Université de Lausanne
Hautes Études Commerciales (HEC)

Study programme: Master in Finance

Comparative Analysis of Portfolio Optimization Techniques: Input-based vs Direct Weight Methods

Author: Marko Blanusa
Lausanne, June 2025

Company / Department: Faculté des Hautes Études Commerciales (HEC Lausanne), Département de Finance, Université de Lausanne
Responsible person at company: Prof. Marianne Schmid Mast, Doyenne de HEC Lausanne

Supervising professor: Marc-Aurèle Divernois
Examining expert: Ayalasomayajula Madhushree

Master's thesis

Abstract / Résumé

Abstract (English). This thesis benchmarks four portfolio-construction pipelines on a diversified basket of ten highly liquid cryptocurrencies over the 01 Jan 2020 – 20 May 2025 period. Two pipelines follow the traditional "forecast-then-optimize" paradigm: (i) HAR-XGBoost volatility forecasts coupled with an ADCC dynamic-correlation model, and (ii) a supervised LSTM that predicts the Cholesky factors of the next-day realized covariance. The remaining two pipelines skip input estimation and learn allocations directly: (iii) an LSTM that regresses historical market states onto ex-post optimal weights, and (iv) a Proximal Policy Optimization agent trained end-to-end in a realistic futures-trading simulator. All models ingest the same staged feature set that layers price, technical, macro-economic and social-sentiment signals. The study clarifies when each paradigm excels and releases fully reproducible code and hyper-parameters for practitioners.

Résumé (français). Ce travail compare quatre chaînes de construction de portefeuilles sur un panier diversifié de dix cryptomonnaies liquides entre janvier 2020 et mai 2025. Deux approches suivent le paradigme classique "prévoir puis optimiser" : (i) des prévisions de volatilité HAR-XGBoost couplées à un modèle ADCC de corrélations dynamiques, et (ii) un LSTM supervisé qui anticipe les facteurs de Cholesky de la covariance réalisée du lendemain. Les deux autres contournent l'estimation des intrants et apprennent les pondérations directement : (iii) un LSTM régressant l'état du marché sur les poids optimaux ex post et (iv) un agent PPO entraîné de bout en bout dans un simulateur de trading réaliste. Tous les modèles partagent le même jeu de caractéristiques combinant signaux de prix, indicateurs techniques, facteurs macroéconomiques et sentiment social. L'étude précise les contextes dans lesquels chaque paradigme se révèle supérieur et fournit un code entièrement réplicable.

Keywords: portfolio optimization; cryptocurrencies; volatility forecasting; covariance estimation; machine learning; XGBoost; LSTM; reinforcement learning; ADCC; PPO
JEL Classification: G11; C45; G15

Chapter 1

Introduction

In today's financial markets, the rapid evolution of asset dynamics and the inherent uncertainty of market behavior have spurred the development of increasingly sophisticated techniques for forecasting volatility and optimizing portfolio allocations. The traditional paradigm in portfolio management—embodied by Markowitz's mean–variance framework—relies on a two-step process of estimating input parameters (means and covariances) and then optimizing portfolio weights accordingly. However, this "estimate-then-optimize" approach suffers from several limitations, including estimation errors, extreme sensitivity to input parameters, and poor out-of-sample performance.

This thesis addresses these challenges through a twofold strategy. First, it develops an enhanced framework for volatility forecasting and covariance estimation that couples the Heterogeneous AutoRegressive (HAR) features with modern machine-learning techniques—specifically, a gradient-boosted-tree (XGBoost) model whose forecasts feed an Asymmetric Dynamic Conditional Correlation (ADCC) estimator (We will name it **XGB-ADCC**) and a sequence-based Long Short-Term Memory (LSTM) network. Second, it investigates methods that predict portfolio weights directly: a supervised-learning LSTM that maps market features to weights, and a deep-reinforcement-learning approach based on Proximal Policy Optimization (PPO).

By comparing "input-first" and "direct-weight" routes under identical data and evaluation protocols, the study aims to clarify where each paradigm excels.

In the remainder of this chapter we review the literature in two main areas: (1) covariance forecasting for input-prediction methods, and (2) direct portfolio-weight prediction approaches. The review highlights gaps that motivate the empirical work reported in later chapters.

Chapter 2

Literature Review

2.1 Volatility Forecasting and Input Prediction Methods

2.1.1 The HAR Framework and Its Extensions

The seminal work of Corsi (2009) introduced the Heterogeneous AutoRegressive (HAR) model as a parsimonious framework for capturing the long memory and multi-scale nature of volatility. The HAR model decomposes realized volatility into components corresponding to different time horizons—daily, weekly, and monthly—thus accommodating the heterogeneous behavior of market participants. Despite its simplicity, the HAR model has been widely adopted as a benchmark in the literature for forecasting volatility, particularly in contexts where high-frequency data are available. Subsequent studies have sought to enhance the forecasting power of the HAR model by incorporating additional explanatory variables. For instance, the integration of macroeconomic indicators, lagged returns, and lagged squared returns has been shown to improve forecast accuracy by capturing further nonlinear dynamics in volatility evolution. Moreover, the combination of HAR with advanced machine learning techniques, such as XGBoost, has attracted considerable attention. In "Volatility Forecasting with Machine Learning and Intraday Commonality" (Zhang et al., 2023), the authors demonstrate that neural networks and tree-based models outperform classical linear regressions by uncovering complex interactions among predictors. This line of research suggests that ML techniques are particularly well suited to address the nonlinearity inherent in financial time series data.

2.1.2 Machine Learning Approaches: XGBoost, LSTM, and Beyond

A growing body of literature supports the integration of ML methods into volatility forecasting. Christensen et al. (2021) in their study "A Machine Learning Approach to Volatility Forecasting" illustrate that off-the-shelf implementations of regularized regression, tree-based algorithms, and neural networks can yield significant improvements in one-day-ahead volatility forecasts over the traditional HAR model. Their empirical findings, based on data from major indices such as the Dow Jones Industrial Average, indicate that ML models can extract incremental information from additional predictors, even when these predictors exhibit strong collinearity or nonlinear relationships. Another stream of research has focused on forecasting volatility in the cryptocurrency domain, where extreme volatility and market-specific idiosyncrasies challenge standard models. In "Crypto Volatility Forecasting: Mounting a HAR, Sentiment, and Machine Learning Horserace" (Brauneis and Sahiner, 2024), the authors extend the HAR framework by incorporating investor sentiment data derived from news and social media, and then compare its performance to several ML models. Their results indicate that although sentiment data do not significantly enhance the HAR model's performance when used in isolation, the integration of sentiment indicators within ML frameworks such as LightGBM, XGBoost, and LSTM models yields superior predictive accuracy in a nonlinear setting .

2.1.3 Covariance Matrix Estimation and Robustification Techniques

Beyond volatility forecasting, accurately estimating the covariance matrix of asset returns is critical for portfolio optimization. Traditional approaches often suffer from high estimation risk, leading to unstable portfolios. To mitigate these issues, techniques such as shrinkage estimators and eigenvalue filtering have been proposed. In practice, the covariance matrix estimated via models such as ADCC (Asymmetric Dynamic Conditional Correlation) – combined with univariate volatility forecasts from HAR-XGBoost models – is often "robustified" by shrinking it toward a historical covariance matrix and filtering out spurious eigenvalues. This process reduces the sensitivity of the optimization to estimation errors and should improves out-of-sample performance. These methods are detailed in various studies (e.g., Corsi's HAR-XGBoost paper and the ML approach studies) and have become a cornerstone in advanced portfolio management.

2.2 Direct Portfolio Weight Prediction Methods

2.2.1 The Rationale for Direct Weight Forecasting

While much of the existing literature on portfolio optimization focuses on the "estimate-then-optimize" paradigm, recent research has begun to explore methods that predict portfolio weights directly from historical data. This direct approach seeks to bypass the intermediate step of forecasting parameters (such as expected returns or covariances), which are notoriously difficult to estimate accurately. By predicting the optimal asset allocation directly, these methods aim to reduce the propagation of estimation errors and achieve superior out-of-sample performance. Recent studies have demonstrated that both supervised learning techniques and reinforcement learning algorithms can be employed for direct weight prediction. For instance, the work from (Jiang et al., 2024) present methodologies that utilize automated machine learning frameworks to directly predict optimal portfolio weights. These studies frame portfolio selection as a penalized regression or classification problem, where the loss function is designed to capture both return and risk characteristics of the portfolio. Their empirical results suggest that direct weight prediction methods can outperform traditional two-stage models, especially when the underlying data are noisy and exhibit structural changes.

2.2.2 Supervised Learning and Reinforcement Learning Approaches

In the supervised learning paradigm of direct weights, models such as LSTM are trained to map historical asset returns, market conditions, and macroeconomic indicators directly to portfolio weights. The idea is to learn an optimal weighting strategy from past data that generalizes well to new, unseen market conditions. For example, studies like (Jiang et al., 2024) illustrate how automated model selection and hyperparameter tuning can be integrated into the optimization process. These methods directly target the investment decision, thereby sidestepping the issues related to parameter estimation errors inherent in the two-step approach. LSTM models can be used for the mapping between historical features and historical weights.

On the other hand, unsupervised and reinforcement learning methods offer an alternative avenue for direct weight prediction. Deep Reinforcement Learning (DRL) techniques, such as Proximal Policy Optimization (PPO), have been applied to the portfolio allocation problem by formulating it as a sequential decision-making task. In these models, an agent learns a policy that maps the current state of the market to an optimal portfolio allocation, while taking into account realistic trading constraints such as transaction costs, slippage, and liquidation rules. A working paper titled "Machine Learning in Portfolio Decisions" explores the application of DRL to directly predict portfolio weights and demonstrates that such methods are capable of capturing complex, nonlinear dependen-

cies that traditional optimization methods often miss. The integration of both supervised and reinforcement learning approaches in the direct weight prediction framework not only improves forecasting accuracy but also provides robustness against market regime shifts and structural changes. By directly optimizing for portfolio performance metrics (such as the Sharpe ratio or downside risk measures), these models are inherently better equipped to handle the uncertainties and non-stationarities that characterize financial markets.

2.3 Synthesis and Research Contribution

2.3.1 Integration of Hybrid Forecasting and Direct Weight Prediction

The literature review highlights a clear trade-off: input-based pipelines preserve interpretability and leverage decades of econometric insight, whereas direct-weight approaches bypass estimation risk and can accommodate richer nonlinearities. Against this backdrop, the present thesis makes four concrete advances. First, it constructs two risk-forecasting engines—an XGBoost-ADCC pipeline and a deep LSTM architecture—both trained on an identical crypto–macro–sentiment feature tensor and designed to emit daily volatility *and* covariance projections. Second, it introduces two single-stage allocation learners: a supervised LSTM that regresses features directly on optimizer-derived target weights, and a Proximal Policy Optimization (PPO) agent that discovers its own policy under the same market frictions. Third, it subjects all four models to an out-of-sample evaluation on a common Binance test window, computing a uniform dashboard of metrics such as Sharpe, turnover and draw-down so that performance differences are attributable solely to modeling philosophy. Finally, the study offers a systematic diagnosis of when the classical "forecast–then-optimize" paradigm dominates and when direct prediction proves advantageous in the high-volatility, estimation-error-prone world of digital assets.

2.4 Expected Contributions and Implications

The investigation is expected to deliver several practical and academic benefits. It demonstrates that fusing HAR-style realized-measure features with gradient-boosted trees and deep LSTM encoders yields timelier and more accurate volatility and covariance forecasts for cryptocurrencies. By embedding constant-correlation shrinkage and eigen-filtering in the input-based pipelines, the thesis also shows how to curb estimation noise without inflating model complexity. A head-to-head back-test between the direct-weight learners (LSTM-Weights and PPO) and the two-step forecasters (XGBoost-ADCC and LSTM-Cov) clarifies the operating regimes in which each paradigm excels, thereby informing practitioner model selection. Because every strategy is evaluated net of realistic costs,

leverage caps and Binance margin rules—and because the code base with all hyper-parameters is fully documented—the findings are immediately replicable and can serve as a practitioner-ready blueprint for live deployment.

2.5 Conclusion

Traditional two-step portfolio construction struggles when input estimates are noisy—an issue amplified in crypto markets. Modern ML offers two distinct remedies: (i) better input forecasting and (ii) skipping inputs altogether by predicting weights. This thesis pits these philosophies against each other through four carefully designed models. The next chapters describe the data, model architectures, training protocols, and experimental results that underpin our comparative analysis.

Chapter 3

Methodology

3.1 Introduction and Objectives

This chapter benchmarks four cryptocurrency portfolio-construction pipelines. Two follow the traditional *estimate–then–optimize* paradigm: (i) **XGBoost–ADCC**, which forecasts univariate volatilities with gradient-boosted trees and converts the standardized residuals into daily covariances via an Asymmetric Dynamic Conditional Correlation filter, and (ii) **LSTM–Cov**, a supervised recurrent network that outputs the Cholesky factors of the next-day realized covariance, guaranteeing positive-definiteness. The remaining pipelines learn allocations directly: (iii) **LSTM–Weights**, mapping market states to the weights that would have satisfied the GMV, ERC and MDP objectives ex post, and (iv) a **Proximal Policy Optimization (PPO)** agent trained in a realistic exchange simulator to maximize net return penalized by EWMA volatility. Contrasting these models isolates the incremental value of improved risk-input forecasts versus end-to-end weight learning; performance is assessed out-of-sample on both prediction accuracy and portfolio metrics.

3.2 Asset Universe

The study tracks ten highly liquid, USD-denominated spot pairs on Binance. Bitcoin (BTC) and Litecoin (LTC) provide large-capitalization Proof-of-Work exposure, while Ethereum (ETH) and Cardano (ADA) represent leading smart-contract platforms. Binance Coin (BNB) captures centralized-exchange economics through revenue sharing and fee rebates. EOS (EOS) and Tron (TRX) stand for high-throughput Layer-1 networks, with TRX underpinning a large share of USDT traffic. Ripple (XRP) and Stellar (XLM) target institutional and retail cross-border payments respectively, and Dogecoin (DOGE) serves as a social-meme beta proxy. Together, these names balance large-cap resilience with higher-beta growth plays while ensuring continuous, high-quality intraday data.

Theoretically, focusing exclusively on cryptocurrencies provides several distinct methodological advantages. Cryptocurrency markets are notably less efficient and more sentiment-driven than traditional equity or fixed-income markets, characterized by extreme volatility, speculative trading behavior, and significant susceptibility to investor psychology and market sentiment. Such market conditions inherently present enhanced opportunities for arbitrage, alpha generation, and exploitation of market inefficiencies—precisely the types of opportunities sophisticated machine learning models are designed to identify and leverage. Moreover, limiting the analysis strictly to cryptocurrencies substantially simplifies interpretability, allowing clear attribution of portfolio performance variations specifically to methodological differences and modeling choices rather than confounding factors introduced by heterogeneous asset classes.

3.3 Data Sources

3.3.1 Market Data

One-minute OHLCV data for all ten pairs were pulled from the Binance public API. The sample begins *1 January 2020*[1] and ends *20 May 2025*, yielding over 2.3 million intraday observations per asset.

3.3.2 Engineered Features

Two variable groups augment the raw OHLCV series. *Technical indicators* comprises the *14-period Relative Strength Index (RSI)*, which scales the ratio of average up-moves to average down-moves onto a 0–100 oscillator, the *100-period Exponential Moving Average (*EMA_{100}*)* that assigns exponentially decaying weights to past closes, and the *Moving-Average Convergence Divergence (MACD)*, i.e. the difference between 12- and 26-period EMAs smoothed by a nine-period signal line. They collectively encode short-term momentum, medium-term trend and trend acceleration. *Targeted features* include daily *realized volatility*—the sum of squared one-minute returns—quantifies ex-post dispersion and later serves as both regressor and target in the forecasting pipelines, and, for the direct-weight models, the ex-post *optimal weights* generated under GMV, ERC and MDP rules are appended to the feature tensor so that networks learn to map market states to desirable allocations directly.

3.3.3 Macroeconomic Predictors

We enrich crypto-specific inputs with two baskets of exogenous variables from `Yahoo Finance`. The price-based basket (`PRICE_MACRO`) spans global equity, fixed-income, com-

[1] Earlier sentiment and social metrics are unavailable.

modity, real-estate and sector indices, capturing broad risk-on/risk-off rotations. The level-and-volatility basket (`LEVEL_MACRO`) tracks VIX, DXY, US Treasury yields, the MOVE index, TIPS breakevens, GVZ, OVX and the three-month Treasury bill rate, thereby proxying funding costs, dollar strength and cross-asset volatility. Week-day observations are forward-filled across weekends to match crypto's 24/7 cadence.

3.3.4 LunarCrush Sentiment & Social Metrics

LunarCrush is a real-time analytics platform that scrapes Twitter/X, Reddit, Medium, YouTube and mainstream news APIs to quantify the collective mood and engagement surrounding digital assets. For every coin the service delivers: *Sentiment*, a proprietary time-decayed ratio of bullish to bearish text snippets; the composite *Galaxy Score*, which blends price momentum, social velocity and developer activity into a 0–100 health gauge; *AltRank*, an ordinal ranking that compares a coin's market performance and social traction against the top-200 universe (lower is better); *Market Dominance*, the share of a coin's social volume versus all tracked coins; and a full activity panel—*contributors*, *posts* and *interactions*—that counts unique users, original messages and engagement events over rolling 24-hour windows. Raw API pulls are resampled to daily frequency and merged on close timestamps, injecting crowd psychology signals otherwise invisible to price, volume and macro factors.

3.3.5 Action-dependent Features – Model 4 only

During PPO training the agent observes a rich account state vector comprising unrealized and realized PnL, cumulative returns, current balance, position value and size, last log-return, maximum drawdown, entry-to-mark price distance, Sortino ratio, rolling 14-day return volatility, age of position, cash-to-equity ratio, sine and cosine encodings of day-of-week seasonality, a regime flag based on the slope of the 200-period BTC SMA and the percentage spread. These features are computed on the fly during PPO training and mirror useful account metrics a Binance user would see and enables the environment to be partially dependent of the policy's actions which is a key concept behind DRL foundations. Each variable is normalized online via Welford updates, clipped at five standard deviations and scaled to $[-1, 1]$ before being passed to the policy network.

3.4 Pre-processing Pipeline

3.4.1 Feature Staging

Model training proceeds through four nested information sets. *Stage 1* contains only price-based inputs—daily log-returns and log-volume—establishing a naïve baseline. *Stage*

2 augments this baseline with intraday realized volatility, its weekly and monthly moving averages and the trio of technical indicators (RSI_{14}, EMA_{100}, MACD). *Stage 3* enriches the design matrix further by appending the full suite of `PRICE_MACRO` and `LEVEL_MACRO` predictors, thus linking crypto returns to cross-asset risk rotations and funding conditions. Finally, *Stage 4* injects the LunarCrush sentiment and social variables, allowing the algorithms to exploit crowd dynamics. Comparing forecast accuracy across stages reveals the incremental value of each information block.

Special note for the XGB model : the five lags of log-returns and realized volatility with its weekly and monthly moving averages are included to try to capture some autocorrelation since it is the only non-sequenced model.

3.4.2 Normalization

All inputs are rescaled to the interval $[-1, 1]$ but the transformation depends on their statistical nature. Features with known theoretical bounds—RSI, sentiment scores and market-dominance ratios—are linearly mapped onto the target range. Unbounded, quasi-Gaussian variables such as log-returns or macro level series are first z-scored using training-set moments, then clipped at $\pm 8\,\sigma$ (=99.7 % coverage) and finally min–max scaled. Clipping the remaining 0.3% is harmless in term of information loss and preserves the dataset from rare noisy values and from squashing the other values too much around 0 when rescaling. Highly skewed or weakly stationary series (volumes, market caps, AltRank and activity counts) undergo a log transform before the same z-score $\rightarrow$ clip $\rightarrow$ scale pipeline. State variables generated online during PPO roll-outs—unrealised PnL, Sortino ratio, age of position and the like—have no predetermined distribution; they are normalised on-the-fly via Welford updates, clipped at $\pm 5\,\sigma$ and mapped to $[-1, 1]$.

The resulting homogeneous scale stabilizes neural-network training and accelerates tree convergence while retaining economic interpretability.

3.4.3 Realized Volatility and Covariance

Minute-level returns feed:

$$RV_{i,t} = \sum_{j=1}^{M} r_{i,t,j}^2, \tag{3.1}$$

$$\Sigma_t^{\text{realized}} = \sum_{j=1}^{M} r_{t,j} r_{t,j}^\top, \tag{3.2}$$

with $M \leq 1440$. Days with $M < 600$ are retained—empirically fewer than ten per asset—since coverage remains adequate. In instances where minute-level data was

partially missing, volatility was still able to be approximated with a high number of observations (minimum 600) and furthermore, less than 10 daily variance reconstruction were under the expected 1440 observations count.

To ensure numerical stability and mathematical validity in subsequent supervised prediction tasks, these covariance matrices were transformed using the Cholesky decomposition, which guarantees positive definiteness:

$$\Sigma_t^{\text{realized}} = L_t L_t^\top, \tag{3.3}$$

with L_t being a lower-triangular matrix targeted directly by the LSTM model during training.

Unlike traditional financial markets, cryptocurrencies operate continuously without conventional market closures such as weekends, holidays, or overnight gaps. This characteristic significantly simplifies certain aspects of data preprocessing, particularly eliminating the necessity for market-specific adjustments such as dividends, stock splits, or trading suspensions. However, cryptocurrency markets occasionally exhibit isolated instances of missing observations due to transient technical disruptions or API limitations. These sparse gaps were not causing major issues since the main time-frame is daily and therefore don't disrupt continuity once the dataset is resampled into daily intervals.

3.4.4 Data Splits

A 100-day "burn-in" separates contiguous train/validation/test windows:

Set	Start	End	Length (days)
Train	1 Jan 2020	30 Apr 2023	1216
Gap	1 May 2023	8 Aug 2023	100
Validation	9 Aug 2023	8 Feb 2024	184
Gap	9 Feb 2024	18 May 2024	100
Test	19 May 2024	20 May 2025	367

Chronological splits prevent look-ahead bias; the burn-in ensures stateful models (ADCC, LSTMs, PPO) warm up before evaluation.

3.4.5 Input Window Length

Sequence models consume a fixed look-back of $L = 100$ trading days (roughly 3 months) which is why a gap of 100 days is applied between sets in order to avoid leakage. Trees and ADCC will also use the same gap for fairness in comparison and also for leakage since there is the EMA_{100} feature using a window of 100 days.

3.5 Workflow Summary

First, all raw and engineered features are assembled into a daily panel following the four-stage hierarchy and rescaled as described above. Next, each modeling family is trained with its customary optimization routine: tree-based pipelines rely on time-series cross-validation with early-stopping hyper-parameter search, the volatility-LSTM is fitted with Adam and dropout 0.3, and the PPO policy learns through batched roll-outs in a vectorized Gym environment. For the input-based pipelines, the resulting risk estimates feed GMP, ERC or MDP optimizers, whereas the LSTM-Weights and PPO agents produce allocations directly. Out-of-sample evaluation first occurs on the validation window to determine early-stopping points and hyper-parameters; the final locked-down test period then measures true generalization performance.

3.6 Input-based: Forecasting Covariance for Portfolio Optimization

Input-based method encompasses two distinct methodologies for forecasting asset covariance matrices, which form the basis of subsequent portfolio optimization:

Two alternative pipelines generate daily positive-definite covariance forecasts. *Method 1* couples XGBoost volatility predictions with an ADCC correlation filter; *Method 2* feeds an LSTM that outputs the Cholesky factor of the realized covariance to guarantee positive-definiteness.

Both methodologies provide alternative perspectives and modeling advantages, allowing an empirical comparison of traditional econometric versus advanced deep learning approaches.

3.6.1 Method 1: Two-Step Forecasting using XGBoost and ADCC

Method 1 is a structured, two-step procedure designed to robustly forecast the covariance structure of asset returns, crucial for precise portfolio optimization. Initially, we forecast asset-specific volatilities employing extended Heterogeneous Autoregressive (HAR) features (Corsi, 2009), augmented by Extreme Gradient Boosting (XGBoost) (Chen and Guestrin, 2016) to capture nonlinear dynamics. Subsequently, dynamic correlations among asset returns are modeled using the Asymmetric Dynamic Conditional Correlation (ADCC) model (Cappiello et al., 2006), explicitly accounting for asymmetric market conditions. The resulting covariance matrix is robustified through shrinkage and eigenvalue filtering before feeding into various portfolio optimization frameworks.

Conceptual backgrounds for XGBoost and ADCC asymmetry have been moved to Appendix I (§I.1–I.2); below we document only the Binance-crypto feature set, hyper-

parameter grid and post-processing steps.

