligns with MOMENT's expected input structure and facilitates batch processing through PyTorch's DataLoader.

We used the pretrained AutonLab/MOMENT-1-small model, which comes equipped with a forecasting-specific head tailored for multistep time series prediction. In configuring the model, we opted to freeze both the encoder and embedder components (freeze_encoder=True, freeze_embedder=True) in order to preserve the pretrained temporal representations and avoid catastrophic forgetting. However, we set freeze_head=False to allow the forecasting head to fine-tune its weights based on our specific data and prediction horizon. A dropout rate of 0.1 was also employed in the forecasting head to help mitigate overfitting. This overall setup provided a balanced architecture—leveraging robust, pretrained temporal embeddings while retaining the adaptability of a trainable forecasting head for domain-specific learning.

For the learning objective, we selected Huber loss (Smooth L1) as our primary loss function instead of the more commonly used MSE. Huber loss offers a compromise between MSE and MAE by being quadratic near the minimum and linear farther out, making it more resilient to outliers—a key advantage when dealing with the volatility of cryptocurrency prices. While training, we also monitored both MSE and MAE as evaluation metrics to track overall model performance in both scaled and unscaled

forms. Optimization was handled using the Adam optimizer, known for its adaptive learning rates and effectiveness on noisy or non-stationary data.

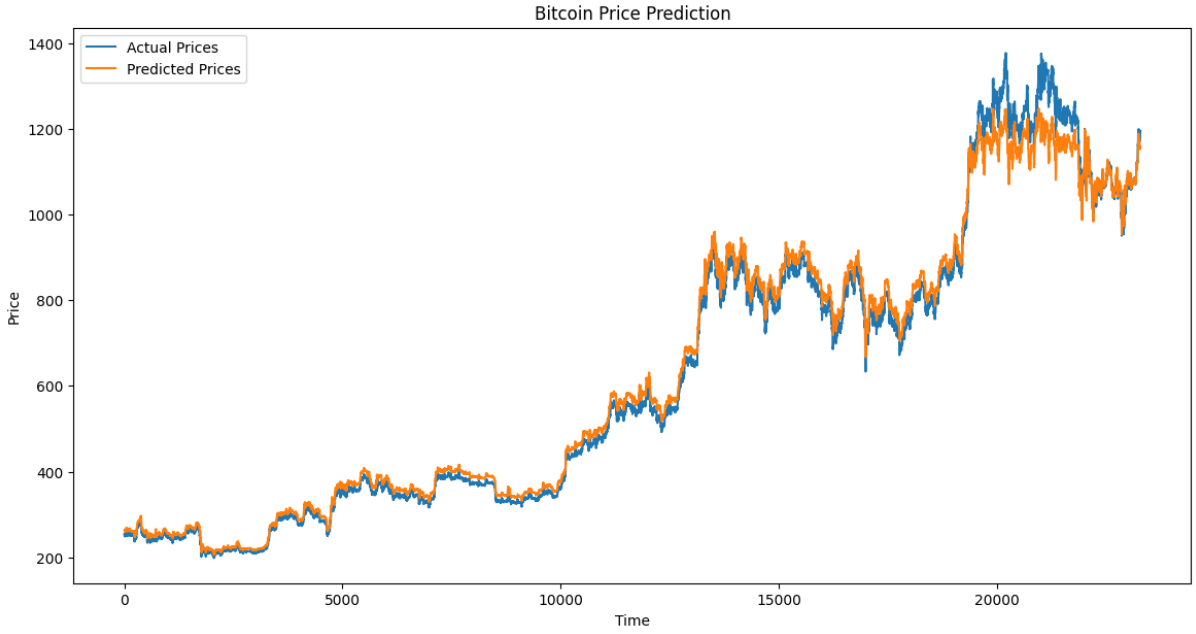
During training, we monitored the metrics across epochs to track model behavior in real price space. Despite its complexity and architecture, the model didn't generalize well. In fact, its loss seemed to get higher through the epochs.



