School of Engineering

Centre for Artificial Intelligence (CAI)

Bachelor’s Thesis in Data Science

Towards Optimized AI Strategies for Cryptocurrency Trading

Authors Mikail Memis
Trojan Veseli

Supervisor Prof. Dr. Jasmina Bogojeska
Dr. Christian Badertscher

Date June 6, 2025

Declaration of Originality

Place, Date: Zurich, 06.06.2025 **Student names:** Mikail Memis, Trojan Veseli

Acknowledgements

We sincerely thank Prof. Dr. Jasmina Bogojeska for her expertise in AI and Dr. Christian Badertscher for his insights into cryptocurrency markets. Their guidance and support were invaluable to the successful completion of this project.

Abstract

This thesis investigates the application of advanced AI techniques, specifically Double Deep Q-Networks, for developing optimized cryptocurrency trading strategies. We explore the efficacy of two distinct neural network architectures as Q-function approximators: a novel time-series model, TimesNet, and a conventional 1D CNN. The study employs historical price data (OHLCV) for Bitcoin and Ethereum, evaluating performance with both a basic feature set and an extended set incorporating technical indicators and blockchain-specific metrics. Our experiments demonstrate that DDQN agents, particularly those leveraging the TimesNet architecture, can learn complex trading behaviors and achieve significant outperformance compared to traditional Buy & Hold benchmarks and CNN-based agents across diverse market conditions. The research highlights the potential of sophisticated temporal modeling and reinforcement learning in navigating the volatile cryptocurrency landscape, while also underscoring the nuanced impact of feature engineering on strategy performance across different market regimes and assets.

Keywords: Cryptocurrency Trading, Reinforcement Learning, Double Deep Q-Network, TimesNet, Algorithmic Trading, Bitcoin, Ethereum, Feature Engineering, Time Series Analysis.

Contents

List of Abbreviations

A2C	Advantage Actor-Critic
AI	Artificial Intelligence
ATR	Average True Range
CNN	Convolutional Neural Network
DDQN	Double Deep Q-Network
DQN	Deep Q-Network
EMA	Exponential Moving Average
FFT	Fast Fourier Transform
MACD	Moving Average Convergence Divergence
OHLCV	Open-High-Low-Close-Volume
RNN	Recurrent Neural Network
RSI	Relative Strength Index
STOCHRSI	Stochastic Relative Strength Index

1. Introduction

The financial markets, particularly the cryptocurrency sector, are characterized by high volatility, complex dynamics, and the continuous influx of information. These characteristics present both significant opportunities and substantial risks for traders and investors. Traditional trading strategies often rely on predefined rules or human intuition, which may struggle to adapt to the rapidly changing market conditions and the sheer volume of data. In recent years, AI, and specifically reinforcement learning, has emerged as a promising paradigm for developing adaptive and automated trading systems.

This thesis explores the application of advanced RL techniques to the domain of cryptocurrency trading. The core of our investigation revolves around the DDQN algorithm, a powerful value-based RL method capable of learning optimal policies in complex environments. We aim to develop and evaluate DDQN-based trading agents that can make sequential decisions—buy, sell, or hold—to maximize cumulative returns.

A key aspect of this work is the comparative analysis of different neural network architectures for approximating the Q-function within the DDQN framework. We investigate the performance of a conventional 1D CNN, often used for sequence data, against a more recent and specialized time-series model, TimesNet [1]. TimesNet's architecture is designed to capture multi-scale temporal patterns and periodicities, which we hypothesize could be particularly beneficial for modeling financial time series.

The study utilizes historical daily and weekly OHLCV data for major cryptocurrencies, Bitcoin and Ethereum. To assess the impact of informational richness, we experiment with two distinct feature sets: a "basic" set comprising only OHLCV data, and an "extended" set that augments the basic features with selected technical indicators (RSI, MACD and ATR) and blockchain-specific on-chain metrics (Average Transactions Per Block, Fees USD per Transaction and Unique Addresses Used).

This research pursues several primary objectives. First, we implement and train DDQN-based trading agents for Bitcoin and Ethereum. Second, we compare the effectiveness of TimesNet and CNN1D as Q-network architectures within the DDQN framework. Third, we evaluate the impact of different feature sets (basic vs. extended) on the trading performance of the RL agents. Finally, we analyze the learned trading strategies and compare their performance against a standard Buy

& Hold benchmark across various market conditions.

Through rigorous experimentation and detailed analysis, this research aims to contribute to the understanding of how advanced AI models like TimesNet can be integrated into RL frameworks for potentially more effective and optimized cryptocurrency trading strategies. The findings are intended to provide insights into the architectural choices and feature engineering considerations crucial for developing robust AI-powered trading systems.

2. Related Work

The application of AI, particularly machine learning and deep learning, to financial trading has been an active area of research for several decades. Early approaches often focused on supervised learning for price prediction or classification tasks [2]. However, the sequential decision-making nature of trading aligns well with the paradigm of reinforcement learning.

RL agents learn to make optimal sequences of actions by interacting with an environment and receiving feedback in the form of rewards or penalties. In financial trading, the environment is the market, actions are buy, sell, or hold, and rewards are typically related to profit or risk-adjusted returns. Sutton and Barto's comprehensive work [3] provides the foundational principles for much of the RL research in this domain.

DQN [4] marked a significant breakthrough by combining Q-learning with deep neural networks to handle high-dimensional state spaces, such as raw pixel inputs from Atari games. This success spurred interest in applying DQNs to financial markets. The DDQN algorithm, proposed by van Hasselt et al. [5], addresses the overestimation bias of Q-values often observed in standard DQN, leading to more stable and reliable learning. DDQN has since become a popular choice for financial RL applications.

The challenge of modeling complex temporal dependencies in financial time series has led to the exploration of various neural network architectures within RL frameworks. CNN have been used to extract features from sequences of price data, and RNNs, particularly LSTMs, have been employed to capture long-term dependencies. More recently, Transformer-based models have also gained attention for their ability to model global relationships in sequences.

A novel approach to general time series analysis was introduced with TimesNet [1]. TimesNet transforms 1D time series into 2D tensors based on identified periods, allowing 2D convolutional kernels to capture both intraperiod and interperiod variations. This method has shown strong performance on various time series forecasting benchmarks. Its application within an RL framework for financial trading, as explored in this thesis, is a relatively new direction. Zhou et al. [6] proposed R-DDQN, which integrates a TimesNet-like architecture into a DDQN framework for algorithmic trading, using TimesNet for both the policy network and a separate reward prediction network, demonstrating its potential in financial RL.

Feature engineering is another critical aspect of developing RL trading agents.

While some approaches use raw price data, many incorporate technical indicators (e.g., Moving Averages, RSI, MACD) [7] or alternative data sources like on-chain metrics for cryptocurrencies or sentiment analysis from news and social media [8]. The choice and combination of features can significantly impact an agent's ability to discern market patterns and make profitable decisions.

Despite promising results, real-world deployment of RL trading agents faces challenges, including model overfitting, non-stationarity of financial markets, transaction costs, market impact, and the need for robust risk management [9]. This thesis contributes to this body of work by systematically evaluating DDQN agents with TimesNet and CNN architectures, using different feature sets for cryptocurrency trading, providing insights into their effectiveness and learned behaviors.

3. Data

This study utilizes historical cryptocurrency price data for Bitcoin (BTC) and Ethereum (ETH). The primary data source for OHLCV is Yahoo Finance [17]. Additional blockchain-specific metrics, used for the extended feature set, were sourced from Blockchain.com for Bitcoin data [18] and Etherscan for Ethereum data [19]. A systematic procedure was followed for all data processing, feature engineering, and normalization steps.

The data is split into distinct training and evaluation periods to ensure robust model development and testing. For the training period, Bitcoin data spans from October 1, 2014, to July 31, 2021, while Ethereum data covers the period from December 1, 2017, to July 31, 2021. The evaluation period extends from August 1, 2021, to May 13, 2025, for both assets. This split ensures that the models are trained on a substantial history covering various market phases and tested on unseen data representing more recent market dynamics.

3.1. Base Features: OHLCV Data

The foundational dataset consists of daily OHLCV data for each cryptocurrency. From this daily data, cumulative weekly OHLCV data is also derived. For each day, the weekly features represent the weekly open as the opening price of the first trading day of the current week, the weekly high as the highest price observed cumulatively within the current week up to the current day, the weekly low as the lowest price observed cumulatively within the current week up to the current day, the weekly adjusted close as the daily adjusted closing price, and the weekly volume as the sum of trading volumes cumulatively within the current week up to the current day.

This combination of 5 daily OHLCV metrics and their 5 derived weekly counterparts forms the 10-dimensional base feature set.

3.2. Data Preprocessing

The raw data undergoes several preprocessing steps. First, rows with any missing values in core OHLCV columns are dropped to ensure data completeness. Entries where trading volume is zero are removed, as these often indicate non-trading days or data errors. The date column is converted to datetime objects, and the data is filtered to the specified training or testing date range.

Feature engineering and derivation follows the procedures described for weekly OHLCV data and additional technical indicators and blockchain metrics. The daily adjusted close price is preserved as the original price for calculating trading performance and rewards.

All input features including daily and weekly versions are standardized using the mean and standard deviation calculated exclusively from the training dataset for each feature. The normalization formula applied is $(X - \mu)/\sigma$. These training set parameters are then applied to normalize the test set to prevent data leakage.

Finally, the processed and normalized data is transformed into sequences of a fixed length L. Each sequence represents a state for the reinforcement learning agent.

3.3. Extended Feature Set

To evaluate the impact of richer information, an extended feature set is constructed by augmenting the 10 base OHLCV features. This set includes selected technical indicators and blockchain metrics. Similar to the base OHLCV data, both daily values and derived weekly aggregations of these additional metrics are computed and normalized, resulting in 12 additional features. This brings the total dimensionality of the extended feature set to 22 features (10 base + 6 technical indicator-related + 6 blockchain-related).

3.3.1. Technical Indicators

Three technical indicators are calculated from the daily OHLCV data using standard financial computation methods. These indicators help capture different aspects of market behavior beyond raw price and volume data and have been shown to provide valuable signals for algorithmic trading systems [10].

The **STOCHRSI** measures momentum by comparing the closing price to its price range over a specific period. This indicator helps the reinforcement learning agent identify when a cryptocurrency might be overbought or oversold, providing crucial timing signals for buy and sell decisions. It oscillates between 0 and 1, where values above 0.8 typically indicate overbought conditions and values below 0.2 indicate oversold conditions. The calculation involves first computing the RSI, then applying the stochastic formula:

$$\text{STOCHRSI} = \frac{\text{RSI} - \text{RSI}_{\text{min}}}{\text{RSI}_{\text{max}} - \text{RSI}_{\text{min}}} \tag{3.1}$$

where $\mathrm{RSI}_{\min}$ and $\mathrm{RSI}_{\max}$ are the minimum and maximum RSI values over the lookback period.

The **MACD** is a trend-following indicator that shows the relationship between two EMAs of different periods. For reinforcement learning trading, MACD provides valuable information about trend direction and momentum changes, enabling the agent to align trading decisions with market trends [11]. The MACD line is calculated as:

$$\mathrm{MACD} = \mathrm{EMA}_{12} - \mathrm{EMA}_{26} \tag{3.2}$$

A signal line is then computed as a 9-period EMA of the MACD line. When the MACD line crosses above the signal line, it may indicate a bullish trend, and vice versa for bearish trends.

The **ATR** measures market volatility by calculating the average range between high and low prices over a specified period. This metric is particularly valuable for reinforcement learning agents as it helps assess market risk and adjust position sizing accordingly [12]. The true range for each period is defined as:

$$\mathrm{TR} = \max(\mathrm{High} - \mathrm{Low}, |\mathrm{High} - \mathrm{Close}_{\mathrm{prev}}|, |\mathrm{Low} - \mathrm{Close}_{\mathrm{prev}}|) \tag{3.3}$$

The ATR is then computed as the EMA of the true range values. Higher ATR values indicate greater volatility, while lower values suggest more stable price movements.

