SAME ERROR, DIFFERENT FUNCTION: THE OPTIMIZER AS AN IMPLICIT PRIOR IN FINANCIAL TIME SERIES

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural networks applied to financial time series operate in a regime of underspecification, where predictors achieve comparable out-of-sample error. Using large-scale volatility forecasting for S&P 500 stocks—twelve architecture–optimizer configurations, thirteen seeds each, evaluated on 564,795 held-out observations—we show that comparable test loss conceals measurably different predictors. In eleven of twelve comparisons, models trained from identical initialization and identical data order diverge 1.3 to 7.8 times further than repeated runs of either optimizer alone, ruling out initialization noise as the explanation. On a common scale, predictions differ by 5–17% of the typical forecast error while the models' error magnitudes differ by less than 0.7% of it. The divergence reshapes non-linear response profiles, temporal dependence, and ultimately decisions: volatility-ranked portfolios trace a Sharpe–turnover frontier with up to 58% turnover dispersion at comparable gross Sharpe ratios. In underspecified settings, optimization is a consequential source of inductive bias, and model evaluation must extend beyond scalar loss to functional and decision-level behavior.

1 INTRODUCTION

Model leaderboard ties. In financial time series forecasting, substantially different predictors often achieve indistinguishable out-of-sample performance under standard loss metrics. Recent benchmark efforts and empirical studies document that deep architectures often match, and sometimes fail to exceed, linear econometric baselines (see Hu et al., 2025 or Table 1).

When loss-based evaluation cannot distinguish among models, the practical question is no longer which model performs best on a leaderboard, but whether these models are meaningfully different. This leads to our first question:

> **Question 1:** *Are models with comparable test loss actually interchangeable? Should we employ deep networks at all?*

In practice, one must still deploy **one** model. On what basis should this choice be made?

Optimizer choice appears inconsequential. In vision and language, similar training losses can coexist with substantially different test performance, making optimizer tuning central to model selection (Zhao et al., 2024; Wilson et al., 2017). In financial forecasting, even validation and test losses are often close. Studies commonly prioritize architecture and data while defaulting to Adam (Gu et al., 2020; Chen et al., 2024). This motivates our second question:

> **Question 2:** *Does the optimizer merely affect training efficiency, or does it materially affect the learned function even when the test loss is comparable?*

Overview and takeaways. We study the fundamental task of volatility forecasting for S&P 500 stocks and empirically investigate the questions introduced above. Using a controlled comparison of architectures (MLP, CNN, LSTM, Transformer) and optimizers (SGD, Adam, Muon) with different initializations and hyperparameters, we establish the following:

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Takeaway 1: *Comparable test loss does not imply interchangeable predictors: the pairs (architecture, optimizer) jointly shape the learned* response surfaces.

We measure prediction disagreement on real held-out observations, compare it with variability across repeated fits, and test predictive equivalence within explicit error tolerances. We further characterize the nature of these differences. We reveal simpler versus more complex behavior across optimizers (Fig. 17a); optimizer-difference surfaces exhibit highly structured patterns for sequence models (Fig. 13); and temporal attributions uncover optimizer-dependent "effective receptive fields" (Fig. 4). Crucially, these differences are not merely aesthetic: when forecasts are used to rank assets by volatility, portfolio turnover varies substantially despite comparable risk-adjusted performance (Fig. 34; Appendix G).

Takeaway 2: *Valid benchmarking and model selection should not rely on minor oscillations of the loss, but on interpretability or further financial metrics.*

The paper proceeds as follows: Sections 2–3 establish the forecasting comparison; Section 4 measures and interprets functional divergence, and Section 5 examines its financial consequences.

2 Experimental Framework

We hold data and evaluation fixed while varying architecture and optimizer configuration, measuring their effects on forecasting error and the learned function.

2.1 Task Definition: Volatility Forecasting

We focus on one-step-ahead volatility forecasting, a task characterized by a low signal-to-noise ratio but substantial temporal dependence. Unlike raw returns, daily volatility exhibits strong autocorrelation and clustering, making it an ideal testbed for examining inductive bias under weak identifiability.

