2. MDP design

A) State Space (S) - 15D:

The state is the set of observations the agent uses at each timestep to make a decision. A comprehensive state space is crucial for effective performance. The following state space is proposed, with all components normalized to ensure numerical stability and improve the learning process. In this methodology, we chose to use a 1 minute time interval, following our core paper, **Yang and Malik [3].**

Core Features

- **Position**: The agent's current portfolio exposure, normalized to a range of [−1,1]. The agent's current position or holding status is an essential state component in DRL trading systems, providing critical context for subsequent actions, especially when considering transaction costs.
- **Z-score**: A standardized mean-reversion signal calculated over a 120-minute rolling window from a 60-minute baseline moving average.The Z-score is a key indicator in pair trading strategies, serving as a critical observation that enables the RL agent to make better-informed decisions based on statistical deviations from a mean.
- $$z_t = \frac{spread_t - \mu_{t,120}}{\sigma_{t,120}}.$$
- **Normalized Zone (zonenorm)**: A discrete trading signal derived from the Z-score and encoded as a continuous value in {−1,−0.5,0,0.5,1}. This feature provides the agent with a clear, strategy-aligned context. Normalizing features derived from primary indicators like the Z-score is a standard and vital practice in DRL to improve model stability and performance.
- **Price Momentum (pmt)**: The last-minute price return, clipped for numerical stability. Momentum is a widely used technical concept in DRL applications that provides information on the rate of price change. Its inclusion is explicitly supported as a key feature in financial forecasting models.
- **Z-score Momentum (zmomt)**: The one-minute change in the Z-score, capturing the velocity of the spread's convergence or divergence. This feature is a logical composite of two individually justified components: Z-score and Momentum. Feature engineering that combines existing signals to enrich the state representation is a common practice for improving model performance.
- **Position Change (Δpost)**: The change in portfolio exposure from the previous minute, representing the agent's last action. This feature is inherently linked to the agent's actions and provides valuable context about its previous decisions and their immediate impact. DRL agents that select an optimal fraction of an asset to trade directly influence this value, making it a key feedback signal.

Technical Indicators (Normalized)

Technical indicators are critical inputs for detecting trends and momentum, and their normalization is a standard preprocessing step.

- **Moving Average Convergence Divergence (MACD)**: Implemented as three separate normalized inputs: MACD_norm, MACD_signal_norm, and MACD_diff_norm. MACD is one of the most frequently cited momentum indicators in DRL studies, used to measure price fluctuations and signal trading opportunities. The MACD line, its signal line, and the histogram (difference) are all collected in DRL studies, with the histogram noted to positively influence agent decisions.
- **Relative Strength Index (RSI)**: A normalized momentum indicator measuring the speed and change of price movements. RSI is a widely used oscillator that helps the agent understand if an asset is overvalued or undervalued. It is a standard component in feature vectors for DRL trading agents.
- **Bollinger Bands**: Implemented as three distinct normalized features: BB_mid_norm, BB_high_norm, and BB_low_norm. Bollinger Bands are recognized technical indicators in DRL applications that provide a multi-faceted view of the market's trend (middle band) and volatility (upper and lower bands).
- **On-Balance Volume (OBV)**: A normalized, cumulative indicator that measures buying and selling pressure. OBV is included as part of rich state representations to capture volume patterns. It uses volume flow to gauge the conviction behind a trend, helping to confirm price moves or identify potential reversals.

Sentiment Data (Normalized)

- **Social Media Sentiment Score (from Reddit)**: A score representing the overall sentiment for the target asset. Sentiment data is strongly justified for inclusion, as it can significantly improve the performance of RL trading systems. Sentiment from Reddit is explicitly supported, with future work suggesting its inclusion and other studies incorporating it as a weighted component (15%) of a "Fear and Greed Index" used for DRL agent inputs.