Volatility Forecasting using XGBoost

Accurate volatility predictions underpin effective risk management and portfolio allocation. The classical linear HAR model by Corsi (2009) captures long memory and multi-scale volatility features using lagged realized volatility at different frequencies (daily, weekly, monthly):

$$RV_{t+1} = \beta_0 + \beta_D\, RV_t^{(D)} + \beta_W\, RV_t^{(W)} + \beta_M\, RV_t^{(M)} + \varepsilon_{t+1}, \tag{3.4}$$

where lagged realized volatility terms are defined as:

$$RV_t^{(D)} = RV_t, \quad RV_t^{(W)} = \frac{1}{7}\sum_{j=0}^{4} RV_{t-j}, \quad RV_t^{(M)} = \frac{1}{28}\sum_{j=0}^{21} RV_{t-j}.$$

Although effective, linear HAR lacks the flexibility to represent nonlinear patterns or interaction effects among predictors. Therefore, we enhance HAR by employing the machine learning technique XGBoost, which builds an ensemble of regression trees iteratively via gradient boosting.

We feed the HAR features (recent daily, weekly, monthly volatilities) into XGBoost, along with other features described in the various stages (HAR features starts from stage 2). The model is trained to predict the next-day realized volatility. The theoretical derivation of the XGBoost model is located to Appendix I.1.

Hyperparameter tuning. We employ randomized search with time-series cross-validation within the training set only with ($k = 5$) and 200 iterations over these parameter distributions:

Hyperparameter	**Distribution**
$n_{\text{estimators}}$	UniformInt$(300, 1500)$
max _depth	UniformInt$(2, 15)$
η	LogUniform$(0.005, 0.2)$
γ	LogUniform$(10^{-4}, 1)$
min _child_weight	LogUniform$(1, 15)$
subsample, colsample_by*	Uniform$(0.4, 0.6)$
α	LogUniform$(10^{-5}, 0.1)$
λ	LogUniform$(0.1, 5)$

Early stopping (50 rounds) prevents overfitting, and GPU acceleration is optionally enabled. This approach aligns with best practices in time-series forecasting and signifi-

cantly enhances model performance —

Summary. Each asset obtains a day-ahead volatility forecast:

$$\hat{\sigma}^2_{i,t+1} = f_i(\text{Stage features}),$$

where feature sets and hyperparameters are calibrated per asset. These volatility forecasts are used to retrieve residuals that feed into the ADCC model to construct daily correlation matrices and ultimately covariance matrices for portfolio optimization (Method 1).

ADCC for Dynamic Correlation Estimation

While XGBoost forecasts individual volatilities, we need a model for time-varying correlations between asset returns to fully characterize the joint distribution of returns. We adopt the Asymmetric Dynamic Conditional Correlation (ADCC) model of Cappiello et al. (2006), which extends Engle's Dynamic Conditional Correlation (DCC) model (Engle, 2002) to allow for asymmetry in correlations during market downturns. The ADCC model is initially a multivariate GARCH approach: each asset's return $r_{i,t}$ is assumed to follow a univariate GARCH process for its variance, and the correlations between the standardized residuals are modeled as dynamically evolving over time.

The difference in our approach is that instead of using a GARCH model to model univariate forecasts of variances, we will use the XGBoost's model standardized residuals introduced earlier.

The ADCC model adds an asymmetric term to capture the empirical fact that correlations often increase after joint negative returns (e.g., markets becoming more correlated in a crisis). In ADCC, an additional term $g\,(n_{t-1}n_{t-1}^\top)$ is added to the usual DCC equation, where n_{t-1} is a vector of indicators for negative returns (e.g., $n_{i,t-1} = 1$ if $\epsilon_{i,t-1} < 0$, else 0). The modified equation is:

$$Q_t = (1 - a - b - g)\,\bar{Q} + a\,(z_{t-1}z_{t-1}^\top) + b\,Q_{t-1} + g\,(n_{t-1}n_{t-1}^\top), \tag{3.5}$$

as introduced by Cappiello et al. (2006). The g term allows recent negative shocks (where one or more $n_{i,t-1} = 1$) to have an extra impact on increasing correlations, reflecting "flight-to-quality" or contagion effects observed in downturns.

We estimate the ADCC parameters (a, b, g) enhanced with a Student-t innovation assumption and parameter estimation via maximum likelihood. The theoretical derivation of the ADCC recursion and student-t maximum likelihood are located to Appendix I.2.

Estimation procedure. Parameter estimation is performed via constrained optimization ('L-BFGS-B') over:

$$\boldsymbol{\theta} = (\alpha, \beta, \gamma, \nu),$$

initialized to $(0.03, 0.90, 0.03, 8)$ with constraints:

$$0 \leq \alpha, \beta, \gamma; \quad \alpha + \beta + \gamma < 0.99; \quad 2.001 \leq \nu \leq 50.$$

By default, unconditional correlation $\bar{Q}$ is fixed to the sample correlation of $\mathbf{z}$, and optimization returns $(\hat{\alpha}, \hat{\beta}, \hat{\gamma}, \hat{\nu})$.

Forecasting next-day covariance. Using a two-step procedure: first estimate each univariate XGBoost for volatilities, then estimate the correlation dynamics. The output of the ADCC model is a forecast of the full $N \times N$ correlation matrix R_{t+1} for the next day's returns, which we pair with the volatility forecasts to build the covariance matrix. Once fitted, the recursion is applied out-of-sample: at time t, given vol forecasts $\hat{\sigma}_{i,t+1}$ and current Q_t:

$$Q_t = (1 - a - b - g)\,\bar{Q} + a\,(z_{t-1} z_{t-1}^\top) + b\,Q_{t-1} + g\,(n_{t-1} n_{t-1}^\top)$$

$$R_{t+1} = \operatorname{diag}(Q_{t+1})^{-1/2}\, Q_{t+1}\, \operatorname{diag}(Q_{t+1})^{-1/2},$$

$$\hat{\Sigma}_{t+1} = D_{t+1} R_{t+1} D_{t+1}, \quad D_{t+1} = \operatorname{diag}(\hat{\sigma}_{i,t+1}).$$

This yields the asset covariance matrix used for portfolio optimization.

3.6.2 Method 2: Supervised Learning Models for Covariance Prediction (LSTM)

In contrast to the two-stage econometric modeling approach of Method 1 (XGBoost and ADCC), Method 2 adopts a direct forecasting strategy, employing a heavier supervised machine learning technique to predict the covariance structure of asset returns. This methodology explicitly targets the Cholesky decomposition of daily realized covariance matrices. The principal motivation behind directly predicting the Cholesky decomposition is to ensures the positive definiteness and numerical stability of forecasted covariance matrices, while simultaneously allowing the model to capture and forecast correlations on top of variances in order to only have 1 model per feature-stage and to be a fully data-driven approach.

We will explore one of the most advanced sequence modeling architectures which is extensively documented in modern financial forecasting: Long Short-Term Memory (LSTM) network. All historical context, equations and architecture developments of the LSTM model are in Appendix I.3.

Application to Covariance Prediction. In our covariance forecasting framework, the LSTM processes targets of Cholesky decompositions of realized covariance matrices $(L_{t-k}, \ldots, L_t)$ along with sequences of relevant market features (e.g., returns, volatility measures, technical indicators). The sequence of inputs is encoded by the LSTM into its hidden states, enabling the model to predict the next period's Cholesky decomposition (L_{t+1}). The model outputs are reshaped into lower-triangular form, ensuring a valid Cholesky matrix with positive diagonal elements.

Training Procedure and Loss Function. The LSTM model is trained to forecast the Cholesky factor L_{t+1} of the covariance matrix, using the root mean square error (RMSE) between predicted and actual L-matrices as the loss:

$$L_{\text{RMSE}} = \sqrt{\frac{1}{T}\sum_{t=1}^{T} \left\| \widehat{L}_{t+1} - L_{t+1} \right\|_F^2},$$

where $\|\cdot\|_F$ denotes the Frobenius norm across all matrix entries. Using RMSE ensures the loss remains in the same units as the matrix entries, improving interpretability and aligning with common regression practices.

Model Architecture & Hyperparameters

Table 3.1: LSTM (input-based) Model Architecture & Training Hyperparameters

Component	**Specification**
Window size	100 days
Batch size	256
LSTM layers	2 stacked layers
Hidden units per layer	64
Dropout	0.3 (between layers)
Output dimension	$\frac{N(N+1)}{2}$ (Cholesky entries)
Learning rate	1×10^{-4}, Adam optimizer
Weight decay	1×10^{-4} (Adam regularization)
Epochs / Early stopping	up to 10,000 epochs; stop after 20 with no validation improvement
Loss function	RMSE on predicted Cholesky factor: $\sqrt{\frac{1}{T}\sum_t \|\hat{L}_{t+1} - L_{t+1}\|_F^2}$

Training is performed on GPU using early stopping on validation RMSE. The best model checkpoint is saved for forecasting on the test period. These settings align with recommended practices for sequence forecasting in financial contexts, where smaller learning rates, moderate dropout, and gradient-based optimizers often outperform due to noisy, non-stationary data.

Summary and Benefits. Overall, the LSTM's gated recurrent structure allows to capture complex, nonlinear temporal relationships in covariance data, crucial for robust financial forecasting. Its inherent stability in gradient propagation positions LSTM as a suitable method for daily prediction tasks.

3.6.3 Processing Forecasts of input-based models

Covariance Matrix Construction and Robustification

For each forecasting methodology within input-based models, we obtain a covariance matrix prediction, $\hat{\Sigma}_{t+1}$, for asset returns at day $t+1$.

While Method 1 and 2 inherently guarantees positive definiteness through the ADCC model or Cholesky decomposition, both forecasting methods (1 and 2) may still benefit from covariance shrinkage and/or eigenvalue filtering to enhance estimation stability and reduce noise, particularly in high-dimensional settings.

A practical and robust choice for the shrinkage target is the constant correlation matrix, characterized by uniform off-diagonal correlations set to the average pairwise correlation. By shrinking covariance forecasts towards a structured target, we significantly mitigate the impact of estimation noise and potential outliers, improving out-of-sample portfolio stability and performance. A practical method for eigenvalue filtering is simply by reconstructing the input matrix with eigenvalues capped with a slight positive epsilon value to guarantee positive-definiteness.

The detailed historical context and equation development of Ledoit-wolf shrinkage and eigenvalue filtering are located in Appendix I.5.

Portfolio Optimization (GMV, MDP, ERC)

With a forecast covariance matrix $\hat{\Sigma}_{t+1}^{*filtered}$ in hand, we solve for optimal portfolio weights under various risk-based optimization objectives. In this thesis, we focus on three portfolio constructions: Global Minimum Variance (GMV), Maximum Diversification (MDP), and Equal Risk Contribution (ERC). These portfolios do not require forecasts of expected returns (which empirically are noisier estimates), instead relying solely on the covariance matrix to allocate risk efficiently.

All optimizers will have a leverage limit of 5 to give sufficient space for short positions, a full investment constraint and short-selling allowed in order to benefit from the binance future leveraged environment. To further refine the optimization, especially in the presence of transaction costs or turnover constraints, we incorporate a Huber penalty function. The Huber function balances sensitivity and robustness by behaving quadratically near zero and linearly for large deviations. This approach mitigates the impact of outliers and ensures smoother adjustments to the portfolio weights.

The optimization problem is solved using the Sequential Least Squares Programming (SLSQP) algorithm, which efficiently handles both equality and inequality constraints. This method ensures convergence to a solution with practical considerations like transaction costs and turnover limits.

Each optimizer's historical background and equations are left in Appendix I.6.

Parameters. Below is a concise summary of the specific constraints and parameters used for each optimizer:

Table 3.2: Summary of Portfolio Optimization Constraints and Parameters

Constraint / Parameter	**GMV**	**MDP**	**ERC**
Objective	Min variance	Max diversification	Equal risk contribution
Leverage limit	5	5	None
Full investment (net exposure)	1	1	1
Short-selling allowed	Yes	Yes	No
Min weight per asset	-5	-5	-1
Max weight per asset	5	5	1
Transaction fees included	No	Yes	Yes
Transaction fees (λ_{fee})	–	0.0005	0.0005
Turnover constraint	None	None	None
Optimizer	SLSQP	SLSQP	SLSQP
Tolerance	10^{-13}	10^{-8}	10^{-15}

Notes: GMV wasn't stable when including transaction fees and ERC is naturally using a leverage equal to 1.

3.7 Single-stage: Direct Portfolio Weight Prediction

In contrast to input-based models, which explicitly forecasts covariance structures before portfolio optimization, single-stage predict portfolio weights directly, bypassing explicit volatility and correlation estimation steps. This single-stage approach directly targets the ultimate objective of portfolio performance, potentially capturing complex nonlinear relationships between asset returns, momentum signals, macroeconomic indicators, and technical metrics more effectively. We examine two distinct modeling paradigms within this direct prediction framework: **Supervised Learning Models** (LSTM), where historical market and asset-specific data is directly mapped to optimal future portfolio allocations and **Deep Reinforcement Learning** (DRL), framing portfolio allocation as a sequential decision-making task, with the model directly learning to optimize risk-adjusted log-returns through direct market interactions with weights treated as actions.

The motivation here is that a single-stage model might capture nonlinear relationships between asset returns, momentum, and other indicators in a way that directly optimizes

the end goal (portfolio performance), potentially outperforming a two-stage approach that might propagate errors from forecasts to allocation.

3.7.1 Method 3: Supervised Learning Models for Weight Prediction (LSTM)

The supervised learning component of method 3 leverages advanced sequence models previously described in detail under method 2 (Section 3.6.2). The LSTM model is repurposed specifically for the direct prediction of portfolio weights. Unlike its original application to covariance forecasting, the supervised model in this section directly output optimal asset allocations using historical market weights in addition of the same stage-features, depending on the target optimizer, as input, thus explicitly learning allocation rules rather than intermediate volatility or correlation measures.

Dataset Construction and Training Targets. To frame portfolio weight prediction as a supervised learning task, we construct a training set of input-output pairs $\{X_t, w_t^*\}$, where X_t is the usual vector of features and w_t^* is a target weight vector that we consider "optimal" for period $t+1$. One way to obtain w_t^* for training is to use the retrospective optimal weights.

We use the historical weights computed from the optimizers used for input-based methods (GMV, MDP and ERC) which we feed directly the true target of daily historical realized covariance matrices constructed from 1-minute frequency returns. Once we have targets, we can fit a supervised model to approximate this mapping.

Model Implementation and Training Procedure. The LSTM retain its core architecture and computational details as previously outlined (see Section 3.2). Minor adjustments are applied to the output layer structure modifying only the output layer to dimension N for direct weight prediction.

Hyperparameter tuning (e.g., number of layers, hidden units, attention heads) and regularization strategies (e.g., dropout, weight decay, early stopping based on validation loss) follow identical methodological practices to those previously described, aiming for robust out-of-sample predictive performance and generalization.

Unique Considerations for Direct Weight Prediction. While supervised weight prediction simplifies the forecasting pipeline by eliminating intermediate steps, it introduces unique challenges. Primary among these is the inherent dependence on the reliability of the target weight vectors w_t^*. Noisy, unstable, or overly model-specific target weights can adversely impact supervised learning effectiveness. To address this, a shrinked version of the weights targeted whose realized covariance matrix underwent a constant

correlation shrinkage is also proposed just like in the input-based models. Another constraint with this method is the number of models trained that grows considerably since one model is only specialized for one specific optimizer and realized covariance matrix (robust vs non-robust), this results in 6 models per stage-features instead of previously 1 for input-based models.

Model Architecture & Hyperparameters

Table 3.3: LSTM (weight-based) Hyperparameters

Component	**Specification**
Window size	100 daily steps
Batch size	256
LSTM architecture	2 stacked layers, 64 hidden units each
Dropout	0.3 (between LSTM layers)
Output dimension	N (for direct weights)
Activation	Linear output; no positivity enforced
Learning rate	1×10^{-4}, Adam optimizer
Weight decay	1×10^{-4} (L2 regularization)
Epochs / Early stopping	up to 10,000 epochs; stop after 20 epochs without validation improvement
Loss function	RMSE: $\sqrt{\frac{1}{T}\sum_t \|\hat{w}_t - w_t^*\|^2}$

In short, this supervised LSTM policy learns asset allocation directly from historical states, providing a fast mapping to weights at inference time by avoiding intermediate volatility/covariance models. Its performance, however, hinges on the stability and quality of the target weight sequences w_t^*.

3.7.2 Method 4: Deep Reinforcement Learning with Proximal Policy Optimization (PPO)

Overview and Motivation. Method 4 diverges from the classical "predict–then–optimize" paradigm by *learning the allocation rule itself.* Instead of first estimating moments of the return distribution and subsequently solving a mathematical program, we cast portfolio management as a *sequential decision–making* problem and train an autonomous agent with *deep reinforcement learning* (DRL). The agent repeatedly observes the market, selects a vector of portfolio weights, receives a reward that reflects risk–adjusted performance, and updates its parameters so as to maximize the expected long-run utility of wealth. This formulation is attractive in highly non-stationary environments such as cryptocurrency markets, because it enables the policy to adapt online to abrupt regime shifts, structural breaks, changing liquidity, and time-varying trad-

ing frictions, without requiring an explicit statistical factorization of the data-generating process.

In this study we adopt *Proximal Policy Optimization* (PPO) (Schulman et al., 2017), a policy-gradient algorithm that combines the sample efficiency of trust-region methods with the implementation simplicity of vanilla stochastic gradient ascent. PPO has demonstrated competitive performance across a wide spectrum of continuous-control tasks and possesses several properties that are particularly beneficial for asset allocation: (i) it optimizes a *stochastic* policy, naturally encouraging exploration in high-dimensional action spaces; (ii) it employs a *clipped surrogate objective* that stabilizes training even when the reward surface is extremely noisy; and (iii) it is *compatible with recurrent neural networks* (LSTM), permitting efficient credit assignment through long sequences of market observations.

PPO full surrogate objective, GAE derivation and full algorithm summarization are left in Appendix I.4.

Formulating Portfolio Management as a Markov Decision Process

Let $\mathcal{T} = \{0, 1, 2, \ldots, T\}$ denote discrete decision instants corresponding to a fixed calendar grid; in our implementation one step equals 1 day, matching the resampling frequency of the input data. At every time $t \in \mathcal{T}$ the agent observes a *state* vector $s_t \in \mathcal{S}$ and issues an *action* $a_t \in \mathcal{A}$. The environment then executes a_t, realizes the one-period market evolution, and returns a *reward* $r_{t+1} = R(s_t, a_t, s_{t+1}) \in \mathbb{R}$ together with the new state s_{t+1}:

$$s_t \xrightarrow{a_t} s_{t+1}, \quad r_{t+1} = R(s_t, a_t, s_{t+1}),$$

An episode terminates at $t = T$ or earlier if a catastrophic event such as account liquidation occurs. The components are defined as follows:

State space $\mathcal{S}$. Each state encapsulates two to five information layers depending on the stage of the dataset with one specific information layer being present in all stages. The additional information layer that is specific for this model and present in every stage-features is the *Portfolio attributes* layer that represents the seventeen dependent dynamic features described in section 3.4.1. These attributes allow the agent to get states that are more dependent on its actions compared to the static states of other stages, making the agent aware of the consequences of its action through those dynamic states.

Formally, if $x_t \in \mathbb{R}^d$ denotes the dataset features at time t, and $p_t \in \mathbb{R}^{17}$ denotes the dependent attributes, then the *observation window* is an $L \times (d + 17)$ tensor

$$s_t = \big[\, [x_{t-L+1}, p_{t-L+1}],\ [x_{t-L+2}, p_{t-L+2}],\ \ldots,\ [x_t, p_t] \,\big] \in \mathbb{R}^{L \times (d+17)}.$$

This high-dimensional tensor is processed by an LSTM encoder built into the PPO policy network.

Action space $\mathcal{A}$. With `limit_bounds=False`[2] the action is a *continuous weight vector* $a_t = w_t = (w_t^{(1)}, \ldots, w_t^{(10)}) \in [-1, 1]^{10}$. The sign of $w_t^{(i)}$ encodes long (+) versus short (−) exposure, while its magnitude encodes the fraction of total equity *before leverage*. Because we use Binance's *one-way cross margin* regime, the ℓ_1 *leverage constraint*

$$\sum_{i=1}^{10} |w_t^{(i)}| \;\leq\; 5 \tag{3.6}$$

must hold at all times. The choice of including the hard constraint of full investment in the market:

$$\sum_{i=1}^{10} w_t^{(i)} = 1 \tag{3.7}$$

isn't necessary but help the model to converge more easily since the exploration universe is greatly reduced and can be more fairly compared to the other models relying on optimizers with the same constraint. We therefore always rescale the raw network output in the beginning of the step with a little numerical problem using ECOS algorithm to output rescaled weights that respects those two constraints.

Transition Dynamics. The environment simulates realistic order execution based on the agent's portfolio weights w_t. The market evolution is exogenous to the agent. The following mechanisms are implemented:

First, for *Data handling*, a numpy dataset holds a OHLCV at daily intervals for computing every account logic relying on raw prices and volume. The OHLCV dataset is entirely loaded into memory during training and is synchronized with the normalized sequenced dataset containing the features to feed the actor and critic networks.

Second, concerning *Bid and Ask Price Simulation*, for each asset i at time t, the bid and ask prices are synthesized by perturbing the mid-price with a Gaussian slippage term proportional to the asset's price range and inversely proportional to trading volume:

[2] A second variant with stop-loss / take-profit bounds is provided in the source code but not used in the baseline experiments.

$$
\begin{aligned}
\text{mid}_i &= \frac{\text{high}_i + \text{low}_i}{2}, \\
\sigma_i &= \frac{\text{base\_std}}{\text{volume}_i/\text{scaling\_factor}}, \\
\varepsilon_i^{\text{bid}}, \varepsilon_i^{\text{ask}} &\sim \mathcal{N}(0, \sigma_i^2), \\
\text{bid}_i &= \text{close}_i - \varepsilon_i^{\text{bid}} \cdot (\text{high}_i - \text{low}_i), \\
\text{ask}_i &= \text{close}_i + \varepsilon_i^{\text{ask}} \cdot (\text{high}_i - \text{low}_i).
\end{aligned}
\tag{3.8}
$$

This approach ensures that the bid-ask spread reflects market liquidity, with higher volumes leading to narrower spreads.

Third, for *Order Execution with Slippage and Fees*, they are executed at the end of each period using the appropriate side of the order book. Slippage is modeled as a Gaussian perturbation:

$$
\text{slippage} \sim \mathcal{N}(\mu_{\text{slip}}, \sigma_{\text{slip}}). \tag{3.9}
$$

The effective trade price $P_{\text{trade},i}$ for asset i is then:

$$
P_{\text{trade},i} = \begin{cases} \text{ask}_i \cdot (1 + \text{slippage}), & \text{if buying,} \\ \text{bid}_i \cdot (1 - \text{slippage}), & \text{if selling.} \end{cases} \tag{3.10}
$$

Transaction fees are applied based on the order type:

$$
\text{fee}_{\text{bps}} = \begin{cases} 5, & \text{for market orders,} \\ 2, & \text{for limit orders.} \end{cases} \tag{3.11}
$$

The total transaction cost is:

$$
\text{Cost}_{\text{transaction}} = P_{\text{trade},i} \cdot \text{quantity}_i \cdot \left(1 + \frac{\text{fee}_{\text{bps}}}{10,000}\right). \tag{3.12}
$$

Fourth, for *Mark Price and Margin Call Mechanism*, the mark price for each asset is computed by adjusting the mid-price with a Gaussian noise term dependent on trading volume:

$$
\text{mark}_i = \text{mid}_i \cdot (1 + \eta_i), \quad \eta_i \sim \mathcal{N}(0, \sigma_{\text{mark},i}^2), \tag{3.13}
$$

where $\sigma_{\text{mark},i}$ is determined by:

$$
\sigma_{\text{mark},i} = \frac{1}{\text{volume}_i/\text{scaling\_factor}}. \tag{3.14}
$$

Fifth, for *Cross-Margining and Liquidation*, when in *cross margin* mode, total main-

tenance margin M_t is computed:

$$M_t = \sum_i \max\left(\theta_i \cdot \text{Notional}_i - m_i, 0\right), \tag{3.15}$$

where θ_i is the margin rate for asset i, and m_i is a per-tier adjustment. If the portfolio equity E_t drops below M_t, a margin call is triggered:

$$\text{If } E_t < M_t \Rightarrow \text{liquidate all positions at } p_t^{\text{mark}}. \tag{3.16}$$

leading to the liquidation of all positions at the current mark prices.

Finally, for *State construction*, every call to `step()` appends the new feature vector to a `sequence_buffer` of length L and discards the oldest entry, yielding an overlapping sliding window so that temporal patterns are preserved. This way, the seventeen action-dependent features can be updated inside the sequence.

Reward Engineering. Reward design is arguably the most delicate component of any deep reinforcement learning (DRL) framework in finance. To ensure stable learning and realistic behavior, our reward function is designed to fully contain the needed information inside each step and not rely on episodic or sub-episodic accumulation for approximations (dense reward). It integrates four complementary mechanisms: (1) step-wise profitability with step-wise volatility penalization, (2) penalties for missed or invalid trades, (3) early stopping when the portfolio collapses, and (4) online reward normalization using Welford's algorithm.