We build a survivorship-bias-free dataset of S&P 500 constituents spanning 2000–2024, constructed from CRSP (see Appendices A and B for further details). The target variable is realized variance, proxied by the Garman–Klass estimator (Garman & Klass, 1980) computed from daily high (H_t), low (L_t), open (O_t), and close (C_t) prices:

$$\sigma^2_{GK,t} = \frac{1}{2}\left(\ln\frac{H_t}{L_t}\right)^2 - (2\ln 2 - 1)\left(\ln\frac{C_t}{O_t}\right)^2. \tag{1}$$

Models are trained to predict the log-variance, $\ln(\sigma^2_{GK,t+1})$ using a lookback window of past daily volatilities. While the model predicts log-variance for numerical stability, we discuss results in terms of volatility, as the two are equivalent up to a monotonic transformation.

2.2 Model–Optimizer Pairs

We define a learning system as a tuple $\mathcal{T} = (\mathcal{A}, \mathcal{O})$, where $\mathcal{A}$ and $\mathcal{O}$ denote the network architecture and the optimizer, respectively. This formulation emphasizes that the learned function is jointly determined by representational capacity and training dynamics.

Architectures ($\mathcal{A}$). We consider four standard neural architectures that impose distinct inductive biases over temporal data. Specifically, we consider: (1) MLPs, (2) CNNs (LeCun et al., 1998), (3) LSTMs (Hochreiter & Schmidhuber, 1997b), and (4) Transformer models (Vaswani et al., 2017). All four architectures have been previously applied to financial forecasting tasks, including volatility and return prediction (Gu et al., 2020; Chen et al., 2024).

Optimizers ($\mathcal{O}$). We contrast three optimization methods that differ in update geometry and implicit regularization. Specifically, we consider: (1) SGD, serving as a baseline for non-adaptive first-order optimization; (2) Adam (Kingma & Ba, 2017), the default adaptive moment estimation algorithm; and (3) Muon (Jordan et al., 2024), a recent matrix-aware optimizer designed for high-dimensional training dynamics. Across all optimizers, we fine-tuned hyperparameters as specified in Appendix C.

Experimental grid. We evaluate 4 architectures $\times$ 3 optimizers $=$ 12 learning systems. For each system we tune learning rate and weight decay on a fixed validation split, then fit on merged training and validation data. We report test NMSE over 13 repeated fits per configuration and use separately identified representative checkpoints for single-model diagnostics. The selected LSTM–SGD representative and its 13 repetitions share the same hyperparameters and fitting budget. Appendix C records these distinctions. Predictive and functional evaluations cover all 564,795 available observations in the test set.

2.3 Functional Diagnostics Beyond Error Metrics

We complement forecasting error with prediction distances, controlled input probes and lag attributions.

Distance between predictions. We measure disagreement on identical held-out observations in a common standardized log-variance scale using empirical L^2 distance (Eq. (4)). Its square estimates integrated squared prediction disagreement. This compares predictions directly, rather than comparing each predictor's forecasting error.

Impulse Response Analysis. To map the model's response surface, we measure its global sensitivity by constructing synthetic input vectors. We define the impulse response $\mathcal{R}(k, \delta)$ as the model output when the k-th lag is set to value δ and all other inputs are held at their mean (zero):

$$\mathcal{R}(k, \delta) = \hat{y}(\delta \cdot \mathbf{e}_k), \tag{2}$$

where $\mathbf{e}_k$ is the standard basis vector corresponding to lag k. We vary $\delta \in [-4, 4]$ (in standard deviation units) to visualize the architecture's inherent nonlinearity, independent of specific market contexts; Figure 1 shows the resulting surface for the LSTM.

Functional Difference Surfaces. To compare optimizer-induced solutions, we compute output difference surfaces, for instance

$$\Delta_{M,A}(x) = \hat{y}_{\text{Muon}}(x) - \hat{y}_{\text{Adam}}(x), \tag{3}$$

evaluated over the same input space. Non-planar $\Delta_{M,A}$ reveals input-dependent disagreement along the probe, without identifying parameter-space minima (see Figure 13).