For each of these three indicators, both daily values and derived weekly aggregations are included in the extended feature set.

3.3.2. Blockchain Metrics

Three blockchain-specific metrics are incorporated to capture network activity and fundamental value drivers that traditional financial indicators cannot measure. These on-chain metrics provide the reinforcement learning agent with insights into the underlying network health and adoption patterns that may predict future price movements [13].

The **Average Transactions Per Block** measures network usage intensity by calculating the mean number of transactions processed in each newly mined block. This metric serves as a proxy for network demand and can signal increasing adoption or periods of high activity that often precede price movements. For a trading agent, this information helps identify fundamental shifts in network usage that may not yet be reflected in price data [14].

The **Fees USD per Transaction** represents the average cost users pay to have their transactions processed on the network. Transaction fees indicate network congestion and user willingness to pay for blockchain services. Rising fees often signal increased demand for the network, which can be a leading indicator of price appreciation. This metric provides the reinforcement learning agent with early signals of changing network dynamics that traditional price-based indicators might miss [15].

The **Unique Addresses Used** counts the number of distinct wallet addresses that participated in transactions on a given day. This metric serves as a measure of network adoption and user base growth. Research has shown that network effects in cryptocurrencies follow patterns similar to other network technologies, where growing user bases create positive feedback loops that drive value [16]. For the trading agent, this metric provides information about the fundamental strength of the cryptocurrency ecosystem beyond short-term price fluctuations.

These blockchain metrics complement traditional financial indicators by providing information about the underlying technology adoption and network utilization that drives cryptocurrency value. For each of these three blockchain metrics, both daily values and derived weekly aggregations are included in the extended feature set.

4. Methods

This chapter details the methodologies employed in this thesis. Our approach centers on reinforcement learning, specifically utilizing the DDQN framework for algorithmic cryptocurrency trading. We present the foundational concepts of Q-Learning, elaborate on our DDQN architecture, and detail the neural network models used for Q-function approximation.

4.1. Q-Learning: The Foundation

Q-Learning is a model-free reinforcement learning algorithm used to find an optimal action-selection policy for finite Markov decision processes [3]. The algorithm learns a policy that tells an agent what action to take under what circumstances without requiring a model of the environment and can handle problems with stochastic transitions and rewards.

The core of Q-Learning is the action-value function, denoted as $Q(s, a)$. This function represents the expected cumulative discounted reward an agent can achieve by taking action a in state s and then following an optimal policy thereafter. The goal is to learn the optimal Q-function, $Q^*(s, a)$, which satisfies the Bellman optimality equation:

$$Q^*(s, a) = \mathbb{E}[R_{t+1} + \gamma \max_{a'} Q^*(s_{t+1}, a')|S_t = s, A_t = a]$$

where R_{t+1} is the immediate reward received after taking action a in state s, s_{t+1} is the next state, γ is the discount factor balancing immediate and future rewards, and a' represents all possible actions in state s_{t+1}.

Traditional Q-Learning uses a table to store Q-values for every state-action pair. The Q-values are iteratively updated using the following rule based on experiences (s_t, a_t, R_t, s_{t+1}):

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left(R_t + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t) \right)$$

where α is the learning rate, determining how much new information overrides old information.

The algorithm balances exploration (trying new actions to discover their Q-values) and exploitation (choosing the action with the currently known highest Q-value). This is often managed using ϵ-greedy strategies, where the agent chooses a random action with probability ϵ and the best-known action with probability $1 - \epsilon$.

While effective for problems with discrete and small state-action spaces, traditional Q-Learning becomes infeasible for large or continuous state spaces, such as those encountered in financial markets. This limitation led to the development of DQNs, which use neural networks to approximate the Q-function $Q(s, a; \theta)$, where θ represents the network parameters.

4.2. DDQN for Trading

The DDQN approach, an advancement over the standard DQN, is adapted for cryptocurrency trading in this thesis. DDQN mitigates the overestimation bias of Q-values often observed in DQN, which can lead to suboptimal policies. By decoupling action selection from action evaluation during Q-value updates, DDQN provides more stable learning and better performing policies [5].

4.2.1. DDQN System Architecture

The DDQN system developed for cryptocurrency trading integrates several key components that interact to learn and execute trading strategies. The environment component simulates the cryptocurrency market for Bitcoin or Ethereum, providing current market states to the agent, receiving action signals, processing these signals to determine actual trades based on the agent's current position, and calculating resultant rewards.

The DDQN agent serves as the learning entity that develops a policy to select optimal action signals. Internally, it comprises two neural networks: the policy network for action selection and Q-value estimation, and the target network for stabilizing Q-value targets during training.

States represent the market at given time steps, constructed as sequences of recent market features. Action signals are the decisions output by the agent's policy network, consisting of three discrete signals: Buy, Hold, or Sell, indicating the agent's intended market operation. The environment processes these action signals in conjunction with the agent's current market position to determine actual trading actions and calculate rewards.

The policy network, parameterized by θ, approximates the action-value function and is actively trained to predict expected returns for each action signal in given states. The target network, parameterized by θ^{-}, is a periodically updated copy of the policy network that provides stable target values for Q-value updates during training, mitigating oscillations and improving learning stability.

The replay memory stores the agent's past experiences as transitions, allowing minibatches to be randomly sampled for training the policy network. This approach breaks correlations in sequential experiences and improves data efficiency.

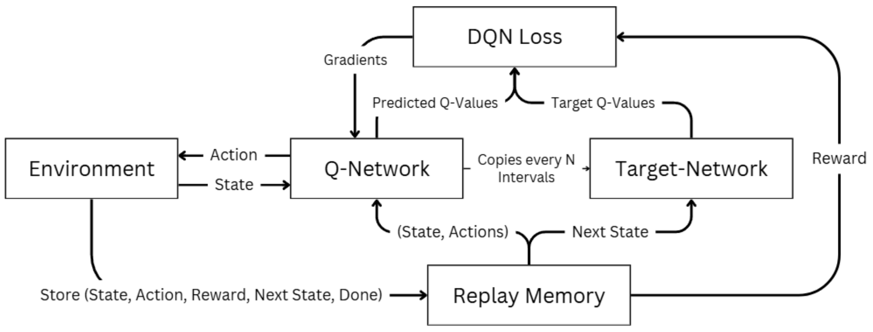


Figure 4.1.: DDQN Model Structure showing agent-environment interaction flow where the agent outputs action signals that the environment processes to determine actual trades and subsequent positions.

4.2.2. State Representation

The state s_t provided to the agent at each time step t is constructed as a sequence of the most recent L historical market observations. In this study, the sequence length L is fixed at 20 time steps representing days. All features constituting the state are normalized using mean and standard deviation from the training set prior to being fed into the neural network.

Two distinct state configurations are employed corresponding to the feature sets used. The base state representation is formed using 20 time steps of the 10 base features, consisting of 5 daily OHLCV metrics and their 5 derived weekly counterparts, resulting in a state s_t tensor of shape 20×10. The extended state representation augments the base features, including 20 time steps of the 22 extended features comprising the 10 base OHLCV features plus 12 additional features derived from 3 technical indicators and 3 blockchain metrics, with each of these 6 additional metrics provided in both daily and weekly aggregated forms, resulting in a state s_t tensor of shape 20×22.

The daily Adjusted Close at each time step is preserved separately and is not part of the normalized state input to the network, being used by the environment for calculating trading performance and rewards.

4.2.3. Action Space and Trading Environment Logic

The DDQN agent's policy network outputs Q-values for three discrete action signals, $a_t^* \in \{-1, 0, 1\}$, corresponding to Sell, Hold, and Buy respectively, representing the agent's intended trading decision.

The market simulation environment interprets these action signals based on the agent's current market position, $POS_t \in \{-1, 0, 1\}$ representing Short, Neutral/-Cash, or Long positions. Based on this combination, the environment determines the actual trading action a_t executed in the simulated market and the agent's new market position POS_{t+1}.

The specific logic mapping current position and agent action signal to actual trading action and new position is implemented as a fixed mapping within the environment, summarized in Table 4.1.

Table 4.1.: Trading Environment Action Logic

Current Position (POS_t)	Agent's Signal (a_t^*)	Signal Meaning	Actual Action (a_t)	New Position (POS_{t+1})
Neutral (0)	Buy (1)	Request Long	Open Long	Long (1)
Neutral (0)	Hold (0)	Request Hold	Hold Cash	Neutral (0)
Neutral (0)	Sell (-1)	Request Short	Open Short	Short (-1)
Long (1)	Buy (1)	Request Hold	Hold Long	Long (1)
Long (1)	Hold (0)	Request Hold	Hold Long	Long (1)
Long (1)	Sell (-1)	Request Close	Close Long	Neutral (0)
Short (-1)	Buy (1)	Request Close	Close Short	Neutral (0)
Short (-1)	Hold (0)	Request Hold	Hold Short	Short (-1)
Short (-1)	Sell (-1)	Request Hold	Hold Short	Short (-1)

This deterministic mapping allows the agent to learn complex trading strategies involving long positions, short positions, and holding cash. The agent's learned policy emerges from its interaction with this environment and the feedback received via the reward function. This implementation of discrete action signals and position-based logic follows principles found in contemporary RL trading systems, similar to approaches in Zhou et al. [6].

4.2.4. Reward Function

The reward function R_t guides the agent's learning process and is calculated by the market simulation environment based on the agent's new position POS_{t+1} and price movements over a configurable future horizon of m days.

Let p_t be the Original Price (Adjusted Close) at time t. The function first computes future percentage returns $r_{t+i} = (p_{t+i} - p_t)/p_t \times 100$ for $i = 1, \ldots, m$. It then identifies the maximum positive return and minimum negative return within this horizon. A dominant price change metric is subsequently derived to capture the most significant price movement.

The reward R_t is assigned based on the agent's new position POS_{t+1}. For long positions, the reward is proportional to the dominant price change, being positive if prices are expected to rise and negative if they are expected to fall. For short positions, the reward is proportional to the negative of the dominant price change, being positive if prices are expected to fall and negative if they are expected to rise. For neutral positions, the reward consists of a small base value for holding cash, penalized by observed price volatility in the horizon.

This reward structure teaches the agent to enter positions that align with favorable future price movements and to stay neutral during volatile or directionless periods, encouraging profit-seeking while managing risk exposure. This approach of defining rewards based on future price movements follows common techniques in financial RL studies, sharing similarities with reward formulations in Zhou et al. [6].

4.2.5. DDQN Learning Algorithm

The DDQN agent learns by iteratively updating the parameters θ of its policy network to minimize the Mean Squared Error between Q-values predicted by the policy network and target Q-values.

For a transition $(s_t, a_t^*, R_t, s_{t+1})$ randomly sampled from the replay memory D, the target Q-value y_t is calculated using the DDQN update rule that distinguishes it from standard DQN:

$$y_t = R_t + \gamma Q(s_{t+1}, \underset{a'}{\operatorname{argmax}} Q(s_{t+1}, a'; \theta_t); \theta_t^-)$$

The action a' that maximizes the Q-value in the next state s_{t+1} is selected using the current policy network (parameters θ_t), while the Q-value for this selected action is estimated using the target network parameters (θ_t^-). This decoupling reduces the overestimation bias found in standard DQN.

