Neural Network-Driven Volatility Drag Mitigation under Aggressive Leverage

Christian Bongiorno
CentraleSupelec
Gif-sur-Yvette, France
christian.bongiorno@centralesupelec.fr

Efstratios Manolakis
Universitá di Catania
Catania, Italy
stratomanolaki@gmail.com

Rosario Nunzio Mantegna
Università degli Studi di Palermo
Palermo, Italy
rosario.mantegna@unipa.it

Abstract

This paper introduces a compact reformulation of a modular, end-to-end neural network for global minimum-variance portfolio optimization that decouples model complexity from both look-back window length and universe size. A five-parameter hyperbolic weighted moving average combined with a saturating exponential replaces the original 2,400-parameter lag-transformation layer, and a bidirectional gated-recurrent-unit eigencleaning module together with a streamlined marginal-volatility network reduce total learnable parameters from 39,586 to just 2,175. In out-of-sample tests against state-of-the-art nonlinear-shrinkage and risk-parity benchmarks, the compact network attains the lowest realized portfolio variance without compromising expected return. Under long-only constraints, the variance reduction supports substantially higher leverage while maintaining comparable drawdown control. Validation in a high-fidelity trading simulator that incorporates realistic margin-call dynamics confirms enhanced over-leverage resilience. These findings demonstrate that end-to-end variance-minimization architectures can achieve substantial parameter efficiency and robust capital-efficiency gains without sacrificing risk-adjusted performance.

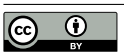
Keywords

Neural Network, Nonlinear Shrinkage, Global Minimum Variance, Leverage

ACM Reference Format:
Christian Bongiorno, Efstratios Manolakis, and Rosario Nunzio Mantegna. 2025. Neural Network-Driven Volatility Drag Mitigation under Aggressive Leverage. In *6th ACM International Conference on AI in Finance (ICAIF '25), November 15–18, 2025, Singapore, Singapore.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3768292.3770370

1 Introduction

In modern quantitative finance, portfolio optimization remains a cornerstone for balancing the trade-off between expected return and risk [22]. By systematically allocating capital across assets, investors seek to construct portfolios that achieve a desired level of return for a given degree of uncertainty. Since Markowitz's foundational mean–variance framework, numerous extensions have targeted improvements in estimation accuracy, model robustness, and practical implementability.



ICAIF '25, Singapore, Singapore

ACM ISBN 979-8-4007-2220-2/25/11
https://doi.org/10.1145/3768292.3770370

A fundamental challenge in mean–variance optimization lies in accurately estimating both asset returns and the covariance matrix [10]. Whereas volatility can often be measured with relatively high precision from high-frequency price data, expected returns exhibit low signal-to-noise ratios and are highly non-stationary. Static estimation techniques, which treat returns as a fixed-parameter estimation problem, fail to adapt to evolving market regimes and may lead to suboptimal allocations [5].

Neural network models offer a promising avenue for addressing non-stationarity by jointly learning estimation and forecasting tasks [17]. Unlike traditional approaches, such as non-linear least-squares methods for covariance estimation, an end-to-end neural architecture can ingest raw price or return series and output optimized weights that implicitly capture time-varying dynamics. In doing so, the model shifts from treating prediction and optimization as separate steps to a unified learning objective that directly minimizes realized portfolio variance.

Lowering realized portfolio volatility has a twofold benefit: it boosts risk-adjusted performance measures (e.g., the Sharpe ratio) and enhances robustness under leverage. Empirical studies have shown that minimum-variance portfolios often outperform alternative rules in both volatility reduction and cumulative return, owing to their implicit structural regularization of asset weights [6].

When leverage is introduced, volatility drag—the erosion of compounded returns due to fluctuating portfolio value—becomes a critical consideration. High-leverage strategies amplify both gains and losses, such that portfolios with elevated variance can suffer disproportionately from drawdowns [15]. Thus, reducing baseline volatility is essential for enabling safe leverage expansion without incurring prohibitive drawdowns.

A further practical constraint arises from margin requirements: leveraged positions may trigger margin calls under adverse price movements, forcing deleveraging at inopportune times. Any optimization method intended for high-leverage applications must therefore account for worst-case price scenarios and ensure that margin buffers remain adequate throughout the investment horizon.

Bongiorno et al. [4] introduced an end-to-end neural network for global minimum-variance optimization that integrated a parameterized lag-transformation layer, an eigencleaning module, and a marginal-volatility network. While effective in reducing realized variance, that model employed approximately $40,000$ learnable parameters and was calibrated for a fixed in-sample window length, limiting its scalability and adaptability to different market universes.

In this paper, we propose a compact reformulation that reduces the total number of learnable parameters to approximately $2,000$

and decouples model complexity from both the number of look-back days and the size of the asset universe. By replacing the high-dimensional lag-transformation weights with a five-parameter formulation based on hyperbolic weighted moving averages and a saturating exponential, we achieve dramatic efficiency gains while preserving variance-minimization performance.

Finally, we validate our approach within a high-fidelity operational simulator that executes market orders at the close price using the most recent available information. The simulator incorporates conservative worst-case price scenarios to model margin-call dynamics, thereby providing a realistic assessment of over-leverage resilience up to 4:1. Through these experiments, we demonstrate that our parameter-efficient network enables higher leverage deployment without sacrificing drawdown control.

2 Volatility Drag under Leverage

In this work, we consider a rebalanced leveraged portfolio in which, at the close of each trading day, holdings are adjusted back to a fixed weight vector $\mathbf{w}$ and the overall leverage factor $\ell > 0$ is reinstated via borrowing or cash adjustments.