Feature Attribution via SHAP. To evaluate the marginal contribution of each input lag to the model's output, we employ SHapley Additive exPlanations (SHAP) (Lundberg & Lee, 2017). This allows us to determine whether different optimizers prioritize distinct temporal windows (e.g., recent momentum at $t-1$ *vs* quarterly cycles at $t-60$) while achieving comparable predictive error.

3 The Phenomenon of Predictive Equivalence

In domains characterized by low signal-to-noise ratios, such as financial time series, standard aggregate performance metrics often fail to identify a unique predictor. This phenomenon has been formalized as *underspecification* (D'Amour et al., 2020), extending Breiman's *Rashomon Effect* (Breiman, 2001). Previous work has focused on high-noise settings, where models with indistinguishable test performance may assign conflicting predictions to individual inputs (Marx et al., 2020; Black et al., 2022). In overparameterized regimes, training dynamics play a central role in selecting among many compatible solutions (Wilson et al., 2017; Zou et al., 2021). For instance, batch size and stochasticity influence the sharpness of the selected solutions (Keskar et al., 2017). See Appendix H for a detailed discussion.

In this section, we provide an empirical characterization of underspecification in financial volatility forecasting as described in Section 2.1. Specifically, we train all combinations of architectures (MLP, CNN, LSTM, Transformer) and optimizers (Adam, SGD, Muon) and document a systematic phenomenon we refer to as *predictive equivalence*: many optimizer–architecture pairs have small error differences, with the extent of equivalence depending on an explicit tolerance.

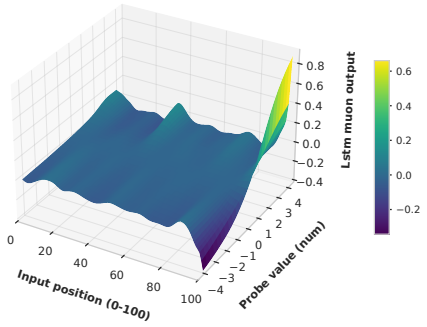


Figure 1: **LSTM Response Surface.** Visualization of the impulse response $\mathcal{R}(k, \delta)$. The LSTM (trained with Muon) exhibits a curved response surface, indicating distinct non-linear sensitivity to volatility shocks at specific lags.

Table 1: **Out-of-Sample Performance (NMSE).** Comparison of generalization error across all Architecture-Optimizer pairs. Mean ± SD over 13 runs. Lower is better.

Panel A: Linear Baselines			
Model	NMSE		
OLS	0.5751		
LASSO	0.5771		
HAR	0.5671		
Panel B: Neural Models			
Architecture	Adam	Muon	SGD
MLP	0.5752 ± 0.0012	0.5770 ± 0.0002	0.5781 ± 0.0040
CNN	0.5747 ± 0.0015	0.5758 ± 0.0007	0.5905 ± 0.0147
LSTM	0.5769 ± 0.0026	0.5783 ± 0.0018	0.5899 ± 0.0203
Transformer	0.5751 ± 0.0016	0.5764 ± 0.0032	0.5761 ± 0.0015

3.1 The Leaderboard Tie

To ensure robustness, we repeat the training procedure for every architecture–optimizer pair using $N = 13$ distinct random seeds for initialization and data shuffling. Table 1 reports the mean NMSE and its standard deviation across seeds. The following observations emerge:

- **Linear baselines remain competitive.** Neural networks do not materially outperform linear benchmarks. OLS and LASSO achieve error levels comparable to even the most complex Transformer models, while HAR (Corsi, 2009)—which encodes hand-crafted multi-horizon aggregates (weekly/monthly/quarterly volatility)—attains the lowest NMSE in the table. This is consistent with prior evidence that parsimonious econometric models with strong domain priors are difficult to beat in low-SNR univariate volatility forecasting (Branco et al., 2024; Vortelinos, 2017).
- **The leaderboard tie persists.** After Holm–Bonferroni correction, only 7 of the 66 comparisons reject equality of expected out-of-sample NMSE. The remaining 59 comparisons (89%) are statistically indistinguishable under this test (Appendix D).
- **Equivalence within stated tolerances.** Two one-sided tests (TOST) assess whether the mean-NMSE difference lies inside $(-\Delta, \Delta)$. At exploratory margins $\Delta = 0.001, 0.002, 0.003, 0.005, 0.010$, respectively $0, 1, 10, 31, 45$ of 66 comparisons pass Holm correction at 5% within each margin. The five margins provide an exploratory sensitivity analysis (Appendix D).
- **Small error differences, distinct predictions.** MLP–Muon has mean NMSE 0.00176 above MLP–Adam: exact equality is rejected (Holm $p = 0.0090$), yet equivalence is supported at $\Delta = 0.005$. Their mean across-optimizer RMS prediction distance is 0.0539, compared with within-optimizer distances 0.0400 for Adam and 0.0119 for Muon (Section 4.2).

The leaderboard leaves room for markedly different predictors. Across a broad set of comparisons, error differences are either statistically unresolved or bounded within explicit tolerances; yet the corresponding functions need not agree on what comes next. We now show that this freedom has structure: optimizer configurations select different responses to the same information, and those differences persist in the forecasts and decisions they produce.

3.2 Robustness via Hyperparameter Optimization

A potential concern in comparing optimization algorithms is that performance parity may arise from suboptimal tuning of one or both methods. To address this concern, we perform optimizer-specific hyperparameter optimization for every architecture–optimizer pair using the Optuna framework (Akiba et al., 2019). We tune the learning rate and weight decay separately for each pair, selecting

the configuration that minimizes validation loss. Search ranges, selected hyperparameters, and implementation details are reported in Appendix C.

The results reported in Table 1 reflect these validation-selected configurations. Strikingly, even after optimizer-specific tuning, performance remains statistically indistinguishable across the pairwise comparisons identified in Section 3.1. Hyperparameter optimization therefore *fails to break the broad leaderboard tie*, in contrast to the pronounced performance differences often observed between optimizers in vision and language tasks.

4 Functional Divergence via Optimization

Section 3 establishes a large matched-loss region. Unlike vision or NLP, where optimizers induce meaningful generalization differences, most pairs here are either statistically unresolved or equivalent within stated tolerances. This predictive equivalence masks deep heterogeneity: qualitatively distinct predictors remain equally compatible with the data (Appendix H).

We investigate the *functions* learned by each pair $(\mathcal{A}, \mathcal{O})$, using the representative model retained for each configuration. Holding architecture fixed, the optimizer acts as a source of inductive bias, selecting qualitatively different decision rules from the same hypothesis class. We call this *functional divergence*: models metrically equivalent in predictive error, yet geometrically and economically distinct in how they map past information into forecasts.

4.1 Functional Divergence

The same forecasting score can conceal markedly different responses to the same input. Figure 2a makes this visible for CNNs: changing the optimizer changes the response to the most recent observation, including how that response bends and dampens large shocks. To see how broadly this disagreement extends, we evaluate the fitted functions on the same held-out observations, rather than only on synthetic probes.

For predictions in a common output scale, their empirical L^2 distance is

$$D(f, g) = \left[\frac{1}{N} \sum_{i=1}^{N} \big(f(x_i) - g(x_i) \big)^2 \right]^{1/2}. \tag{4}$$

Its square estimates $\int (f - g)^2 \, dP_x$. Unlike an error difference, which compares each predictor with the target, D compares their predictions directly. Figure 2b evaluates all $\binom{12}{2} = 66$ representative pairs on all $564{,}795$ test observations. Small differences in NMSE coexist with substantial prediction disagreement: closeness on the leaderboard does not imply closeness of the learned functions.

The response surfaces reveal the structure of this disagreement. Figure 13 in Appendix E plots $\Delta_{M,A}(x) = \hat{y}_{\mathrm{Muon}}(x) - \hat{y}_{\mathrm{Adam}}(x)$ for CNN, LSTM and Transformer. These single-impulse slices are structured across both lag and impulse size: the optimizers select different sensitivities to past information, even within the same architecture.