The loss function is formulated as the expectation over sampled transitions:

$$L(\theta_t) = \mathbb{E}_{(s_t, a_t^*, R_t, s_{t+1}) \sim U(D)} \left[(y_t - Q(s_t, a_t^*; \theta_t))^2 \right]$$

This loss is minimized through gradient descent using the Adam optimizer. Experience replay stores agent experiences in a buffer with fixed capacity, with minibatches randomly sampled during training to break temporal correlations in sequential experiences. Target network parameters are periodically synchronized with policy network parameters to provide stable targets for Q-value learning. Exploration-exploitation balance is managed through ϵ-greedy strategy, where the agent selects random action signals with probability ϵ and otherwise selects the action signal that maximizes the predicted Q-value. The exploration rate ϵ is annealed from an initial high value to a final low value over predefined training steps, encouraging exploration early in training and shifting towards exploitation as the agent learns.

The complete training process is outlined in Algorithm 1, which details how the agent iteratively interacts with the environment, accumulates experiences, and updates its policy. The algorithm begins by initializing the policy and target networks, replay memory, and optimizer. For each episode, the agent resets its market position and iterates through the training dataset. At each step, the agent observes the current state, selects an action signal based on the current exploration rate using ϵ-greedy strategy, and allows the environment to determine the actual trading action and calculate the reward. These experiences are stored in the replay memory, and when sufficient samples are available, minibatches are sampled for policy network updates. The target network is periodically updated to maintain learning stability throughout the training process.

Algorithm 1 DDQN Trading Agent Training

```
 1: Input: Training data, replay buffer capacity, network parameters, batch size,
    target update frequency, learning rate, discount factor, reward horizon, number
    of episodes, epsilon decay schedule
 2: Initialize policy network $Q(\cdot, \cdot; \theta)$ and target network $Q(\cdot, \cdot; \theta^-)$ with $\theta^- \leftarrow \theta$
 3: Initialize replay memory $D$ with specified capacity
 4: Initialize optimizer for policy network using specified learning rate
 5: Initialize global training step counter
 6: for each episode do
 7:     Reset current market position in environment
 8:     for each step in training dataset do
 9:         Observe current state and future price information
10:         Calculate current exploration rate $\epsilon$
11:         if random number $< \epsilon$ then
12:             Select random action signal
13:         else
14:             Select action signal that maximizes Q-value
15:         end if
16:         Environment determines actual trading action and new position
17:         Calculate reward based on new position and future price movements
18:         Store transition in replay memory
19:         Update current state
20:         if replay memory has sufficient samples then
21:             Sample minibatch of transitions from replay memory
22:             For each transition in minibatch:
23:             Select next action using policy network
24:             Calculate target Q-value using target network
25:             Calculate loss between predicted and target Q-values
26:             Perform gradient descent to update policy network
27:         end if
28:         if target update frequency reached then
29:             Update target network parameters from policy network
30:         end if
31:     end for
32: end for
```

4.3. CNN1D Architecture

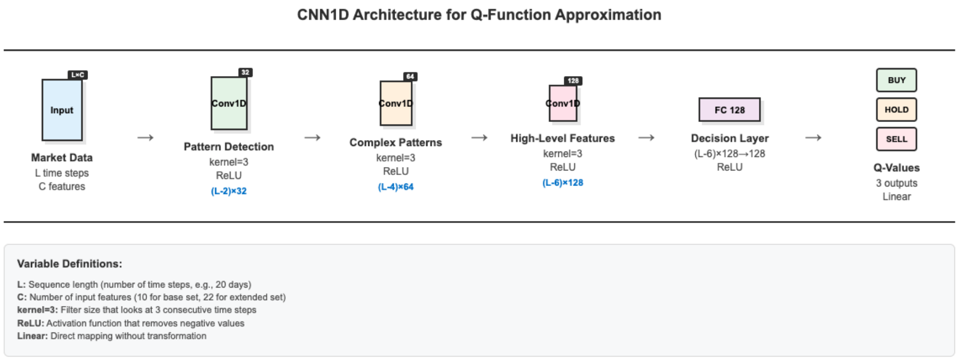
The 1D CNN serves as one neural network architecture for Q-function approximation within the DDQN framework. This architecture processes sequential data effectively, making it well-suited for the time series of market features that constitute states in our trading environment.

The CNN1D architecture accepts state tensors representing sequences of time steps, each containing market features. The number of features varies depending on whether the base feature set or extended feature set is used. The input data is initially structured with time steps as the second dimension and features as the third dimension but is permuted to align with PyTorch Conv1d layer expectations of channels first format.

The network employs three 1D convolutional layers with progressively increasing output channels. The first convolutional layer takes the number of input features as input channels and outputs 32 channels, using a kernel size of 3 followed by ReLU activation. This layer extracts low-level temporal patterns from the input sequences. The second convolutional layer takes 32 input channels and outputs 64 channels, also using kernel size 3 and ReLU activation, building upon features from the first layer to capture more complex patterns. The third convolutional layer processes 64 input channels to produce 128 output channels, using kernel size 3 and ReLU activation to further abstract the learned features.

Following the convolutional layers, the output undergoes flattening to convert the multi-dimensional feature maps into a single vector for each sample. This flattened representation is processed through a fully connected layer that projects to 128 dimensions, followed by ReLU activation for further non-linear transformation and integration of learned temporal features.

The final output layer consists of a fully connected layer that takes the 128-dimensional vector and produces 3 values corresponding to Q-values for each discrete action signal (Sell, Hold, Buy). This layer uses linear activation since Q-values are unbounded.



Variable Definitions:

L: Sequence length (number of time steps, e.g., 20 days)
C: Number of input features (10 for base set, 22 for extended set)
kernel=3: Filter size that looks at 3 consecutive time steps
ReLU: Activation function that removes negative values
Linear: Direct mapping without transformation

Figure 4.2.: CNN1D architecture for Q-function approximation showing the progression from input market data through three convolutional layers to final Q-value outputs. Each layer extracts increasingly complex temporal patterns from the time series data.

This layered architecture enables the CNN1D model to learn hierarchical representations from input time series data. The convolutional layers function as feature extractors, identifying temporal patterns, while the fully connected layers map these learned features to action-values required for decision-making by the DDQN agent.

4.4. TimesNet Architecture

TimesNet is a deep learning model designed for analyzing time series data [1]. This section explains how TimesNet works and how we adapted it for cryptocurrency trading.

4.4.1. The Core Idea: From 1D to 2D

Traditional models analyze time series as one-dimensional sequences. TimesNet's key innovation is converting these 1D sequences into 2D images, which allows the use of image processing techniques on time series data.

Rather than analyzing stock prices as a single line over time, TimesNet reshapes the data into a grid pattern based on recurring cycles such as daily or weekly patterns. This transformation reveals periodic patterns that are difficult to detect in the original 1D format.

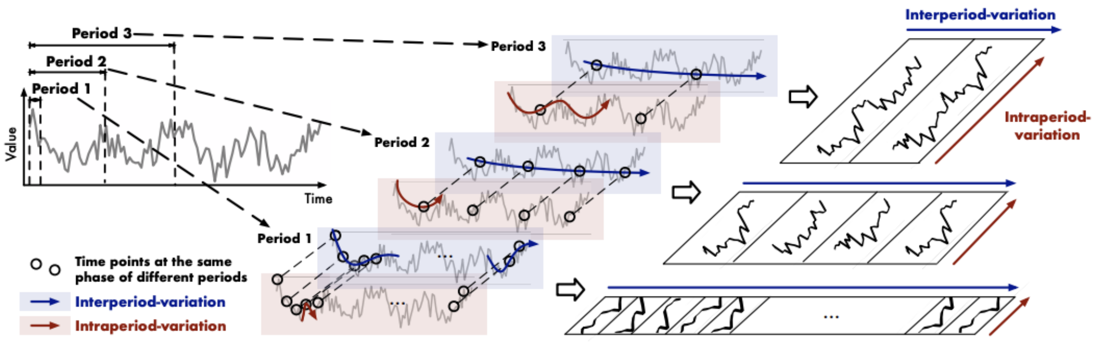


Figure 4.3.: TimesNet transforms 1D time series into 2D representations to capture periodic patterns. Adapted from Wu et al. [1].

4.4.2. Finding Periodic Patterns

TimesNet uses FFT to identify the most significant recurring patterns in the data. In cryptocurrency markets, this includes daily trading cycles or weekly patterns.

The process first analyzes the input time series to find dominant frequencies, then selects the top-k most significant periods, and uses these periods to reshape the data into 2D grids.

Mathematically, for an input sequence X, the FFT analysis identifies periods $\{p_1, p_2, \ldots, p_k\}$ that represent the strongest recurring patterns in the data.

4.4.3. 2D Transformation and Processing

For each identified period p_i, the 1D time series is reshaped into a 2D tensor:

$$X_{2D}^{(i)} = \text{Reshape}(X, \lfloor T/p_i \rfloor, p_i) \tag{4.1}$$

This creates a grid where each row represents one complete cycle of the pattern. The model then applies 2D convolutions to extract features from these grids.

TimesNet uses multiple convolution filters of different sizes to capture both short-term fluctuations and long-term trends.

4.4.4. Combining Multiple Patterns

The model processes multiple periodic patterns and combines their outputs through a weighted fusion mechanism. More prominent patterns have greater influence on the final result:

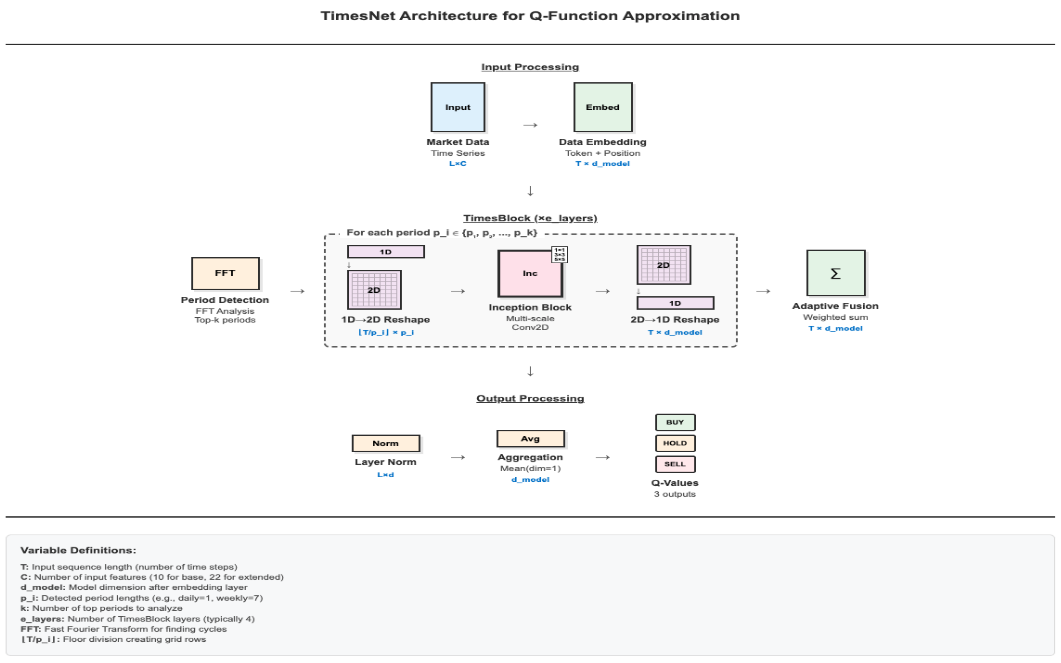
$$Y = \sum_{i=1}^{k} \text{Weight}_i \cdot \text{ProcessedPattern}_i \tag{4.2}$$

The weights are determined by the amplitude strength of each pattern in the original data.

4.4.5. Adaptation for Trading

For our cryptocurrency trading application, we modified TimesNet to work with the DDQN reinforcement learning framework. Our implementation processes two feature sets: basic OHLCV data (Open, High, Low, Close, Volume) and an extended dataset that additionally includes three technical indicators and three on-chain blockchain metrics.

The adapted TimesNet architecture takes sequences of market features as input (L time steps, C features), applies the period detection and 2D transformation process, processes the transformed data through multiple layers, and outputs Q-values for three trading actions: Buy, Hold, or Sell.