Denoting by $\mathbf{r}_t$ the vector of (unlevered) asset returns over day t, the first two moments of the portfolio returns are

$$\mathbb{E}[\mathbf{w}^\top \mathbf{r}_t] = \mu, \qquad \mathbb{V}[\mathbf{w}^\top \mathbf{r}_t] = \mathbf{w}^\top \Sigma \mathbf{w} = \sigma^2, \tag{1}$$

where Σ is the population covariance matrix. Introducing a constant leverage factor ℓ, assuming a zero risk-free rate, the compounded portfolio equity (net asset value) after Δt periods is

$$m_{\Delta t} = m_0 \prod_{t=1}^{\Delta t} \left(1 + \ell\, \mathbf{w}^\top \mathbf{r}_t\right) = m_0 \exp\left[\sum_{t=1}^{\Delta t} \ln\left(1 + \ell\, \mathbf{w}^\top \mathbf{r}_t\right)\right]. \tag{2}$$

When returns are small relative to 1, a second-order expansion $\ln(1+x) \simeq x - \frac{1}{2}x^2$ yields

$$\mathbb{E}\left[\ln(1 + \ell\, \mathbf{w}^\top \mathbf{r}_t)\right] \;\simeq\; \ell\, \mu - \tfrac{1}{2}\, \ell^2 \sigma^2. \tag{3}$$

The second term in Eq. (3) is the volatility drag, i.e., the reduction in compound growth caused by return variability. Practically, volatility drag scales as $\frac{1}{2}\ell^2\sigma^2$, so doubling leverage quadruples the drag. Minimising σ^2 is therefore a prerequisite for sustainable high leverage.

3 Model Architecture

The end-to-end neural network for global minimum-variance optimization consists of four sequential stages. First, raw historical returns for n assets over a look-back window of length Δt_{in} are fed into a parameterized lag-transformation block, which produces a temporally enriched feature tensor (Sec. 3.1). Second, this tensor branches into two parallel modules: a correlation-denoising network that outputs a cleaned inverse eigenvalue spectrum (Sec. 3.2) and a marginal-volatility network that estimates inverse asset-specific volatilities (Sec. 3.3). Third, the outputs of these branches are recombined to assemble an approximate inverse covariance matrix

$$\Sigma_{\text{NN}}^{-1} = \mathbf{D}_{\text{NN}}^{-1} \mathbf{C}_{\text{NN}}^{-1} \mathbf{D}_{\text{NN}}^{-1}. \tag{4}$$

Fourth, we compute the analytic global minimum-variance weights

$$\mathbf{w} \;=\; \frac{\Sigma_{\text{NN}}^{-1}\, \mathbf{1}}{\mathbf{1}^\top \Sigma_{\text{NN}}^{-1}\, \mathbf{1}}, \tag{5}$$

where $\mathbf{1}$ is the n-vector of ones. Importantly, by using fixed-size parameterizations in each module, the total number of learnable parameters is independent of both the temporal depth Δt_{in} and the cross-sectional dimension n, enabling immediate transfer to new asset universes or sampling frequencies without retraining.

We train the entire network by minimizing the realized variance of its portfolio returns. Denote by $\Sigma_{\text{out}} \in \mathbb{R}^{n \times n}$ the future realized covariance matrix of asset returns $\mathbf{R}_{\text{out}} \in \mathbb{R}^{\Delta t_{\text{out}} \times n}$. The loss function is defined as

$$\mathcal{L}(\mathbf{w}, \Sigma_{\text{out}}) \;=\; n\, \mathbf{w}^\top \Sigma_{\text{out}} \mathbf{w}, \tag{6}$$

which corresponds to the out-of-sample variance of the model's portfolio returns. No explicit expected-return term is included, allowing the network to focus solely on variance reduction. Detailed parameterizations and learning dynamics for each module are described in the following subsections.

It is worth stressing that the trained covariance Σ_{NN} can be extracted from the calibrated model and tested with constrained optimizations, such as in this work, where we apply it to the long-only portfolios.

3.1 Lag-Transformation Module

Let the input return series be denoted by $\mathbf{R} \in \mathbb{R}^{\Delta t_{\text{in}} \times n}$, where each $\mathbf{r}_t \in \mathbb{R}^n$ is the vector of asset returns at lag t, with $t = 1$ corresponding to the most recent observation. The lag-transformation module produces an output matrix $\widetilde{\mathbf{R}} \in \mathbb{R}^{\Delta t_{\text{in}} \times n}$ by applying a parametric temporal weighting and nonlinear clipping to each lag. Crucially, this module is parameterized by only five scalars $\boldsymbol{\theta} = \{\theta_1, \theta_2, \theta_3, \theta_4, \theta_5\} \in \mathbb{R}^5_{>0}$, making its complexity independent of both Δt_{in} and n. These five scalars are treated as model parameters and are learned jointly with the rest of the network via backpropagation and gradient descent optimization.

For each lag $t \in \{1, \ldots, \Delta t_{\text{in}}\}$, define the hyperbolic weight

$$\alpha_t \;=\; \theta_1\, t^{-\theta_2}, \tag{7}$$

and the saturating exponential threshold

$$\beta_t \;=\; \theta_3 \;-\; \theta_4\, e^{-\,\theta_5\, t}, \tag{8}$$

where all operations are applied elementwise across assets. The transformed feature for asset i at lag t is given by

$$\widetilde{r}_{t,i} \;=\; \frac{\alpha_t}{\beta_t}\, \tanh\left(\beta_t\, r_{t,i}\right) \quad \text{with} \quad i = 1, \ldots, n. \tag{9}$$

This formulation captures the empirically observed hyperbolic decay and exponential saturation behaviors [4] while requiring only five learned parameters, and thus scales trivially to different window lengths and asset universes.