4.2 Functional Differences across Repeated Fits

Could the observed differences simply reflect initialization? We compare disagreement across optimizers with disagreement among repeated fits of the same configuration. Let $f_{A,i}$ be fit i from optimizer configuration A, with $n_A = 13$ repetitions. Define

$$W_A = \frac{2}{n_A(n_A - 1)} \sum_{i<j} D(f_{A,i}, f_{A,j}), \qquad C_{AB} = \frac{1}{n_A n_B} \sum_{i,j} D(f_{A,i}, f_{B,j}), \tag{5}$$

$$\boxed{R_{AB} = \frac{C_{AB}}{(W_A + W_B)/2}}. \tag{6}$$

W_A measures typical disagreement between repeated fits of the same configuration; C_{AB} measures disagreement across the two configurations. Thus $R_{AB} = 2$ means that across-optimizer disagreement is twice the average within-optimizer disagreement. Because this denominator is an average, a

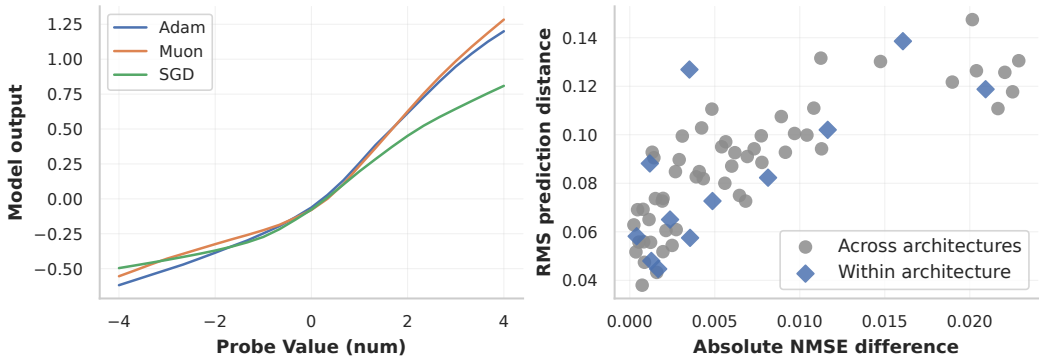


Figure 2: **Similar error, different functions.** (a) CNN response to an impulse at $t-1$, with other inputs held at their standardized means. (b) Absolute NMSE difference versus RMS prediction distance for all 66 representative pairs on the same held-out observations. Diamonds identify comparisons within an architecture. Prediction distances use standardized log-variance units.

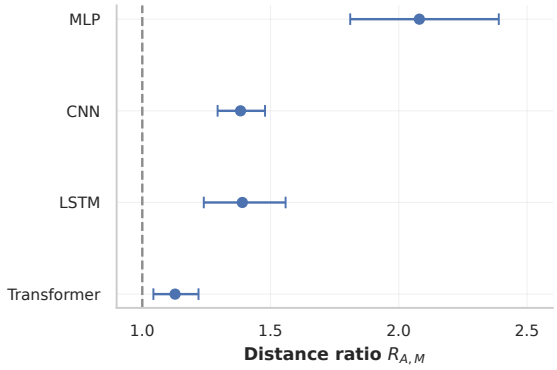


Figure 3: **Functional separation beyond repeated-fit variability.** Adam–Muon ratios $R_{A,M}$, with pointwise 95% seed-level jackknife intervals. The dashed line marks $R = 1$.

Table 2: Components of the Adam–Muon ratio, in RMS prediction units.

	W_A	W_M	$C_{A,M}$
MLP	0.0400	0.0119	0.0539
CNN	0.0475	0.0364	0.0580
LSTM	0.0477	0.0519	0.0693
Transf.	0.0461	0.0505	0.0545

cohort with unusually reproducible fits can inflate R_{AB}; we therefore also report the stricter ratio $R_{AB}^{\max} = C_{AB}/\max(W_A, W_B)$, which compares across-optimizer disagreement with the larger of the two within-optimizer levels (Appendix E.1); Table 2 gives the components it is built from.