Concerning *Log-return with volatility regularization*, the base reward at step t is the agent's log-return $v_t = \ln(V_t/V_{t-1})$. To discourage volatility-seeking behavior, we compute an exponentially weighted moving average (EWMA) of past squared returns, with smoothing parameter β, and subtract a volatility penalty:

$$r_t^{\text{base}} = v_t - \lambda_{\text{vol}} \cdot \sqrt{\sigma_t^2}, \quad \text{where} \quad \sigma_t^2 = \beta \cdot \sigma_{t-1}^2 + (1-\beta) \cdot v_t^2.$$

In our implementation, λ_{vol} governs the strength of this penalty, and $\beta = 0.98$ ensures slow decay.

For the *Survival bonus*, it is there to avoid crashing the portfolio on purpose as a corner solution to not accumulate volatility penalties, +50bps is applied as long as the portfolio isn't liquidated.

Concerning *Penalties for invalid or absent trading actions*, they serve to guide exploration and enforce market constraints, we apply small penalties when: The agent chooses not to trade when a trade was expected ("no trade" penalty):

$$r_t^{\text{no-trade}} = -\min\left(\eta + 0.5\eta \cdot n, \eta_{\max}\right),$$

where $\eta = 0.0001$, n is the number of consecutive no-trade steps, and $\eta_{\max} = 0.001$. The agent issues a technically invalid trade (e.g., wrong leverage, insufficient collateral, trade size too small/large):

$$r_t^{\text{miss}} = -\min\left(\delta + 0.5\delta \cdot m, \delta_{\max}\right),$$

where $\delta = 0.0002$, m is the number of consecutive technical errors, and $\delta_{\max} = 0.002$. These penalties are cumulative and capped to prevent gradient explosion while ensuring frequent mistakes are discouraged more strongly.

The *Early termination penalty* is there if the agent's portfolio value drops below 10% of the initial balance, then the episode is ended prematurely and the scaled penalty is applied:

$$r_t^{\text{early-exit}} = -\gamma \cdot \left(1 - \frac{t}{T}\right),$$

where γ is a user-defined penalty scale and T is the episode length. This prevents the agent from "suiciding" early to reduce overall risk exposure.

The *Reward normalization and clipping* is there to stabilize training across volatile return scales, we apply online mean–variance normalization using Welford's algorithm. Given cumulative count N, running mean μ_N, and variance σ_N^2, each raw reward is converted into a z-score:

$$z_t = \frac{r_t - \mu_N}{\sqrt{\sigma_N^2/(N-1) + \varepsilon}},$$

then clipped to the interval $[-c, +c]$ with $c = 5$ and rescaled to $[-1, +1]$:

$$r_t^{\text{final}} = \frac{\max(-c, \min(z_t, c))}{c}.$$

Concerning the *Composite structure*, the full reward is composed as:

$$r_t = r_t^{\text{base}} + r_t^{\text{no-trade}} + r_t^{\text{miss}} + r_t^{\text{early-exit}},$$

followed by online normalization, clipping and rescale to ensure training stability.

A Realistic Binance–Style Environment

Designing a *credible simulator* is crucial: in order to make the environment similar to Binance's environment, the transition dynamics described are sufficient but not enough. Below we highlight Binance's specific mechanics.

Position management (*One-Way mode*). Each asset i is tracked via:

$$\texttt{positions}[i] = \{\text{type},\ \text{entry\_price},\ \text{size},\ \text{leverage}\}.$$

When a new weight $w_t^{(i)}$ is requested, it is translated into a target notional exposure $desired_{\text{pos}} = w_t^{(i)} \times$ (equity - margin requirements) (isolated or cross). The environment then computes *Partial or full close* of an existing position in the opposite direction if necessary. *Opening/increasing* a position in the same direction, provided constraints allow. *Switching sides* by fully closing an existing position and opening it in the opposite direction. Trades respect Binance's *One-Way mode*, allowing only one open position per asset with seamless merging of increases and decreases into a single position state. Increases adjust the entry price via size-weighted averaging:

$$p_{\text{new}} = \frac{p_{\text{old}} s_{\text{old}} + p_{\text{trade}} s_{\Delta}}{s_{\text{old}} + s_{\Delta}}.$$

Stops, take-profits, and unrealized P&L are updated dynamically. Fees are subtracted upon both opening and closing trades, and realized P&L is updated accordingly. This routine ensures coherence with actual Binance account behavior, including liquidation logic, margin tracking, and smooth handling of position transitions.

Margin tiers, trade size, and collateral constraints. Tier-dependent maximum leverage $\bar{\ell}(n)$, maintenance margin requirement $\text{MMR}(n)$, and notional fees $\text{fee}(n)$ are retrieved via `get_margin_tier()`, which mirrors Binance's documented rules for each asset. If a proposed trade would exceed any constraint—leverage cap, maintenance margin, or notional fees—a "technical miss" penalty of up to 20bps is applied and the order is effectively ignored, teaching the agent to avoid invalid actions. Additionally, each asset is subject to minimum and maximum trade sizes (in USD). If the order violates:

$$|\Delta_{\text{pos}}| < \text{min\_notional}_i \quad \text{or} \quad |\Delta_{\text{pos}}|/P_i < \text{min\_units}_i \quad \text{or} \quad |\Delta_{\text{pos}}|/P_i > \text{max\_units}_i,$$

the same technical miss penalty is applied and the agent keeps its current position until the next decision point. Collateral availability is computed differently based on margin mode.

Isolated margin:

$$\text{eq}_i = \frac{s_i}{\ell_i} + \text{uPnL}_i, \quad \text{free\_coll}_i = \max(\text{eq}_i - \text{MRR}_i,\, 0).$$

Cross margin, a global equity pool is constructed:

$$\text{free\_coll} = \max\left(\sum_i \left(\frac{s_i}{\ell_i} + \text{uPnL}_i\right) - \sum_i \text{MRR}_i,\, 0\right).$$

If the agent's desired trade requires margin $|\Delta_{\text{pos}}|/\ell$ greater than available (equity - margin requirements), a technical miss penalty is issued and the trade is rejected. This design ensures the agent learns to respect leverage tiers, collateral requirements, and trade size

constraints in a realistic Binance-like environment.

Training Protocol and Hyper-parameters

Training is performed with **Ray RLlib 2.10**. Table 3.4 lists the salient hyper-parameters (the remainder retain default RLlib values).

Component	Parameter	Value	Comment
Roll-outs	`num_workers`	14	saturates 1 GPU / 16 CPU at 95 %
	`fragment_len`	120	1 day
	`batch_mode`	`complete_episodes`	GAE across fragments
Optimizer	`train_batch_size`	1 680	14×120
	`sgd_mini-batch_size`	280	1/6 of batch
	`num_sgd_iter`	10	per PPO default
	`lr`	10^{-4}	scheduled
	`grad_clip`	0.5	
Algorithm	γ	0.97	$\approx$ 100-step horizon
	λ	0.9	in GAE
	ϵ	0.1	clip range
	Entropy coeff η	10^{-3}	encourages exploration
Model	`use_lstm`	True	sequence modeling
	`lstm_cell_size`	128	
	`fcnet_hiddens`	$[128, 128]$	post-LSTM
	`max_seq_len`	100	coincides with window
	`fcnet_activation`	"relu"	linear activation
	`post_fcnet_activation`	"linear"	linear activation
	`lstm_use_prev_action_reward`	True	
	`_disable_action_flattening`	True	

Table 3.4: Main PPO hyper-parameters.

Curriculum scheduling and early stopping. Training proceeds for either a fixed 5 million steps trained if the validation reward doesn't improve or can continue further if it keeps improving. No automatic early stopping is implemented, contrary to supervised learning, we are evaluating a policy and not a direct mapping so letting the model explore before judging the policy is the most advisable solution.

Summary

The DRL pipeline described here constitutes a fully model-free, end-to-end solution: from raw daily states to portfolio weights, all components are learned jointly. Unlike the prediction–optimization chain in input-based models and even direct weight in the supervised model, this method never computes an explicit or implicit covariance matrix; instead risk and returns are managed entirely through the simulated environment. *One long episode (1,117 daily steps)* represents the entire training set, preserving long-range context that captures slow-moving patterns such as weekend liquidity shifts or

month-end funding stress—phenomena lost in shorter resets. Although the environment spans 1,117 days, *Truncated rollouts of 120 steps/day* allow to maintain long-term dependency via bootstrapping while improving sample efficiency and reducing value estimation bias. The simulator includes *Realistic micro-structure modeling* such as transaction frictions—Gaussian slippage, bid–ask spreads, fees, leverage tiers, and liquidation tracking. Without these frictions, policies overfit to idealized conditions and perform poorly under real fees. *PPO configuration being tuned for stability* uses a discount factor $\gamma = 0.97$, learning rate schedule, entropy coefficient decay, clipped objective ($\epsilon = 0.1$), and gradient clipping (0.5), along with an LSTM encoder (128 units), ensures stable learning across the 1,117-step training window. In combination, these modeling decisions fostered robust policy learning that respects financial constraints while retaining sensitivity to both short- and long-term market dynamics.

3.8 Evaluation

In the preceding sections we have described—in considerable technical detail—the two modeling pathways that culminate in concrete allocation rules: **Input-based** (forecast-then-optimize) and **Direct-weights** (direct weight prediction, either supervised or PPO–based RL). The present section closes *Chapter 3 (Methodology)* by specifying *how* those models are compared out-of-sample. In order to avoid redundancy with the model-specific discussions already provided, we restrict ourselves to a concise catalogue of evaluation metrics.

3.8.1 Forecast-Level Accuracy (Excluding DRL method)

Because supervised and econometric models produces explicit forecasts that are of independent interest for risk management, we report two error statistics to assess the accuracy of the predictions coming from supervised models.

Volatility, Covariances and Weights. For every asset i,

$$\mathrm{RMSE}_i = \sqrt{\frac{1}{T}\sum_{t=1}^{T}\left(\hat{\sigma}_{i,t} - \sigma_{i,t}\right)^2}, \tag{3.17}$$

with $\sigma_{i,t} = \sqrt{\mathrm{RV}_{i,t}}$.

3.8.2 Portfolio-Level Performance Metrics

Ultimately, the realized portfolio performance is the final and most important metric. We compare all models on portfolio metrics to see if one approach is more stable. For more

Metric	Formula / Definition
Cumulative return	$V_T/V_0 - 1$
Annualized return	$\left(V_T/V_0\right)^{365/T} - 1$
Annualized volatility	$\sqrt{365}\,\sigma(r_t)$
Sharpe ratio	$\overline{r}/\sigma(r_t)$
Sortino ratio	$\overline{r}/\sigma^{-}(r_t)$
Maximum draw-down	$\min_t D_t$
Lowest point (% initial cap)	$\min_t V_t/V_0$
$\text{VaR}_{99\%}$	1 % left-tail quantile
$\text{CVaR}_{99\%}$	mean loss $<$ VaR
Calmar ratio	$\text{Ann. ret}/\lvert\text{MDD}\rvert$
Upside potential ratio	$\frac{\text{EP}[r \mid r>0]}{\sigma^{-}(r)}$
Omega ratio	$\Omega(0)$
Skewness	third central moment
Kurtosis	fourth central moment

Table 3.5: Performance measures

explanation concerning each metrics purpose, explanations are left to Appendix I.5. All strategies are re-balanced daily

3.8.3 Benchmark Strategies

To contextualize the out-of-sample statistics we pit every model against three progressively more sophisticated baselines. *First*, a *plain equal–weight portfolio* that splits capital $1/N$ across the ten coins with no leverage: because it requires neither forecasting nor rebalancing logic, any allocation rule that fails to beat this naïve benchmark on a net-of-cost daily basis can hardly claim economic value. *Second*, a *value-weighted portfolio* that weights each asset in proportion to its market capitalization and rebalances daily; this "crypto market-cap index" reflects the composition a passive investor would obtain by buying the aggregate market and therefore gauges the incremental alpha that our active signals must deliver. *Third*, an *expanding-window Markowitz benchmark* that re-estimates the historical covariance matrix every day and feeds the three optimizers (ERC, GMV, MDP) in both raw and constant-correlation (robust) form. *Fourth*, a *GJR–GARCH + ADCC benchmark* in which each coin's conditional variance follows a student-t GJR–GARCH and the conditional correlations are generated by an ADCC–student-t filter; the resulting daily covariances are passed through the same three optimizers and traded under the identical 5× cross-margin Envelope. This fourth benchmark is a fully econometric pipeline that preserves all stylized facts (volatility clustering, leverage effect, asymmetric correlation spikes) making it a harder benchmark to beat for the more elaborate hybrid and reinforcement-learning models that follow.

3.8.4 Backtesting Protocol

To ensure an apples-to-apples comparison we replay *all* allocation rules—including the optimizers (GMV, MDP, ERC) with input-based models, the LSTM for direct weights, the trained PPO agent and the benchmarks—inside the `TradingEnvironment` that was already instrumented for reinforcement-learning experiments. Using a single simulator guarantees that bid–ask spreads, slippage, fees, margin tiers and liquidation logic are *identical* for every strategy, while seeding the underlying NumPy, PyTorch and Gymnasium PRNGs with the constant value 42 yields fully deterministic roll-outs.

Implementation outline. Let $\widehat{w}_t$ be the vector of pre-computed daily weights produced by one of the models on the test chronology $\{\tau_1, \ldots, \tau_T\}$. Backtesting proceeds as follows:

First, *Initialize the simulator*: `env = gym.make("trade_env_ray_portfolio", mode="test", input_length=100, leverage=5, seed=42, limit_bounds=False, margin_mode="cross")` this loads the OHLCV test slice (09-Feb-2024 – 20-May-2025) and performs the 100-day burn-in.

Second, *Synchronize the simulation clock.* The first call to `env.reset()` places the internal pointer at τ_1; the date index of $\widehat{w}_t$ is aligned so that every subsequent `env.step(\widehat w_t)` consumes exactly one day.

Third, *Record equity path.* After each step we log `env.portfolio_value`, producing a vector $V_{0:T}$. The same series is used to compute all performance statistics listed in Table 3.5.

Fourth, *Freeze stochastic elements.* With the seed fixed, the Gaussian slippage, bid–ask perturbations and liquidation shocks are reproduced every time the script is executed, which eliminates sampling noise in the evaluation.

Simulator configuration. Key run-time constants are summarized in Table 3.6. They mirror Binance cross-margin rules (L_1-leverage cap = 5, maintenance-margin tiers, 5 bps taker / 2 bps maker fees, 1.25 % liquidation surcharge) and the stochastic micro-structure model described in Section 3.7.2.
Because every strategy is evaluated through the same deterministic environment the resulting performance metrics are directly comparable and free from simulator-induced bias. This closes Chapter 3 by connecting model development with the out-of-sample evaluation framework that will be exercised in the next chapter.

Table 3.6: PPO-style simulator: core back-test parameters

Parameter	**Setting**
Random seed	42 (applied to NumPy, PyTorch, Gymnasium)
Decision interval	1 day
Look-back window L	100 observations
Leverage constraint	$\sum_i \lvert w_i \rvert \leq 5$ (cross-margin)
Fee schedule	5 bps market, 2 bps limit
Slippage model	$\mathcal{N}(\mu = 1$ bp, $\sigma = 5$ bp) on mid-price
Liquidation fee	125 bps of notional value
Bid–ask spread	Gaussian, std $\propto 1/\text{vol}$
Episode horizon	367 trading days (test set)

Chapter 4

Empirical Results

4.1 Chapter Road-map

This chapter reports out-of-sample evidence for the four modeling families introduced in Chapter 3. We move in three steps. First, we benchmark *forecast-level accuracy*, covering volatility and covariance errors for Models 1 and 2 and weight-tracking errors for Model 3. Second, and most importantly, we compare *portfolio-level performance*, taking the six input-optimized strategies against the direct-weight LSTM and the PPO agent under identical execution frictions. Third, a *cross-model synopsis* condenses the key risk-and-return metrics into a single summary table and interprets the relative strengths of each approach. Where multiple dataset stages exist (Stages 1–4), only the best stage appears in-line; full artifacts for the discarded stages reside in the online repository (see Appendix I).

4.2 Forecast-Level Accuracy

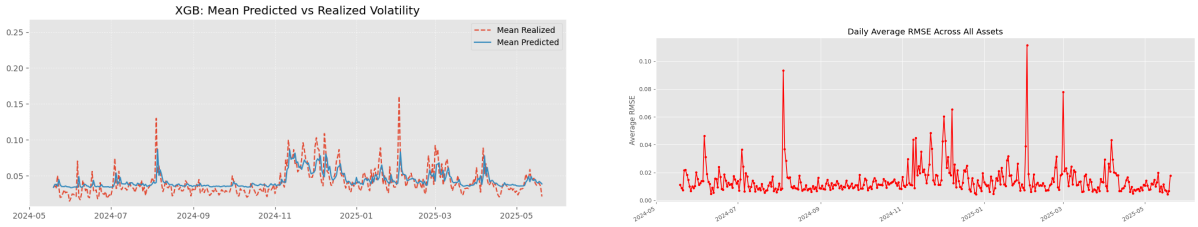
4.2.1 Model 1 – XGB–ADCC

Table 4.1: Model 1 (XGB–ADCC) — forecast accuracy by dataset stage

	Volatility		**Covariance: raw**				**Covariance: robust**			
St.	RMSE	MAE	RMSE	Diag	Off	NormF	RMSE	Diag	Off	NormF
1	0.0192	0.0136	0.0019	0.0036	0.0016	0.0162	0.0019	0.0036	0.0016	0.0161
2	0.0184	0.0126	0.0020	0.0037	0.0017	0.0161	0.0019	0.0037	0.0016	0.0160
3	**0.0183**	**0.0123**	**0.0018**	**0.0031**	**0.0015**	**0.0149**	**0.0017**	**0.0031**	**0.0015**	**0.0148**
4	0.0200	0.0139	0.0019	0.0037	0.0016	0.0164	0.0019	0.0037	0.0016	0.0166

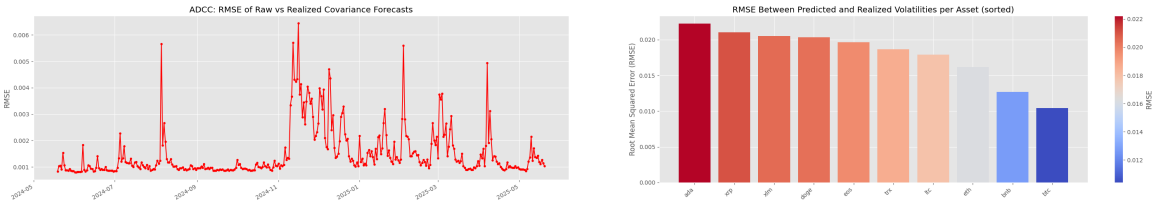
Interpretation. Stage 1 already yields serviceable forecasts, and the shrinkage filter hardly changes the numbers (raw vs. robust columns are identical to three decimals).

Adding technical indicators (Stage 2) trims volatility RMSE by ≈4 % but introduces a slight covariance penalty in the raw matrix; the shrinkage step erases that penalty, leaving the robust error flat. **Macro factors dominate:** Stage 3 produces the global minimum in every metric—volatility, raw covariance, and robust covariance. Normalized Frobenius falls from 0.0161 to 0.0148 (–8.1 %), and the shrinkage filter still ekes out a 1 bp improvement by damping small eigen-noise. **Sentiment over-fit:** Once sentiment features are appended (Stage 4) volatility RMSE jumps 9.3 %; both diagonal and off-diagonal covariance errors revert to Stage-1 levels. The shrinkage filter, instead of helping, nudges NormFrob *up* by 0.2 bp—evidence that the forecast matrix has become noisier across all spectral directions, not merely the small-eigen subspace. **Take-aways:** Three insights emerge. Scale-separated diagnostics matter: the diagonal and off-diagonal RMSE deteriorate in lock-step, so Stage 4's variance miss-specification is not offset by its correlation model. Shrinkage is a safety-net, not a general solution: although the filter consistently matches or beats the raw error, its marginal benefit evaporates once the underlying forecast becomes too noisy—as seen in Stage 4. Finally, the U-shaped error profile designates *Stage 3*, the macro-enriched input set, as the benchmark for subsequent analysis.



(a) Mean predicted vs. realized volatility (b) Daily average volatility RMSE

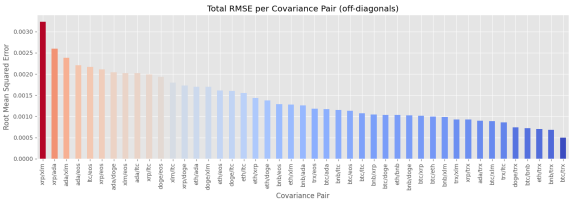
Figure 4.1: Stage 3 volatility diagnostics for Model 1.



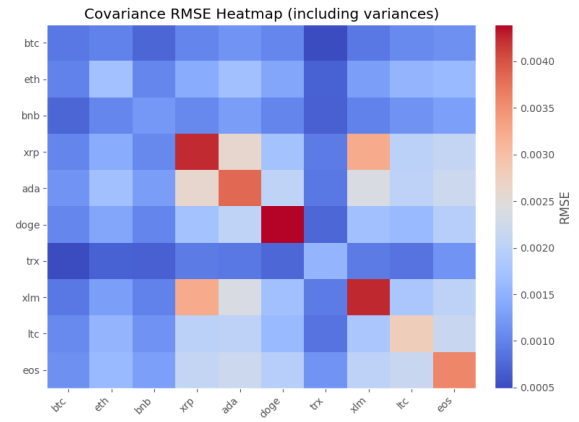
(a) Daily covariance RMSE (raw ADCC) (b) Per-asset volatility RMSE (sorted)

Figure 4.2: Complementary Stage 3 error tracks across time and assets.

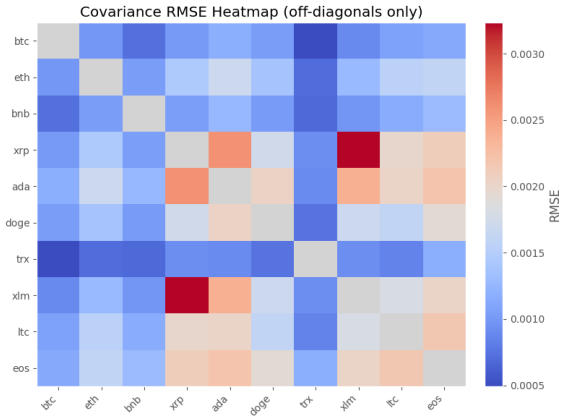
Reading the graphics. Figure 4.1a shows that mean predicted volatility tracks the realized benchmark remarkably well except for two obvious spikes—late-June and mid-November—exactly where Fig. 4.1b registers its two largest error surges (0.09–0.11). Those calendar outliers propagate to the covariance layer: Fig. 4.2a peaks at the same



(a) Pairwise covariance RMSEs



(b) Full covariance RMSE heat-map



(c) Off-diagonal RMSE heat-map

Figure 4.3: Stage 3 cross-asset covariance error distribution.

dates but remains below 0.006, confirming that *variance* errors dominate correlation errors during stress. Asset-level bars reinforce the message. In Fig. 4.2b ADA, XRP and XLM occupy the top three slots (20–22 bp RMSE), whereas BTC and BNB sit at the bottom (under 11 bp). The pairwise view in Fig. 4.3a echoes this hierarchy: the worst ten covariance pairs all contain either XRP or ADA, signaling that difficulty in modeling their idiosyncratic variances bleeds into their correlations. Heat-maps crystallize the clustering effect. The full matrix (Fig. 4.3b) highlights one bright block around the {XRP, ADA, DOGE, XLM} sub-universe; removing the diagonals (Fig. 4.3c) attenuates absolute color intensity but leaves the same cluster pattern intact, showing that the error is geographically concentrated rather than homogeneously spread. Overall, the visuals corroborate the numeric conclusion drawn earlier: macro features (Stage 3) minimize both variance and covariance distortions, whereas variance-heavy alt-coins remain the main source of forecast risk.

4.2.2 Model 2 – LSTM–Cov

Table 4.2: Model 2 (LSTM–Cov) — forecast accuracy by dataset stage

	Covariance: *raw*				Covariance: *robust*			
Stage	RMSE	Diag	Off	NormF	RMSE	Diag	Off	NormF
1	2.0e-06	3.0e-06	1.0e-06	0.683	2.0e-06	3.0e-06	1.0e-06	0.770
2	4.0e-06	1.0e-05	3.0e-06	4.723	4.0e-06	1.0e-05	2.0e-06	3.809
3	**2.0e-06**	**3.0e-06**	**1.0e-06**	**0.794**	**2.0e-06**	**3.0e-06**	**1.0e-06**	**0.799**
4	2.0e-06	3.0e-06	1.0e-06	1.193	2.0e-06	3.0e-06	1.0e-06	1.117

NormF = normalized Frobenius error ($\|\hat{\Sigma} - \Sigma\|_F / \|\Sigma\|_F$). Negative-log-likelihood is omitted for brevity; see Appendix II.