3.2 Correlation Cleaning Module

We denote by $\mathbf{C} \in \mathbb{R}^{n \times n}$ the empirical correlation matrix computed from the lag-transformed features of Eq. (9). Writing its eigendecomposition as

$$\mathbf{C} \;=\; \mathbf{Q}\, \boldsymbol{\Lambda}\, \mathbf{Q}^\top, \qquad \boldsymbol{\Lambda} \;=\; \mathrm{Diag}(\lambda_1, \ldots, \lambda_n), \tag{10}$$

we seek a rotation-invariant, permutationally equivariant mapping $\mathbf{\Lambda} \mapsto \mathbf{\Lambda}_{\mathrm{NN}}^{-1}$ that produces cleaned inverse eigenvalues [4]. In our compact model, this mapping is still implemented by a bidirectional Recurrent Neural Network (RNN).

Interpreting the ordered spectrum as a sequence, we embed each eigenvalue at rank i into a unified feature vector

$$\mathbf{x}_i \;=\; \left\{\lambda_i,\; q,\; \sqrt{n},\; \sqrt{\Delta t_{\mathrm{in}}}\right\} \quad \text{with} \quad i = 1, \ldots, n. \tag{11}$$

The first component conveys the raw eigenvalue, while the remaining entries respectively encode the aspect ratio $q = n/\Delta t_{\mathrm{in}}$ and the two natural scales $\sqrt{n}$ and $\sqrt{\Delta t_{\mathrm{in}}}$, which together capture both the matrix shape and the magnitude of sampling noise. Although $n/\Delta t_{\mathrm{in}}$ it is algebraically redundant given $\sqrt{n}$ and $\sqrt{\Delta t_{\mathrm{in}}}$, its explicit inclusion empirically accelerates convergence during training.

Then, we process the sequence $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ with a Bidirectional Gated Recurrent Unit (BiGRU) network having $k = 16$ hidden units per direction. Specifically, for $i = 1, \ldots, n$ we compute

$$\boldsymbol{h}_i^{\rightarrow} \;=\; \mathrm{GRU}_k^{\rightarrow}\big(\boldsymbol{h}_{i-1}^{\rightarrow},\, \mathbf{x}_i\big), \qquad \boldsymbol{h}_i^{\leftarrow} \;=\; \mathrm{GRU}_k^{\leftarrow}\big(\boldsymbol{h}_{i+1}^{\leftarrow},\, \boldsymbol{x}_i\big), \tag{12}$$

which we concatenate as $\boldsymbol{h}_i = [\boldsymbol{h}_i^{\rightarrow}; \boldsymbol{h}_i^{\leftarrow}]$. A shared affine projection followed by a softplus activation then yields

$$\lambda_{i,\mathrm{NN}}^{-1} \;=\; \mathrm{softplus}\big(\boldsymbol{\gamma}^{\top}\boldsymbol{h}_i + \omega\big), \quad \text{with} \quad i = 1, \ldots, n. \tag{13}$$

The cleaned inverse correlation is reconstructed as

$$\mathbf{C}_{\mathrm{NN}}^{-1} \;=\; \mathbf{Q}\,\mathrm{Diag}\big(\lambda_{1,\mathrm{NN}}^{-1}, \ldots, \lambda_{n,\mathrm{NN}}^{-1}\big)\,\mathbf{Q}^{\top}. \tag{14}$$

By replacing the original bidirectional LSTM of Ref. [4] with a GRU of $k = 16$ units per direction, we reduce the parameter count of this module from over 30 000 to under 2 000, without degrading expressivity.

Although classical NonLinear Shrinkage (NLS) theory prescribes an analytical form for the optimal shrinkage in the high-dimensional limit, our BiGRU learns an implicit shrinkage function end-to-end under the realized-variance loss. This architecture has been shown to be highly effective in Ref. [4], where a BiRNN-based spectral denoiser outperformed standard NLS estimators in high-dimensional covariance cleaning.

3.3 Marginal Volatility Module

In our architecture, the marginal-volatility module converts each asset's sample standard deviation $\{\widetilde{\sigma}_1, \ldots, \widetilde{\sigma}_n\}$ computed over the lag-transformed return series $\widetilde{\mathbf{R}} \in \mathbb{R}^{\Delta t_{\mathrm{in}} \times n}$ of Eq. (9) into an inverse volatility scale by means of a small, per-asset neural network. Concretely, for each asset i we feed its empirical standard $\widetilde{\sigma}_i$ deviation into a MultiLayer Perceptron (MLP) with a single hidden layer of 8 neurons using a leaky ReLU activation, followed by a softplus output to enforce positivity. Because this network is applied independently to each asset, no cross-asset information is exchanged in this step. The resulting vector of inverse volatilities

$$\mathbf{D}_{\mathrm{NN}}^{-1} = \mathrm{Diag}\left(\sigma_{1,\mathrm{NN}}^{-1}, \ldots, \sigma_{n,\mathrm{NN}}^{-1}\right) \tag{15}$$

4 Real-World Experimental Setup

The experimental framework that follows is designed for direct deployment in production environments. Sec 4.1 presents the universe-selection protocol, which applies market-capitalization and liquidity filters at each rebalancing date and relies exclusively on information available at selection time to eliminate any look-ahead bias. Sec. 4.2 outlines the training pipeline on historical data crafted to ensure robust out-of-sample generalization of the neural network. Sec. 4.3 describes the high-fidelity simulator that reproduces the complete lifecycle of an Interactive Brokers cash-and-margin account. Finally, Sec. 5 specifies the alternative benchmark portfolios employed in this study.