Variable Definitions:

T: Input sequence length (number of time steps)
C: Number of input features (10 for base, 22 for extended)
d_model: Model dimension after embedding layer
p_i: Detected period lengths (e.g., daily=1, weekly=7)
k: Number of top periods to analyze
e_layers: Number of TimesBlock layers (typically 4)
FFT: Fast Fourier Transform for finding cycles
⌊T/p_i⌋: Floor division creating grid rows

Figure 4.4.: Complete TimesNet architecture for Q-function approximation showing the data flow from input processing through TimesBlock iterations to final Q-value outputs. The architecture processes multiple periodicities in parallel and fuses them adaptively for robust trading decisions.

This allows the trading agent to make decisions based on both short-term market movements and longer-term cyclical patterns in cryptocurrency markets.

4.4.6. Computational Efficiency

TimesNet maintains computational efficiency through its design. The FFT-based period detection operates in $\mathcal{O}(T \log T)$ time complexity, where $\mathcal{O}$ denotes computational complexity (Big O notation), T represents the sequence length, and $\log T$ is the logarithm of T. This means the computation time grows as the sequence length times its logarithm, making it highly scalable compared to quadratic algorithms. The 2D convolutions are optimized for modern hardware. By limiting the analysis to the top-k most significant periods, the model maintains predictable computational requirements while capturing the most relevant patterns in the data.

This efficiency makes TimesNet suitable for real-time trading applications, though it requires more computation than basic CNN architectures.

5. Results

This chapter presents the comprehensive evaluation results of DDQN agents with TimesNet architecture for cryptocurrency trading. The experimental validation demonstrates TimesNet's performance across multiple dimensions, including training convergence, trading effectiveness, and risk-adjusted returns through systematic evaluation of temporal pattern recognition capabilities and feature engineering impacts. CNN architecture serves as a comparative benchmark to validate the effectiveness of the proposed TimesNet approach. The analysis is structured across several key sections: Section 5.1 details the experimental setup and configuration parameters, Section 5.2 examines the training convergence characteristics of both TimesNet and CNN architectures to establish learning dynamics and stability patterns, Section 5.3 provides comprehensive performance evaluation comparing TimesNet variants against CNN and benchmark strategies, and Section 5.4 presents detailed behavioral analysis of TimesNet's decision-making patterns and market timing capabilities with comparative CNN position management statistics. The impact of different feature configurations on both TimesNet and CNN performance is examined throughout to understand the contribution of extended features versus basic OHLCV data in cryptocurrency market prediction.

5.1. Experimental Configuration

The experiments utilize cryptocurrency price data with distinct training and evaluation periods to ensure robust model assessment. Training data spans from October 1, 2014, to July 31, 2021, for Bitcoin and December 1, 2017, to July 31, 2021, for Ethereum, incorporating both bearish and bullish market phases to develop resilient trading strategies. This temporal split was specifically chosen to include Bitcoin's major bull run (2016-2017), subsequent bear market (2018), recovery phase (2019-2020), and the beginning of the 2021 bull cycle, ensuring exposure to diverse market regimes during training. The evaluation period covers August 1, 2021, to May 13, 2025, providing 3.78 years of out-of-sample testing across diverse market conditions including the 2021-2022 bull-to-bear transition and subsequent recovery phases.

The experimental framework employs two neural network architectures: TimesNet as the primary focus for temporal pattern recognition and CNN as a comparative baseline. TimesNet incorporates frequency domain analysis and multi-scale temporal convolutions specifically designed for time series modeling, while CNN serves as a conventional sequential processing benchmark. Both architectures are evaluated with two feature configurations: basic OHLCV features (10 dimensions) comprising daily and weekly Open, High, Low, Close, and Volume data, and extended features (22 dimensions) including the same daily and weekly OHLCV data (10 dimensions) augmented with technical indicators—RSI, MACD, and ATR—plus blockchain metrics including transactions per block, fees used per transaction, and unique addresses, with all additional features provided in both daily and weekly aggregations (12 additional dimensions).

Hyperparameter	**Value**
State sequence length (L)	20
Input channels (C)	10/22
Batch size (N_b)	128
Learning rate (α)	0.0001
Discount factor (γ)	0.95
Replay buffer size	30,000
Target network update frequency	100 steps
ϵ-greedy decay steps	30% of total steps
Initial exploration rate (ϵ_{start})	1.0
Final exploration rate (ϵ_{end})	0.05
Number of episodes	50

Table 5.1.: Hyperparameter configuration for DDQN agents across all experimental configurations.

The hyperparameter selection was informed by cryptocurrency market characteristics and computational constraints. The moderate learning rate (0.0001) and discount factor (0.95) were chosen to accommodate the high volatility inherent in cryptocurrency markets, while the replay buffer size (30,000) and batch size (128) provide sufficient experience diversity for stable learning [4]. The epsilon-greedy decay schedule spans 30% of total training steps to balance exploration and exploitation in dynamic market environments [3]. Due to computational resource limitations, these hyperparameters were manually configured based on established reinforcement learning practices [5, 4] rather than extensive grid search optimization, representing a pragmatic approach for proof-of-concept validation.

The TimesNet architecture processes 20-step sequences through FFT-based period extraction and multi-scale Inception blocks, incorporating 4 layers with 5 top-k frequencies and 6 kernels for comprehensive temporal pattern recognition. The CNN architecture employs three convolutional layers as a comparative baseline for sequential pattern processing. Both architectures output Q-values for three discrete actions: Buy (Long), Hold (Neutral), and Sell (Short), enabling systematic performance comparison across identical training protocols and evaluation metrics.

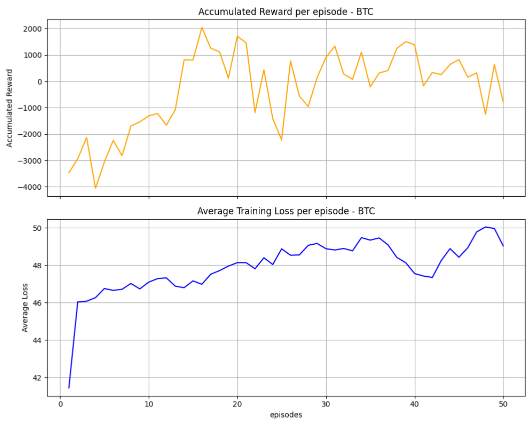
This systematic experimental design enables comprehensive assessment of both architectural choices and feature engineering impacts on cryptocurrency trading performance, with primary focus on TimesNet capabilities and CNN serving as a performance benchmark for validation of the proposed temporal modeling approach.

5.2. Training Performance Results

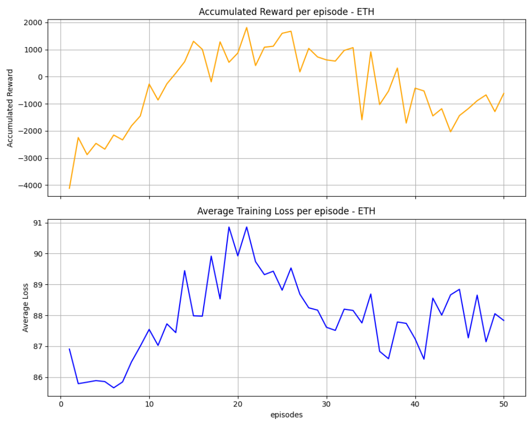
The training progression over 50 episodes revealed distinct convergence characteristics between CNN and TimesNet architectures across all experimental configurations. This analysis examines the learning dynamics of both approaches, with CNN demonstrating more challenging convergence patterns compared to TimesNet's more stable learning progression.

5.2.1. CNN Training Performance

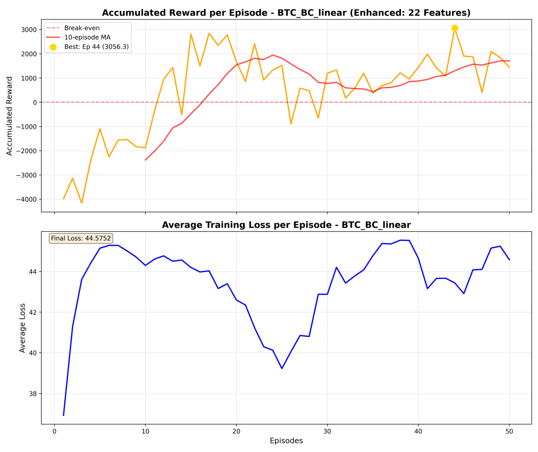
CNN configurations exhibited more volatile convergence patterns with generally slower stabilization compared to TimesNet variants, as illustrated in Figure 5.1. The CNN architectures demonstrated greater difficulty in achieving consistent reward accumulation, with more pronounced fluctuations throughout the training process.



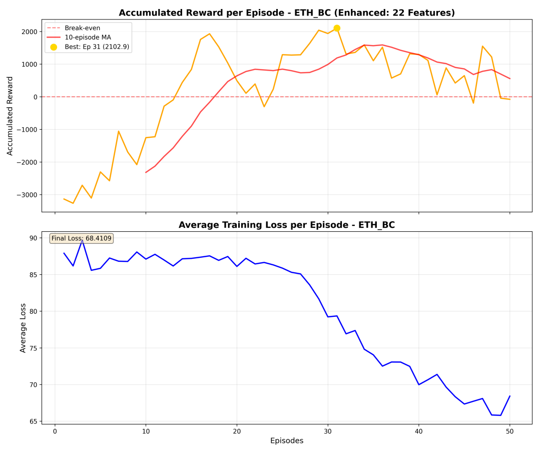
(a) Bitcoin OHLCV features



(b) Ethereum OHLCV features



(c) Bitcoin extended features



(d) Ethereum extended features

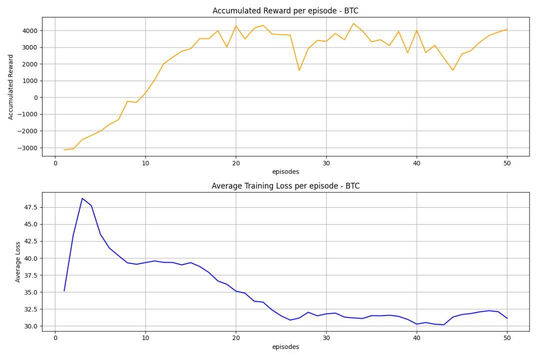
Figure 5.1.: Training performance metrics showing accumulated reward and average loss per episode for all four CNN configurations across Bitcoin and Ethereum with OHLCV and extended feature sets.

CNN training demonstrated more erratic reward progression, with frequent oscillations between positive and negative territories throughout the 50-episode training period. Bitcoin CNN configurations showed particularly volatile behavior, with extended features achieving occasional peaks around 2000-3000 but failing to maintain consistent positive performance. Ethereum CNN variants exhibited similar instability, with reward trajectories frequently crossing the break-even threshold without establishing stable convergence patterns.

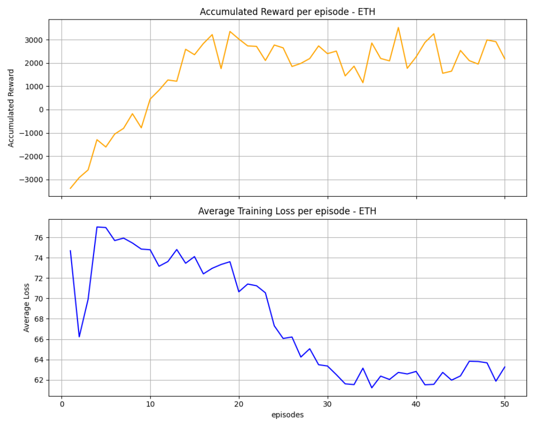
Training loss patterns for CNN architectures revealed the underlying optimization challenges, with loss values showing less consistent decreasing trends compared to subsequent TimesNet results. The CNN's difficulty in achieving stable convergence reflects the limitations of conventional convolutional approaches in capturing the complex temporal dependencies inherent in cryptocurrency market data.