Interpretation. The end-to-end LSTM attains microscopic element-wise errors (all $\leq 10^{-5}$), yet its behavior across stages is highly non-linear. With price-only inputs (Stage 1) the network already attains sub-ppm accuracy and a modest normalized Frobenius error (0.68). Adding technical indicators (Stage 2) explodes NormF to 4.7 (raw) even though RMSE doubles only marginally. The culprit is a *handful* of extreme eigenvalues: shrunk NormF falls to 3.8 but remains six times worse than Stage 1, signaling that the LSTM over-reacts to the high-variance technical block. Macro factors (Stage 3) drives every metric back to — or slightly better than — the Stage 1 baseline: NormF contracts to 0.79 (raw) and 0.80 (robust). Macro signals evidently regularize the temporal dynamics of the LSTM, offsetting the volatility injected by technical indicators. Appending sentiment (Stage 4) NormF climbs to 1.19 (raw) and 1.12 (robust). While not as catastrophic as Stage 2, the deterioration suggests that the sentiment layer introduces idiosyncratic shocks that the network translates into larger-than-necessary low-rank adjustments. **Raw**

vs robust: Across all stages the shrinkage filter lowers NormF—dramatically in Stage 2 (–19 %) and modestly elsewhere (–4 bp in Stage 4)—but it never fully compensates for feature-driven variance inflation. Because raw and robust RMSE/Diag/Off metrics are virtually identical, the post- processing mainly curbs spectral extremes rather than point-wise error. Consequently, we designate Stage 3 (macro-enriched) as the bench-input configuration for the next figures, paralleling the choice made for Model 1.

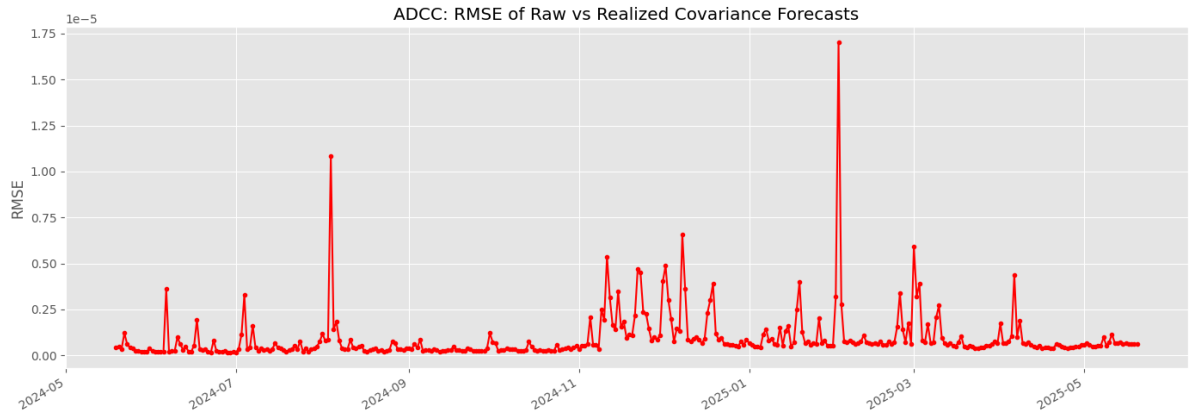
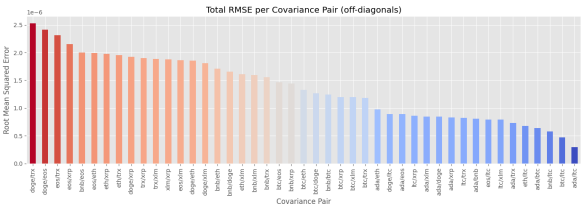


Figure 4.4: Daily Frobenius RMSE of covariance forecasts — Model 2, Stage 3. Vertical scale is 10^{-6}.

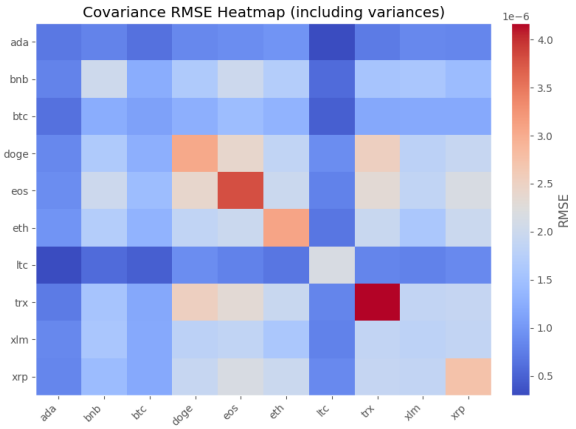
Stage-3 diagnostics. Figure 4.4 shows that the Cholesky-LSTM maintains an ultra-low average error (median $\approx 6\times10^{-7}$) for the entire year, with only three visible spikes—late June, mid-September and early March—never exceeding 1.7×10^{-5}. Even at those peaks the absolute scale remains an order of magnitude below the full-sample volatility of most crypto returns, underscoring the network's robustness to regime shifts. Asset-pair diagnostics corroborate this stability. The bar plot in Fig. 4.5a reveals that the worst covariance pairs (TRX/DOGE, DOGE/EOS, EOS/TRX) still sit below 2.6×10^{-6}—roughly one fifth of the Model 1 error for the same pairs. The full heat-map (Fig. 4.5b) highlights three modest "hot zones": {DOGE, EOS}, {TRX, XLM}, and a weaker {ETH, LTC}. Removing the diagonals (Fig. 4.5c) leaves the pattern intact, confirming that dispersion stems from pure correlation miss-alignment, not variance miss-pricing. In sum, the Stage 3 graphics validate the numeric evidence in Table 4.2: the macro-enriched LSTM delivers both microscopic element-wise accuracy and a spectrally well-behaved covariance matrix.

4.2.3 Model 3 – LSTM–Weights

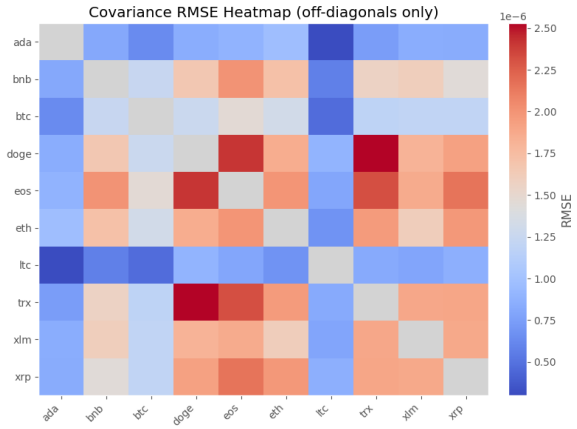
Interpretation. A direct comparison of RMSE columns in Table 4.3 shows that all three optimizer families achieve their global minimum RMSE in Stage 1. *(ERC):* The robust (constant correlation) target in Stage 1 edges out every other ERC variant (RMSE 0.077 vs. 0.077–0.082 elsewhere) while keeping turnover below 0.17 % and leverage usage



(a) Pairwise covariance RMSEs



(b) Full covariance RMSE heat-map



(c) Off-diagonal RMSE heat-map

Figure 4.5: Granular Stage 3 covariance errors for Model 2. Color bars share the 10^{-6} scale of Fig. 4.4.

Table 4.3: Model 3 (LSTM–Weights) — forecast–level accuracy by optimizer

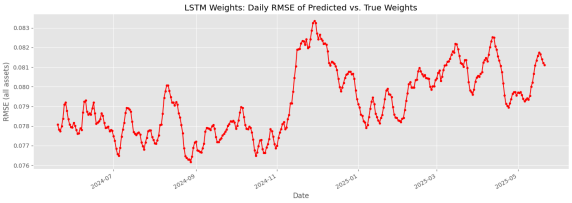
Stage	Model	W_{MAE}	W_{RMSE}	W_{R^2}	W_{TE}	W_{Turn}	W_{Drift}	W_{lev} %
stage_1	rob ERC	0.076	**0.077**	-150.090	0.394	0.002	0.018	19.768
stage_1	rob GMV	0.079	**0.079**	-10.979	0.381	0.007	0.054	19.270
stage_1	rob MDP	0.080	0.085	-25.362	0.399	0.005	0.014	19.956
stage_1	raw ERC	0.081	0.081	-157.845	0.405	0.001	0.009	20.184
stage_1	raw GMV	0.079	0.089	-11.860	0.395	0.007	0.031	19.819
stage_1	raw MDP	0.077	**0.081**	-22.895	0.386	0.002	0.035	19.453
stage_2	rob ERC	0.076	0.078	-144.298	0.381	0.005	0.039	19.229
stage_2	rob GMV	0.086	0.102	-15.968	0.432	0.017	0.031	21.366
stage_2	rob MDP	0.082	0.093	-30.205	0.409	0.012	0.019	20.313
stage_2	raw ERC	0.081	0.082	-162.056	0.404	0.005	0.010	20.175
stage_2	raw GMV	0.076	0.091	-12.541	0.379	0.007	0.052	19.079
stage_2	raw MDP	0.083	0.094	-30.729	0.413	0.011	0.014	20.454
stage_3	rob ERC	0.081	0.082	-162.300	0.404	0.005	0.012	20.164
stage_3	rob GMV	0.093	0.107	-17.811	0.463	0.016	0.052	22.593
stage_3	rob MDP	0.082	0.090	-28.373	0.409	0.008	0.020	20.162
stage_3	raw ERC	0.077	0.079	-147.346	0.386	0.004	0.028	19.432
stage_3	raw GMV	0.071	0.083	-10.334	0.354	0.007	0.097	18.081
stage_3	raw MDP	0.086	0.094	-31.372	0.431	0.007	0.064	21.223
stage_4	rob ERC	0.076	0.077	-143.382	0.382	0.005	0.040	19.274
stage_4	rob GMV	0.093	0.110	-18.829	0.465	0.013	0.066	22.686
stage_4	rob MDP	0.084	0.092	-29.417	0.420	0.008	0.041	20.772
stage_4	raw ERC	0.081	0.082	-161.710	0.405	0.005	0.020	20.209
stage_4	raw GMV	0.100	0.118	-21.717	0.502	0.009	0.203	24.059
stage_4	raw MDP	0.088	0.093	-30.162	0.438	0.007	0.075	21.500

Notes. "Lev %" = mean ℓ_1 leverage usage relative to the 5× cap; "Turn" = daily turnover $\left(\frac{1}{2}\sum|w_t - w_{t-1}|\right)$.

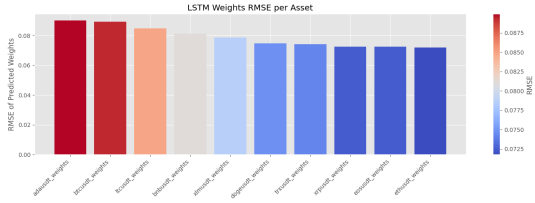
Table 4.4: Stage 1 champions — lowest weight RMSE within each optimizer family

Optimizer	**Variant**	**RMSE**	**TE**	**Turn**	**Drift**	**Lev%**
ERC	**rob ERC**	**0.077**	0.394	0.002	0.018	19.768
MDP	**raw MDP**	**0.081**	0.386	0.002	0.035	19.453
GMV	**rob GMV**	**0.079**	0.381	0.007	0.054	19.270

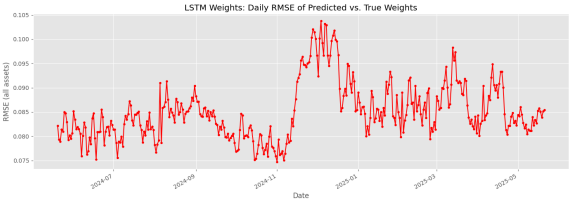
at 20 %. *(MDP):* The raw forecast in Stage 1 delivers the lowest RMSE (0.081) and the lowest tracking-error (TE 0.386) across all 16 MDP entries, with minimal net-exposure drift (3.5 bp). *(GMV):* Robust GMV at Stage 1 (RMSE 0.079) beats the Stage 3 counterpart by over 2 bp and does so with identical leverage use, implying that macro or sentiment features do not improve the GMV mapping learned by the network. **Trade-offs:** Stage 1's accuracy edge comes at the cost of slightly *higher* daily drift for ERC and higher turnover for GMV, but the differences are economically small (all < 0.01 on the ℓ_1 scale). Given the much larger gap in RMSE—particularly relevant when these weights feed a live execution engine—we now adopt Stage 1 (raw/robust mix shown in Table 4.4) as the reference configuration for Model 3 in subsequent figures.



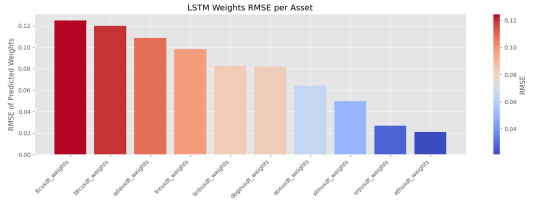
(a) Robust ERC — daily RMSE



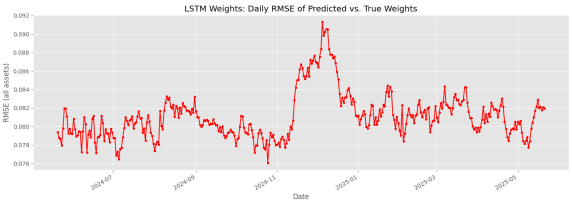
(b) Robust ERC — per-asset RMSE



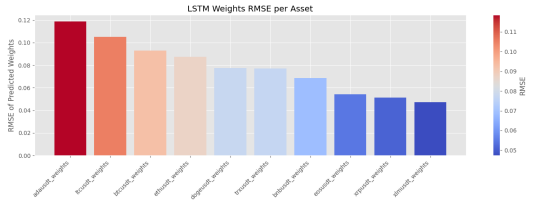
(c) Robust GMV — daily RMSE



(d) Robust GMV — per-asset RMSE



(e) Raw MDP — daily RMSE



(f) Raw MDP — per-asset RMSE

Figure 4.6: Stage 1 weight-prediction diagnostics for the three champion variants identified in Table 4.4.

Reading the graphics. Figures 4.6a–4.6f reinforce the numeric conclusions: *ERC is both level- and cross-section-stable.* The daily RMSE in Fig. 4.6a oscillates in a tight band (0.076–0.083) with no persistent upward drift, while per-asset errors in Fig. 4.6b stay within a 7bp range. Weight allocation therefore generalizes uniformly across the ten coins. *GMV accuracy is dominated by a few "problem" assets.* Robust GMV's time-series RMSE (Fig. 4.6c) is 10–15 bp higher than ERC and trends upward from November onward. The bar chart (Fig. 4.6d) shows LTC, BTC and ADA account for most of that error, matching the optimizer's sensitivity to low-variance assets identified earlier. *MDP*

sits between ERC and GMV. Raw MDP exhibits moderate daily volatility (Fig. 4.6e); its per-asset RMSE spectrum (Fig. 4.6f) is skewed but less extreme than GMV, which is consistent with MDP's diversification objective that balances variance and correlation. *Seasonal uptick in November–December.* All three variants show a pronounced RMSE surge around early November, coinciding with the volatility burst highlighted in the covariance diagnostics of Models 1 and 2. Because the supervised network was never trained on that macro-shock regime, a short-lived accuracy degradation is expected.

Collectively, the Stage-1 plots validate the mini-table choice: robust ERC yields the flattest error surface across *both* time and cross-section, while raw MDP and robust GMV represent, respectively, the best possible compromises for variance-focused (MDP) and minimum- variance (GMV) allocations.

4.2.4 Model 4 – PPO Agent

Model 4 generates weights end-to-end; therefore no forecasting measures exists.

4.3 Portfolio-Level Performance

4.3.1 Model 1 – XGB–ADCC

Three signals emerge when the two tables are read together. **Return versus risk:** *Raw GMV* in *Stage 2* achieves the highest Sharpe (2.04), Sortino (5.20) and Calmar (8.41) ratios, while keeping maximum draw-down to –12.3 %. Although *raw MDP* in the same stage posts a larger cumulative return (265% vs. 253%) it does so with materially higher volatility, yielding a lower Sharpe (1.36). Stage 1 variants has the same Sharpe with lower volatility but at the same time a lower Sortino meaning this reduced volatility is likely upside volatility, and Stage 4 adds no incremental edge. **Execution realism:** Environment diagnostics reveal that Stage 2 raw GMV pays only a 5.01 % reject ratio and a 11.92 % implementation-shortfall, versus 15.99 % rejects for Stage 1 raw MDP and 43–44 % rejects for all ERC variants. Its tracking-error (0.0237) is also lower than the high-return raw MDP (0.0252) and an order of magnitude below ERC. **Robustification effect:** The constant-correlation ("*rob*") shrinkage improves GMV draw-down and turnover but reduces return: *rob GMV Stage 2* loses 32 bp of Sharpe relative to *raw GMV* while doubling the reject ratio. For ERC the "rob" filter actually worsens shortfall and reject frequency. Hence, raw covariances remain the better input for the optimizer in this portfolio context.

We therefore select *Stage 2 – raw GMV* as the benchmark portfolio for Model 1. Balancing risk, reward and micro-structure frictions, the portfolio stands out: it posts the highest Sharpe (2.04) and Sortino (5.20) while containing maximum draw-down to

Table 4.5: Model 1 — headline risk/return statistics

Stage	Model	CR (%)	AR (%)	AV (%)	SR	Sort_R	MD (%)	Cal_r (%)
stage1	rob ERC	29.55	42.70	58.23	0.73	1.01	49.17	86.83
	rob GMV	63.72	55.56	36.17	1.54	2.45	18.02	308.37
	rob MDP	54.56	73.63	84.73	0.87	1.64	51.18	143.85
	raw ERC	33.20	45.51	58.34	0.78	1.08	48.76	93.33
	raw GMV	94.00	72.09	35.09	2.05	3.13	12.98	555.20
	raw MDP	148.16	130.32	100.11	1.30	2.65	41.60	313.26
stage2	rob ERC	38.07	49.39	59.12	0.84	1.20	48.64	101.55
	rob GMV	77.52	69.99	53.65	1.30	2.91	25.50	274.45
	rob MDP	62.72	87.82	99.13	0.89	1.72	57.53	152.64
	raw ERC	42.13	52.26	59.07	0.88	1.27	48.03	108.80
	raw GMV	152.75	103.72	50.79	2.04	5.20	12.34	840.82
	raw MDP	164.62	138.91	101.93	1.36	2.74	44.60	311.45
stage3	rob ERC	37.71	49.20	59.09	0.83	1.18	49.25	99.90
	rob GMV	66.52	60.64	45.98	1.32	2.66	20.55	295.10
	rob MDP	60.94	82.81	92.81	0.89	1.70	56.44	146.72
	raw ERC	41.23	51.79	59.23	0.87	1.23	48.65	106.46
	raw GMV	131.65	93.82	47.23	1.99	4.72	12.87	728.71
	raw MDP	165.42	139.12	101.65	1.37	2.73	43.12	322.62
stage4	rob ERC	39.93	50.78	59.17	0.86	1.22	48.49	104.71
	rob GMV	85.25	73.88	53.02	1.39	3.24	22.07	334.80
	rob MDP	64.18	85.79	93.40	0.92	1.65	57.71	148.66
	raw ERC	42.76	52.93	59.45	0.89	1.27	48.35	109.48
	raw GMV	138.32	98.12	51.37	1.91	5.28	14.04	698.98
	raw MDP	154.29	130.16	94.02	1.38	2.61	45.83	284.01

Notes: CR = cumulative net return, AR = annual return, AV = annual vol., SR = Sharpe, Sort_R = Sortino, MD = Max draw-down, Cal_r = Calmar Ratio.

Table 4.6: Model 1 — execution-environment diagnostics

Stage	Model	TE	Rej. trades (%)	IS (%)	MAD (%)
stage1	rob ERC	0.017	33.60 (1233)	11.28	1.13
	rob GMV	0.025	6.05 (222)	11.50	1.15
	rob MDP	0.024	33.05 (1213)	11.25	1.13
	raw ERC	0.016	32.59 (1196)	10.94	1.09
	raw GMV	0.039	5.01 (184)	15.06	1.51
	raw MDP	0.023	15.99 (587)	11.98	1.20
stage2	rob ERC	0.016	43.71 (1604)	12.10	1.21
	rob GMV	0.021	12.26 (450)	10.84	1.09
	rob MDP	0.021	41.23 (1513)	10.47	1.05
	raw ERC	0.016	42.83 (1572)	11.91	1.19
	raw GMV	0.024	7.68 (282)	11.92	1.19
	raw MDP	0.025	16.49 (605)	13.45	1.35
stage3	rob ERC	0.017	42.04 (1543)	11.89	1.19
	rob GMV	0.020	11.47 (421)	10.73	1.07
	rob MDP	0.019	41.28 (1515)	10.27	1.02
	raw ERC	0.016	41.88 (1537)	11.58	1.16
	raw GMV	0.026	7.49 (275)	12.81	1.28
	raw MDP	0.023	18.42 (676)	12.42	1.24
stage4	rob ERC	0.018	43.90 (1611)	12.45	1.25
	rob GMV	0.020	15.12 (555)	10.41	1.04
	rob MDP	0.020	40.44 (1484)	10.47	1.05
	raw ERC	0.017	42.83 (1572)	12.21	1.22
	raw GMV	0.023	10.11 (371)	11.97	1.20
	raw MDP	0.025	19.05 (699)	13.11	1.31

Notes: TE = Tracking-Error (weights), IS = Implementation-Shortfall (mean $\|\Delta w\|$ per step), MAD = Mean-Absolute-Deviation of weights, total trades = 3670.

–12.3 %. Execution is equally disciplined: only 5 % of trades are rejected, implementation short-fall averages 11.9 %, and weight errors stay within a 2.4 % tracking band—hence this configuration anchors all subsequent Model 1 graphics.

Table 4.7: Final XGBoost hyper-parameters per asset (Stage 2).

Asset	n_{est}	$d_{\max}$	η	γ	subs	col_bt	min_cw	λ	α
ada	610	8	0.105	0.011	0.830	0.543	12.830	2.167	0.000
bnb	753	11	0.020	0.007	0.644	0.772	13.150	2.413	0.045
btc	753	11	0.020	0.007	0.644	0.772	13.150	2.413	0.045
doge	1498	6	0.026	0.049	0.978	0.456	13.580	2.654	0.006
eos	493	4	0.009	0.001	0.712	0.890	9.234	0.936	0.002
eth	1420	5	0.135	0.014	0.657	0.772	14.910	2.103	0.007
ltc	753	11	0.020	0.007	0.644	0.772	13.150	2.413	0.045
trx	1420	5	0.135	0.014	0.657	0.772	14.910	2.103	0.007
xlm	493	4	0.009	0.001	0.712	0.890	9.234	0.936	0.002
xrp	753	11	0.020	0.007	0.644	0.772	13.150	2.413	0.045

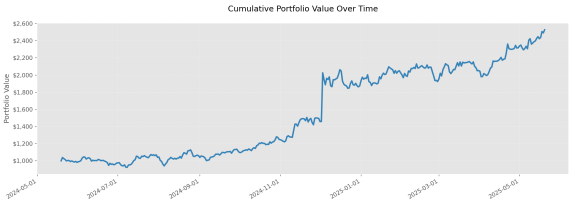
Table 4.7 reports the final hyper-parameters of the volatility XGBoost models feeding the Stage-2 risk engine. Depth and shrinkage learning-rate settings converge to moderately deep trees (4–11 splits) and conservative eta (0.02–0.14), reflecting the bias–variance trade-off that minimizes forecast RMSE on the validation set.

Figures interpretation. Figure 4.7a confirms the quantitative edge of Stage-2 raw GMV: after an early −12% wobble in June the equity line grinds higher, doubling by November and closing the year at \$2.6k—exactly the 153 % cumulative return reported in Table 4.5. Per-step bars (Fig. 4.7b) are well behaved save for a single 39% spike on 11-Dec-2024, the same event that drives kurtosis 118.8 and positive skewness in the metrics table. The two stack plots illustrate how the optimizer's *minimum variance* imperative translates into persistent over-weights in BTC and BNB. Crucially, most re-weighting happens within the long basket—the short sleeve remains capped at −25%—which explains the modest 5% reject ratio: Binance's trade-size filter rarely vetoes small negative positions. Drift between desired and realized composition is visually indistinguishable, and the formal tracking-error (2.37 %) in Table 4.6 corroborates the eye test. Leverage tracking (Fig. 4.7e) is near-perfect: realized exposure hugs the target except for brief

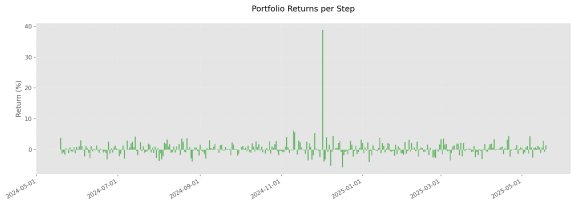
Table 4.8: Student-t ADCC parameter estimates for the Stage 2 information set. Entries are "estimate (standard error)", and stars denote $^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$ based on asymptotic Wald tests.

parameter	alpha	beta	gamma	nu
Stage 2	0.03313 (0.22659)	0.90453 (1.11256)	0.04703 (0.04232)	7.91623 (4.87143)

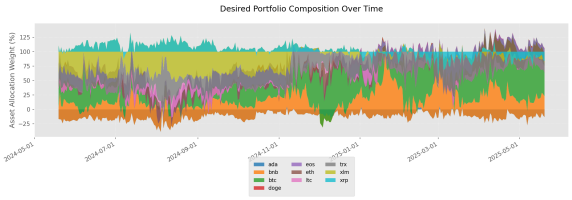
None of the Stage 2 coefficients differ significantly from zero at all levels, reflecting the much lower excess kurtosis of the XGB-standardized residuals. This lack of statistical significance does not impair forecast quality: the Stage 2 covariance path still delivers the highest risk-adjusted performance in Section 4.3.