For Adam–Muon, $R_{A,M}$ is 2.08 for MLP, 1.38 for CNN, 1.39 for LSTM and 1.13 for Transformer (Figure 3). Their pointwise 95% intervals are $[1.81, 2.39]$, $[1.29, 1.48]$, $[1.24, 1.56]$ and $[1.04, 1.22]$, respectively. Under the stricter denominator the MLP, CNN and LSTM comparisons retain intervals excluding one ($R^{\max} = 1.35\,[1.13, 1.61]$, $1.22\,[1.05, 1.42]$ and $1.33\,[1.18, 1.51]$), whereas Transformer does not ($1.08\,[0.89, 1.31]$): for these three architectures the optimizer moves predictions further than rerunning either optimizer on its own, not merely further than their average. This connects directly to the MLP Adam–Muon predictive comparison: a small but statistically detectable mean-error difference satisfies the exploratory equivalence margin $\Delta = 0.005$ NMSE after Holm correction, while the learned functions remain well separated.

The size of the effect depends on the architecture, and we report where it is largest and where it is not: in LSTM Adam–SGD the across-optimizer distance falls between the two within-optimizer levels, and the interval for R includes one. Read together, these comparisons locate the optimizer as a substantial and measurable source of functional separation, on a scale set by repeated fitting itself. Full results and the uncertainty construction appear in Appendix E.1.

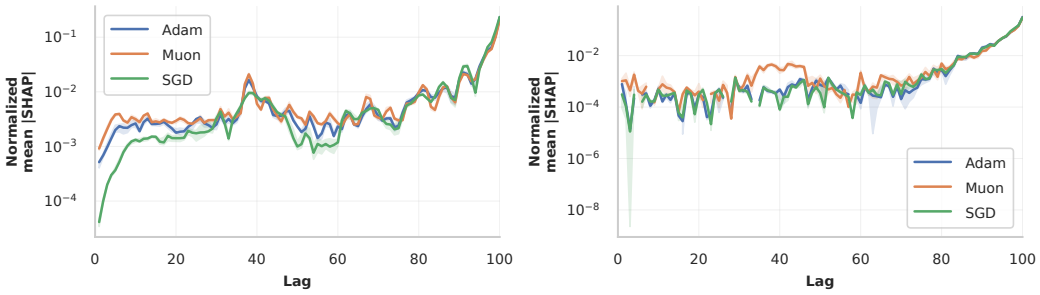


Figure 4: **Optimizer-dependent lag importance.** CNN (left) and LSTM (right). Normalized mean absolute SHAP values across the 100 input lags. Lines show medians over 13 repeated fits; shaded bands show interquartile ranges.

Functional divergence persists under matched initialization and data order. The decisive test removes randomness from the comparison altogether. We refit every architecture from three shared initializations under a common data order (36 runs), so that two optimizers see identical starting weights and identical batches. *In eleven of twelve comparisons the resulting predictors diverge* 1.3 *to* 7.8 *times further than repeated runs of either optimizer alone*—a margin the unpaired seed cohorts cannot resolve, since there the initialization variance sits on both sides of the comparison. The single exception is Transformer Adam–Muon, where the paired and repeated-fit distances coincide. Putting error and disagreement on one scale makes the gap concrete: predictions differ by 5.0–16.7% of the typical forecast error, while the two models' error magnitudes differ by less than 0.7% of it, a ratio of 21 to 2859. MLP SGD–Adam is the extreme case, matching to 2×10^{-5} in RMSE while sitting 0.0593 apart in prediction space. Initialization noise cannot produce this: it is the smaller effect in every single comparison. Appendix E.2 reports all configurations and puts the loss and functional gaps on a common scale.

4.3 Optimizer-Driven Feature Attribution

To understand which parts of the input history distinguish these functions, we analyze lag attribution using SHAP (Lundberg & Lee, 2017). The resulting temporal profiles give structure to the prediction distances: optimizers differ not only in the magnitude of their forecasts, but also in the past information associated with those forecasts.