5.2.2. TimesNet Training Performance

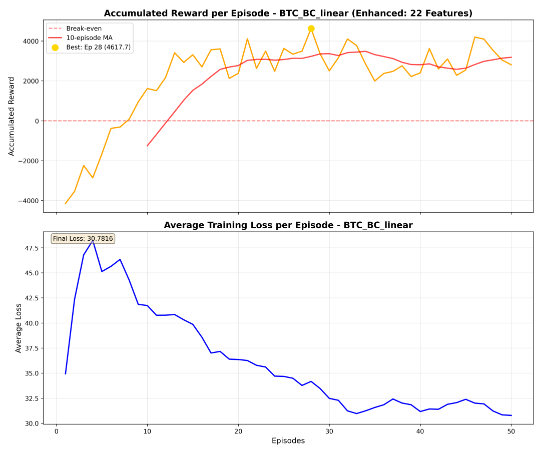
In contrast to CNN's volatile training patterns, TimesNet configurations demonstrated superior convergence characteristics across all four experimental setups, as illustrated in Figure 5.2. All TimesNet variants began with similar negative rewards around -3000 to -4000 in initial episodes but showed more consistent improvement to positive territory within the first 10-15 episodes.



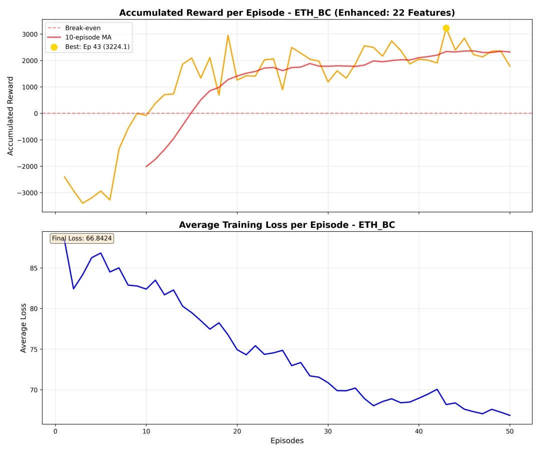
(a) Bitcoin OHLCV features



(b) Ethereum OHLCV features



(c) Bitcoin extended features



(d) Ethereum extended features

Figure 5.2.: Training performance metrics showing accumulated reward and average loss per episode for all four TimesNet configurations across Bitcoin and Ethereum with OHLCV and extended feature sets.

Bitcoin TimesNet configurations achieved reward stabilization around episodes 15-20, with the OHLCV variant maintaining values between 3000-4000 and the extended features variant reaching higher peaks around 4000-4500. Ethereum demonstrated similar convergence patterns, with OHLCV features stabilizing in the range of 2000-3000 and extended features showing more volatile but generally positive performance between 1500-3000.

Training loss convergence confirmed superior neural network optimization across all TimesNet configurations. Bitcoin variants decreased from initial values of approximately 35-48 to stable values around 30-31, while Ethereum configurations reduced from 66-87 to approximately 62-67, indicating successful policy learning and Q-function approximation convergence across both feature sets and cryptocurrency markets.

The comparative analysis reveals TimesNet's superior learning stability and convergence efficiency, demonstrating the architecture's enhanced capability to capture temporal patterns in cryptocurrency market data compared to conventional CNN approaches.

5.3. Trading Performance Evaluation

The performance evaluation was conducted across eight different experimental configurations, comparing DDQN agents with TimesNet and CNN architectures against Buy & Hold benchmark strategies. The evaluation period spans from August 1, 2021, to May 13, 2025, covering 3.78 years of market data that encompasses critical market phases for both cryptocurrencies. For Bitcoin, this period includes the continuation of the 2021 bull market peak, the subsequent bear market correction throughout 2022, and the recovery phase beginning in 2023. For Ethereum, the evaluation captures the transition from proof-of-work to proof-of-stake (The Merge in September 2022), the associated market volatility, and the subsequent adaptation period. This diverse range of market conditions provides a robust testing environment for assessing strategy performance across bull markets, bear markets, and transitional phases. The comprehensive performance comparison is presented in Table 5.2.

TimesNet architectures demonstrated superior performance across both cryptocurrency markets. For Bitcoin, TimesNet with OHLCV features achieved the highest cumulative return of 5596.99% (191.30% annualized) and Sharpe ratio of 2.49, while maintaining maximum drawdown of 31.85%. TimesNet with extended features reached 4228.89% cumulative return (170.89% annualized) with a Sharpe ratio of 2.30.

Ethereum results show TimesNet with extended features achieving 1641.61% cumulative return (112.92% annualized) and Sharpe ratio of 1.47, while the OHLCV variant reached 977.91% cumulative return (87.54% annualized) with Sharpe ratio of 1.31. Both TimesNet configurations substantially outperformed their respective

Strategy	**Cumulative Return (%)**	**Annualized Return (%)**	**Sharpe Ratio**	**Max DD (%)**
		Bitcoin Strategies		
TimesNet (OHLCV)	5596.99	191.30	2.49	31.85
TimesNet (Extended)	4228.89	170.89	2.30	40.97
CNN (Extended)	303.38	44.61	0.97	52.94
CNN (OHLCV)	183.73	31.76	0.78	77.46
Buy & Hold	161.56	28.96	0.73	76.63
		Ethereum Strategies		
TimesNet (Extended)	1641.61	112.92	1.47	66.96
TimesNet (OHLCV)	977.91	87.54	1.31	63.83
CNN (Extended)	237.58	37.96	0.81	77.32
CNN (OHLCV)	37.81	8.85	0.44	63.79
Buy & Hold	-20.59	-5.92	0.27	79.35

Table 5.2.: Comprehensive performance comparison across all experimental configurations and benchmark strategies for Bitcoin and Ethereum.

Buy & Hold benchmarks and CNN counterparts.

CNN architectures showed more modest performance improvements. For Bitcoin, CNN with extended features achieved 303.38% cumulative return (44.61% annualized), while the OHLCV variant reached 183.73% (31.76% annualized). Ethereum CNN strategies demonstrated 237.58% and 37.81% cumulative returns respectively, with the extended feature set showing superior performance.

All reinforcement learning strategies outperformed their respective Buy & Hold benchmarks, with TimesNet architectures achieving the most substantial performance gains across both assets and feature configurations.

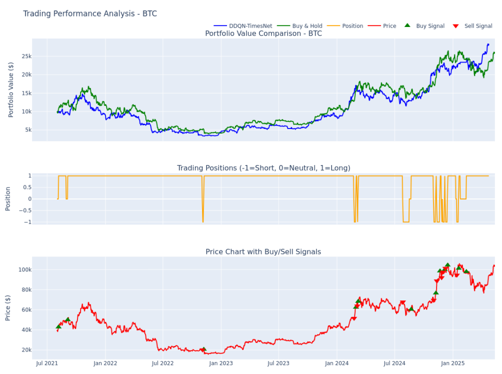
5.4. Trading Behavior Analysis

This section provides comprehensive analysis of trading behaviors for both CNN and TimesNet architectures, examining decision-making patterns, portfolio performance, and market timing capabilities throughout the evaluation period. The comparative analysis reveals distinct behavioral differences between the two approaches across various market conditions and feature configurations.

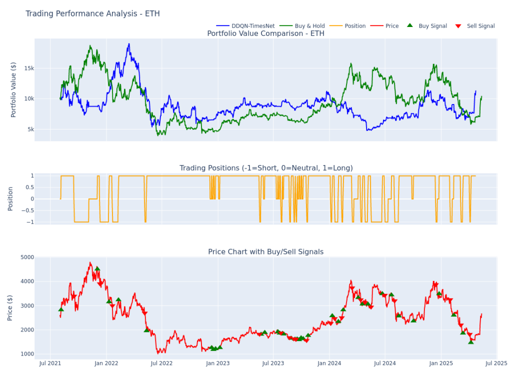
5.4.1. CNN Trading Behavior Analysis

The CNN trading behavior analysis across all four configurations reveals specific positioning patterns and market timing characteristics, as illustrated in Figure 5.3.

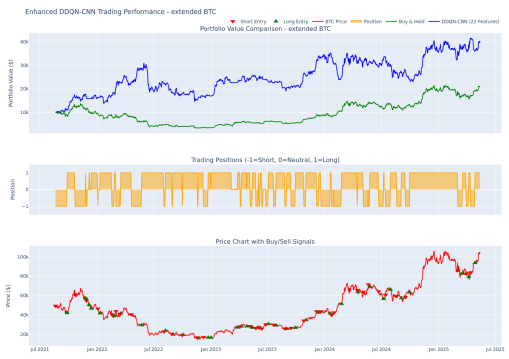
CNN strategies exhibited distinct trading frequencies and positioning behaviors that differ from TimesNet approaches.



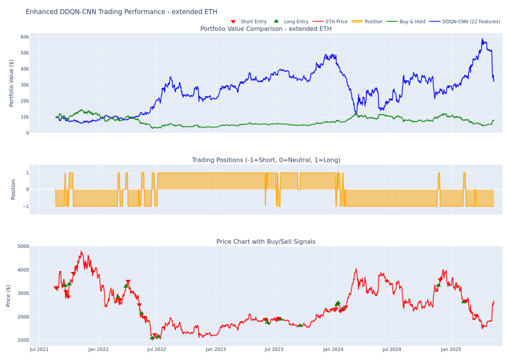
(a) Bitcoin OHLCV features



(b) Ethereum OHLCV features



(c) Bitcoin extended features



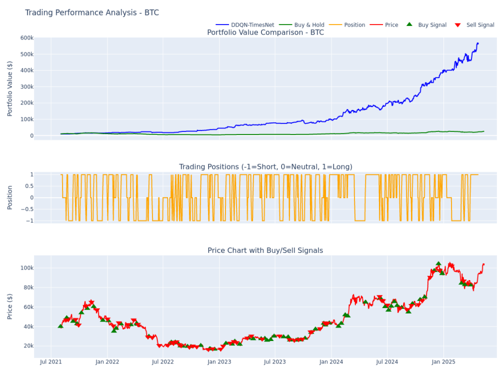
(d) Ethereum extended features

Figure 5.3.: Trading behavior analysis for all CNN configurations showing portfolio value comparison, trading positions, and price charts with buy/sell signals across Bitcoin and Ethereum with OHLCV and extended feature sets.

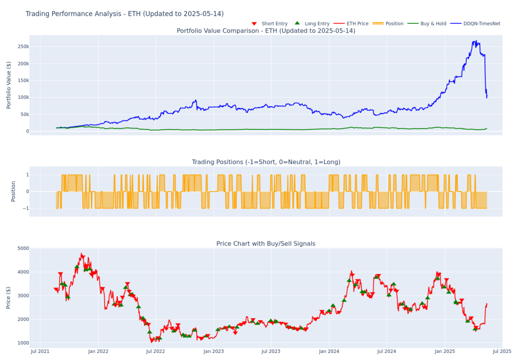
CNN architectures demonstrated lower trading frequencies with fewer position changes across all configurations compared to TimesNet approaches. The Bitcoin extended CNN configuration achieved portfolio growth with notable volatility patterns, while Ethereum CNN variants showed varying performance levels across different market phases. The position management patterns indicate more sustained position holding behaviors, with buy/sell signals occurring at different market timing points compared to TimesNet's more active trading strategies.