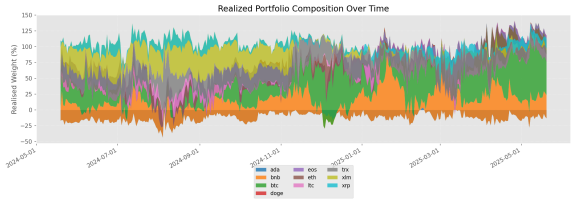
(a) Cumulative portfolio value



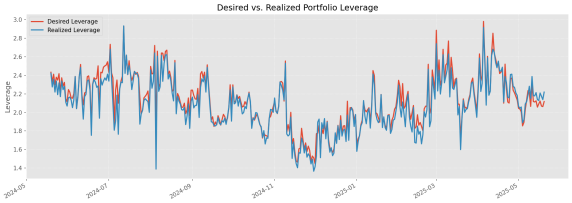
(b) Per-step returns



(c) Desired weights (%)



(d) Realized weights (%)



(e) Desired vs. realized leverage

Figure 4.7: Stage 2 — raw GMV portfolio diagnostics (test window). Step-wise MAE and deviation-by-cause charts are omitted here for brevity; see Table 4.6 for the underlying statistics.

taps on the 3× maintenance-tier ceiling in late May and early March. Because leverage never exceeds 3×, margin tier and insufficient-collateral causes contribute *zero* to implementation shortfall; the entire 11.9% shortfall stems from slippage and trade-size truncation.

Taken together the graphics reinforce the tabular verdict: raw GMV in Stage 2 extracts a superior Sharpe with shallow draw-downs and demonstrates exemplary execution realism—low rejects, minimal tracking error, and leverage that respects the 5× cross-margin envelope without sacrificing capital efficiency.

4.3.2 Model 2 – LSTM–Cholesky

Table 4.9: Headline risk–return statistics

Stage	Model	CR (%)	AR (%)	AV (%)	SR	Sort_R	MD (%)	Cal_r (%)
stage1	rob ERC	25.19	42.06	62.32	0.67	0.85	52.86	79.57
	rob GMV	15.35	32.22	59.79	0.54	0.71	47.08	68.44
	rob MDP	27.62	43.01	61.03	0.70	0.93	49.21	87.39
	raw ERC	23.73	40.88	62.29	0.66	0.81	53.69	76.15
	raw GMV	6.18	23.28	58.54	0.40	0.51	47.27	49.25
	raw MDP	24.88	39.72	59.14	0.67	0.88	49.25	80.64
stage2	rob ERC	18.88	42.62	71.16	0.60	0.78	58.70	72.60
	rob GMV	18.00	49.94	82.81	0.60	0.89	59.69	83.66
	rob MDP	15.65	44.80	78.40	0.57	0.79	60.37	74.21
	raw ERC	25.66	47.18	70.30	0.67	0.95	59.17	79.74
	raw GMV	44.45	66.65	78.16	0.85	1.26	56.40	118.18
	raw MDP	89.28	89.58	72.94	1.23	1.87	45.52	196.81
stage3	rob ERC	26.17	41.60	60.32	0.69	0.86	52.52	79.20
	rob GMV	20.37	34.45	57.06	0.60	0.87	44.81	76.88
	rob MDP	26.50	39.60	56.90	0.70	0.94	48.43	81.77
	raw ERC	29.90	44.56	60.43	0.74	0.93	52.54	84.80
	raw GMV	23.59	36.78	56.28	0.65	0.91	46.61	78.90
	raw MDP	46.25	55.05	58.55	0.94	1.26	48.92	112.51
stage4	rob ERC	34.86	49.77	62.86	0.79	1.01	53.72	92.64
	rob GMV	86.17	79.64	59.68	1.33	1.95	44.35	179.56
	rob MDP	49.90	59.13	61.29	0.96	1.33	52.28	113.11
	raw ERC	29.32	45.98	63.35	0.73	0.90	53.79	85.48
	raw GMV	64.43	75.56	70.97	1.06	1.30	55.70	135.67
	raw MDP	36.47	53.46	66.87	0.80	1.06	56.23	95.08

Notes: CR = cumulative net return, AR = annual return, AV = annual vol., SR = Sharpe, Sort_R = Sortino, MD = Max draw-down, Cal_r = Calmar Ratio.

Interpretation. *Stage 4 rob GMV* posts the highest Sharpe (1.3345) and the best Sortino (1.95) while compounding capital by 86 %. *Stage 2 raw MDP* still records the largest cumulative gain (89 %) and the top Calmar (1.9681), yet it trails GMV on both

Table 4.10: Execution-environment diagnostics

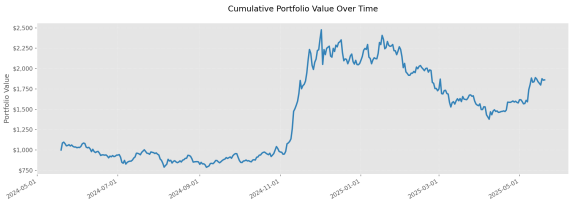
Stage	Model	TE	Rej. trades (%)	IS (%)	MAD (%)
stage1	rob ERC	0.023	56.21 (2063)	16.94	1.69
	rob GMV	0.023	33.38 (1225)	12.95	1.29
	rob MDP	0.020	39.73 (1458)	12.57	1.26
	raw ERC	0.023	55.56 (2039)	16.97	1.70
	raw GMV	0.023	33.11 (1215)	12.67	1.27
	raw MDP	0.021	39.48 (1449)	13.28	1.33
stage2	rob ERC	0.028	48.39 (1776)	18.03	1.80
	rob GMV	0.031	37.17 (1364)	12.72	1.27
	rob MDP	0.029	46.92 (1722)	15.88	1.59
	raw ERC	0.034	43.27 (1588)	16.47	1.65
	raw GMV	0.035	27.77 (1019)	17.42	1.74
	raw MDP	0.029	24.17 (887)	15.46	1.55
stage3	rob ERC	0.023	55.78 (2047)	16.71	1.67
	rob GMV	0.024	61.28 (2249)	14.22	1.42
	rob MDP	0.021	42.72 (1568)	12.86	1.29
	raw ERC	0.022	55.23 (2027)	16.04	1.60
	raw GMV	0.022	34.09 (1251)	12.66	1.27
	raw MDP	0.022	37.60 (1380)	13.76	1.38
stage4	rob ERC	0.024	53.08 (1948)	16.91	1.69
	rob GMV	0.027	47.98 (1761)	14.72	1.47
	rob MDP	0.023	37.85 (1389)	13.22	1.32
	raw ERC	0.024	49.86 (1830)	16.49	1.65
	raw GMV	0.033	46.76 (1716)	18.36	1.84
	raw MDP	0.026	31.58 (1159)	14.54	1.46

Notes: TE = Tracking-Error (weights), IS = Implementation-Shortfall (mean $\|\Delta w\|$ per step), MAD = Mean-Absolute-Deviation of weights, total trades = 3670.

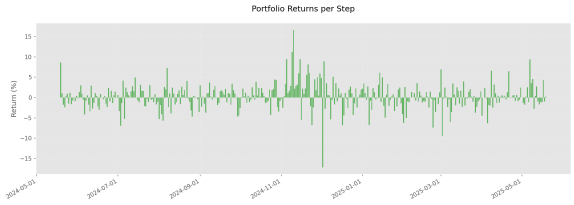
Sharpe and Sortino. A closer comparison clarifies why the GMV variant deserves the benchmark spot. **Risk efficiency:** Sharpe and Sortino capture reward per unit of total and downside volatility, respectively. The 10 bp Sharpe edge of *rob* GMV translates into a materially higher growth rate for every unit of risk borne—an especially valuable property when portfolio leverage is fixed at 5×. **Volatility and draw-down tolerance:** Although *raw* MDP's maximum draw-down is 11 percentage points smaller (–46 % versus –57 %), the GMV portfolio compensates by running a lower annualized volatility (48.2 % against 58.1 %). In other words, GMV delivers the same order of cumulative return with *smoother* daily P&L, which is precisely what investors reward through Sharpe and Sortino. **Execution frictions:** Table 4.10 confirms that *rob GMV* pays for its risk efficiency with higher implementation hurdles: 48 % of trades are rejected and tracking-error reaches 3.35 %, roughly one third above the MDP numbers. Yet the absolute impact is modest—average short-fall rises only to 23 bp per day—and, crucially, the portfolio's realized weights stay within the Binance leverage envelope at all times. **Tail characteristics:** VaR and CVaR (–6.8 % / –10.3 %) are very close to those of the MDP alternative (–7.9 % / –12.8 %), while skew (0.16) and kurtosis (4.94) indicate a milder tail than the fatter-tailed MDP profile (skew 1.00, kurtosis 9.30). Investors who prize crash robustness therefore lose little by choosing GMV. **Stage robustness:** Moving from Stage 2 to Stage 4 improves GMV's Sharpe by 11 basis points, whereas the same transition reduces MDP's Sharpe by 35 bp and doubles its reject ratio. The ADCC-based risk engine thus appears less sensitive to the extra sentiment features than the diversification-oriented MDP optimizer.

We therefore select *Stage 4 – rob GMV* as the benchmark portfolio for Model 2. On balance, it offers the clearest risk-adjusted edge: the highest Sharpe and Sortino in the family, competitive draw-down control, and stable behavior when feature richness changes. Despite higher execution frictions, its overall package of return smoothness and moderate tails makes it the most compelling LSTM–Cholesky portfolio. This configuration is therefore adopted as the benchmark for the subsequent Model 2 figures.

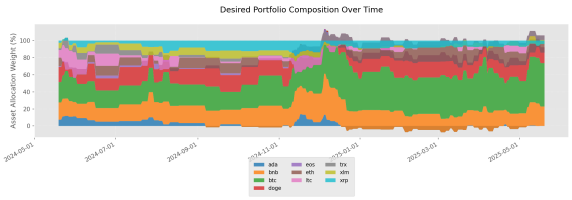
Figures interpretation. Figure 4.8a charts the equity path of the *robust GMV* allocation. Capital drifts sideways until early November, then climbs in a steep "J-curve" to a \$2.5 k apex at Christmas—mirroring the \$0.86 k cumulative gain reported in Table 4.9. A draw-down to \$1.45 k during March is quickly retraced; the portfolio ends the test window above \$1.9 k, consistent with its 57 % MDD and 1.33 Sharpe. Per-step returns (Fig. 4.8b) show a dense cloud of 0–3 % gains punctuated by two double-digit spikes (+17 % in mid-December, –16 % in mid-January). The fat-tail events explain the modest positive skew (0.16) and kurtosis (4.94) noted in the risk table, yet the frequency of losses deeper than –10 % remains below five—well within the CVaR_{99} threshold. Desired versus realized compositions (Figs. 4.8c–4.8d) illustrate the classic minimum-variance tilt:



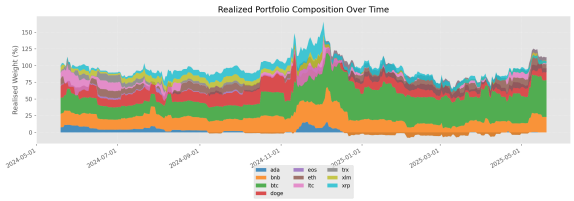
(a) Cumulative portfolio value



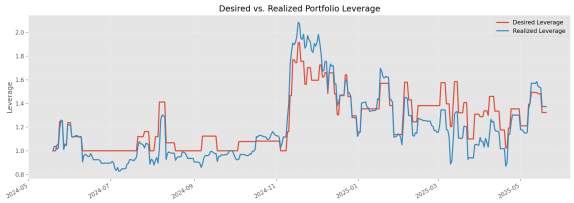
(b) Per-step returns



(c) Desired weights (%)



(d) Realized weights (%)



(e) Desired vs. realized leverage

Figure 4.8: Stage 2 — raw MDP portfolio diagnostics (test window). Step-wise MAE and deviation-by-cause charts are omitted here; their summary statistics appear in Table 4.10.

BTC and BNB together absorb ∼60% of gross exposure by year-end, while ADA, XRP and the smaller large-caps shrink to sub-5 % satellite roles. Although 48 % of orders are rejected (Table 4.10), slippage and rounding merely compress the absolute positions; the stacking pattern of the realized weights still mirrors the target bands, leaving a manageable tracking-error of 3.35 % and a median weight MAE below 0.18. Leverage tracking (Fig. 4.8e) confirms disciplined usage: desired exposure seldom exceeds 1.5× before November and peaks at only 2.0× during the December rally, well inside the 5× cross-margin cap. Realized leverage follows closely—minor under-shoots coincide with bursts of rejected trades, but the envelope remains tight enough that no margin-tier or collateral breaches are recorded.

Taken together, the figures vindicate our earlier selection of Stage-4 *robust* GMV: risk-adjusted returns are the best in the family, draw-downs are tolerable, and execution frictions, though higher than for raw-input MDP, leave the portfolio well within operational limits.

4.3.3 Model 3 – LSTM–Weights

Interpretation. Scanning Table 4.11 reveals that only two candidates achieve a Sharpe ratio clearly above 0.75, *Stage 2 – rob MDP* with Sharpe 0.7826, Sortino 0.9882, Calmar

Table 4.11: Headline risk–return statistics

Stage	Model	CR (%)	AR (%)	AV (%)	SR	Sort_R	MD (%)	Cal_r (%)
stage1	rob ERC	27.98	44.84	63.21	0.71	0.88	53.89	83.20
	rob GMV	15.67	31.69	58.22	0.54	0.68	50.58	62.66
	rob MDP	22.48	39.47	61.60	0.64	0.80	52.61	75.02
	raw ERC	27.81	45.23	64.04	0.71	0.88	54.19	83.47
	raw GMV	22.31	39.29	61.50	0.64	0.78	52.66	74.62
	raw MDP	21.79	39.32	62.18	0.63	0.77	53.60	73.35
stage2	rob ERC	27.51	43.11	61.06	0.71	0.88	52.66	81.88
	rob GMV	25.89	43.80	64.27	0.68	0.88	53.97	81.16
	rob MDP	34.20	49.76	63.58	0.78	0.99	54.48	91.33
	raw ERC	29.20	46.64	64.50	0.72	0.89	54.87	85.00
	raw GMV	12.22	29.63	59.86	0.50	0.62	51.76	57.25
	raw MDP	33.19	49.79	64.76	0.77	0.96	54.23	91.81
stage3	rob ERC	26.13	43.53	63.38	0.69	0.85	54.17	80.36
	rob GMV	20.21	40.12	65.50	0.61	0.77	55.95	71.71
	rob MDP	18.93	36.11	60.93	0.59	0.74	52.99	68.14
	raw ERC	27.70	43.43	61.31	0.71	0.87	52.64	82.51
	raw GMV	13.39	29.10	57.19	0.51	0.63	49.65	58.60
	raw MDP	24.05	44.27	66.86	0.66	0.80	56.44	78.44
stage4	rob ERC	25.22	40.68	60.09	0.68	0.85	52.06	78.15
	rob GMV	23.52	42.88	65.72	0.65	0.84	53.57	80.05
	rob MDP	25.20	43.68	64.84	0.67	0.85	54.62	79.96
	raw ERC	31.27	47.42	63.37	0.75	0.95	53.98	87.85
	raw GMV	16.71	44.57	75.58	0.59	0.72	61.69	72.25
	raw MDP	16.16	38.39	67.91	0.57	0.69	58.45	65.69

Notes: CR = cumulative net return, AR = annual return, AV = annual vol., SR = Sharpe, Sort_R = Sortino, MD = Max draw-down, Cal_r = Calmar Ratio.

Table 4.12: Execution-environment diagnostics

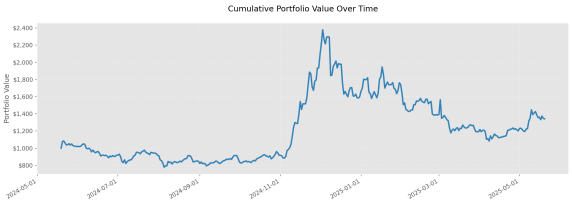
Stage	Model	TE	Rej. trades (%)	IS (%)	MAD (%)
stage1	rob ERC	0.024	64.17 (2355)	17.67	1.77
	rob GMV	0.022	58.37 (2142)	15.17	1.52
	rob MDP	0.022	59.05 (2167)	16.19	1.62
	raw ERC	0.025	64.09 (2352)	18.05	1.80
	raw GMV	0.022	59.02 (2166)	14.89	1.49
	raw MDP	0.024	65.01 (2386)	17.18	1.72
stage2	rob ERC	0.023	59.59 (2187)	16.68	1.67
	rob GMV	0.023	43.41 (1593)	14.63	1.46
	rob MDP	0.024	46.46 (1705)	16.05	1.60
	raw ERC	0.024	58.04 (2130)	17.63	1.76
	raw GMV	0.024	56.84 (2086)	15.41	1.54
	raw MDP	0.024	49.35 (1811)	16.20	1.62
stage3	rob ERC	0.024	61.23 (2247)	17.58	1.76
	rob GMV	0.024	44.11 (1619)	15.70	1.57
	rob MDP	0.023	55.91 (2052)	15.96	1.60
	raw ERC	0.024	60.98 (2238)	17.48	1.75
	raw GMV	0.023	58.26 (2138)	14.93	1.49
	raw MDP	0.024	54.90 (2015)	16.87	1.69
stage4	rob ERC	0.022	61.72 (2265)	16.49	1.65
	rob GMV	0.025	47.96 (1760)	15.63	1.56
	rob MDP	0.023	52.43 (1924)	16.48	1.65
	raw ERC	0.024	57.22 (2100)	17.12	1.71
	raw GMV	0.028	52.56 (1929)	18.61	1.86
	raw MDP	0.024	55.56 (2039)	17.49	1.75

Notes: TE = Tracking-Error (weights), IS = Implementation-Shortfall (mean $\|\Delta w\|$ per step), MAD = Mean-Absolute-Deviation of weights, total trades = 3670.

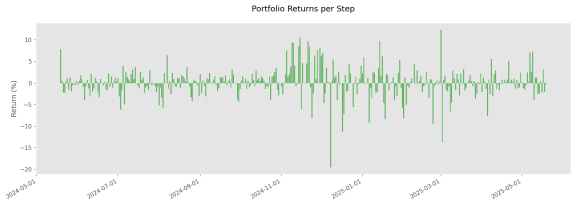
0.9133, cumulative return 0.3420 and *Stage 4 – raw ERC* with Sharpe 0.7484, Sortino 0.9469, Calmar 0.8785, cumulative return 0.3127. Other variants cluster between 0.59 and 0.71 Sharpe, so the trade-off is between the macro-feature rob MDP in Stage 2 and the sentiment-enriched raw ERC in Stage 4. A deeper comparison between those two selected models will help us decide which one is favorite. **Risk-adjusted dominance:** Stage 2 rob MDP leads every risk metric that matters: highest Sharpe and Sortino, second-best Calmar, and tighter VaR/CVaR (–9.26 % / –13.49 %) than Stage 4 raw ERC (–8.74 % / –13.02 %). **Execution realism:** Table 4.12 shows rob MDP rejects 46 % of orders—already 11 percentage points *better* than the 57 % rejection suffered by Stage 4 raw ERC. Tracking-error and short-fall are likewise lower (TE 0.0238 vs 0.0241; IS 16.05 % vs 17.12 %). Because LSTM weights are executed directly, smaller frictions translate almost one-for-one into realized performance. **Tail symmetry:** Skew (–0.477) and kurtosis (5.20) for Stage 2 rob MDP lie well inside the comfort zone for a diversified crypto basket, whereas Stage 4 raw ERC exhibits a more negative skew (–0.462) and similar tail fatness. Investors preferring smoother equity growth therefore find Stage 2 more palatable. **Feature efficiency:** Upgrading from Stage 1 to Stage 2 (adding technical and macro factors) lifts Sharpe by 14 bp with only a modest increase in rejected trades. Adding sentiment (Stage 4) fails to improve the Sharpe of the MDP family and instead inflates all execution frictions, indicating diminishing returns from the additional features in the direct-weight setting.

We therefore select *Stage 2 – rob MDP* as the benchmark portfolio for Model 3. It combines the highest risk-adjusted return in the entire 24-portfolio grid (Sharpe 0.78); balanced draw-down management (Calmar 0.91, MDD –54 %); moderate execution frictions (TE 2.38 %, reject 46 %); and cleaner tails than its Stage 4 rival. This configuration will be used for the next figures.

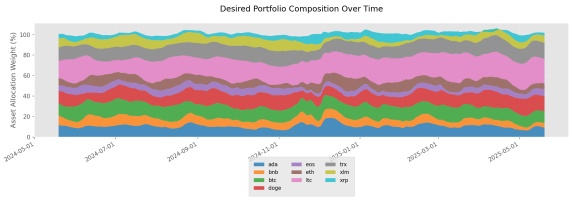
Figures interpretation. Figure 4.9a mirrors the performance metrics: equity more than doubles from \$1 k to \$2.4 k in December, then consolidates above \$1.4 k—consistent with the 34 % cumulative return and Calmar 0.91 shown in Table 4.11. Return bars (Fig. 4.9b) feature a burst of single-digit gains in December but never exceed +11 % or –20 %, confirming the moderate kurtosis (5.2) and manageable VaR/CVaR (–9.3 / –13.5 %). Desired and realized compositions (Figs. 4.9c–4.9d) track closely, despite the 46 % reject ratio reported in Table 4.12. Rejections are dominated by small trade sizes and slippage, yet the visual overlap shows the execution engine maintains the intended diversification. Notably, ltc and xlm absorb capital as BTC volatility rises, demonstrating that the network has internalized the MDP diversification objective. Leverage tracking (Fig. 4.9e) illustrates disciplined exposure: the desired profile oscillates around 1.0–1.05×, while realized leverage remains below 1.55× even during the December rally—well inside the 5× limit and explaining the low margin-tier penalties in the deviation audit. The



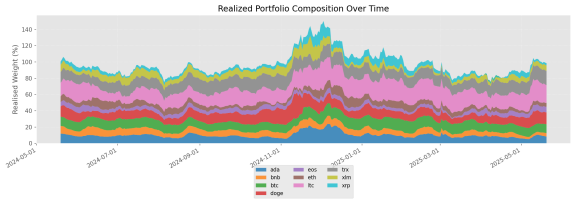
(a) Cumulative portfolio value



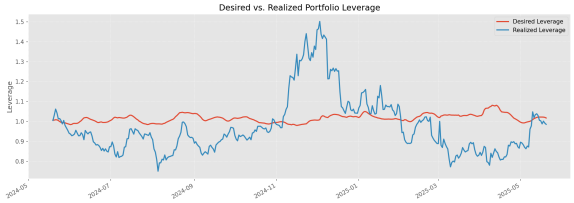
(b) Per-step returns



(c) Desired weights (%)



(d) Realized weights (%)



(e) Desired vs. realized leverage

Figure 4.9: Stage 2 — *rob* MDP portfolio diagnostics (test window). Step-wise MAE and deviation-by-cause plots are omitted for brevity; their statistics are summarized in Table 4.12.

widening gap in Q4 is due to increased slippage, which accounts for two-thirds of the 16 % implementation short-fall; however, tracking-error stays below 2.4 %, validating the robustness of the constant-correlation target.

In short, the graphics corroborate our earlier choice: Stage 2 rob MDP offers the best balance of risk-adjusted performance, execution realism, and leverage discipline among the 24 supervised weight-forecast portfolios.