4.1 Stock Selection

We define our investable universe to comprise all U.S. common equities and ADRs listed on the NYSE or NASDAQ between January 1, 1990 and December 31, 2024, explicitly excluding funds and ETFs. To eliminate look-ahead bias, at each rebalance date t we use only information available up to $t - 1$ and drop any security with a delisting within the next $\Delta t_{\mathrm{out}} = 5$ trading days. From this broad set, we apply a two-tiered filter that enforces (i) homogeneous long-term trading records over our five-year calibration window ($\Delta t_{\mathrm{in}} = 1200$ days) and (ii) robust short-term liquidity over the most recent 5–20 trading days.

In the first tier, we require that in every rolling one-year subperiod of the calibration window, a security participates in at least 95% of closing auctions, ensuring continuous price histories. In the second tier, designed to capture extreme liquidity shortfalls immediately prior to trading, we mandate that over the past five days (a) 100% of closing auctions execute, (b) average daily volume represents at least 1% of shares outstanding, and (c) average daily notional turnover represents at least 1% of market capitalization. We further exclude any stock whose shares outstanding fall below 5 million or whose closing price lies outside the \$10–\$2 000 range one day before each rebalance.

To guard against univariate outliers and redundant listings, we then remove all securities whose log-standard deviation of returns falls below the 1.5 Inter-Quartile Range (IQR) threshold, computed separately over the most recent 5-day and 20-day windows; thus avoiding assets with anomalously low risk. This filter prevents the optimization from degenerating into a trivial univariate variance minimization and preserves the truly multivariate nature of our method, ensuring that training and testing leverage cross-asset interactions rather than isolated low-risk outliers. Eventually, we collapse multiple share classes by retaining only the class with the highest market capitalization as of the prior close, and we eliminate one member of any pair whose in-sample return correlation exceeds 0.95 [8]. Finally, we rank the remaining candidates by previous-day market capitalization and select the top $n = 1000$ to form the daily rebalancing universe.

4.2 Training Procedure

The training protocol largely follows that of Ref. [4], with the sole modification that the in-sample window length Δt_{in} is sampled uniformly from the integer interval $\Delta t_{\mathrm{in}} \in [250, 1200]$ for each batch. For each training batch, we uniformly sample the cross-sectional dimension $n \in [50, 350]$ and the in-sample window length Δt_{in} to promote model robustness across aspect ratios $q = n/\Delta t_{\mathrm{in}}$.

Given a randomly drawn reference date t such that the interval $[t - \Delta t_{\mathrm{in}} - 1,\; t + \Delta t_{\mathrm{out}}]$ is contained within the calibration window for the validation year, we select n assets from the filtered investable

universe available at time $t-1$. The input to the model is the matrix of adjusted close-to-close returns over the period $[t-\Delta t_{\text{in}},\ t-1]$. We compute the realized out-of-sample covariance matrix Σ_{out} from returns over the subsequent window $[t,\ t+\Delta t_{\text{out}}]$, where $\Delta t_{\text{out}} = 5$. A one-day shift between in- and out-of-sample windows prevents data leakage from end-of-day prices.

We optimize the portfolio weight $\mathbf{w}$ by minimizing the end-to-end loss of Eq. (6) using the Adam optimizer with an initial learning rate of 10^{-4} and an exponential decay factor of $0.99^{1/500}$ per batch. We employ gradient-norm clipping at a threshold of 1.0 to enhance numerical stability. Training proceeds for 100 epochs, each consisting of 500 gradient steps with a batch size of 32. We repeat this procedure for each of the 24 yearly validation windows (2000-2024), yielding a total of 24 trained networks.

4.3 High-Fidelity Simulator

We simulate an Interactive Brokers cash-and-margin account in continuous calendar time, aligning market events (price moves, dividends, corporate actions) to the exchange calendar supplied with our price data. At any date t, the account holds a vector of shares $s_{t-1} = \{s_{t-1,i}\}_{i=1}^{n}$ and a cash balance c_{t-1}. Overnight and intraday cash dividends are credited to c_t. If c_t ever becomes negative, the account incurs debit interest at a daily rate equal to the prevailing Fed Funds effective rate [1] plus the broker's spread, using a 360-day convention [14]. Positive cash balances above the unpaid \$100 k tranche accrue credit interest according to the broker's tiered schedule [13].

Each rebalance date, before submitting trades, we compute a pre-trade estimate of Net Liquidation Value (NLV)

$$\widehat{\text{NLV}}'_t = c_{t-1} + \sum_{i=1}^{n} s_{t-1,i}\,\hat{p}_{t,i}, \tag{16}$$

where $\hat{p}_{t,i}$ is the primary-exchange opening price on day t (or, if unavailable, the prior adjusted close). Target shares are then set by

$$s_{t,i} = \text{round}\Big(\ell\, w_{t,i}\, \widehat{\text{NLV}}'_t / \hat{p}_{t,i}\Big) \quad \text{with} \quad w_{t,i} \geq 0, \tag{17}$$

and executed as end-of-day market orders at the actual closing price $p_{t,i}$, yielding the post-trade NLV

$$\text{NLV}_t = c_t + \sum_{i=1}^{n} s_{t,i}\, p_{t,i}, \tag{18}$$

with the cash balance c_t adjusted for the day's dividend cash flows and trading fees. We emphasize that, in this work, we do not model any market impact: execution is assumed to occur at the displayed prices without affecting the price formation process.