The deliberate omission of raw price data from the state space is a methodological choice designed to enhance the DRL agent's learning stability and provide more decision-relevant features. Raw financial price series are inherently non-stationary, a statistical property that can impede the development of a robust trading policy. To mitigate this, the state space is engineered with more stationary, derived features, such as price momentum (returns), the standardized Z-score, and bounded oscillators like the RSI. These features transform the absolute price into contextual, relational information. Instead of a raw price level, the agent perceives the market's state through signals indicating if the price is statistically overextended (Z-score), trending with momentum (MACD), or operating within a specific volatility regime (Bollinger Bands). These indicators also act as engineered filters, improving the low signal-to-noise ratio common in financial data and presenting a clearer representation of market dynamics to simplify the agent's learning task.

B) Action Space (A)

A continuous action space in the range [-1, 1] is appropriate for TD3/A2C and for this trading task. The action can be interpreted as follows:

- -1: Take a full short position.
- 0: Hold a neutral position (no holdings).
- 1: Take a full long position.

We will set a maximum position shift per minute as 0.1 to stop the agent from suddenly moving positions too quickly and encourage split TP and SL.

The values in between represent fractional positions (e.g., 0.5 for a half-sized long position). This allows for nuanced trading decisions. This action space design is inspired by **Yang and Malik [3].**

C) Reward Function (R)

The reward function is designed as a comprehensive, hybrid signal that integrates multiple financial objectives to guide the agent toward a robust and risk-aware trading policy. The final reward at each timestep:

Overall Reward Function

$$R_{\text{final},t} = w_{\text{PnL}} \cdot R_t^{\text{PnL}} + w_{\text{risk}} \cdot R_t^{\text{risk\_adj}} - w_{\text{trans}} \cdot P_t^{\text{trans}} - w_{\text{drawdown}} \cdot P_t^{\text{drawdown}} + w_{\text{hold}} \cdot R_t^{\text{hold}}$$

Primary Reward: Profit-and-Loss: The core of the reward function is the realized profit and loss, calculated at each timestep to provide the agent with dense feedback on its performance. This component directly incentivizes the agent to maximize portfolio wealth. Defining the reward as the change in portfolio value at each step is a standard and widely accepted approach in financial RL. The reward signal is based on the difference in the gross value of the account before and after the agent takes an action.

$$R_t^{\text{PnL}} = (\text{Gross\_Portfolio\_Value}_t - \text{Gross\_Portfolio\_Value}_{t-1}) \times \nu$$
$$\text{Gross\_Portfolio\_Value}_t = \text{Asset\_Value}_t + \text{Cash\_Balance}_t$$

Here, v is a normalization scalar.

Risk-Adjusted Return (Differential Sharpe Ratio): A reward based on the Differential Sharpe Ratio (DSR) is included to encourage the agent to optimize for risk-adjusted returns incrementally. The Sharpe Ratio is a common focus for reward functions, and its differential form (DSR) has been found to yield consistent results and is well-suited for efficient online optimization. This reward aligns the agent's optimization with risk-adjusted returns, effectively penalizing volatility and encouraging more stable reward profiles.

$$DSR_t = \frac{\beta_{t-1}\Delta\alpha_t - \frac{1}{2}\alpha_{t-1}\Delta\beta_t}{(\beta_{t-1} - \alpha_{t-1}^2)^{3/2}}$$
$$\Delta\alpha_t = R_t^{\text{PnL\_unnormalized}} - \alpha_{t-1}$$
$$\Delta\beta_t = (R_t^{\text{PnL\_unnormalized}})^2 - \beta_{t-1}$$
$$\alpha_t = \alpha_{t-1} + \lambda \cdot \Delta\alpha_t$$
$$\beta_t = \beta_{t-1} + \lambda \cdot \Delta\beta_t$$

Transaction Costs Penalty: A penalty proportional to the size of each trade is subtracted from the reward to account for commissions and slippage. This is a crucial component for realism, as many strategies perform poorly once costs are considered. This penalty is effective at discouraging over-trading and frequent changes in position.