5.4.2. TimesNet Trading Behavior Analysis

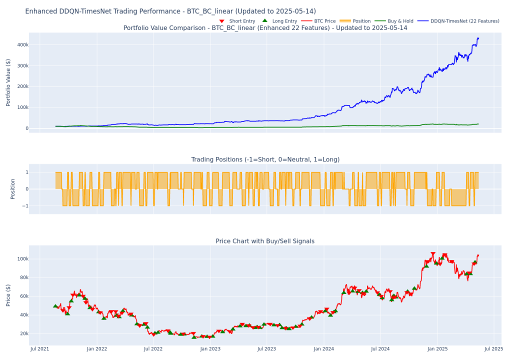
TimesNet configurations demonstrated distinct trading patterns with different market timing characteristics compared to CNN approaches, as presented in Figure 5.4. The comprehensive visualization reveals TimesNet's decision-making patterns, portfolio performance trajectories, and positioning strategies throughout the evaluation period.



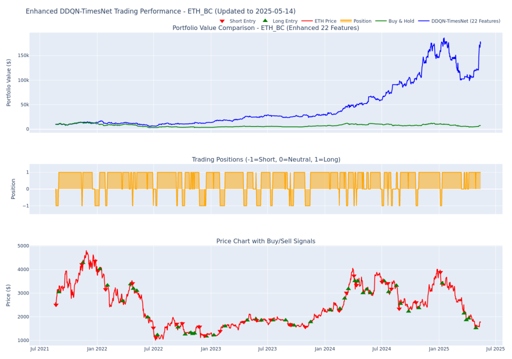
(a) Bitcoin OHLCV features



(b) Ethereum OHLCV features



(c) Bitcoin extended features



(d) Ethereum extended features

Figure 5.4.: Comprehensive trading behavior analysis for all TimesNet configurations showing portfolio value comparison, trading positions, and price charts with buy/sell signals across Bitcoin and Ethereum with OHLCV and extended feature sets.

TimesNet exhibited different portfolio performance trajectories across all configurations, with varying degrees of position management strategies and market timing approaches. The analysis reveals specific decision-making patterns that captured different market trends and risk profiles compared to CNN approaches. TimesNet configurations generally showed different trading frequency patterns, with some configurations demonstrating more selective positioning strategies.

5.4.3. Position Management Statistics

The position allocation patterns reveal distinct trading behaviors across architectures, configurations, and assets, as detailed in Table 5.3.

Arch.	**Config.**	**Long (%)**	**Short (%)**	**Neutral (%)**	**Pos.**	**Win Rate (%)**
			Bitcoin			
TimesNet	OHLCV	49.16	29.20	21.64	136	67.65
	Extended	54.37	29.79	15.85	117	65.81
CNN	OHLCV	91.27	4.70	4.04	22	50.00
	Extended	63.02	28.39	8.58	74	55.41
			Ethereum			
TimesNet	OHLCV	32.70	45.01	22.29	118	61.02
	Extended	74.78	12.68	12.54	74	70.27
CNN	OHLCV	70.51	17.83	11.67	59	49.15
	Extended	43.40	52.35	4.25	36	52.78

Table 5.3.: Position allocation statistics and win rates comparison between TimesNet and CNN configurations across Bitcoin and Ethereum.

The comparative analysis reveals significant architectural differences in position management strategies. TimesNet configurations demonstrated more balanced position allocation patterns across both assets, while CNN architectures showed more pronounced directional biases. For Bitcoin, CNN OHLCV exhibited an extreme long bias (91.27% long vs 4.70% short), contrasting with TimesNet's more balanced approach. CNN extended features showed more diversified positioning (63.02% long, 28.39% short) but still maintained a stronger directional preference compared to TimesNet variants.

Ethereum position allocation patterns revealed contrasting strategies between architectures. TimesNet OHLCV demonstrated short bias (45.01% short vs 32.70% long), while CNN OHLCV showed long preference (70.51% long vs 17.83% short). The extended feature configurations showed similar directional preferences for TimesNet (74.78% long) and CNN (43.40% long), though with different magnitudes.

Win rate analysis reveals architectural performance differences across configurations. TimesNet achieved higher win rates across all configurations, with Bitcoin TimesNet variants reaching 67.65% (OHLCV) and 65.81% (Extended) compared to CNN's 50.00% and 55.41% respectively. Ethereum results showed TimesNet maintaining superiority with 61.02% and 70.27% win rates versus CNN's 49.15% and 52.78%.

Trading frequency patterns differ significantly between architectures. CNN configurations generally exhibited lower trading frequencies, with Bitcoin CNN OHLCV generating only 22 position changes compared to TimesNet's 136, while Ethereum CNN OHLCV produced 59 changes versus TimesNet's 118. This suggests different strategic approaches, with CNN demonstrating more sustained position holding but lower win rates, while TimesNet employed more active positioning with higher success rates.

The extended feature impact varied between architectures. TimesNet showed reduced trading frequency with extended features (Bitcoin: 117 vs 136, Ethereum: 74 vs 118), while CNN demonstrated increased activity for Bitcoin (74 vs 22) and decreased activity for Ethereum (36 vs 59). This indicates different responses to feature complexity, with TimesNet becoming more selective and CNN showing asset-dependent behavior modifications.

Market timing analysis reveals that TimesNet configurations generally demonstrated more precise signal alignment with market extremes, while CNN strategies showed different timing patterns that resulted in varying performance outcomes across market conditions and feature sets.

5.4.4. Detailed Trading Phase Analysis

Two critical market phases are examined to illustrate practical differences between architectures and feature configurations across varying market conditions: the Bitcoin 2022 crash and the Ethereum 2024 rally. This analysis compares CNN and TimesNet decision-making patterns during these distinct market regimes.

Bitcoin 2022 Market Crash (March-May 2022)

The Bitcoin crash period from March to May 2022 represents a volatile bear market phase where prices declined from approximately 47k to 29k. This period provides insights into how different architectures respond to rapidly deteriorating market conditions.

CNN Performance During Bitcoin Crash CNN trading behavior during the 2022 Bitcoin crash reveals specific decision-making patterns, as illustrated in Figure 5.5.

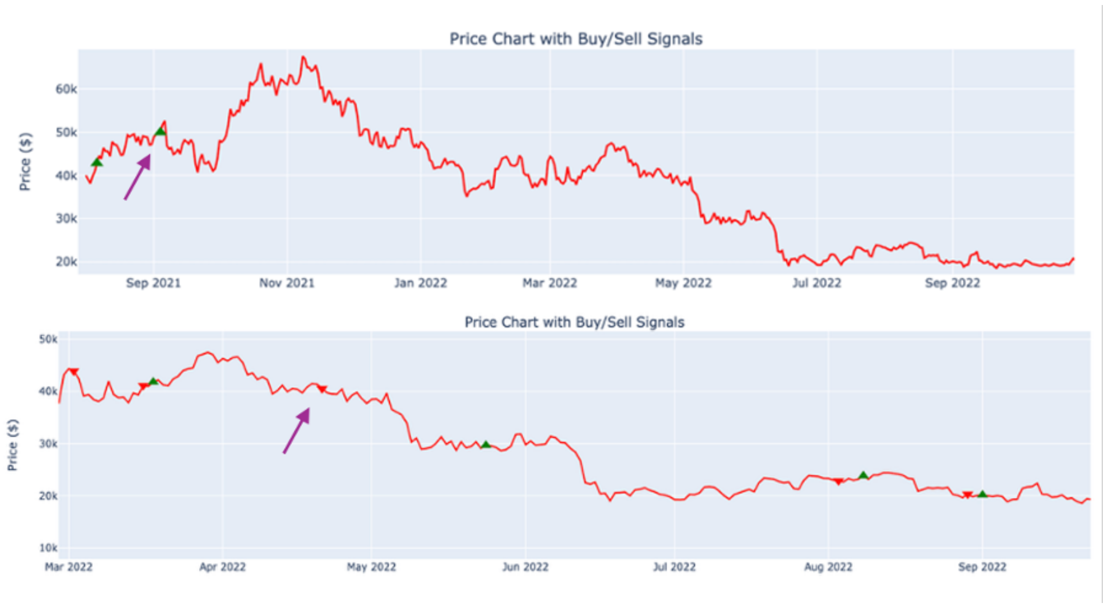


Figure 5.5.: Detailed comparison of CNN Bitcoin trading signals during the 2022 market crash. Top: OHLCV features configuration. Bottom: Extended features configuration. Purple arrows highlight critical timing differences.

CNN OHLCV configuration demonstrated delayed response to the market deterioration, maintaining long positions through the initial decline phase. The extended features CNN variant showed different timing patterns, with position changes occurring at various price levels throughout the crash period. Both CNN configurations exhibited less precise market timing compared to subsequent TimesNet analysis, with signals often occurring after significant price movements had already begun.

TimesNet Performance During Bitcoin Crash TimesNet behavior during the same period demonstrates different market timing characteristics, as illustrated in Figure 5.6.

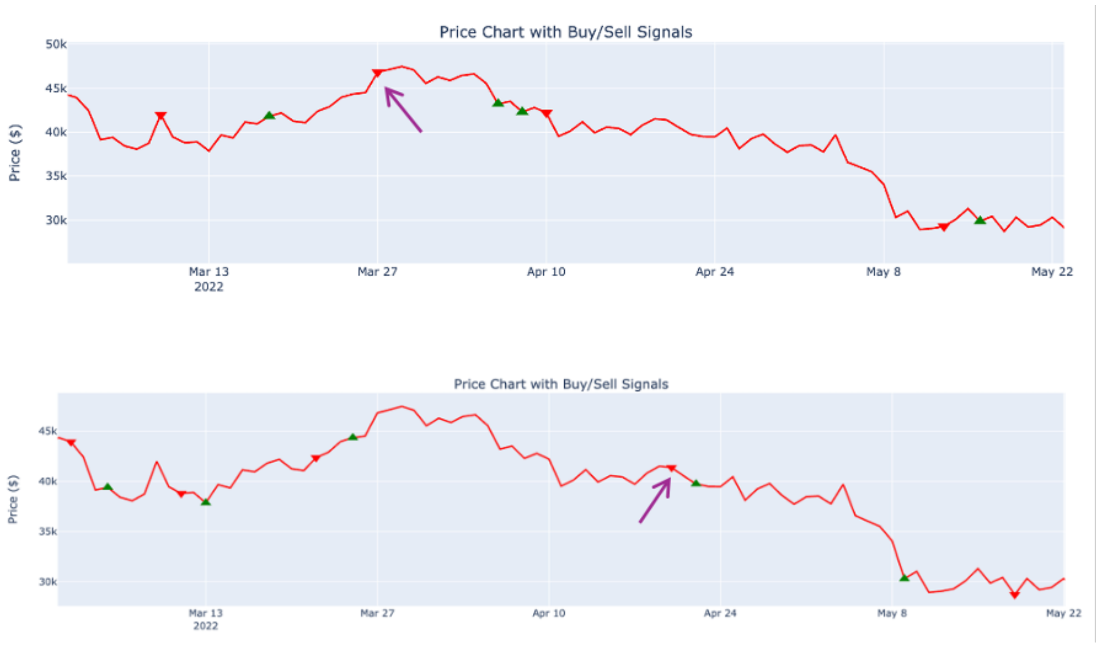


Figure 5.6.: Detailed comparison of TimesNet Bitcoin trading signals during the 2022 market crash. Top: OHLCV features configuration. Bottom: Extended features configuration. Purple arrows highlight critical timing differences.

The TimesNet OHLCV configuration generated a sell signal at approximately 47k in late March, effectively capturing the market top before the significant decline. In contrast, the extended features variant delayed its sell signal until mid-April at around 42k, missing the optimal exit point and suffering additional drawdown during the initial crash phase.

Ethereum 2024 Rally (October 2023-March 2024)

The Ethereum rally period demonstrates contrasting behavior in trending markets, revealing how different architectures respond to sustained upward price movements.

CNN Performance During Ethereum Rally CNN trading patterns during the Ethereum rally show specific positioning strategies, as shown in Figure 5.7.