The portfolio weights $\mathbf{w}_t$ are obtained by performing an external Quadratic Programming (QP) optimization with long-only constraints $w_{t,i} \geq 0$ from the NN internal representation of the covariance matrix Σ_{NN} by inverting Eq. (4). It is important to note that a long-short strategy with a realistic fee structure, although interesting for academic purposes, is less relevant for a practical investing strategy when a reliable guess of the expected returns is not provided.

Trades incur three categories of costs: per-share commissions following the broker's tiered schedule (0.035 ¢/share below 300 k shares traded in the month, 0.020 ¢ above, with a 0.35 \$ minimum per ticket); exchange, clearing, and regulatory fees totalling 0.0845 % of executed notional; and Section 31 SEC fees on sells at 1.157 bp of sell notional [13]. All fees and interest charges are debited immediately from the cash.

Because we permit up to 4:1 gross leverage (maintenance margin requirement of 25 %), we enforce an intraday margin-maintenance check on each trading day. For this purpose, we compute the maintenance margin ratio using each position's intraday low price, thereby adopting a conservative worst-case valuation that assumes all stocks simultaneously realize their intraday minima. To avoid overly pessimistic valuations driven by extreme intraday swings (e.g., during the 2010 flash crash [7]), we floor the low price at

$$\max\big(\text{low},\ 0.85\,\min(\text{open}, \text{close})\big), \tag{19}$$

so that no more than a 15 % loss relative to the open or close is recognized. If this intraday-low valuation would breach the 25 % maintenance threshold, we would immediately liquidate sufficient shares, proportionally to the portfolio allocation, at those intraday lows to restore the margin ratio to 27 % (i.e. 2 pp above the requirement), incurring the same per-share commissions and regulatory fees described above. Although somewhat pessimistic, this low-price–based enforcement mechanism serves as a valuable stress test for the model.

5 Compared Estimators

We benchmark our compact neural estimator against three broad families of allocation techniques: univariate-based rules, risk-parity approaches, and covariance-filtering methods solved via long-only quadratic programming. A naive Equally-Weighted (EW) strategy, which assigns identical weight to each asset, is included too as a simple diversification baseline.

The first, and probably, most popular univariate rule is the Market-Capitalization Weighting (MCW), which allocates in proportion to each asset's float-adjusted market value [12]; this practice is very common since it is the basis of the composition of many indices and most of the ETFs. The Equal-Risk Budget (ERB) scales each asset by the reciprocal of its estimated variance [18], thereby equalizing univariate risk exposures.

Equal Risk Contribution (ERC) extends the ERB principle by solving for weights that equalize marginal risk contributions under the full covariance structure [20]. Relative to the GMV portfolio, ERC typically yields more evenly distributed weights across assets and, therefore, ensures exposure to a broader set of systematic risk factors; this balanced risk allocation is often valued in practice for its enhanced diversification benefits and greater robustness to market regime shifts. Hierarchical Risk Parity (HRP) further exploits the empirical clustering of assets by constructing a dendrogram from the sample correlation matrix and then recursively allocating capital within each cluster so as to balance cluster-level variances [19], potentially capturing sectoral relationships and reducing estimation noise.

On the covariance-filtering side, each estimator first produces a filtered covariance matrix $\hat{\Sigma}$, which we then plug into a long-only QP to obtain the GMV weights under the constraints $w_i \geq 0$ and $\sum_i w_i = 1$. The Maximum Likelihood Estimator (MLE) of the sample covariance serves as a high-noise baseline prior to any

shrinkage. The state-of-the-art Quadratic-Inverse Shrinkage (QIS) estimator of Ledoit–Wolf [16] efficiently regularizes the sample eigenvalues according to Random Matrix Theory (RMT), delivering a well-conditioned Σ_{QIS} estimator that, when fed into the same long-only QP, yields markedly improved out-of-sample risk forecasts. By contrast, the Average Oracle (AO) method [3] abandons time-varying adjustments entirely: it calibrates a fixed set of eigenvalues over an extended historical window (1990–2000) in a stock-agnostic, time-invariant fashion and then applies this identical eigen-spectrum unchanged to all future covariance estimates. Remarkably, this deceptively simple static correction, when combined with the same long-only QP solver, consistently outperforms more sophisticated filtering schemes, achieving higher Sharpe ratios, lower portfolio turnover, and broader effective diversification in long-only backtests [2, 4, 9].

6 Results

We evaluate our model within a rolling–annual calibration framework: at each trading year, the network is retrained on all data up to the previous year, after which portfolio weights are computed every day under long-only constraints and a fixed leverage ℓ is held constant at each daily rebalance. The simulation starts at 2000-01-01 with an initial cash capital $c_1 = 1,000,000$\$, which is compatible with a small hedge fund, and no initial investment. Then the simulation is carried on up to 2024-12-31, and the performances are tracked daily. The following backtesting considers at any point in time t the full investible universe of $n = 1,000$ stocks available at t as described in Sec. 4.1. If on a given day a stock exits from the investable universe, then the position must be closed that day.

Fig. 1 compares the NLV of the neural estimator (upper panel) and the HRP benchmark (lower panel) at $\ell = 3.4$. Vertical red dotted lines denote intraday margin-call-induced liquidations. Despite operating at high leverage, the neural strategy delays its first forced liquidation and experiences markedly fewer subsequent liquidations than HRP, proving better drawdown control under extreme market stress. It is important to note that our simulator assumes exogenous execution prices and therefore does not model market impact. While such effects are negligible for highly liquid, large-cap US stocks in closing auctions [23], their influence grows with larger NLV. Thus, the full NLV trajectory should be interpreted with caution, while the slope is a more reliable measure when rescaled to lower NLV levels.