$$P_t^{\text{trans}} = C^{\text{trans}} \times |\text{Position\_Size}_t - \text{Position\_Size}_{t-1}|$$

Drawdown Penalty: A large negative reward is triggered if the portfolio's current drawdown exceeds a predefined risk threshold. RL's flexibility allows for complex reward schemes that can "send a large negative reinforcement signal, in case a certain drawdown is exceeded," directly training the agent to avoid catastrophic

losses. This encourages the agent to learn strategies that avoid breaching the drawdown threshold, effectively biasing it toward behaviors similar to applying a stop-loss.

$$P_t^{\text{drawdown}} = \begin{cases} C^{\text{drawdown\_penalty}} & \text{if Current\_Drawdown}_t \geq \text{Threshold}_{\text{Drawdown}} \\ 0 & \text{otherwise} \end{cases}$$

Holding Reward/Penalty: A small penalty is applied for prolonged inactivity or for holding a position for an excessive duration. This penalty is explicitly supported as a mechanism to encourage active trading rather than remaining static.

$$R_t^{\text{hold}} = \begin{cases} C^{\text{hold\_profit\_bonus}} & \text{if position is profitable and held} \\ C^{\text{hold\_loss\_penalty}} & \text{if position is losing and held} \\ C^{\text{idle\_penalty}} & \text{if agent is inactive} \\ C^{\text{action\_bonus}} & \text{if a Buy/Sell action occurs} \end{cases}$$

We use automated scripts to generate combinations of hyperparameters and use gridsearch to find the best combination on the validation set.

3. Data Splitting and Training Protocol

This protocol is designed to ensure a robust evaluation of the DRL agent, mitigate backtest overfitting, and promote generalization to unseen, real-time market data by incorporating best practices for handling non-stationary financial time series.

Overall Data Splitting Strategy: Strict Chronological Order

The 3.7-year Ethereum dataset will be partitioned into three distinct, non-overlapping sets in strict chronological order to prevent any form of data leakage from the future into the past.

- **Training Set**: The first **70%** of the data (~2.6 years). This set is used for the initial training and subsequent retraining in the rolling window.
- **Validation Set**: The next **15%** of the data (~7 months). This set is used exclusively for hyperparameter tuning and model selection via purged cross-validation.
- **Test Set**: The final **15%** of the data (~7 months). This is a hold-out set, used only once for the final evaluation of the trained model's performance on completely unseen data.

Model Adaptation Strategy: Rolling Window Training

To address the non-stationary nature of financial markets, a single training on a fixed dataset is insufficient. A rolling window approach will be implemented to ensure the model continuously adapts to recent market conditions.

The model will be trained on a rolling window of the preceding **6-12 months** of data from the training set. It will then be evaluated in the subsequent **1-month** period. This process will be repeated, rolling forward one month at a time, to simulate a realistic deployment scenario.

Overfitting Mitigation: Combinatorial Purged Cross-Validation (CPCV)

During the hyperparameter tuning phase on the validation set, standard cross-validation will be avoided. Instead, Combinatorial Purged Cross-Validation (CPCV) will be used to select the most robust hyperparameters.

The validation set will be divided into N equal-sized segments. Multiple unique training-validation splits will be generated by combining these segments. Crucially, **purging** and **embargoing** techniques will be applied to remove overlapping data points and prevent information leakage between the splits. This method provides a more reliable estimate of a model's performance across various market conditions, reducing the risk of overfitting to a specific historical path and selecting hyperparameters that generalize better.

DRL Episode and State Design

Training and Evaluation Episode Length: Each training and evaluation episode will consist of, which corresponds to **28 days (40320 steps)** of minute data. This length is sufficient to capture meaningful monthly market dynamics while being short enough to manage non-stationarity and computational constraints.

State Representation: At each step within an episode, the agent's input state will be constructed from a lookback window of the **past 2 hours (120 steps)** of market data (features), providing the necessary temporal context for its decisions. This is the minimum amount needed for our z-score calculation.