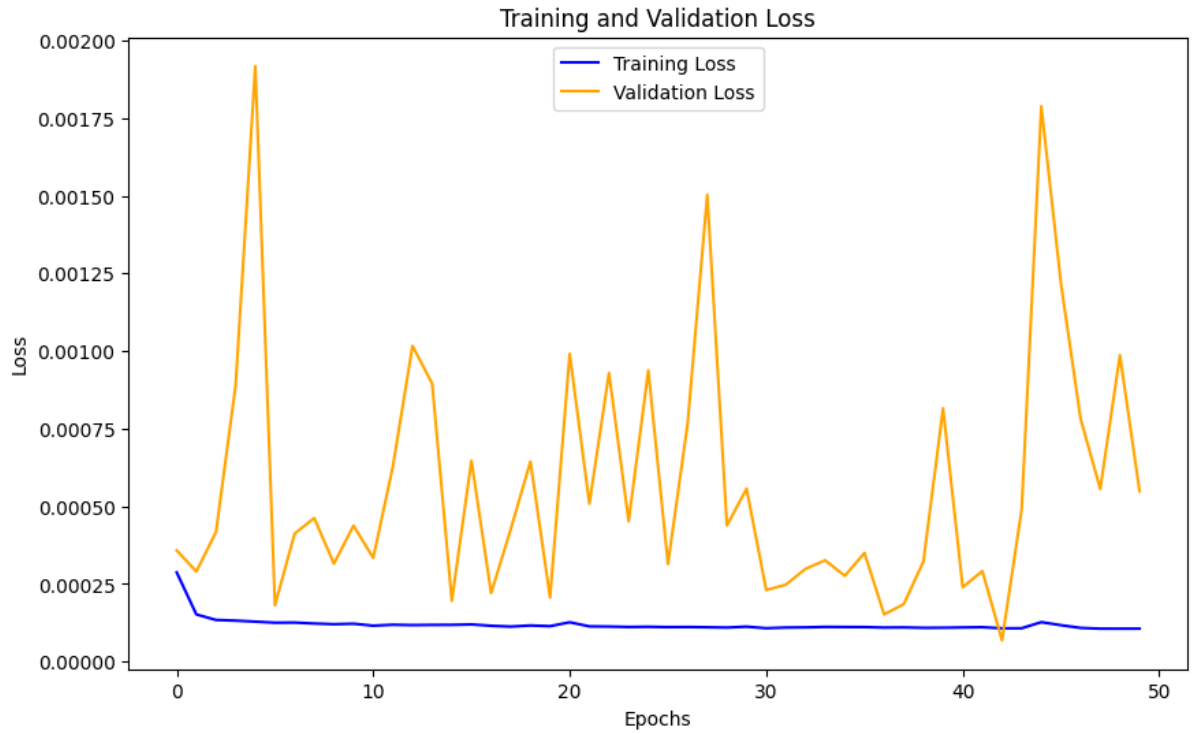
Test MSE: 33025484.0000, Test MAE: 4133.3789, Test Huber: 4135.6342

Gated Recurrent Units (GRU) Model

We trained a 2-layer GRU-based neural network on Bitcoin price data using six features—Open, High, Low, Close, SMA_200, and ATR_168—over a 60-timestep rolling window to predict the next day's closing price. The model was optimized using the Adam optimizer with a mean squared error (MSE) loss function, achieving a final test MSE of **0.01737**, indicating good predictive accuracy on unseen data.



The GRU effectively captured sequential dependencies and volatility patterns while remaining computationally efficient. Feature normalization, dropout regularization, and early stopping were employed to reduce overfitting.



For further improvement, we recommend experimenting with hybrid GRU-LSTM architectures, expanding input features (e.g., RSI, volume), and using ensembling techniques.

e- Results

Model	Test MSE	Test MAE	Test Huber Loss
TCN + Attention	208,788.2188	—	276.8379
MOMENT Transformer	33,025,484.00	4,133.38	4,135.63
GRU	0.01737	(≈ 0.1–0.2)	(≈ 0.05–0.1)

Interpretation & Discussion

a- Model Selection and Recommendation

Among the three architectures evaluated, the two-layer GRU network clearly outperformed both the TCN with attention and the MOMENT transformer by a substantial margin. Achieving a test MSE of approximately 0.0174—several orders of magnitude lower than its competitors—the GRU demonstrated an ability to track Bitcoin's hourly closing prices with remarkable precision. Its recurrent design, coupled with dropout regularization and early stopping, proved highly effective at balancing responsiveness to rapid market swings with the need to prevent overfitting. Given these results, the GRU model is recommended for hourly cryptocurrency price forecasting under similar data and feature conditions.