Fig. 2 plot the compound log-growth rate and the Sharpe ratio (computed on log returns) as functions of leverage over a 25-year simulation. In both metrics, the neural estimator consistently outperforms all competing methods, converting additional leverage into incremental return more efficiently while containing volatility drag.

An exhaustive analysis over the full grid of approximately 400 leverage points ($0.01 \le \ell \le 3.99$ in increments of 0.01) confirms and materially extends the preliminary findings illustrated in Figs. 2–1. Three quantitatively robust regularities emerge by analyzing Tab. 1

First, the neural estimator delays the first forced liquidation to $\ell_{\text{liq}}^{(\text{NN})} = 2.77$, whereas all competing covariance filters trigger margin calls between 2.61 and 2.73. Although the gap appears

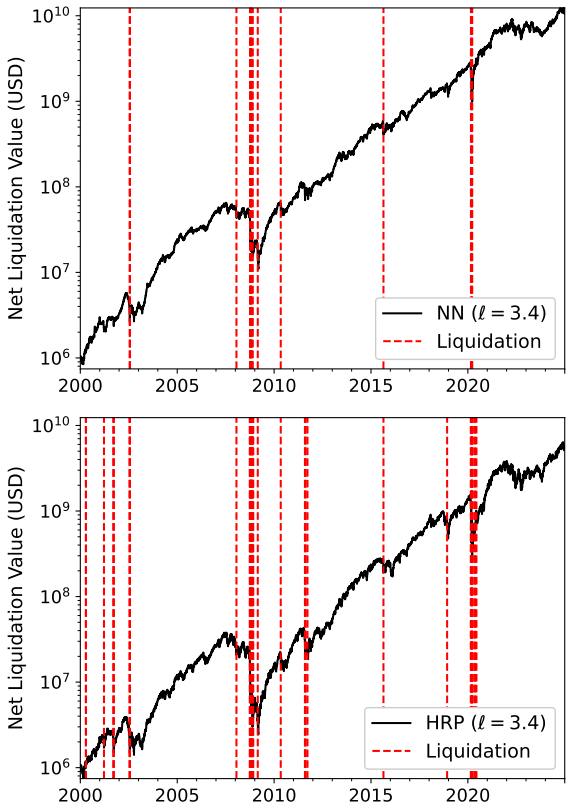


Figure 1: NLV of two high-leverage strategies for NN and HRP on the top 1000 high-capitalized US stocks. The dotted lines represent intraday liquidations. The simulations do not include any market impact.

modest, the nonlinearity of liquidation costs and the convex capital-efficiency penalty beyond $\ell = 2.5$ leverage render this extension economically meaningful, corroborating the visual separation between the solid and dashed curves in Figure 1.

Second, within the frictionless region $0.5 \le \ell \le 2.5$, realised volatility grows approximately linearly with leverage, $\sigma(\ell) \simeq a\,\ell$. A least-squares fit yields the smallest slope for our model, $a_{\text{NN}} = 0.1208$, followed by QIS (0.1284) and AO (0.1275); for comparison, EW requires 0.2089. Mean return increases with slope $b = \partial\mu/\partial\ell$, and only the neural estimator satisfies $b > a$, yielding an incremental efficiency ratio $b_{\text{NN}}/a_{\text{NN}} = 1.08 > 1$. All rival methods fall below unity, confirming that the advantage visible in Fig. 2 in converting leverage into excess return rather than into incremental risk.

Third, the superiority becomes more pronounced in the long-only leverage band ($2.5 \le \ell \le 3.5$). At $\ell = 3$, the neural strategy realises a Sharpe ratio of 1.12 with annualised volatility of 0.36, while the nearest alternative (AO) attains a Sharpe ratio of 0.99 and 0.38, respectively. Maximum Draw-Down (MDD) improves from -0.83 to -0.77 despite identical liquidation counts. To verify that these differences are not attributable to sampling variability, we construct a Model Confidence Set (MCS) [11] on a Sharpe-based

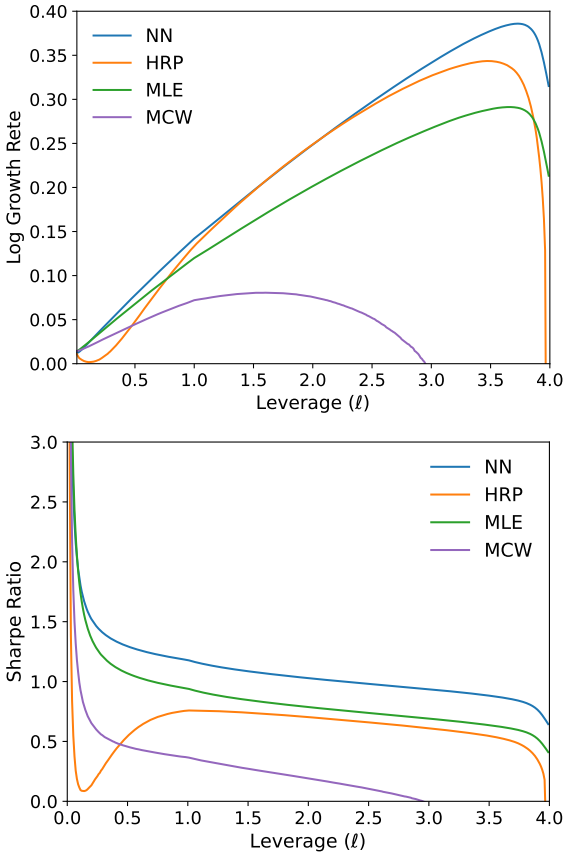


Figure 2: The upper panel represents the out-of-sample log growth rate (see Eq. (3)) measured from the realized NLV of Eq. 18 for different values of the leverage. The lower panel is the Sharpe ratio computed from log returns of the realized NLV as a function of the leverage. Both plots refer to a 25-year simulation of the top 1000 high-capitalized US stocks.

loss: for each method, we compute a daily Sharpe proxy by standardising returns with an exponentially weighted moving-average volatility with memory 0.94 [24], and form the 5% MCS using a stationary block bootstrap with optimally selected block lengths [21] and 100,000 resamples. Under this procedure, only the NN strategy belongs to the superior set at $\ell = 3$ (Table 1). When the legacy NN [4] is added to the candidate set, the MCS does not reject equal performance and, therefore, does not distinguish the two architectures.

