Ethereum Price Movement Prediction using a GRU Neural Network through a Pathway Framework

Yash Modi : 2401ME59

October 22, 2025

Abstract

This report details the development and evaluation of a deep learning model designed to predict the price movement of Ethereum (ETH). The methodology leverages a Gated Recurrent Unit (GRU) neural network, a type of Recurrent Neural Network (RNN) adept at processing sequential data. To enhance predictive power, the model is trained not on raw price data, but on a rich set of engineered features, including common financial technical indicators like the Relative Strength Index (RSI), Moving Average Convergence Divergence (MACD), and On-Balance Volume (OBV). The model's primary objective is to predict the daily percentage price change, which is then used to determine the direction of price movement (up or down). The final evaluation focuses on Directional Accuracy, a practical metric for financial forecasting.

1 Introduction

1.1 Problem Statement

Cryptocurrency markets, particularly Ethereum, are characterized by high volatility and complex, non-linear dynamics. Accurately forecasting their price movements presents a significant challenge. Traditional statistical models often fall short as they struggle to capture the intricate temporal dependencies and market sentiment embedded in historical data.

1.2 Objective

The objective of this project is to construct a sophisticated time-series forecasting model capable of predicting the next-day price movement of Ethereum. The primary goal is not to predict the exact future price, but to accurately forecast the *direction* of the price change, a more attainable and commercially valuable outcome.

1.3 Technology Stack

The project is implemented in Python, utilizing the following core libraries:

- **TensorFlow/Keras:** For building and training the deep learning model.
- **Pandas:** For data manipulation and analysis.
- **NumPy:** For numerical operations.
- **Scikit-learn:** For data scaling.
- **Matplotlib:** For data visualization.

2 Methodology

The project follows a systematic pipeline from data acquisition to model evaluation.

2.1 Data and Feature Engineering

The model utilizes a historical dataset of Ethereum's daily price. To provide the model with a deeper understanding of market dynamics, the following features were engineered:

- **Momentum Indicators (RSI, MACD):** To capture the speed, change, and trend of price movements.
- **Volatility Indicators (ATR, HLT):** To quantify the magnitude of daily price fluctuations.
- **Volume Indicator (OBV):** To integrate trading volume as a proxy for market pressure.

2.2 Target Variable Transformation

A crucial methodological decision was to transform the prediction target. Instead of predicting the non-stationary price series directly, the model is trained to predict the daily **percentage change**. This creates a more stationary target variable, which is easier for machine learning models to learn.

2.3 Data Preparation and Sequencing

The dataset was split into an 80% training set and a 20% testing set. All features were normalized using `MinMaxScaler` to a range of (0, 1). A `create_sequences` function was employed to reformat the data. Using a look-back period of 30 days, the model is provided with the features of the last 30 days to predict the target for the 31st day.

2.4 Model Architecture

A stacked GRU architecture was chosen for its effectiveness in capturing temporal dependencies.

- **Layer 1:** A GRU layer with 100 units processes the input sequences.
- **Layer 2:** A second GRU layer with 50 units consolidates learned patterns.
- **Regularization:** Dropout layers (rate=0.2) are applied after each GRU layer to prevent overfitting.
- **Output:** The sequence data is passed through a Dense layer (25 units, ReLU) and then to a final output neuron that predicts the percentage change.

2.5 Training Protocol

The model was compiled using the `Adam` optimizer and `Mean Squared Error` loss. An `EarlyStopping` callback was implemented to monitor validation loss, halting training if no improvement was observed for 10 epochs and restoring the best weights.

3 Evaluation and Results

3.1 Prediction and Interpretation

The trained model was used to generate predictions on the unseen test set. These predictions, representing percentage changes, were then converted back into absolute price values using the previous day's closing price.

$$\text{PredictedPrice}_{\text{today}} = \text{ActualPrice}_{\text{yesterday}} \times (1 + \text{Predicted\%Change}_{\text{today}})$$

3.2 Performance Metric: Directional Accuracy

The model's performance was primarily evaluated using **Directional Accuracy**. This metric measures the percentage of time the model correctly predicted whether the price would increase or decrease compared to the previous day. This is a superior metric for this task because it aligns directly with the goal of informing a trading strategy, where direction is paramount.

3.3 Visualization

A time-series plot was generated to visually compare the predicted Ethereum prices against the actual prices. This visualization (Figure 1) confirmed that the model was able to capture the major trends and turning points in the price data effectively.

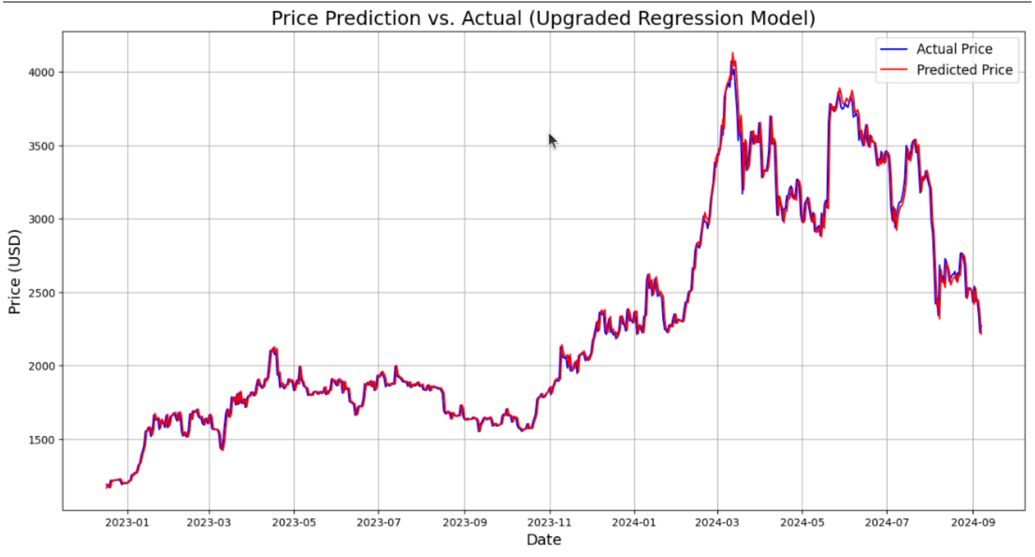

Figure 1: Price Prediction vs. Actual Price

4 Conclusion and Future Work

This project successfully demonstrates the efficacy of a GRU-based deep learning model for predicting Ethereum price movements. The combination of state-of-the-art feature engineering, predicting percentage change instead of absolute price, and evaluating with directional accuracy provides a robust framework for financial time-series forecasting.

Potential areas for future work include:

- **Hyperparameter Tuning:** Systematically optimizing parameters like the look-back period, batch size, and network architecture.

- **Alternative Architectures:** Exploring more advanced models like Transformers, which can capture longer-term dependencies.
- **Exogenous Variables:** Incorporating external data, such as social media sentiment or broader macroeconomic indicators, to further enrich the feature set.