For CNNs, recent lags dominate, but optimizers differ in their secondary attribution structure. For LSTMs, Muon develops a pronounced mid-horizon band, whereas Adam concentrates more strongly on recent history (Figure 4). The architecture supplies the capacity to use the full input history; the optimization procedure helps determine which temporal structure the fitted predictor actually uses. From a financial perspective, recent-lag emphasis is consistent with short-horizon volatility persistence, while attribution at longer horizons reflects a different temporal weighting of past information (Engle, 1982; Andersen et al., 2007). These describe the predictors' dependence on observed history rather than identifying market mechanisms. The finding stands regardless: at matched forecasting error, optimizers select different effective receptive fields over the past, and so different representations of volatility persistence. Appendix E.3 gives the estimators, shared evaluation samples and attribution checks.

4.4 Mechanism: Curvature Constraints

Why do these optimizers induce functional divergence? Their update rules—plain mini-batch gradients, coordinatewise adaptive scaling, and orthogonalized matrix-valued momentum (Jordan et al., 2024)—can impose different constraints on the regions explored during fitting.

A curvature plateau need not mean that optimization has settled at a fixed minimum: it may instead be held in place by the update dynamics. The two accounts make opposite predictions when the step size is cut. Around a fixed quadratic minimum, a smaller learning rate contracts fluctuations and leaves the Hessian alone (Mandt et al., 2017); under a dynamical constraint, it releases renewed sharpening

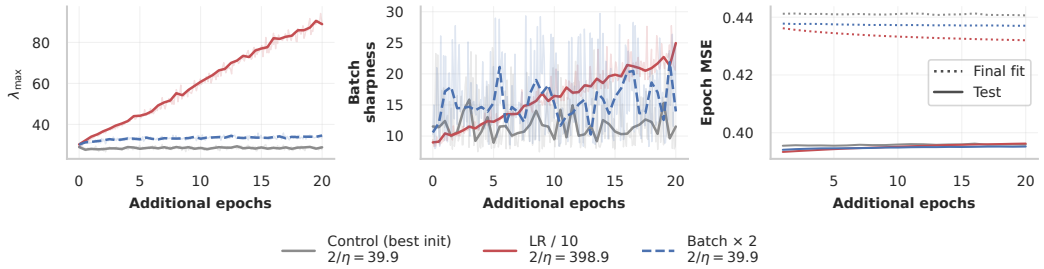


Figure 5: **Releasing the curvature constraint.** MLP–SGD continuations from common fitted weights and optimizer state: unchanged control, LR/10 and batch×2. Left to right: raw $\lambda_{\max}$, batch sharpness, and epoch final fit and test MSE over the same 20 additional epochs. The legend reports $2/\eta$ for each arm. Light curvature traces are individual measurements; dark curvature traces are half-epoch medians. Dotted loss curves show final fit; solid curves show test MSE. Curvature is measured on 4,096 observations; diagnostic batch size is 512, or 1,024 for batch×2.

(Andreyev & Beneventano, 2025). We therefore continue each architecture–optimizer configuration from a common fitted checkpoint and its saved optimizer state, comparing an unchanged control against a tenfold learning-rate reduction and, separately, a doubling of batch size.

The SGD configurations answer clearly. For MLP–SGD (Figure 5), cutting the learning rate lifts the median leading eigenvalue from approximately 29 to 85 over the last fifth of the common window, while the control holds its earlier level—and the loss keeps falling. Sharpening resumes, and the fitted solution was therefore not a minimum the optimizer had settled into: the step size itself was the binding constraint. The CNN diagnostic responds more strongly still. Adaptive optimizers behave differently, and the raw Hessian does not capture their effective update geometry (Cohen et al., 2022a); Appendix F reports every configuration, including those that respond weakly or not at all. Update dynamics, not the loss surface alone, select the region a training run ends in.