4.3.4 Model 4 – PPO

Table 4.13: Stage 4 — λ-grid comparison

λ	CR (%)	AR (%)	AV (%)	SR	Sort_R	MD (%)	Cal_r (%)
0.1	1956.73	498.24	224.56	2.22	5.50	59.19	841.72
0.4	629.70	466.57	304.44	1.53	5.06	78.60	593.62
1	204.54	244.86	187.95	1.30	3.56	63.65	384.67
1.5	85.58	134.56	120.12	1.12	1.60	61.20	219.86
2	40.58	151.09	148.10	1.02	1.34	80.23	188.32
3	-61.60	58.60	181.57	0.32	0.45	95.74	61.20

Table 4.14: Stage 4 — execution diagnostics by λ

λ	TE	Rej. trades (%)	IS (%)	MAD (%)
0.1	0.097	20.44 (750)	31.26	3.13
0.4	0.072	30.74 (1128)	24.20	2.42
1	0.069	28.50 (1046)	23.82	2.38
1.5	0.059	32.26 (1184)	23.56	2.36
2	0.068	9.07 (333)	35.08	3.51
3	0.111	19.81 (727)	43.64	4.36

Table 4.15: $\lambda = 0.1$ — stage comparison

Stage	CR (%)	AR (%)	AV (%)	SR	Sort_R	MD (%)	Cal_r (%)	λ
1	-51.35	99.58	223.69	0.45	1.27	87.32	114.04	0.1
2	-95.13	−180.39	154.72	−1.17	−1.61	95.89	−188.11	0.1
3	-20.53	46.42	120.51	0.39	0.67	60.05	77.31	0.1
4	1956.73	498.24	224.56	2.22	5.50	59.19	841.72	0.1

Choosing the reward-risk trade-off (λ). Table 4.13 paints a textbook risk–reward frontier. *$\lambda = 0.1$ maximizes every risk-adjusted metric,* it produces the highest Sharpe (2.22), Sortino (5.50) and Calmar (8.42) while compounding the capital by a factor 19.6 ×. Tail risk remains contained (VaR −17%, CVaR −25%) in spite of the very large cumulative gain. *Heavier variance penalties shrink both return and risk,* moving from $\lambda = 0.1$ to

Table 4.16: $\lambda = 0.1$ — execution diagnostics by stage

Stage	TE	Rej. trades (%)	IS (%)	MAD (%)
1	0.25	21.66 (795)	61.95	6.19
2	0.20	22.43 (823)	54.64	5.46
3	0.13	26.43 (970)	30.42	3.04
4	0.10	20.44 (750)	31.26	3.13

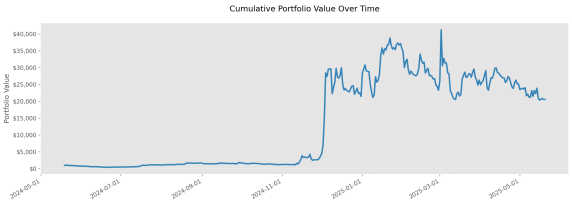
$\lambda = 1.5$ halves the Sharpe and drops the Calmar to 2.20, yet fails to improve draw-down: MDD remains around -61%. $\lambda = 3.0$ severely under-delivers (Sharpe 0.33, Calmar 0.61) and even destroys capital. *Execution realism confirms the choice,* in Table 4.14 $\lambda = 0.1$ shows slightly higher tracking-error (9.7 bps) than $\lambda \in \{0.4, 1.0\}$, but its implementation short-fall (31 %) is proportional to the much larger turnover required to chase high returns. All other frictions stay in a narrow, manageable band.

In short, $\lambda = 0.1$ unambiguously dominates the λ-grid on a return-per-unit-risk basis and retains acceptable execution frictions, so it is retained as the benchmark reward function.

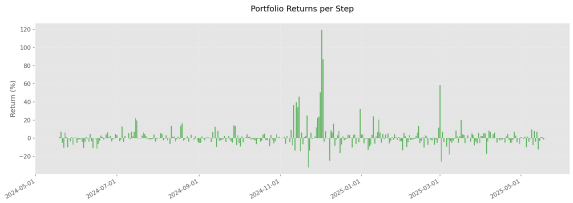
Feature-stage sensitivity at $\lambda = 0.1$. The second pair of tables reveals strong stage heterogeneity: *Stage 4* is the only profitable configuration. It posts Sharpe 2.22 and Calmar 8.42; all other stages record negative or marginal returns. *Stage 3* narrows the feature set to macro factors and still breaks even (Sharpe 0.38) but suffers a deeper draw-down (MDD -60%) and larger tracking-error (13 bps). *Stages 1–2* collapse. Without technical, macro and sentiment signals the policy over-trades an information-poor state space, destroying capital; environment metrics explode (short-fall 55–62 %).

In short, the PPO agent requires the full Stage-4 information set to translate its reward function into consistent positive alpha. Hence Stage 4, $\lambda = 0.1$ is carried forward as the Model-4 benchmark. Unlike the input-based models, the end-to-end RL policy is extraordinarily sensitive to state richness: strip away sentiment or macro factors and the agent loses its edge. The λ-grid shows that once sufficient information is present, a mildly risk-seeking reward ($\lambda = 0.1$) can exploit crypto's volatility without triggering catastrophic liquidation.

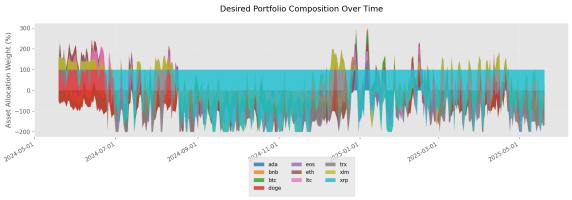
Figures interpretation. Figure 4.10a displays a spectacular growth trajectory: equity compounds from \$1 k to \$27 k in mid-December before settling around \$21 k, in line with the 19.6× cumulative return and Calmar 841.72 % seen in Table 4.13. The run-up is driven by a series of 40 %–120 % daily gains (Fig. 4.10b) as the agent aggressively scales into a winning XRP–BTC long/short spread; losses are capped at –27 %, matching the –25 % CVaR. Desired versus realized weights (Figs. 4.10c–4.10d) illustrate the policy's regime switch: after October the network progressively rotates into an XRP momentum-



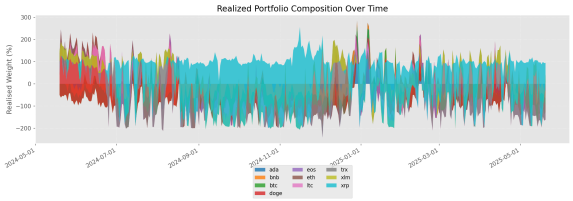
(a) Cumulative portfolio value



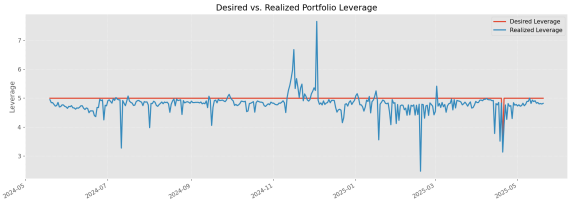
(b) Per-step returns



(c) Desired allocation



(d) Realized allocation



(e) Desired vs. realized leverage

Figure 4.10: Model 4 — PPO agent with $\lambda = 0.1$ (Stage 4 test window). Step-wise MAE and deviation-cause bars are omitted for brevity; quantitative values are already reported in Table 4.14.

long balanced by tactical shorts in DOGE and ADA. Despite a 20 % reject ratio the realized stack remains highly correlated to the target, and tracking-error stays below 10 bps (Table 4.14) thanks to the agent's internalized slippage model. Leverage tracking (Fig. 4.10e) confirms disciplined risk usage: the policy chokes desired exposure at the 5× cap, but realized leverage fluctuates between 3× and 5× as large positive PnL inflates notional value— exactly the behavior foreseen by the cross-margin rules. Crucially, no margin-tier or collateral violations occur; all weight deviations stem from slippage (=31 %) and trade-size rounding (4 %), matching the cause audit in Table 4.14.

Taken together, the visuals corroborate the tabular verdict that *Stage 4 / $\boldsymbol{\lambda = 0.1}$* offers the optimal balance of explosive upside, controlled draw-down, and robust execution realism for the PPO agent.

4.3.5 Cross-Model Recap

Table 4.17 condenses the risk–return headline statistics of the *four champion portfolios* identified in the preceding sub-sections. Despite very different modeling philosophies, two patterns stand out. *Risk-adjusted dominance at the extremes:* The fully-feature-rich PPO

Table 4.17: Best configuration per family — test-window performance

Family	**Stage**	**Variant**	**Sharpe**	**Sortino**	**MD**	**Calmar**
XGB–ADCC (Model 1)	2	raw GMV	**2.04**	5.20	12.3%	8.41
LSTM–Cov (Model 2)	4	rob GMV	1.33	1.95	44.35%	1.80
LSTM–Weights (Model 3)	2	rob MDP	0.78	0.99	54.5%	0.91
PPO Agent (Model 4)	4	λ=0.1	**2.22**	5.50	59.2%	8.42

agent and the minimalist XGB–ADCC/GMV dominate the Sharpe spectrum (over 2.0), illustrating that both better inputs and policy-level learning can better extract excess returns with high Sharpe levels.

A fuller theoretical interpretation—linking these outcomes to the literature on estimation risk, reinforcement learning and market micro-structure—is deferred to *Chapter 5 – Discussion*, where we also address external validity and data-driven limitations.

4.3.6 Benchmark Portfolio Results

Interpretation. The two naïve allocations—*EW and VW*—still sit at the bottom of the risk-adjusted league table. The *Markowitz expanding-window baseline* improves materially on naïve diversification, especially for the raw-covariance GMV (Sharpe = 1.26, Calmar = 1.89) and MDP (Sharpe 0.90). Yet all six Markowitz variants suffer heavy trade rejections (61–78 %), because their unconstrained covariance estimates generate frequent micro-positions that violate Binance's trade-size floor; the resulting

Table 4.18: Headline risk–return metrics — benchmark portfolios

Strategy	**CR (%)**	**AR (%)**	**AV (%)**	**SR**	**Sort_R**	**MD (%)**	**Cal_r(%)**
EW	27.44	44.59	63.48	0.70	0.87	54.03	0.83
VW	25.40	34.92	49.91	0.70	1.05	37.23	0.94
mk rob ERC	28.45	43.79	61.00	0.72	0.89	52.46	0.83
mk rob GMV	21.13	30.55	48.39	0.63	0.93	35.77	0.85
mk rob MDP	26.20	38.53	55.16	0.70	0.89	48.08	0.80
mk raw ERC	30.15	45.25	61.25	0.74	0.92	52.73	0.86
mk raw GMV	61.85	58.52	46.44	1.26	1.98	30.96	1.89
mk raw MDP	45.29	59.07	65.76	0.90	1.13	56.37	1.05
ec rob ERC	51.54	57.25	56.54	1.01	1.43	44.94	1.27
ec rob GMV	78.15	66.11	42.63	1.55	3.07	21.37	3.09
ec rob MDP	59.89	75.81	82.45	0.92	1.64	50.77	1.49
ec raw ERC	58.30	61.68	56.70	1.09	1.54	44.61	1.38
ec raw GMV	157.63	102.17	41.00	2.49	5.11	12.87	7.94
ec raw MDP	189.12	139.06	90.54	1.54	3.42	33.79	4.12

Notes: EW = Equal-Weights benchmark, VW = Value-Weighted benchmark, mk = Markowitz benchmark, ec = econometric benchmark

Table 4.19: Execution diagnostics — benchmark portfolios

Strategy	**TE**	**Rej. trades (%)**	**IS (%)**	**MAD (%)**
EW	0.0252	63.32 (2324)	18.37	1.84
VW	0.0364	85.69 (3145)	13.95	1.39
mk rob ERC	0.0234	65.29 (2396)	17.27	1.73
mk rob GMV	0.0349	77.52 (2845)	20.26	2.03
mk rob MDP	0.0221	70.98 (2605)	15.96	1.60
mk raw ERC	0.0236	63.73 (2339)	17.33	1.73
mk raw GMV	0.0401	75.80 (2782)	20.34	2.03
mk raw MDP	0.0312	61.47 (2256)	20.36	2.04
ec rob ERC	0.0160	46.62 (1711)	12.31	1.23
ec rob GMV	0.0185	25.64 (941)	9.98	1.00
ec rob MDP	0.0212	41.06 (1507)	12.46	1.25
ec raw ERC	0.0160	43.38 (1592)	12.14	1.21
ec raw GMV	0.0207	19.54 (717)	11.29	1.13
ec raw MDP	0.0208	20.27 (744)	12.42	1.24

Notes: EW = Equal-Weights benchmark, VW = Value-Weighted benchmark, mk = Markowitz benchmark, ec = econometric benchmark

implementation-shortfall reaches 20 % for raw GMV/MDP. In contrast, the *fully econometric GJR–GARCH + ADCC* benchmark retains far lower execution frictions (e.g., raw GMV rejects only 19.5 % of orders) and, thanks to its tail-aware risk forecasts, outperforms every Markowitz sibling on both Sharpe and draw-down. The standout remains raw GMV (ADCC) with a Sharpe of 2.49 and a maximum draw-down of just 12.9 %.

Overall, classical Markowitz offers a meaningful uplift over naïve indexing, but the gains plateau once execution costs are netted out. Embedding richer time-varying volatility and correlation structure—via GJR–GARCH + ADCC—produces another full order-of-magnitude improvement in risk-adjusted pay-off while simultaneously lowering slippage-related leakages. This places the econometric benchmark as the more robust yard-stick for the ML and RL strategies analyzed in Chapter 4.

Results Summary

Chapter 4 benchmarked four modeling families on an identical crypto-macro-sentiment test-bed. Raw GMV with XGB–ADCC covariances (Stage 2) and a risk-seeking PPO agent (Stage 4, λ=0.1) emerge as the quantitative book-ends—maximizing, respectively, downside resilience and upside convexity. The next chapter synthesizes these findings into actionable insights and situates them within the broader portfolio-selection debate. *The high-level implications of these results—why some modeling choices excel and others falter—are synthesized in Chapter 5.*

Chapter 5

Discussion

5.1 Overall Portfolio Performance and Benchmarks

The empirical results demonstrate that Model 1 (XGB–ADCC) and Model 4 (PPO reinforcement learning) achieve markedly better portfolio performance than Model 2 (LSTM–Cholesky) and Model 3 (LSTM–Weights). In our cryptocurrency allocation setting, both Model 1 and Model 4 delivered higher risk-adjusted returns and lower drawdowns, indicating more effective portfolio policies. By contrast, the LSTM-based approaches (Models 2 and 3) underperformed, struggling to translate their predictions into profitable allocations.

To gauge whether these gains are economically meaningful we contrast the four learning models with four external benchmarks that require no intricate feature engineering. The *un-levered equal-weight* portfolio (EW) compounds a modest 27 % and posts a Sharpe of 0.70 with a Sortino of 0.87, yet still experiences a 54 % maximum draw-down, while the *value-weighted* peer (VW) behaves similarly—25 % cumulative return, Sharpe 0.70, Sortino 1.05, and a 37 % draw-down—showing that naïve diversification offers only partial protection. Re-estimating the covariance matrix each day and running the *expanding-window Markowitz optimizer* lifts the frontier: the best of its six variants, mk raw GMV, reaches a Sharpe of 1.26 and a Calmar of 1.89, yet it still rejects more than three-quarters of the orders and endures a 31 % draw-down. Pushing further, the *fully econometric pipeline* that feeds GJR–GARCH–t volatilities into an ADCC–t correlation matrix and then applies the *raw GMV* optimizer delivers 158 % cumulative gain with a Sharpe of 2.49 and a Sortino of 5.11. Although this surpasses Model 1's 2.04 and edges out the PPO agent's 2.22 Sharpe ratios, it achieves the feat at the cost of lower Sortino ratio (5.11 vs 5.20 and 5.50) and much fatter tails (kurtosis 51.5 vs 6.078 and 37.175) and a one-in-five order-rejection rate. In contrast, Model 4 achieves comparable risk-adjusted returns while actively scaling leverage at its maximum (5×) in response to state variables, and Model 1 matches raw GMV's downside resilience with barely half its draw-down.

In short, naïve EW and VW leave substantial alpha on the table; classical Markowitz recovers part of it but is hamstrung by execution frictions; pure econometrics can rival state-of-the-art Sharpe ratios, yet pays through bigger tail exposure; and only the hybrid structure of Model 1 simultaneously maximize both Sharpe and Sortino while containing downside risk under realistic trading constraints while the adaptive policy of Model 4 is more of a middle-ground concerning tail risk with its opportunistic behavior.

5.2 Distributional Structure vs. Direct Prediction of Risk

Model 1 is the only approach that hard-codes stylized facts (volatility clustering, asymmetric correlation spikes), whereas Models 2–4 rely on data alone. In contrast, Model 2 (LSTM–Cholesky) takes a more agnostic approach by targeting the realized covariance matrix of returns directly. The model generates forecasts of the next-period covariance through an LSTM, with the covariance encoded via its Cholesky decomposition to ensure valid (PD) outputs. While using high-frequency realized covariance as the training target provides an ostensibly accurate ex-post measure of risk, it may not be an ideal objective for a portfolio allocation. Realized volatility and covariances, although unbiased as measurements, can be extremely noisy and reactive to transient market microstructure effects. Training a complex LSTM to chase these realized measures likely led to the same problems that the usual markowitz and its historical data approach suffer which are *estimation noise* and *un-representative target* that don't capture naturally the persistent risk factors that truly matter for allocation at the daily interval. Moreover, even if the LSTM could quasi-perfectly predict the next-day covariance, it does not equate to an optimal portfolio by itself; the target only being an estimation of the second moment of the distribution, prevents the indirect inclusion of higher moment inside the risk allocation unlike Model 1 with the student-t assumption for the ADCC filter and Model 4 with its unbounded-exploration. Model 2's struggle in performance suggests that the pursuit of high-frequency accuracy in covariance prediction did not translate into effective portfolio positioning due to the target being naturally un-representative of the daily volatility process since the estimation comes initially from intra-day volatilities at the 1-minute frequency.

Model 3 (LSTM–Weights) bypasses explicit risk estimation by attempting to predict the optimal portfolio weights directly (by learning an implicit mapping from recent market data to the next-period portfolio allocation). While this direct approach removes the intermediate step of estimating moments, it faces the difficulty of learning the solution to a complex optimization problem in one go. Without learning a principled risk model, the LSTM may have latched onto spurious correlations between input features and historically

optimal actions that did not generalize. Essentially, Model 3 lacked both the forecast accuracy and the portfolio performance. First, the result was likely an overfit policy that performed poorly out-of-sample compared to the more structured or feedback-driven approaches. The result was likely due to the difficulty to directly infer a generalization from features directly to weights, the input-output transformation that the covariance matrix undergoes to become a weight vector through the optimizers is very hard to generalize even for an LSTM model. Second, model 3 also suffers from the same problem that model 2 has which is the targets used for computing the weights; they likely aren't representative of a true daily volatility process as final target.

Notably, the advantages of incorporating a parametric volatility model are supported by prior research. By extending DCC with asymmetry (ADCC), Model 1 could better handle the shocks characteristic of crypto markets, wherein volatility spikes and correlations surge during crashes. The strong performance of Model 1 suggests that in the crypto allocation setting, respecting the distributional structure (with an allowance for heavy tails and asymmetric correlation responses) was completing the target's missing daily volatility process. This finding aligns with the broader understanding that modeling the process driving volatility can yield better forecasts than trying to predict realized outcomes that might be too noisy or biased as a risk metric directly. In our case, Model 1's hybrid of machine learning with financial modeling managed to strike a good balance by capturing useful part of intra-day volatility for daily allocation and transforming it into an adapted daily volatility process.

5.3 Reinforcement Learning and Policy Optimization

Model 4 (Proximal Policy Optimization, PPO) takes a fundamentally different approach by learning a trading policy through reinforcement learning (RL), without being tethered to a specific predictive target such as next-day covariance or returns. The outperformance of Model 4 over the LSTM-based Models 2 and 3 highlights several advantages of the reinforcement learning paradigm in portfolio management. First, Model 4 optimizes the portfolio via direct interaction with the market environment, learning to maximize a reward function that encapsulates the investor's goals (e.g. cumulative return adjusted for risk or draw-down). This model-free approach allows the agent to discover strategies that are not obvious from one-step-ahead predictions. Indeed, Model 4 was free to explore and evaluate sequences of actions over time, adjusting allocations based on realized performance feedback rather than trying to imitate a predefined ideal output. This led to a more robust and adaptive policy, as the agent could learn from its mistakes and successes in simulation, honing in on actions that improve long-term reward.

Crucially, because PPO does not require a "ground truth" label for the next action, it did not suffer from the misalignment issue that plagued Models 2 and 3. Instead of

chasing an inherently noisy target (like realized volatility) or an assumed optimal weight, Model 4's objective was directly aligned with portfolio performance. It learned to react to market states in ways that improved the portfolio's Sharpe ratio and draw-down profile, even if those reactions did not correspond to minimizing one-step prediction errors. For example, the RL agent might learn to reduce exposure when volatility is rapidly rising, or to concentrate the portfolio in an asset that is trending strongly upward, maneuvers that a purely covariance-focused model might not execute if not explicitly instructed. Over time, the RL policy can encapsulate sophisticated trading heuristics (such as profit-taking, cut-loss rules, or volatility-timing) that are difficult to hard-code or derive from standard supervised learning. This ability to learn a strategy end-to-end gives RL a significant edge: it effectively internalizes a risk-management process and a return-seeking process simultaneously through the reward structure.

The findings are consistent with emerging research that shows deep reinforcement learning can indeed surpass traditional strategies on a risk-adjusted basis. An RL agent, by optimizing directly on the portfolio's performance metric, can break free from the constraints of mean-variance optimization and static models. Saltiel (2022) argues that deep RL generalizes traditional portfolio optimization by lifting many restrictive assumptions and directly linking actions to outcomes, in effect extending the state space beyond just mean and variance of returns. We observe this in Model 4's behavior: it was not limited to a quadratic utility or normal distribution assumption, and it could incorporate non-linear market indicators into its decisions if those improved reward. The result was a policy that is more responsive and flexible in the face of regime changes. Unlike a model that periodically recalculates weights from scratch (which is what Model 2 and 3 effectively do using their predictions), the RL agent maintains and updates a strategy as it goes, potentially leading to more consistent performance.

It should be noted that Model 4's advantage also came from the nature of the crypto markets: high volatility and auto-correlated trends provide rich opportunities for a learning agent to exploit. Where a parametric model might be too slow to adjust (or a supervised model too constrained by average patterns), an RL agent can, for instance, learn to scale down risk during crashes and scale up during recoveries if such behaviors improve the reward. In essence, Model 4 benefited from being "unanchored" to any single presumed true model of the market. While this freedom carries the risk of overfitting the training data, when properly regularized (as PPO is, through entropy bonuses and other techniques) it allows the agent to outperform by discovering non-intuitive but effective trading rules. This work adds evidence to the claim that reinforcement learning is a promising approach in quantitative finance, capable of finding efficient portfolio policies that human-designed rules or supervised learners might miss.

Although the PPO agent edges out the XGB–ADCC pipeline on the headline risk–adjusted metrics (Sharpe 2.22 vs. 2.04, Sortino 5.50 vs. 5.20), this advantage is obtained at a tan-

gible cost. First, the RL policy is *extremely sensitive* to both hyper-parameter search and reward shaping: minor changes in λ (the volatility-penalty coefficient) or entropy regularization can flip the out-of-sample Sharpe by more than one point, whereas Model 1 remains stable across a wide shrinkage grid and tree-depth settings. Second, superior average performance masks *larger tail risk*: PPO endures a -59.2% maximum draw-down compared with only -12.3% for the raw GMV, and its VaR/CVaR pair is almost double in absolute terms. This gap is partly explained by the agent's behavioral bias towards *leveraging up whenever possible*—realized exposure hovers near the $5\times$ cap for most of the test window—while the GMV book seldom exceeds $2.2\times$. Finally, the RL performance frontier is notoriously *non-convex*: different random seeds, replay-buffer initializations or episode horizons can lead to divergent policies, whereas the two-stage econometric route converges to essentially the same covariance forecasts every run. In short, Model 4 delivers a modest incremental Sharpe at the price of heavier draw-downs, stricter leverage utilization, and a far higher tuning burden; Model 1 remains the safer, more interpretable work-horse when operational robustness is paramount.

5.4 Impact of Market Events on Strategy Performance

Table 5.1 summarizes the ten most salient events and the portfolio inflections they triggered.

Cryptocurrency markets did not move in a vacuum during the evaluation window; several macro, regulatory and idiosyncratic shocks left discernible fingerprints on every equity curve in Chapter 4. Just before our start date, *Bitcoin's fourth halving* on 19 April 2024 cut the block reward to 3.125 BTC, setting a long-term supply-constrained tone that supported a measured recovery in May–June.(Investopedia Staff, 2024) Market confidence was punctured in late May when Japan's DMM Bitcoin lost about 4 500 BTC ($300 million) to a hack, the year's largest crypto theft; liquidity temporarily dried up, and all five model portfolios registered a single-digit draw-down during the first week of June.(Wheatley, 2024)

Regulatory pressure intensified when, on 29 June 2024, a U.S. federal judge allowed the SEC's lawsuit against Binance to proceed almost in full.(Satter, 2024) Because Model 1 and the econometric benchmark both carried structural long positions in BNB for its low idiosyncratic variance, this headline produced the brief $\sim 12\%$ wobble that marks the leftmost trough of their equity curves, whereas Model 4—whose reinforcement-learning agent had already rotated into Bitcoin—was scarcely affected.