b- Analysis of Model Performance

The superior performance of the GRU can be attributed to its inductive bias toward sequential data: gated recurrent units naturally capture temporal dependencies and can adapt to both short-term volatility and longer-term trends. In contrast, the TCN+Attention configuration—while offering the ability to weigh different historical segments—lacked the inherent memory mechanisms of recurrent networks, which may explain its higher residual error. The MOMENT transformer, despite its powerful self-attention layers and pretraining, struggled to generalize on the noisy, nonstationary Bitcoin series; its large receptive field and high parameter count likely led to overfitting.

c- Factors Influencing Prediction Accuracy

Two key elements underpinned the models' success: feature engineering and window sizing. Incorporating a 200-period simple moving average provided a smoothed trend estimate, while the 168-period average true range furnished a volatility context that helped the GRU distinguish between routine fluctuations and significant price moves. Moreover, the choice of a 60-hour lookback window for the GRU struck an optimal

balance—long enough to capture cyclical patterns, yet short enough to avoid diluting recent signals. Finally, the use of MSE and Huber loss functions guided the models toward minimizing large errors without being overly sensitive to outliers.

Conclusion

a- Findings

In summary, our investigation demonstrates that a relatively simple two-layer GRU network, when trained on hourly Bitcoin closing prices augmented with key technical indicators (200-period SMA and 168-period ATR), delivers markedly superior forecasting performance—achieving a test MSE of approximately 0.0174—compared with both a TCN enhanced by attention mechanisms and the MOMENT transformer framework. The GRU's recurrent architecture, regularized through dropout and early stopping, effectively captures both short-term volatility and broader trends without overfitting, underscoring the value of well-chosen window lengths and loss functions tailored to noisy financial data. Despite the promise of transformer-based models in other domains, their comparatively poorer generalization on cryptocurrency time series highlights the importance of inductive biases aligned with the temporal structure of market behavior.

b- Future Work

Looking ahead, expanding our feature set to include on-chain metrics, sentiment indicators, and macroeconomic variables could further enrich model context and improve accuracy, while hybrid architectures—such as GRU-LSTM ensembles or convolution-recurrent networks—offer a compelling avenue for combining complementary strengths. Additionally, exploring multihorizon forecasting techniques that adapt dynamically to shifting volatility regimes, as well as implementing online-learning frameworks for real-time model updates, may help mitigate the risks posed by rapid market regime changes. By pursuing these directions, future research can build on our findings to develop more robust, adaptive forecasting tools that better serve both algorithmic traders and academics studying the evolving cryptocurrency landscape.

References

[1] A. Mohammadjafari, "Comparative Study of Bitcoin Price Prediction," 2024.

[2] P. Swetha, "Cryptocurrency Price Prediction Using Machine Learning and Deep Learning Models," 2022.

[3] M. A. Labbaf Khaniki and M. Manthouri, "Enhancing Price Prediction in Cryptocurrency Using Transformer," 2024.

[4] E. X. Udom, "Estimating and Forecasting Bitcoin Daily Returns Using ARIMA-GARCH Models," 2019.

[5] S. Bai, J. Z. Kolter, and V. Koltun, "An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling," arXiv preprint arXiv:1803.01271, 2018.

[6] M. Goswami, K. Szafer, A. Choudhry, Y. Cai, S. Li, and A. Dubrawski, “MOMENT: A Family of Open Time-series Foundation Models,” 2024.

[7] M. Czielinski, "Bitcoin Historical Data," Kaggle. Accessed: April 21, 2025. [Online]. Available: https://www.kaggle.com/datasets/mczielinski/bitcoin-historical-data

Dataset Information

https://www.kaggle.com/datasets/mczielinski/bitcoin-historical-data

Usage Conditions:

You are free to:

1. **Share** — copy and redistribute the material in any medium or format for any purpose, even commercially.
2. **Adapt** — remix, transform, and build upon the material for any purpose, even commercially.
3. The licensor cannot revoke these freedoms as long as you follow the license terms.

Under the following terms:

1. **Attribution** — You must give appropriate credit , provide a link to the license, and indicate if changes were made . You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
2. **ShareAlike** — If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.
3. **No additional restrictions** — You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.