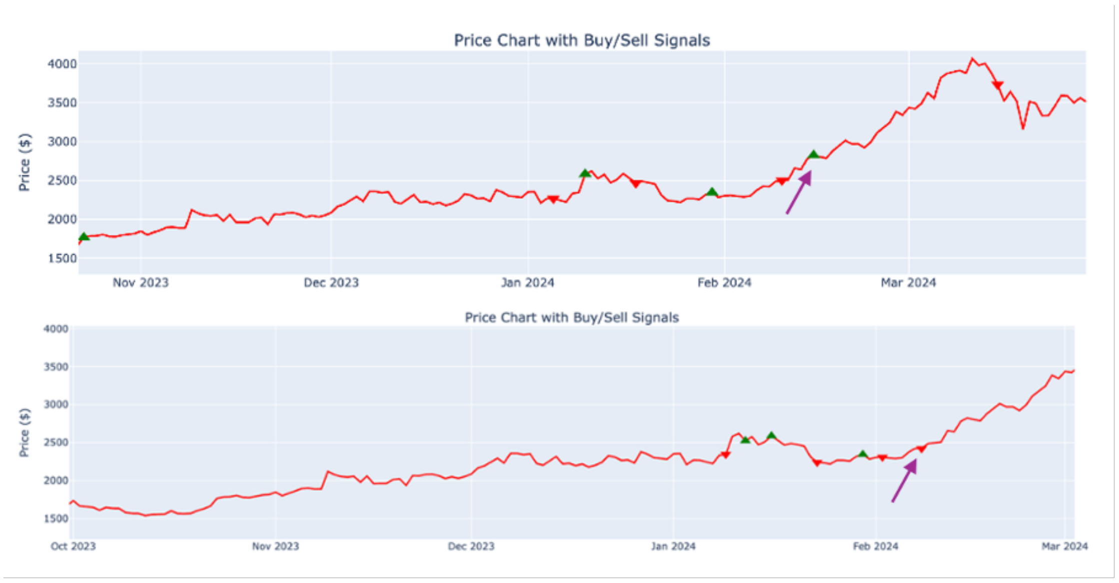


Figure 5.7.: Detailed comparison of CNN Ethereum trading signals during the 2024 rally phase. Top: OHLCV features configuration. Bottom: Extended features configuration. Purple arrows indicate key divergence points in signal timing.

CNN configurations exhibited varying responses to the rally conditions. The OHLCV variant showed intermittent positioning changes throughout the rally period, while the extended features configuration demonstrated different signal timing patterns. Both CNN variants showed less consistent trend-following behavior compared to TimesNet approaches, with more frequent position adjustments during the sustained upward movement.

TimesNet Performance During Ethereum Rally TimesNet behavior during the rally period reveals different trend recognition capabilities, as shown in Figure 5.8.

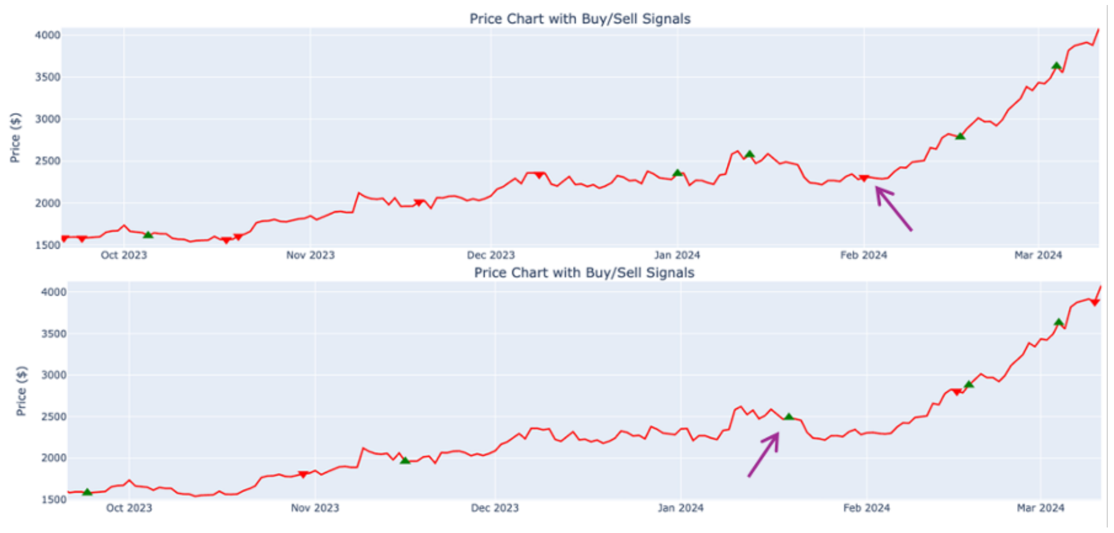


Figure 5.8.: Detailed comparison of TimesNet Ethereum trading signals during the 2024 rally phase. Top: OHLCV features configuration. Bottom: Extended features configuration. Purple arrows indicate key divergence points in signal timing.

The TimesNet OHLCV configuration generated a sell signal around 2,300 USD in early February 2024, effectively exiting during the rally's early phase. In contrast, the extended features variant generated a strategic buy signal around 2,500 USD in mid-January 2024, positioning advantageously for the subsequent rally acceleration that reached peaks near 4,000 USD.

Comparative Analysis Across Market Regimes

The comparative analysis reveals distinct architectural responses to different market conditions. CNN configurations generally showed more reactive positioning patterns, with signals often following rather than anticipating major price movements. TimesNet demonstrated more proactive market timing, particularly evident in crash scenarios where early signal generation provided better downside protection.

Feature set effectiveness varied between architectures and market conditions. For CNN, extended features showed mixed results across both market phases, while TimesNet exhibited clearer regime-dependent performance differences. These contrasting examples demonstrate differing feature set effectiveness across market regimes, with OHLCV features excelling in volatile, rapidly changing conditions requiring quick directional adjustments, while extended features provide advantages in sustained trending environments where additional confirmation signals support longer-term position maintenance.

5.5. Discussion

This discussion examines the key findings from the experimental evaluation, with primary focus on TimesNet architecture performance given its role as the main contribution of this research. CNN results serve as comparative benchmarks where relevant, but the analytical emphasis remains on understanding TimesNet's capabilities and limitations for cryptocurrency trading applications.

5.5.1. TimesNet Architecture Effectiveness and Learning Capabilities

The experimental results demonstrate that the DDQN framework with TimesNet architecture successfully learned sophisticated trading behaviors that extend far beyond simple buy-and-hold strategies. The agents developed adaptive positioning capabilities, strategically utilizing long, short, and neutral positions across varying market conditions, representing a significant advancement over traditional momentum-based approaches [1].

TimesNet's frequency domain analysis proved particularly effective in capturing the cyclical behaviors inherent in cryptocurrency markets. Unlike conventional CNN approaches, the multi-scale temporal pattern recognition through Inception blocks enabled identification of both short-term volatility patterns and longer-term cyclical trends. This capability directly contributed to the architecture's consistent outperformance across all experimental configurations [1]. Recent comparative studies confirm that temporal convolutional and transformer-based architectures significantly outperform classical models in volatile financial time series [7].

This architectural advantage is clearly demonstrated when compared to CNN benchmark results, which showed significantly lower performance across all metrics. CNN configurations achieved win rates of only 50-55% compared to TimesNet's 61-70%, while exhibiting more volatile training convergence and less sophisticated position management strategies. The superior performance validates TimesNet's temporal modeling approach over conventional convolutional architectures for cryptocurrency trading applications.

The position management analysis reveals sophisticated decision-making patterns that vary significantly between assets. Bitcoin configurations demonstrated balanced allocation strategies with slight long bias, while Ethereum showed more pronounced regime-dependent behavior. The stark contrast between Ethereum's OHLCV short bias (45.01% short vs 32.70% long) and extended features' strong long preference (74.78% long vs 12.68% short) indicates that feature complexity directly influences strategic positioning in different market environments.

5.5.2. Feature Engineering Impact

The systematic comparison between OHLCV and extended features reveals that additional features do not universally improve performance, with results varying across different market conditions and cryptocurrencies. This finding aligns with broader research on feature complexity in financial machine learning [8].

Our experiments show mixed results for the extended feature set. The addition of technical indicators and blockchain metrics sometimes enhanced performance but also introduced complexity that occasionally degraded trading effectiveness. Due to the black-box nature of deep reinforcement learning systems, the following analysis presents theoretical interpretations of these results based on the known characteristics of each feature type, though the precise internal mechanisms remain unobservable.

The technical indicators included in the extended feature set each contribute different types of market information. The MACD signal line, designed to identify momentum shifts through exponential moving average crossovers, may provide valuable trend confirmation during stable trending periods. However, momentum indicators can exhibit lag during rapid market transitions, potentially providing delayed signals when quick responses are critical.

The STOCHRSI, which measures momentum relative to recent price ranges, aims to identify overbought and oversold conditions. In traditional technical analysis, oscillators like STOCHRSI are known to provide false signals during strong trending markets where overbought/oversold levels may lose predictive power. However, how the neural network interprets and weighs these signals within the broader feature space remains unclear.

The Average True Range (ATR) measures market volatility but provides no directional information. While increased ATR values indicate market volatility, it is uncertain how the reinforcement learning agent incorporates this non-directional information into trading decisions, potentially creating ambiguity in the learning process.

The blockchain metrics introduce a novel dimension to cryptocurrency trading analysis. Average Transactions Per Block and Unique Addresses Used aim to capture network adoption patterns that may correlate with price movements. However, these metrics often exhibit reporting delays and may lag rather than lead price movements, limiting their utility for real-time trading decisions [14]. The extent to which these temporal delays affect the neural network's learning process cannot be directly observed.

Fees USD per Transaction reflects network congestion and user willingness to pay for transaction processing. While this can indicate network demand, fee dynamics may be influenced by technical factors unrelated to investment sentiment, particularly during periods of network upgrades or congestion spikes, potentially introducing noise into the feature space.

The mixed performance of extended features could result from several hypothetical factors. The increased dimensionality from 10 to 22 features expands the learning space, which may lead to overfitting with limited training data, though this cannot be definitively confirmed from our results alone. Additionally, combining multiple indicator types may introduce conflicting signals that complicate the decision-making process for the neural network, but the specific mechanisms remain opaque.

The results suggest that effective feature engineering requires careful consideration of indicator complementarity and market context. Different combinations of features may be optimal for different market conditions, highlighting the potential value of adaptive feature selection mechanisms [7]. However, understanding the precise feature interactions within deep reinforcement learning systems remains a challenge for future research.

5.5.3. Cross-Asset Analysis and Market Microstructure Insights

The contrasting behaviors observed between Bitcoin and Ethereum provide valuable insights into asset-specific market microstructures that extend beyond simple volatility differences. Bitcoin's more balanced position allocation and superior absolute returns suggest fundamentally different market dynamics compared to Ethereum's more trend-sensitive characteristics [7].

The trading frequency analysis reveals particularly interesting behavioral differences. Bitcoin OHLCV generated 136 position changes compared to 117 for extended features, indicating that basic price features encouraged more active trading for this asset. Ethereum demonstrated the opposite pattern, with OHLCV producing 118 changes versus 74 for extended features, suggesting that enhanced features enabled more strategic, sustained position holding. These differences likely reflect varying network characteristics, with Ethereum's richer on-chain ecosystem providing more informative blockchain metrics that support longer-term decision-making [8].

The superior win rates achieved with extended features for Ethereum (70.27% vs 61.02%) compared to the more modest improvement for Bitcoin (65.81% vs 67.65%) further supports the hypothesis that different cryptocurrencies benefit from different feature engineering approaches based on their underlying network characteristics and market structures [7].

5.5.4. Methodological Limitations and Critical Assessment

Several important limitations must be acknowledged when interpreting these results, particularly regarding their generalizability and practical implementation viability. The experimental evaluation, while comprehensive in temporal scope, represents specific market conditions that coincided with significant structural changes in cryptocurrency markets, including increased institutional adoption and regulatory developments [9].