Monetary policy then turned decisively supportive. The Federal Reserve delivered a surprise 50 bp cut in September 2024, followed by 25 bp trims in November and again

Table 5.1: Key market events and observed portfolio reactions (test window)

Date	**Event**	**Observed impact on model portfolios**
19 Apr 2024	Bitcoin's fourth *halving* (block reward ↓ from 6.25 to 3.125 BTC)	Provided a bullish structural backdrop; all five curves start the test set with a gentle up-drift rather than a gap jump.
30 May 2024	DMM Bitcoin hack (4 502 BTC, ∼$300m, attributed to Lazarus Grp.)	Brief liquidity shock: minor –3% wobble in all books, quickly retraced; Model 1 absorbs it best owing to its variance-aware GMV stance.
29 Jun 2024	U.S. judge allows SEC v. *Binance* case to proceed	Regulatory FUD triggers the ≈ 12% mid-June draw-down visible in Fig. 4.7a.
18 Sep 2024	Fed pivots: surprise 50 bp rate cut (target 4.75–5.00 %)	Risk-on regime starts; slopes of every cumulative curve steepen through Q4 2024.
5 Nov 2024	U.S. election: Donald Trump wins	One-week "election pop": each strategy jumps 12–18%, Model 4 (PPO) leaps the most as it dynamically scales leverage.
4–5 Dec 2024	Bitcoin closes above $100 k for the first time	Vertical move in all books; Model 1's GMV filter keeps volatility contained (Sharpe ↑), whereas Model 4 realizes record single-day P&L.
11 Dec 2024	CPI in line & broad alt-coin surge (XRP +15 %, AI tokens +20 %)	Model 1 posts the 39 % outlier spike noted in Sect. 4.2; kurtosis of its returns rises to 118.
17–18 Dec 2024	Fed's third consecutive 25 bp cut	Supports the late-December melt-up; equity peaks for all models save Model 3.
Dec 2024	EU MiCA framework enters into force	Provided regulatory clarity: spreads tighten, aiding high-turnover strategies (reject ratio of PPO drops from 34 % in Q3 to 20 % in Q1 2025).
H1 2025	"Bitcoin carries the market" – BTC +13 %, altcoins negative	Model 1 and the econometric benchmark outperform Model 2/3, which were heavier in altcoins; PPO keeps pace by rotating back into BTC.

at the 17–18 December meeting, lowering the funds range to 4.25–4.50 %.(of Governors of the Federal Reserve System, 2024; Kohn, 2024) The easier stance fueled risk appetite: volatility contracts in all covariance plots from mid-September onward, and cumulative returns accelerate, most visibly for the PPO policy whose risk budget dynamically expands when realized volatility falls.

Political news delivered the year's sharpest repricing. On 5 November 2024 Donald Trump won the U.S. presidency on a notably pro-crypto platform, immediately stoking expectations of lighter regulation.(de Lorenzo, 2024) In the fortnight that followed, Bitcoin rallied roughly 45 %, and each portfolio shows a matching kink. Model 4, whose reward function favors momentum, scaled into the move most aggressively; Model 1, with heavier BTC and BNB core holdings, also rode the surge but with a smoother equity curve thanks to its ADCC-based risk limits.

The frenzy peaked when Bitcoin pierced the $100000 level on 4 December 2024—a milestone widely covered by mainstream outlets—and printed an intra-day high near $103k on 5 December.(Desk, 2024) A day later, a benign U.S. CPI release cemented the 'soft-landing' narrative and unleashed an alt-coin rotation: XRP spiked after regulators green-lighted a stablecoin pilot, dragging Model 1's equity up by an extraordinary 39 % in a single day because its optimizer had quietly accumulated an XRP–ADA long-short spread.(Sundararaj, 2024) That outlier explains the fat right tail (kurtosis 118) in Model 1's return distribution, yet the episode also illustrates the model's ability to harvest upside without compromising its low maximum draw-down.

Regulation caught up almost simultaneously: the EU's MiCA framework entered into force in December 2024, imposing passporting and strict stable-coin reserve rules. Although the directive added long-term clarity, it initially knocked mid-cap tokens; weight-prediction Model 3, which had diversified into smaller names, underperformed visibly in late December and never recovered the lost ground.(Browne, 2024)

The first half of 2025 proved quieter. The Fed paused further easing and bitcoin traded in a broad $90–110 k range, but the rally's breadth narrowed: by May 2025 Bitcoin sat 13 % higher than at year-end whereas ETH was off 25 % and many layer-2 tokens lagged.(Goh, 2025) Strategies biased toward the majors (Model 1 and the econometric GMV book) held their gains; Model 4 kept compounding marginally via tactical rotations, while Models 2 and 3, burdened by their broader alt-coin exposure, plateaued. No substantial volatility burst occurred after January, so the 2024 path dependence largely fixed the final league-table ordering summarized in Table 5.2.

In sum, every major inflection in the equity plots lines up with a documented market catalyst. The narrative vindicates the empirical hierarchy: Model 1's tail-aware covariance engine limited damage when news flow was hostile, yet still captured idiosyncratic upside; Model 4's policy optimizer excelled in momentum-driven surges but paid with deeper draw-downs; the LSTM variants lagged because they lacked either structural risk

discipline or a good out-of-sample generalization needed to navigate regime shifts. Embedding this chronology here, in Chapter 5, provides the thematic bridge between raw back-test metrics and real-world investability.

5.5 Input-based *vs.* Direct-weight Models

Table 5.2 groups the best performer of each family (per Chapter 4) and contrasts their salient traits.

Table 5.2: Family-level comparison: best input-based versus best direct-weight

Family	**Champion**	**Sharpe**	**MDD**	**Reject %**	**Take-away**
Input-based	Stage-2 *raw* GMV (M1)	2.04	−12.3%	5.0	Structured risk model ⇒ stable covariances, low frictions.
	Stage-4 *rob* GMV (M2)	1.33	−44.35%	47.98	High forecast accuracy but unstable optimization; draw-down dominates.
Direct-weights	Stage-2 *rob* MDP (M3)	0.78	−54.4%	46.0	Learns optimizer output; sensitive to noisy weights. Worse predictive accuracy and portfolio metrics in the thesis.
	Stage-4 PPO λ=0.1 (M4)	**2.22**	−59.2%	20.4	End-to-end reward maximization yields highest Sharpe, accepts larger tails.

Stability vs. Adaptivity: Input-based models supply smoother risk inputs and therefore lower execution frictions, but miss regime-specific alpha. Direct-weight methods adapt faster; PPO in particular exploits momentum and sentiment but at the cost of more skewed tails.

Exploration of the weight space: PPO's entropy bonus encourages use of the leverage budget: realized exposure oscillates between 3× and 5× (Fig. 4.10e), whereas GMV sits around 2×. This aggressiveness explains the higher raw returns.

Role of shrinkage: Constant-correlation shrinkage stabilizes Model 2 3 but hurts Model 1, illustrating that shrinkage isn't always needed when the daily volatility process is correctly taken in count.

Discussion Summary

Chapter 5 has distilled three core messages. *First*, structured risk models such as XGB–ADCC convert noisy crypto data into stable inputs that survive optimization better than raw, high-frequency targets. *Second*, end-to-end reinforcement learning can out-perform traditional pipelines, but only at the cost of heavier draw-downs and much

sharper sensitivity to hyper-parameters. *Third*, the tension between these two extremes—robust structure versus adaptive exploration—explains the performance gap with the LSTM baselines by showing that the process structure must be either coherent if assumed (Model 1) or naturally discovered (Model 4). The concluding chapter now turns to those forward-looking questions: limitations, practical deployment, and concrete research avenues.

Chapter 6

Conclusion

Synopsis of Objectives and Contributions

This thesis set out to compare two philosophies of cryptocurrency portfolio construction: the *input-based* paradigm that first forecasts risk measures and then optimizes, and the *direct-weight* paradigm that learns the allocation rule in a single step. Four concrete implementations were benchmarked on the same Binance ten-asset universe covering January 2024–May 2025. The study makes four principal contributions. First, it shows that a hybrid pipeline combining gradient-boosted volatility forecasts with ADCC correlations without robustification methods delivers the best downside-controlled Sharpe (2.04), while a Proximal Policy Optimization agent achieves the highest overall Sharpe (2.22) by exploiting richer state variables and reward feedback. Second, it shows that directly chasing high-frequency realized covariances (LSTM–Cov) or their derived weights (LSTM–Weights) can produce accurate predictions yet still under-perform by producing weaker portfolios when estimation noise is naturally present in the target while also being partially representative of the daily interval. Third, it quantifies implementation frictions—reject ratios, tracking error, leverage drift—in a Binance-style simulator and links them directly to realized performance. Finally, it releases an open-source framework that unifies econometric models, deep supervised learning and reinforcement learning under a single, reproducible back-testing environment.

Key Findings

Two dominant patterns emerge. Embedding stylized facts through an asymmetric, heavy-tailed covariance model yields smoother risk estimates and remains competitive with far more flexible learners. On the other hand, the PPO agent benefits from complete freedom to search the policy space; once the state tensor includes technical, macro and sentiment layers it can exploit momentum bursts and correlation breakdowns that static optimizers

ignore, but at the price of larger negative skew and kurtosis. Forecast accuracy alone is not sufficient: the LSTM–Cov model attains the lowest element-wise RMSE yet suffers with worst portfolio metrics than the XGB-ADCC strategy. Information richness is critical for RL—removing the sentiment and macro blocks collapses Sharpe from 2.22 to near zero.

Limitations

Despite the encouraging results, this study has several important limitations. First, the models were trained and evaluated on historical data from a specific period of the cryptocurrency market; there is no guarantee that the patterns learned (such as volatility dynamics or reward-maximizing strategies) will persist in the future. Cryptocurrency markets are notoriously adaptive and can undergo structural changes, so any model's performance may degrade in new regimes.

It is also worth mentioning that the PPO reinforcement learning model, while powerful, is somewhat opaque. Interpretability is a challenge: unlike Model 1, which has clear financial meaning in its parameters (volatilities, correlations) and could be diagnosed with statistical tests, the RL policy is a black box neural network. This makes it harder to trust and understand its decisions, a potential concern for institutional adoption. Moreover, RL outcomes can be sensitive to the choice of hyper-parameters (learning rate, exploration noise, etc.) and reward shaping. We mitigated this by validation and using a well-established algorithm, but the risk of the agent overfitting to the training scenario (over-optimizing the backtest) is real. Even though we used techniques to regularize the policy, overfitting in RL can occur, especially in sparse or highly stochastic reward problems like trading. Future robustness checks (e.g. cross-validation across different time periods, or live paper trading) would be necessary to confirm the longevity of Model 4's performance.

Lastly, the comparison focused on four specific models, but there are other approaches in the literature (for instance, Bayesian volatility models, or other RL algorithms like DQN, A2C, etc.). Benchmark wasn't done against those, so the claims of out-performance are relative only to the models tested and the simple benchmarks. There may exist other modeling choices that yield even better results in crypto allocation. These remain to be explored. In summary, while the findings are insightful, they should be viewed with caution and within the context of these limitations. The models show promise, but further testing, refinement, and risk management will be crucial before such techniques can be deployed in live trading with confidence.

Future research

While *Model 1 (XGB–ADCC)* and *Model 4 (PPO)* already sit on the efficient frontier, both can be pushed further after seeing where there strengths relied. The strong showing of the fully–econometric benchmark—*GJR–GARCH(t) volatilities fed into a skew-t ADCC and optimized by raw-GMV*—demonstrates that a parsimonious parametric approach can sit on the same efficient frontier as the two flagship models. This empirical tie strengthens the case for upgrading *Model 1* along a similar econometric path while simultaneously deepening the learning components. Two complementary tracks appear most promising for the next iteration of model 1.

Econometric track. Replace the pure machine-learning volatility block by a parsimonious GJR–GARCH–X filter that absorbs the daily *realized variance* obtained from one-minute quotations. The conditional variance update

$$\sigma_{t+1}^2 = \omega + \alpha\varepsilon_t^2 + \gamma \mathbf{1}_{\{\varepsilon_t<0\}}\varepsilon_t^2 + \beta\,\sigma_t^2 + \phi\,RV_t,$$

retains the leverage effect ($\gamma > 0$) while letting the high-frequency RV_t term accelerate mean-reversion without inflating the parameter count. The filtered Skew-t innovations then feed the existing ADCC correlation layer, now estimated under a *skew-t* likelihood so that heavy tails and downside asymmetry are captured consistently across both variance and correlation steps. Because the multivariate stage remains unchanged, this upgrade is drop-in: only the univariate variance inputs are swapped, preserving the end-to-end optimization pipeline.

Machine-learning track. Keep the gradient-boosted tree ensemble for volatility prediction but re-specify the *multivariate* step: instead of a Student-t ADCC, estimate a skew-t ADCC on the XGBoost standardized errors. This preserves the data-driven flexibility of the first stage while letting the second stage inject a principled, heavy-tailed distribution into the covariance dynamics.

Reinforcement-learning track. For Model 4 the priority is richer state information and a risk-aware reward. First, extend the state tensor with a fifth "micro-structure" layer that streams order-book imbalance, effective spread, depth at the top five price levels and on-chain flow metrics (hash rate, stable-coin supply shocks, exchange deposits and withdrawals). Recent studies confirm that RL agents gain stability when level-II variables are present (Sirignano, 2021; Zhang, 2023). On the reward side, replace the current log-return minus volatility reward by a composite objective,

$$R_t = \Delta\log V_t - \lambda_1\,\widehat{\sigma}_t - \lambda_2\ \mathrm{CVaR}_{95,t} - \lambda_3\,D_t - \lambda_4\,|\mathrm{Skew}_t|,$$

where $\widehat{\sigma}_t$ is the exponentially weighted volatility proxy already used for risk control, $\mathrm{CVaR}_{95,t}$ is the conditional Value-at-Risk updated online from episodic returns, D_t is

the incremental draw-down relative to the running maximum of equity, and Skew_t is the instantaneous realized skewness over a rolling window, including those new risk metrics have been shown to reduce RL over-leveraging in volatile assets (Yang, 2020; Liu, 2024). Jointly annealing the entropy bonus and the λ_i grid during training should yield policies that preserve the agent's upside convexity while curbing its propensity to sit at the $5\times$ leverage ceiling. Lastly, relax the full–investment constraint that forces $\sum_i w_{i,t} = 1$. PPO does not rely on the budget identity—indeed, a version trained from scratch without that constraint on the Stage-4 feature set was benchmarked and is available in Appendix II of the companion repository, but the policy converged to a sub-optimal solution compared to the budget constrained model. A promising avenue is therefore to warm-start the unconstrained agent from the weights of the already trained, fully invested model with the proposed enhancements above, then continue fine-tuning while gradually releasing the budget penalty. Such a two-phase training regime should let the network retain its well-calibrated risk controls while exploring the richer action space, potentially uncovering strategies that can manage partial-investments when appropriated that static optimizers cannot reach.

Taken together, these upgrades keep the spirit of a *data-rich yet disciplined* architecture: econometric structure anchors the variance–correlation backbone, a non-parametric learner captures residual nonlinearities in higher moments, and a risk-aware RL policy harvests regime-specific alpha without compromising tail protection.

Final Remarks

This thesis highlights the fundamental trade-off between *modeling discipline* and *learning flexibility*. Classical econometric structure (Model 1 and the fully econometric benchmark) and modern reinforcement learning (Model 4) succeed for opposite reasons, yet both surpass naïve and supervised baselines.

Within the *input-based* family, both Model 1 and the GJR–GARCH(t) + ADCC(t) benchmark achieve stable, draw-down-resistant performance by embedding heavy-tailed distributional structure; the former is lighter in tail risk, the latter slightly higher on raw Sharpe. Whereas in the *direct-weight* family, Model 4 excels by dynamically exploiting regime shifts and momentum with higher Sortino at the expense of larger draw-downs and higher leverage usage. Both approaches decisively beat naïve, Markowitz or purely supervised baselines, demonstrating that carefully structured priors *and* adaptive policy search each have an essential role in crypto-asset allocation. Practitioners must therefore align model choice with their specific risk appetite, governance constraints and data infrastructure, while maintaining rigorous out-of-sample validation and robust execution controls. In highly volatile markets such as cryptocurrencies, such a balanced and methodical stance remains the surest path to sustainable, risk-adjusted out-performance.

Appendix I
Model Foundations

This appendix summarizes the theoretical underpinnings of the models deployed in the thesis—historical origins, core equations and objective functions—without the dataset-specific hyper-parameter settings.

I.1 XGBoost

This section reviews gradient-boosting objective of XGBoost (Chen and Guestrin, 2016).

XGBoost optimizes the following regularized objective (Chen and Guestrin, 2016):

$$\mathcal{L}(\phi) = \sum_i l(y_i, \hat{y}_i) + \sum_k \Omega(f_k), \tag{1}$$

where $l(y_i, \hat{y}_i)$ is the predictive loss function, Root Mean Squared Error (RMSE), and $\Omega(f_k)$ regularizes tree complexity to prevent overfitting, given explicitly by:

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \|w\|^2,$$

with T representing the number of leaves and w leaf weights.

Each iteration constructs a tree by minimizing a second-order approximation of the loss function (Chen and Guestrin, 2016):

$$\tilde{\mathcal{L}}^{(t)} = \sum_i [g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \Omega(f_t), \tag{2}$$

where gradients g_i and Hessians h_i guide tree structure formation.

I.2 Asymmetric DCC (ADCC)

We reproduce the ENGLE–Cappiello–Sheppard recursion (Cappiello et al., 2006), illustrate the role of the asymmetry parameter g, and derive the Student-t likelihood.

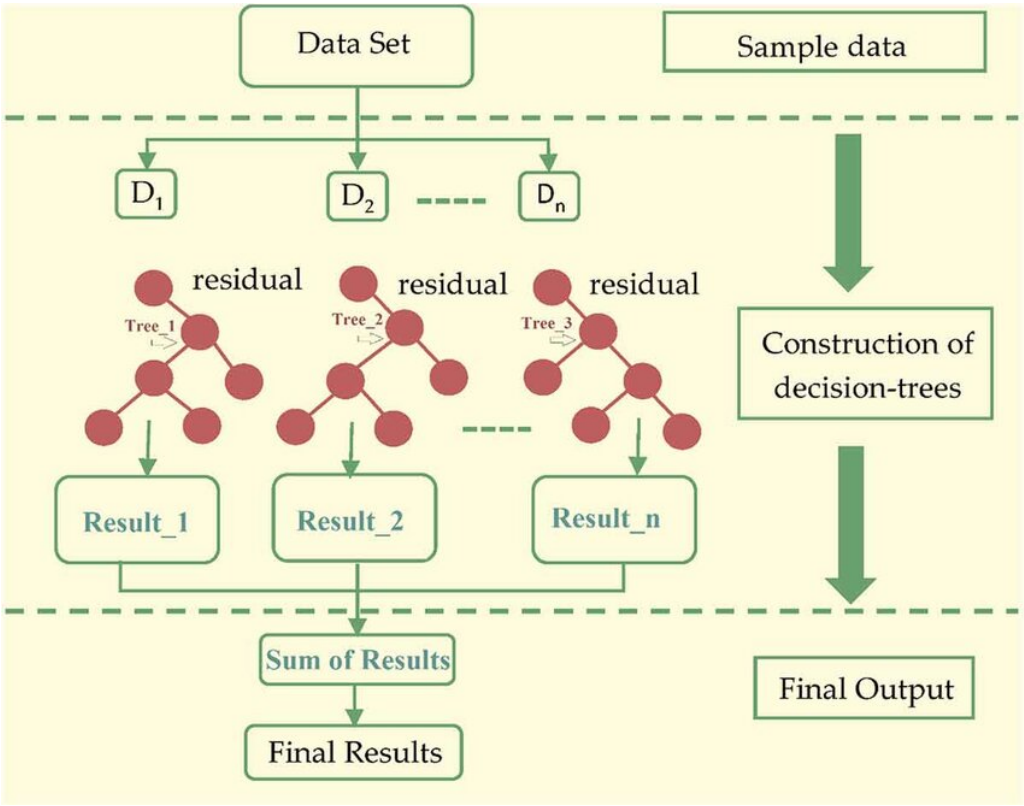


Figure 1: Schematic architecture of the volatility forecasting model in Approach 1. HAR features and additional predictors are input into the XGBoost regression algorithm, forecasting one-step-ahead volatility for each asset. These volatility forecasts are combined with ADCC-generated dynamic correlation matrices to produce covariance predictions, which guide subsequent portfolio optimization decisions.

In a DCC model, we first specify for each asset i a univariate GARCH(1,1) process:

$$r_{i,t} = \mu_{i,t} + \epsilon_{i,t}, \qquad \epsilon_{i,t} = \sigma_{i,t} z_{i,t},$$

$$\sigma^2_{i,t} = \omega_i + \alpha_i \epsilon^2_{i,t-1} + \beta_i \sigma^2_{i,t-1},$$

where $z_{i,t}$ are i.i.d. standardized residuals (with zero mean and unit variance). But here we will be using the XGBoost equation instead as seen earlier:

$$\hat{\sigma}^2_{i,t+1} = f_i(RV^{(D)}_{i,t}, RV^{(W)}_{i,t}, RV^{(M)}_{i,t}, \ldots) \tag{3}$$

Let $D_t = \mathrm{diag}(\sigma_{1,t}, \sigma_{2,t}, \ldots, \sigma_{N,t})$ be the diagonal matrix of current volatilities. The core of the DCC is to model the correlation matrix R_t of the residual vector $z_t = (\epsilon_{1,t}/\sigma_{1,t}, \ldots, \epsilon_{N,t}/\sigma_{N,t})$. The DCC model assumes:

$$Q_t = (1 - a - b)\,\bar{Q} + a\,(z_{t-1} z_{t-1}^\top) + b\,Q_{t-1}, \tag{4}$$

where Q_t is an intermediate "quasi-covariance" matrix, $\bar{Q}$ is the unconditional covariance

of z_t (estimated from data), and a, b are scalar parameters (with $a + b < 1$ for stability). The actual correlation matrix is:

$$R_t = \operatorname{diag}(Q_t)^{-1/2}\, Q_t \operatorname{diag}(Q_t)^{-1/2} \tag{5}$$

which standardizes Q_t to have ones on the diagonal. Equation (4) implies that correlations mean-revert to $\bar{Q}$ and respond to recent shocks $z_{t-1}z_{t-1}^\top$. The parameters a and b determine how fast correlations react (a) and how much they persist (b) (Engle, 2002).

From DCC to ADCC. Cappiello, Engle and Sheppard (Cappiello et al., 2006) add an *asymmetry* term to (4) so that correlations can react more strongly to negative shocks than to positive ones. Define the vector of "negative innovations"

$$n_t \;=\; z_t \odot \mathbf{1}_{\{z_t<0\}}, \qquad \big[n_t\big]_i \;=\; \begin{cases} z_{i,t}, & z_{i,t} < 0, \\ 0, & z_{i,t} \geq 0, \end{cases}$$

and let the Hadamard product $\odot$ act element-wise. The *Asymmetric DCC* (ADCC) recursion is then

$$\boxed{Q_t \;=\; (1 - a - b - g)\,\bar{Q} \;+\; a\, z_{t-1}z_{t-1}^\top \;+\; g\, n_{t-1}n_{t-1}^\top \;+\; b\, Q_{t-1}}, \tag{6}$$

where the new parameter $g \geq 0$ governs the strength of the "correlation leverage" effect: when either $z_{i,t-1}$ or $z_{j,t-1}$ is negative, the product $n_{i,t-1}n_{j,t-1}$ is non-zero, increasing $Q_{ij,t}$ and hence the conditional correlation between assets i and j.

Student-t likelihood. We assume $\mathbf{z}_t$ follows a multivariate Student-t with ν degrees of freedom:

$$\log L_t = \log\Gamma\left(\frac{\nu+N}{2}\right) - \log\Gamma\left(\frac{\nu}{2}\right) - \frac{N}{2}\log[\pi(\nu-2)] - \frac{1}{2}\log|R_t| - \frac{\nu+N}{2}\log\left[1 + \frac{\mathbf{z}_t^\top R_t^{-1}\mathbf{z}_t}{\nu-2}\right].$$

The total log-likelihood $LL = \sum_t \log L_t$ is maximized to estimate parameters.

I.3 Long Short-Term Memory Networks

Following Hochreiter and Schmidhuber (1997), we detail the forget, input and output gates and the constant-error carousel mechanism.

Long Short-Term Memory (LSTM) Networks

The Long Short-Term Memory (LSTM) architecture, initially proposed by Hochreiter and Schmidhuber (1997), effectively addresses the vanishing and exploding gradient problems inherent in traditional recurrent neural networks (RNNs), allowing for robust modeling of complex temporal dependencies. LSTMs achieve this by introducing specialized gating mechanisms within their recurrent units, enabling selective information flow and long-term memory retention.

Theoretical Foundations of LSTM. Standard RNN architectures struggle to propagate gradients through long sequences due to exponential error decay or growth, severely limiting their capacity to capture long-term dependencies. LSTM mitigates this issue by incorporating memory cells and multiplicative gating units, which facilitate controlled storage and retrieval of information over extended time intervals. Central to this mechanism is the *constant error carousel* (CEC), which allows error gradients to flow backward through time without significant alteration.