Taken together, these results confirm that the proposed compact neural estimator not only minimises variance at unit leverage but also extends the safe-leverage corridor, improves incremental risk-adjusted efficiency, and contains drawdowns at high leverage, thereby translating its parameter efficiency into genuine capital-efficiency gains.

Method	ℓ_{liq}	b/a	Sharpe_3	Vol_3	MDD_3	p-val_3
NN	**2.77**	**1.08**	**1.12**	**0.36**	**−0.77**	**1.000**
AO	2.73	0.95	0.99	0.38	−0.83	0.013
HRP	2.66	0.94	0.88	0.53	−0.89	0.001
ERC	2.63	0.89	0.89	0.57	−0.90	0.002
EW	2.62	0.86	0.86	0.63	−0.92	0.001
MLE	2.69	0.85	0.89	0.38	−0.86	0.002
QIS	2.70	0.82	0.87	0.39	−0.85	0.002
ERB	2.63	0.78	0.79	0.55	−0.91	0.000
MCW	2.61	0.32	0.28	0.59	−0.98	0.000

Table 1: Key risk metrics by covariance estimator. ℓ_{liq} is the lowest leverage at which the first forced liquidation occurs; b/a is the incremental efficiency ratio (slope of mean return divided by slope of volatility) estimated over $0.5 \le \ell \le 2.5$; Sharpe_3, Vol_3, and MDD_3 are the realised Sharpe ratio, annualised volatility, and maximum draw-down at $\ell = 3$. The last column reports the MCS inclusion p-value at $\ell = 3$, computed on the Sharpe-based loss; larger values indicate stronger evidence that the method belongs to the superior set. Boldface marks the best value in each column; for the MCS column, boldface denotes inclusion superior set at 95% confidence level.

7 Discussions

In this paper, we have introduced a compact end-to-end neural network architecture for global minimum-variance portfolio optimization that decouples model complexity from both the look-back window length and the size of the asset universe. By replacing a high-dimensional lag-transformation layer with a five-parameter hyperbolic weighted moving average combined with a saturating exponential, and by substituting a bidirectional gated-recurrent-unit eigencleaning module and a streamlined marginal-volatility network for larger, more cumbersome components, we reduce the total number of learnable parameters from nearly 40,000 to approximately 2,000. This parameter efficiency enables immediate transfer to new universes or sampling frequencies without retraining, while preserving the network's ability to directly minimize realized portfolio variance.

Extensive out-of-sample experiments against state-of-the-art benchmarks—including nonlinear shrinkage estimators and risk-parity rules—demonstrate that our compact network not only attains the lowest realized variance at unit leverage but also extends the safe-leverage corridor, improving the first forced-liquidation threshold from approximately 2.61–2.73 under competing filters to 2.77 under our model. Within the frictionless leverage band, the network achieves an incremental efficiency ratio $b/a > 1$, indicating that incremental mean return grows faster than volatility, and at high-leverage $\ell = 3$, it delivers a Sharpe ratio in excess of 1.1 with tighter drawdown control. High-fidelity simulator tests that incorporate realistic margin dynamics and transaction costs confirm the network's enhanced over-leverage resilience up to 4:1 gross leverage. We note, however, that our simulator currently assumes

exogenous execution prices and does not model market impact; under large orders or stressed liquidity conditions, this simplification may understate realized costs and drawdown effects.

While the compact architecture is purpose-built for variance minimization on daily data and excels in that setting, the original network remains preferable when the objective, constraints, or application context are expected to change. In particular, if one intends to modify the loss function (e.g., to incorporate return-predictive components or alternative risk measures), operate at different sampling frequencies, or adapt to domain-specific structure, the original model's higher parameterization and more general lag-transformation afford greater flexibility. We regard the two architectures as complementary: the compact model is an efficient, task-optimized variance minimizer for daily horizons, whereas the original network provides a more elastic platform for extending the objective or porting to nonstandard environments.

Future work could address extending the portfolio-variance loss to incorporate higher-order co-moments, particularly co-kurtosis, so as to control joint tail risk and potentially allow for more aggressive leverage under stringent extreme-event constraints. Integrating a co-kurtosis term into the objective would enable the optimizer to penalize large simultaneous deviations more directly, thereby improving resilience in crises. Additional avenues include embedding context-dependent eigenvector adjustments to enhance signal extraction across regimes, coupling module parameters for joint adaptation to market states, unifying variance-based risk control with return forecasting via differentiable optimization layers, and explicitly incorporating market-impact models to capture liquidity costs under aggressive trading. Pursuing these directions promises to further strengthen the robustness, tail-risk management, and practical applicability of neural network–based portfolio optimization models.

Code Availability

The source code implementing the proposed architecture is publicly available at https://github.com/bongiornoc/Compact-RIEnet. In addition, upon request, we can provide the calibrated neural network models used in this study to facilitate replication and further research.