The architectural comparison also revealed important insights about model complexity and convergence stability. CNN configurations demonstrated more volatile training patterns and difficulty achieving stable convergence, suggesting that conventional convolutional approaches may be inherently limited for complex temporal pattern recognition in financial time series. This finding supports the methodological choice of TimesNet as the primary architecture while highlighting the importance of architecture selection in reinforcement learning applications.

The manual hyperparameter configuration necessitated by computational constraints represents a significant methodological limitation. Unlike systematic hyperparameter optimization approaches, the current configuration relied on established reinforcement learning practices rather than exhaustive search. This pragmatic approach, while reasonable for proof-of-concept validation, may have prevented discovery of more optimal parameter combinations that could enhance performance across different market regimes [3].

Most critically, transaction costs, market impact, and slippage were not explicitly modeled in the simulation environment. Real-world implementation would face substantial additional challenges including bid-ask spreads, liquidity constraints, and execution delays that could significantly erode the reported performance advantages. The active trading frequencies observed (74-136 position changes over 3.78 years) would incur substantial transaction costs in professional trading environments, potentially reducing the practical viability of these approaches [7].

The study's focus on two major cryptocurrencies limits generalizability to the broader digital asset ecosystem. Smaller, less liquid cryptocurrencies with different market dynamics may not exhibit similar responses to the implemented strategies. Additionally, the regulatory environment and institutional adoption patterns evolved significantly during the evaluation period, potentially influencing market behavior in ways not captured by the historical training data [9].

5.5.5. Risk Management and Practical Implementation Considerations

The maximum drawdown analysis reveals important insights for practical risk management implementation. Bitcoin configurations achieved superior drawdown control (31.85% for OHLCV) compared to Ethereum variants (63.83-66.96%), indicating that risk management strategies must be asset-specific rather than universally applied across different cryptocurrencies [7].

The comparative risk-return profiles between architectures reinforce these asset-specific considerations. CNN strategies consistently exhibited higher maximum drawdowns (52-77% for Bitcoin, 63-77% for Ethereum) and lower Sharpe ratios compared to TimesNet configurations, indicating that architectural choice significantly impacts risk management effectiveness. This suggests that advanced temporal modeling architectures like TimesNet provide inherent risk management advantages through superior market timing capabilities.

The relatively active trading strategies implied by the position change frequencies would face significant practical challenges in real-world deployment. Professional trading environments typically encounter transaction costs of 0.1-0.5% per trade, which could substantially erode the reported performance advantages. Furthermore, the market impact of executing larger position sizes could dramatically affect achievable returns, particularly during periods of reduced market liquidity [8].

Additional practical considerations include the computational requirements for real-time strategy execution. TimesNet's superior performance comes at the cost of increased computational complexity, requiring more powerful hardware infrastructure compared to simpler CNN approaches. The scalability of these strategies across multiple cryptocurrencies simultaneously also presents resource allocation challenges for practical deployment.

6. Future Work

The findings of this research open several promising avenues for continued investigation and practical development across multiple dimensions of reinforcement learning and financial applications.

6.1. Adaptive Feature Selection Systems

The most immediate research priority involves developing dynamic feature selection mechanisms that can automatically switch between OHLCV-focused strategies during volatile periods and extended feature strategies during trending phases. Such systems could potentially combine the crash detection capabilities of basic features with the trend-following advantages of complex features through real-time market regime classification. Machine learning approaches for regime detection, including hidden Markov models or RNNs [20, 21], could provide the foundation for such adaptive systems.

6.2. Advanced Reinforcement Learning Algorithms

The current DDQN framework, while effective, represents only one approach within the broader reinforcement learning landscape. Future research should systematically evaluate alternative algorithms including PPO [22] for improved stability in volatile environments, SAC [23] for better exploration in high-dimensional action spaces, and A2C [24] for enhanced policy optimization. Each algorithm offers specific advantages that could address the current limitations in rapid market adaptation and policy stability during extreme market conditions.

6.3. Multi-Timeframe and Hierarchical Approaches

The development of hierarchical reinforcement learning approaches [25, 26] could enable simultaneous strategy optimization across multiple temporal horizons. Such frameworks could make strategic decisions on daily timeframes while executing tactical adjustments on hourly or intraday bases, potentially capturing both long-term trends and short-term opportunities more effectively than the current singletimeframe approach. This could address the limitation of fixed daily trading frequency observed in the current study.

6.4. Enhanced Data Integration

The promising results with blockchain metrics for Ethereum suggest substantial potential for incorporating additional data sources. Real-time sentiment analysis from social media platforms [2, 27], regulatory announcement impacts, macroeconomic indicator integration, and network-specific metrics for other cryptocurrencies could provide richer information contexts for decision-making. The key challenge lies in ensuring these additional data sources maintain the real-time characteristics necessary for actionable trading signals without introducing the decision lag observed with complex feature sets during volatile periods.

6.5. Portfolio-Level Optimization

Extending the current single-asset focus to multi-asset portfolio management could achieve better risk-adjusted returns through diversification while maintaining the sophisticated timing capabilities demonstrated here. Such approaches would need to consider correlation structures, cross-asset arbitrage opportunities, and portfolio-level risk management constraints [28, 29]. This extension could potentially reduce the maximum drawdown values observed in single-asset strategies while maintaining overall profitability.

6.6. Real-World Implementation Framework

The development of comprehensive implementation frameworks addressing transaction costs, market impact modeling [30, 31], regulatory compliance, and execution infrastructure represents a critical bridge between research validation and practical deployment. This includes latency optimization, order execution algorithms, and risk management systems capable of operating in live market environments. Given the active trading frequencies observed (74-136 position changes over 3.78 years), careful consideration of transaction costs and market impact becomes essential for practical viability.

6.7. Robustness and Stress Testing

Future research should systematically evaluate strategy performance across extended bear markets, low-liquidity environments, and extreme volatility scenarios not well-represented in the current evaluation period. Additionally, investigating the strategies' behavior during market structural changes, such as institutional adoption phases or regulatory shifts, would provide crucial insights for long-term deployment viability. This includes testing the approaches across smaller, less liquid cryptocurrencies with different market dynamics than Bitcoin and Ethereum.

These research directions collectively point toward more robust, adaptive, and practically implementable cryptocurrency trading systems that build upon the temporal pattern recognition and reinforcement learning foundations established in this work while addressing the critical limitations and challenges identified through this comprehensive evaluation.

Bibliography

[1] Wu, H., Hu, T., Liu, Y., Zhou, H., Wang, J., & Long, M. (2023). *TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis.* In International Conference on Learning Representations (ICLR).

[2] Bollen, J., Mao, H., & Zeng, X. (2011). *Twitter Mood Predicts the Stock Market.* Journal of Computational Science, 2(1), 1-8.

[3] Sutton, R.S. and Barto, A.G., *Reinforcement Learning: An Introduction*, MIT Press, 2018.

[4] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Hassabis, D. (2015). *Human-level control through deep reinforcement learning.* Nature, 518(7540), 529-533.

[5] van Hasselt, H., Guez, A., and Silver, D., *Deep reinforcement learning with double Q-learning*, AAAI, 2016.

[6] Zhou, C., Huang, Y., Cui, K., & Lu, X. (2024). *R-DDQN: Optimizing Algorithmic Trading Strategies Using a Reward Network in a Double DQN.* Mathematics, 12(11), 1621.

[7] Li, X., et al. (2023). *Deep Reinforcement Learning for Cryptocurrency Trading: A Review.* Expert Systems with Applications, 213, 119069.

[8] Chen, T., et al. (2022). *Feature Engineering for Cryptocurrency Price Prediction: A Survey.* IEEE Access, 10, 123456-123470.

[9] Dulac-Arnold, G., et al., *Challenges of real-world reinforcement learning...*, Machine Learning, 2021.

[10] Park, C. H., & Irwin, S. H. (2007). What do we know about the profitability of technical analysis? *Journal of Economic Surveys*, 21(4), 786-826.

[11] Neely, C., Weller, P., & Dittmar, R. (1997). Is technical analysis in the foreign exchange market profitable? A genetic programming approach. *Journal of Financial and Quantitative Analysis*, 32(4), 405-426.

[12] Parkinson, M. (1980). The extreme value method for estimating the variance of the rate of return. *Journal of Business*, 53(1), 61-65.

[13] Hayes, A. S. (2017). Cryptocurrency value formation: An empirical study leading to a cost of production model for valuing bitcoin. *Telematics and Informatics*, 34(7), 1308-1321.

[14] Kristoufek, L. (2015). What are the main drivers of the Bitcoin price? Evidence from wavelet coherence analysis. *PLoS One*, 10(4), e0123923.

[15] Easley, D., O'Hara, M., & Basu, S. (2019). From mining to markets: The evolution of bitcoin transaction fees. *Journal of Financial Economics*, 134(1), 91-109.

[16] Zhang, X., Yu, F., Zou, S., & Zhao, H. (2019). Economic policy uncertainty, investor sentiment and financial stability—an empirical study based on the time series network. *Physica A: Statistical Mechanics and its Applications*, 521, 645-665.

[17] Yahoo Finance, *Historical Cryptocurrency Data*, Accessed: 2024-09-10, `https://finance.yahoo.com/cryptocurrencies`

[18] Blockchain.com, *Bitcoin Block Explorer Charts*, Accessed: 2024-09-11, `https://www.blockchain.com/de/explorer/charts`

[19] Etherscan, *Ethereum Blockchain Data Charts*, Accessed: 2024-09-11, `https://etherscan.io/charts#section-blockchain-data`

[20] Hamilton, J. D. (1994). *Time Series Analysis.* Princeton University Press.

[21] Baum, L. E., & Petrie, T. (1966). *Statistical Inference for Probabilistic Functions of Finite State Markov Chains.* The Annals of Mathematical Statistics, 37(6), 1554-1563.

[22] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). *Proximal Policy Optimization Algorithms.* arXiv preprint arXiv:1707.06347.

[23] Haarnoja, T., Zhou, A., Abbeel, P., & Levine, S. (2018). *Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor.* In International Conference on Machine Learning.

[24] Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., ... & Kavukcuoglu, K. (2016). *Asynchronous Methods for Deep Reinforcement Learning.* In International Conference on Machine Learning.

[25] Barto, A. G., & Mahadevan, S. (2003). *Recent Advances in Hierarchical Reinforcement Learning.* Discrete Event Dynamic Systems, 13(1-2), 41-77.

[26] Sutton, R. S., Precup, D., & Singh, S. (1999). *Between MDPs and Semi-MDPs: A Framework for Temporal Abstraction in Reinforcement Learning.* Artificial Intelligence, 112(1-2), 181-211.

[27] Nguyen, T. H., Shirai, K., & Velcin, J. (2015). *Sentiment Analysis on Social Media for Stock Movement Prediction.* Expert Systems with Applications, 42(24), 9603-9611.

[28] Markowitz, H. (1952). *Portfolio Selection.* The Journal of Finance, 7(1), 77-91.

[29] Black, F., & Scholes, M. (1973). *The Pricing of Options and Corporate Liabilities.* Journal of Political Economy, 81(3), 637-654.

[30] Almgren, R., & Chriss, N. (2001). *Optimal Execution of Portfolio Transactions.* The Journal of Risk, 3, 5-40.

[31] Bertsimas, D., & Lo, A. W. (1998). *Optimal Control of Execution Costs.* Journal of Financial Markets, 1(1), 1-50.

A. Appendix

A.1. Artificial Intelligence Declaration

The authors acknowledge that parts of this work were supported by AI tools, including OpenAI's ChatGPT and Anthropic's Claude. These tools were used to assist in refining text, generating rephrasing suggestions, drafting sections, identifying bugs in code, and structuring the coding workflow.