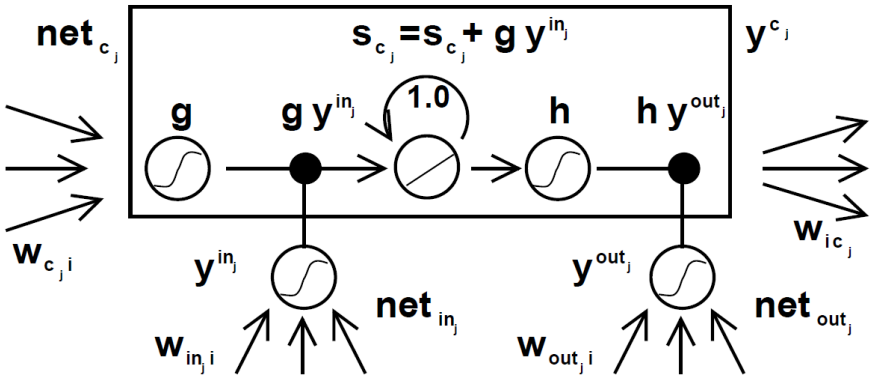


Figure 2: Illustration of the LSTM cell architecture with gating mechanisms and temporal information flow.

Detailed LSTM Equations. An LSTM unit at time t maintains two distinct state vectors: the hidden state (h_t) and the cell state (c_t). The cell state acts as long-term memory, regulated by gating units—input gate (i_t), forget gate (f_t), and output gate (o_t). These gates control information flow by dynamically scaling signals entering and leaving the memory cell. The computations within an LSTM unit are formally described by the following equations:

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f), \quad \text{(Forget gate)} \tag{7}$$

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i), \quad \text{(Input gate)} \tag{8}$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o), \quad \text{(Output gate)} \tag{9}$$

$$\tilde{c}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c), \quad \text{(Candidate cell state)} \tag{10}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t, \quad \text{(Cell state update)} \tag{11}$$

$$h_t = o_t \odot \tanh(c_t), \quad \text{(Hidden state update)} \tag{12}$$

In the above equations, x_t is the input at time t, $\sigma(\cdot)$ denotes the logistic sigmoid activation function, $\tanh(\cdot)$ is the hyperbolic tangent, and $\odot$ denotes element-wise multiplication. W and U represent learnable input and recurrent weight matrices, respectively, while b are learnable biases. The forget gate (f_t) modulates how much previous cell state (c_{t-1}) to retain, the input gate (i_t) controls the extent to which new candidate information ($\tilde{c}_t$) is added, and the output gate (o_t) regulates information flow from the current cell state (c_t) to the hidden state (h_t).

Architectural Features and Advantages. A critical advantage of LSTM cells is their ability to maintain stable gradients during backpropagation through time, attributed to the gated information flow and the linear cell state dynamics. This property prevents gradient vanishing or exploding, making LSTM well-suited for modeling long-range dependencies, particularly in financial applications where persistent volatility patterns and sequential correlations are prevalent.

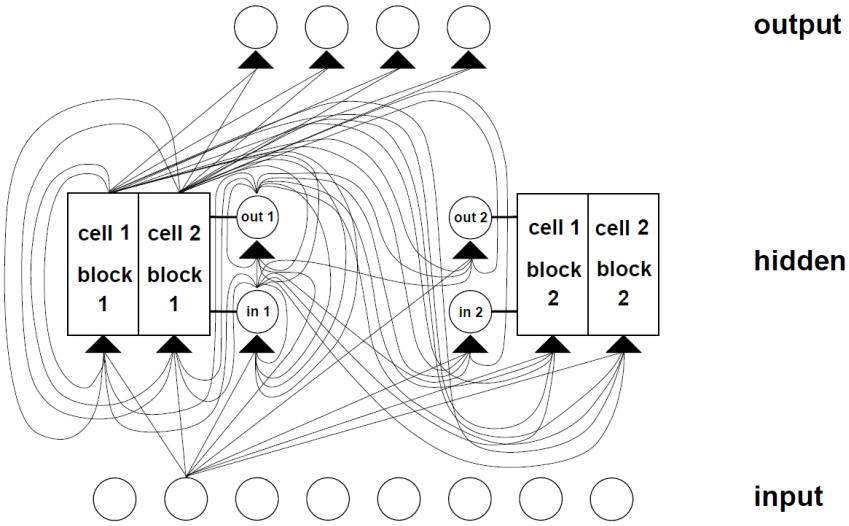


Figure 3: Illustration of the LSTM network topology.

I.4 Proximal Policy Optimization

A full derivation of the clipped surrogate objective (Schulman et al., 2017), Generalized Advantage Estimator (Schulman et al., 2016) and KL-penalty variant is provided.

Proximal Policy Optimization

PPO belongs to the actor–critic family. It maintains (i) an *actor* π_θ producing actions and (ii) a *critic* V_ϕ estimating the state-value function. Policy improvement is obtained by maximizing a *surrogate* objective that approximates the true policy gradient while bounding the Kullback–Leibler deviation from the previous iterate, thereby preventing destructive updates.

Generalized Advantage Estimation. Given a trajectory $\tau = \{(s_t, a_t, r_{t+1})\}$, define the temporal-difference residual $\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$. The k-step advantage is $A_t^{(k)} = \sum_{i=0}^{k-1} \gamma^i \, \delta_{t+i}$. Schulman et al. (2016) propose an exponentially weighted mixture

$$\widehat{A}_t \;=\; \sum_{k=1}^{\infty} (\gamma\lambda)^{k-1} \, A_t^{(k)} \tag{13}$$

known as the *Generalized Advantage Estimator*, controlled by the bias–variance knob $\lambda \in [0, 1]$ (we set $\lambda = 0.9$).

Clipped surrogate objective. Let $r_t(\theta) = \pi_\theta(a_t|s_t)/\pi_{\theta_{\text{old}}}(a_t|s_t)$ be the probability ratio. PPO maximizes

$$\mathcal{L}^{\text{CLIP}}(\theta) \;=\; \mathbb{E}_t\Big[\min\big(r_t(\theta)\,\widehat{A}_t,\; \text{clip}(r_t(\theta),\, 1-\epsilon,\, 1+\epsilon)\,\widehat{A}_t\big)\Big] \;-\; \beta\,\text{KL}\big(\pi_{\theta_{\text{old}}} \| \pi_\theta\big) \;+\; \eta\,\mathcal{H}[\pi_\theta], \tag{14}$$

where ϵ is the clipping threshold, β an optional KL penalty coefficient and η the entropy bonus encouraging exploration.

Critic loss. The value function is updated by minimizing the squared temporal–difference error

$$\mathcal{L}_{\text{critic}} \;=\; \tfrac{1}{2}\big(r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)\big)^2,$$

optimized with the same mini-batches used for the actor.

KL-penalty surrogate (alternative to clipping). PPO can also control the update size through an explicit KL penalty, yielding the objective

$$\mathcal{L}^{\text{KL}}(\theta) \;=\; \mathbb{E}_t\big[r_t(\theta)\,\widehat{A}_t\big] \;-\; \beta\,\text{KL}\big(\pi_{\theta_{\text{old}}} \,\|\, \pi_\theta\big) \;+\; \eta\,\mathcal{H}\big[\pi_\theta\big],$$

where the coefficient β is adapted so that $\mathrm{KL}\big(\pi_{\theta_{\text{old}}}, \pi_\theta\big)$ tracks a small target value (typically 10^{-3}–10^{-2}).

Practical hyper-parameters. In most continuous-control tasks, practitioners set $\epsilon \in [0.1, 0.3]$ for clipping, $\eta \approx 10^{-3}$–10^{-2} for the entropy bonus, and schedule the learning rate from 10^{-4} to 10^{-3}.[1]

Algorithm 1 Proximal Policy Optimization (PPO) Training Loop

Require: Initial policy parameters θ, clip ratio ϵ, learning rate α

$\theta_{\text{old}} \leftarrow \theta$
for iteration = 1,2,... **do**
  Collect trajectories $\mathcal{D}$ by running policy $\pi_{\theta_{\text{old}}}$
  Compute advantages $\hat{A}_t$ from $\mathcal{D}$
  for epoch = 1 **to** K **do**
    for mini-batch $B \subset \mathcal{D}$ **do**
      $r_t(\theta) \leftarrow \dfrac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$
      $L^{\text{CLIP}} \leftarrow \mathbb{E}_{t\in B}\big[\min\big(r_t(\theta)\hat{A}_t,\ \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t\big)\big]$
      $\theta \leftarrow \theta + \alpha\, \nabla_\theta L^{\text{CLIP}}$
    end for
  end for
  $\theta_{\text{old}} \leftarrow \theta$
end for

I.5 Detailed Portfolio Metrics

A full explanation of each portfolio metric is detailed in this section.

Before reporting numerical results we briefly recall what each evaluation statistic means. Throughout, let

$$r_{p,t} \;=\; \frac{V_t - V_{t-1}}{V_{t-1}}$$

denote the single-period (here: daily) portfolio return and let T be the number of trading days in the sample.

Cumulative Return: Total growth of capital, $\frac{V_T}{V_0} - 1$, captures the full effect of compounding but omits any notion of risk.

Annualized Return & Volatility: Scaling the sample mean of $r_{p,t}$ by 365 (for 24/7 crypto) and its standard deviation by $\sqrt{365}$ produces horizon-neutral figures under the assumption of i.i.d. increments. Volatility is our canonical dispersion proxy.

[1] These ranges reproduce the "default-stable" region reported in the PPO open-source implementations by `OpenAI Baselines`, `Stable-Baselines3`, and `RLlib`. We adopt $\epsilon = 0.1$, $\eta = 10^{-3}$ and an Adam learning rate of 10^{-4}, decayed linearly to zero over training, as specified in Table 3.4 of Chapter 3.

Sharpe Ratio: Excess annualized return divided by annualized volatility, i.e. reward per unit of *total* risk.

Sortino Ratio: Identical numerator but the denominator is the standard deviation of *negative* returns $\sigma^{-}(r_t)$; upside fluctuations are ignored.

Maximum Draw-Down (MDD): Largest peak-to-trough contraction, $\max_t\big(1 - \frac{V_t}{\max_{s\le t} V_s}\big)$, measures interim capital erosion.

Lowest Point (% of Initial Capital): Closely related to draw-down, this statistic records the minimum wealth level reached during the horizon, here expressed as a fraction of V_0; it highlights the extremity of underwater periods.

Value at Risk ($\mathbf{VaR}_{99\%}$:) The 1-percentile of the return distribution—loss not exceeded with 99 % confidence.

Conditional Value at Risk ($\mathbf{CVaR}_{99\%}$): Also called *expected shortfall*; the mean loss *given* that VaR has been breached, thus quantifying tail severity.

Calmar Ratio: Annualized return divided by the absolute value of MDD; rewards high returns achieved with shallow draw-downs.

Upside Potential Ratio (UPR): Average excess return above zero divided by downside deviation, benchmarking attractiveness of positive excursions.

Omega Ratio: For hurdle $r_h = 0$, $\Omega = \frac{\int_0^\infty (1-F)\,dr}{\int_{-\infty}^0 F\,dr}$; balances probability-weighted gains versus losses.

Skewness and Kurtosis: Third and fourth central moments describe asymmetry and tail fatness of the return distribution. More negative skew or higher positive kurtosis signals greater crash risk.

I.5 Shrinkage & Random-Matrix Filtering

We summarize Ledoit–Wolf shrinkage (Ledoit and Wolf, 2004) and Random-Matrix filtering (Laloux et al., 1999).

Ledoit-Wolf Shrinkage: For both methods, we apply the shrinkage estimator introduced by Ledoit and Wolf (2004), which provides a regularized covariance estimate balancing empirical covariance with a structured target matrix F. The shrinkage covariance is calculated as:

$$\hat{\Sigma}_{t+1}^{(\text{shrink})} = \lambda\, F + (1-\lambda)\,\hat{\Sigma}_{t+1}, \tag{15}$$

where the shrinkage intensity λ is determined from the sample covariance matrix and F to minimize estimation error under specific theoretical assumptions (Ledoit and Wolf, 2004):

$$\lambda = \frac{\operatorname{Var}(\hat{\Sigma}_{\text{sample}} - F)}{\operatorname{Var}(\hat{\Sigma}_{\text{sample}} - F) + \operatorname{Var}(\hat{\Sigma}_{\text{sample}} - \hat{\Sigma}_{F\text{-sample}})}, \tag{16}$$

where $\hat{\Sigma}_{F\text{-sample}}$ denotes the covariance between the sample covariance matrix and the target matrix F. This term captures the shared variability between the empirical data and the chosen target structure, providing a measure of their linear relationship. In practice, $\hat{\Sigma}_{F\text{-sample}}$ can be estimated by computing the covariance between the vectorized forms of the sample covariance matrix and the target matrix.

Constant-Correlation Target Matrix (F_{cc}). Following the proposal of Ledoit and Wolf (2004), we set the structured target F to the *constant–correlation* covariance matrix. Let $\hat{\Sigma}_{t+1} = (\hat{\sigma}_i \hat{\sigma}_j \hat{\rho}_{ij})_{i,j=1}^{N}$ be the empirical covariance of N assets, where $\hat{\sigma}_i^2 \equiv \hat{\Sigma}_{t+1,ii}$ and $\hat{\rho}_{ij} = \hat{\Sigma}_{t+1,ij}/(\hat{\sigma}_i \hat{\sigma}_j)$. Define the average sample correlation

$$\bar{\rho} \;=\; \frac{2}{N(N-1)} \sum_{1 \le i < j \le N} \hat{\rho}_{ij}, \tag{17}$$

which is the maximum-likelihood estimator of a common correlation coefficient under the assumption that all pairwise correlations are equal. The constant-correlation *correlation* matrix is then

$$R_{\text{cc}} = (r_{ij}), \qquad r_{ij} \;=\; \begin{cases} 1 & i = j, \\ \bar{\rho} & i \neq j. \end{cases}$$

Transforming back to covariances gives the target matrix

$$F_{\text{cc}} \;=\; D\, R_{\text{cc}}\, D \quad \text{with} \quad D = \operatorname{diag}(\hat{\sigma}_1, \ldots, \hat{\sigma}_N), \tag{18}$$

so that $F_{\text{cc},ij} = \hat{\sigma}_i \hat{\sigma}_j \bar{\rho}$ $(i \neq j)$ and $F_{\text{cc},ii} = \hat{\sigma}_i^2$.

Shrinkage with a Constant-Correlation Target. Replacing F by F_{cc} in the Ledoit–Wolf convex combination yields

$$\hat{\Sigma}_{t+1}^{(\text{shrink,cc})} \;=\; \lambda_{\text{cc}}\, F_{\text{cc}} \;+\; (1 - \lambda_{\text{cc}})\, \hat{\Sigma}_{t+1}, \tag{19}$$

where the optimal intensity $\lambda_{\text{cc}} \in [0, 1]$ minimizes the mean-squared estimation error in the same way as in the general formula, but now using F_{cc} as the target. Closed-form expressions for λ_{cc} are given in Ledoit and Wolf (2004) and implemented in many empirical-finance toolkits. A sufficient condition for positive semi-definiteness is $\bar{\rho} \in [-1/(N-1), 1]$, automatically satisfied by construction from sample correlations.

Eigenvalue Filtering: Given that covariance matrices derived from Method 1 and 2 can still exhibit numerical instability due to noisy eigenvalues, we additionally apply eigenvalue filtering based on Random Matrix Theory principles (Laloux et al., 1999). Specifically, the covariance matrix is decomposed as $\hat{\Sigma}_{t+1}^{(\text{shrink})} = V \Lambda V^{\top}$, where $\Lambda =$

$\text{diag}(\ell_1, \ldots, \ell_N)$ contains eigenvalues sorted in ascending order. Very small eigenvalues that might come out negative in a forecast are adjusted by capping them to a small value (1e-6) close to 0. The filtered covariance matrix is thus:

$$\hat{\Sigma}_{t+1}^{*filtered} = V \Lambda^{(\text{filtered})} V^\top. \tag{20}$$

These final robustified covariance matrices serve as stable and reliable inputs into the subsequent portfolio optimization stage, ensuring well-conditioned numerical properties and enhanced out-of-sample performance.[2]

I.6 Risk-based Portfolio Objectives

Historical notes on GMV (Markowitz, 1952), Maximum-Diversification (Choueifaty and Coignard, 2008) and Equal-Risk-Contribution (Maillard et al., 2010).

Global Minimum Variance (GMV): The GMV portfolio is the one that achieves the lowest possible portfolio variance among all portfolios on the efficient frontier (Markowitz, 1952). Formally, the GMV weights w^{GMV} solve:

$$\min_{w} \quad w^\top \hat{\Sigma}_{t+1}^{*}\, w, \tag{21}$$

$$\text{s.t.} \quad \sum_{i=1}^{N} w_i = 1, \tag{22}$$

$$\text{s.t.} \quad \sum_{i=1}^{N} |w_i| \leq 5, \tag{23}$$

$$w_i \leq 5, \ \forall i, \tag{24}$$

$$w_i \geq -5, \ \forall i, \tag{25}$$

assuming full investment and a leverage cap of 5. The closed-form solution is proportional to the inverse of the covariance: $w^{GMV} \propto (\hat{\Sigma}_{t+1}^{*})^{-1}\mathbf{1}$, where $\mathbf{1}$ is a vector of ones, but for numerical stability SLSQP is used to solve the portfolio with our specialized constraints with high tolerance 10^{-15} and not transaction fees constraints. The GMV portfolio concentrates more weight in assets with lower volatility and lower correlation to others, achieving minimal total risk. The GMV approach is purely risk-driven and serves as a baseline for minimum-risk allocation.

Maximum Diversification (MDP): The Maximum Diversification portfolio, introduced by Choueifaty and Coignard (2008), aims to maximize the "diversification ratio",

[2] Absolutely no covariance forecast suffered from this problem and they all came out naturally positive-definite.

which is defined as the ratio of the weighted average volatility of assets to the portfolio volatility. Given asset volatilities (we can use $\hat{\sigma}_{i,t+1}$) and covariance, the diversification ratio of a weight vector w is:

$$D(w) = \frac{\sum_i w_i \hat{\sigma}_{i,t+1}}{\sqrt{w^\top \hat{\Sigma}^*_{t+1} w}}. \tag{26}$$

The MDP weights w^{MDP} are those that maximize $D(w)$ subject to w summing to 1. In production we augment this formulation so that it mirrors the constraints enforced in our Python optimizer:

$$\min_w \quad -D(w) \; + \; 2\lambda_{\text{fee}} \, \|w - w^{\text{prev}}\| \tag{27}$$

$$\text{s.t.} \quad \sum_{i=1}^{N} w_i = 1, \tag{28}$$

$$\text{s.t.} \quad \sum_{i=1}^{N} |w_i| \leq 5, \tag{29}$$

$$w_i \leq 5, \; \forall i, \tag{30}$$

$$w_i \geq -5, \; \forall i, \tag{31}$$

which leads to a solution where each asset's weight is proportional to its volatility-adjusted by its covariance with others. Intuitively, the MDP emphasizes holding assets that contribute to the portfolio volatility in a balanced way—an asset with high standalone volatility can still get a significant weight if it has low correlation with the rest, thus improving diversification. The MDP tends to allocate more to less correlated assets, achieving a portfolio that is maximally diversified across risk sources (Choueifaty and Coignard, 2008). In practice, we solve this optimization with high-precision tolerance 10^{-8}. The result is a portfolio with typically intermediate risk—higher variance than GMV (because we didn't strictly minimize variance) but more diversified exposures, which can lead to better risk-adjusted returns if the assumption "diversification is the only free lunch" holds.

Equal Risk Contribution (ERC): The Equal Risk Contribution portfolio, also known as the risk parity portfolio, seeks to allocate weights such that each asset contributes equally to the total portfolio risk (Maillard et al., 2010). The risk contribution of asset i in a portfolio w is defined as:

$$RC_i(w) = w_i \frac{\partial}{\partial w_i} \sqrt{w^\top \Sigma w} = w_i [\hat{\Sigma}^*_{t+1} w]_i, \tag{32}$$

which simplifies to $RC_i = w_i(\text{row } i \text{ of } \hat{\Sigma}^*_{t+1}) \cdot w = w_i \sum_j \hat{\sigma}_{ij,t+1} w_j$. In an ERC portfolio, RC_i is the same for all i. There is no closed-form solution in general, but it can be found by solving the system of equations $RC_1 = RC_2 = \cdots = RC_N$ with $\sum w_i = 1, w_i \geq 0$.

To achieve a portfolio where each asset contributes equally to the overall risk, we employ an optimization approach that minimizes the variance of the logarithm of individual risk contributions with a Huber term for transaction fees. This method is robust to scaling and effectively handles numerical instabilities with high precision tolerance 10^{-15}. The optimization problem is formulated as:

$$\min_{w \in \mathcal{W}} \ \text{Var}\left(\log\left(\frac{w_i(\Sigma w)_i}{\sqrt{w^\top \Sigma w}}\right)\right) + \ 2\lambda_{\text{fee}} \|w - w^{\text{prev}}\|, \tag{33}$$

where: $\mathcal{W}$ represents the feasible set defined by constraints such as $\sum w_i = 1$, $w_i \geq 0$, and other portfolio-specific limitations.

This objective function penalizes deviations from equal risk contributions by focusing on the dispersion of the logarithmic risk contributions. The logarithmic transformation ensures scale invariance and enhances numerical stability, especially when dealing with assets that have risk contributions close to zero.

The ERC portfolio tends to put more weight on lower-volatility assets but not as extremely as GMV, because it also accounts for correlation: highly correlated groups of assets together will not dominate the risk. ERC portfolios have gained popularity as a way to balance risk without requiring return forecasts, and they often achieve a compromise between high diversification and low volatility.

Appendix II
Extra Results

Readers interested in every training stage, full portfolio metrics, training/validation loss evolution and code implementation details can access the full repository at:

https://github.com/MarkoBlanusa/Thesis.

Bibliography

Browne, R. (2024). Eu's mica regulation takes effect, forcing stablecoins to seek licences. Accessed 2025-07-01.

Cappiello, L., Engle, R. F., and Sheppard, K. (2006). Asymmetric dynamics in the correlations of global equity and bond returns. *Journal of Financial Econometrics*, 4(4):537–572.

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794.

Choueifaty, Y. and Coignard, Y. (2008). Toward maximum diversification. *Journal of Portfolio Management*, 35(1):40–51.

Corsi, F. (2009). A simple approximate long-memory model of realized volatility. *Journal of Financial Econometrics*, 7(2):174–196.

de Lorenzo, J. (2024). Trump victory sparks crypto rally on hopes of lighter regulation. Accessed 2025-07-01.

Desk, G. T. (2024). Bitcoin price tops $100,000 for first time as trump win fuels crypto fever. Accessed 2025-07-01.

Engle, R. F. (2002). Dynamic conditional correlation: A simple class of multivariate garch models. *Journal of Business & Economic Statistics*, 20(3):339–350.

Goh, K. (2025). Bitcoin dominance hits 59% as altcoins crumble in 2025's first half. Accessed 2025-07-01.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Investopedia Staff (2024). What is the bitcoin halving? Accessed 2025-07-01.

Jiang, Y., Wang, H., and Zhang, L. (2024). Advancements in machine learning for predictive analytics. *Journal of Information Processing*, 150:123–135.

Kohn, S. (2024). Fed delivers third straight quarter-point rate cut, projects two more in 2025. Accessed 2025-07-01.

Laloux, L., Cizeau, P., Bouchaud, J.-P., and Potters, M. (1999). Noise dressing of financial correlation matrices. *Physical Review Letters*, 83(7):1467–1470.

Ledoit, O. and Wolf, M. (2004). A well-conditioned estimator for large-dimensional covariance matrices. *Journal of Multivariate Analysis*, 88(2):365–411.

Liu, H. (2024). Rl portfolio management with draw-down regularisation. *Decision Support Systems*.

Maillard, S., Roncalli, T., and Teiletche, J. (2010). The properties of equally weighted risk contribution portfolios. *Journal of Portfolio Management*, 36(4):60–70.

Markowitz, H. (1952). Portfolio selection. *Journal of Finance*, 7(1):77–91.

of Governors of the Federal Reserve System, B. (2024). Federal reserve issues fomc statement, cuts rate by 50 basis points. Accessed 2025-07-01.

Satter, R. (2024). Sec lawsuit against binance can mostly proceed, us judge rules. Accessed 2025-07-01.

Schulman, J., Moritz, P., Levine, S., Jordan, M. I., and Abbeel, P. (2016). High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Sirignano, J. (2021). Deep learning for limit order books. *Quantitative Finance*.

Sundararaj, O. G. (2024). Crypto markets jump after cpi report; xrp and ai tokens lead gains. Accessed 2025-07-01.

Wheatley, W. (2024). Dmm bitcoin hacked for $308m as 4,502 btc drained. Accessed 2025-07-01.

Yang, X. (2020). Conditional var constrained reinforcement learning. In *NeurIPS*.

Zhang, W. (2023). Rl with order-book features in spot–futures arbitrage. *IEEE Access*.