Acknowledgments

This work was performed using HPC resources from the "Mésocentre" computing center of CentraleSupélec and École Normale Supérieure Paris-Saclay supported by CNRS and Région Île-de-France (http://mesocentre.centralesupelec.fr/). E.M. acknowledges a fellowship funded by PNRR for the PhD DOT1608375 in Sistemi complessi per le scienze fisiche, socio-economiche e della vita of Catania University.

References

[1] Board of Governors of the Federal Reserve System. 2025. Federal Funds Effective Rate. https://fred.stlouisfed.org/series/FEDFUNDS. Accessed: 2025-06-29.
[2] Christian Bongiorno and Damien Challet. 2024. Covariance matrix filtering and portfolio optimisation: the Average Oracle vs Non-Linear Shrinkage and all the variants of DCC-NLS. *Quantitative Finance* (2024), 1–8.
[3] Christian Bongiorno, Damien Challet, and Grégoire Loeper. 2023. Filtering time-dependent covariance matrices using time-independent eigenvalues. *Journal of Statistical Mechanics: Theory and Experiment* 2023, 2 (2023), 023402.
[4] Christian Bongiorno, Efstratios Manolakis, and Rosario N. Mantegna. 2025. End-to-End Large Portfolio Optimization for Variance Minimization with Neural Networks through Covariance Cleaning. *arXiv preprint arXiv:2507.01918* (2025).
[5] Vijay K. Chopra and William T. Ziemba. 1993. The Effect of Errors in Means, Variances, and Covariances on Optimal Portfolio Choice. *Journal of Portfolio Management* 19, 2 (1993), 6–11.
[6] Roger G. Clarke, Harindra de Silva, and Steven Thorley. 2006. Minimum-Variance Portfolios in the U.S. Equity Market. *Journal of Portfolio Management* 33, 1 (2006), 10–24.
[7] Commodity Futures Trading Commission and Securities and Exchange Commission. 2010. *Findings Regarding the Market Events of May 6, 2010.* Technical Report. U.S. Commodity Futures Trading Commission and U.S. Securities and Exchange Commission. https://www.sec.gov/files/marketevents-report.pdf
[8] Robert F Engle, Olivier Ledoit, and Michael Wolf. 2019. Large dynamic covariance matrices. *Journal of Business & Economic Statistics* 37, 2 (2019), 363–375.
[9] Jean-David Fermanian, Benjamin Poignard, and Panos Xidonas. 2024. Model-based vs. agnostic methods for the prediction of time-varying covariance matrices. *Annals of Operations Research* (2024), 1–38.
[10] Lorenzo Garlappi, Raman Uppal, and Tan Wang. 2007. Portfolio Selection with Parameter and Model Uncertainty: A Multi-Prior Approach. *Review of Financial Studies* 20, 1 (2007), 41–81.
[11] Peter Reinhard Hansen, Asger Lunde, and James M. Nason. 2011. The Model Confidence Set. *Econometrica* 79, 2 (2011), 453–497. doi:10.3982/ECTA5771
[12] Jason C Hsu. 2004. Cap-weighted portfolios are sub-optimal portfolios. *Journal of investment Management* 4, 3 (2004).
[13] Interactive Brokers. 2025. Commissions & Fees . https://www.interactivebrokers.com/en/pricing/commissions-home.php. Accessed: 2025-06-19.
[14] Interactive Brokers. 2025. Interest Rates. https://www.interactivebrokers.com/en/accounts/fees/pricing-interest-rates.php. Accessed: 2025-06-19.
[15] Bruce I. Jacobs and Kenneth N. Levy. 2014. The Unique Risks of Portfolio Leverage: Why Modern Portfolio Theory Fails and How to Fix It. *Journal of Financial Perspectives* (2014), 113–126.
[16] Olivier Ledoit and Michael Wolf. 2022. Quadratic shrinkage for large covariance matrices. *Bernoulli* 28, 3 (2022), 1519–1547.
[17] Yongjae Lee, Jang Ho Kim, Woo Chang Kim, and Frank J Fabozzi. 2024. An Overview of Machine Learning for Portfolio Optimization. *Journal of Portfolio Management* 51, 2 (2024).
[18] Raul Leote, Xiao Lu, and Pierre Moulin. 2012. Demystifying equity risk-based strategies: A simple alpha plus beta description. *Journal of Portfolio Management* 38, 3 (2012), 56–70.
[19] Marcos López de Prado. 2016. Building Diversified Portfolios that Outperform Out-of-Sample. *Journal of Portfolio Management* 42, 4 (2016), 59–69. doi:10.3905/jpm.2016.42.4.059
[20] Sébastien Maillard, Thierry Roncalli, and Jérôme Teïletche. 2010. The properties of equally weighted risk contribution portfolios. *Journal of Portfolio Management* 36, 4 (2010), 60.
[21] Andrew Patton, Dimitris N. Politis, and Halbert White. 2009. Correction to "Automatic Block-Length Selection for the Dependent Bootstrap" by D. Politis and H. White. *Econometric Reviews* 28, 4 (2009), 372–375. doi:10.1080/07474930802459016
[22] Mark Rubinstein. 2002. Markowitz's "Portfolio Selection": A Fifty-Year Retrospective. *Journal of Finance* 57, 3 (2002), 1041–1060.
[23] Mohammed Salek, Damien Challet, and Ioane Muni Toke. 2023. Price impact in equity auctions: zero, then linear. *arXiv preprint arXiv:2301.05677* (2023).
[24] Gilles Zumbach. 2007. *RiskMetrics 2006 Methodology (RM2006).* Technical Report. RiskMetrics Group. Research report; accessed 2025-09-30.