5 Behavioral Divergence in Trading Strategies

Two MLP predictors with gross Sharpe ratios of 0.420 and 0.411 demand 16.3% and 10.5% daily turnover respectively: one requires roughly 55% more trading than the other to express what is, on the leaderboard, the same forecast. Functional divergence becomes economic divergence as soon as forecasts drive decisions (Rudin et al., 2022; Fisher et al., 2019). We use volatility-ranked portfolios to measure this decision multiplicity, with construction details and the Sharpe–turnover frontier in Appendix G.

Using the same representative predictors, we construct daily equal-weight volatility-quintile portfolios over May 2000–December 2024. This retrospective diagnostic includes fitting observations; Appendix G reports the held-out comparison. Gross risk-adjusted performance remains close, but daily turnover among neural predictors spans 14.8–20.8% in the low-volatility quintile (Q1) and 10.5–16.6% in the high-volatility quintile (Q5), versus 20.3% and 15.0% for OLS.

The ordering itself depends on architecture: SGD trades least for CNN and LSTM, Muon least for MLP. No optimizer is uniformly the cheaper one to run, which is precisely the point—the leaderboard cannot tell you which allocation rule you are buying, and similar gross performance does not make the induced rules interchangeable.

Turnover reflects changes in forecast-induced rankings, not response-surface smoothness alone, and under common proportional trading costs greater turnover incurs greater costs. These are gross comparisons, and we do not claim net profitability or capacity. What they establish is sharper than that: a predictor's error tells you almost nothing about the trading it will demand, so model evaluation must account for the decisions a predictor induces, not only the loss it attains.

6 CONCLUSION

Predictive equivalence is real and misleading. Financial volatility forecasting exhibits a *Rashomon regime*: many architecture–optimizer comparisons have close mean errors, with equivalence supported within explicitly reported tolerances. But similar NMSE does not imply similar predictions: held-out RMS distances quantify functional disagreement relative to variability across repeated fits. Impulse-response profiles, optimizer-difference surfaces, and lag attributions reveal materially different input–output mappings, which translate into different stability and turnover once forecasts are embedded into trading rules.

Question 1 (interchangeability / "should we use deep nets?"). No: tied predictors are not interchangeable, because "same loss" need not mean "same behavior" downstream. **Yes**: deep networks still matter—not as leaderboard winners, but as flexible hypothesis classes that realize a *menu of admissible signals* (stable *vs* reactive, smooth *vs* aggressive, short- *vs* long-memory).

Question 2 ("is the optimizer just engineering?"). No. When loss is indifferent, optimization is an *implicit prior*: holding data and architecture fixed, optimizer configurations can select *which* admissible function is returned, often determining implementability (e.g., turnover) even when predictive accuracy is comparable. Matched-checkpoint interventions trace this to the training dynamics themselves: in the SGD configurations, reducing the step size releases renewed sharpening and further fitting, so the fitted solution was a constraint of the optimizer rather than a minimum of the loss. **Takeaway.** In underspecified time series, model selection is function selection. When the leaderboard is a tie, choose the function that matches the downstream economic objective—or the optimizer will choose it for you. In finance, the optimizer is part of the model.

AI USE STATEMENT

LLMs assisted with research discussions, implementation, evaluation scripts, literature checking, and manuscript revision, including the presentation of mathematical arguments. No synthetic observations were generated for empirical evaluation. The authors are responsible for reviewing the analyses, interpretations, and final content before submission.

REPRODUCIBILITY STATEMENT

Every result in this paper is generated from recorded artifacts rather than re-derived at write-up time. Appendix B gives the exact CRSP WRDS query defining the survivorship-bias-free panel, and Appendix C records the preprocessing pipeline, the architectures, the validation-selected learning rates, weight decay and batch sizes for all twelve configurations, and the numerical settings used for every curvature measurement. Repeated-fit cohorts are frozen: all thirteen runs per configuration are retained, including loss outliers, and the accompanying reproducibility records identify each checkpoint and fitting budget. Prediction distances are computed on one shared panel of $564{,}795$ test observations with a single preprocessing and output scale, and the paired-initialization runs of Appendix E.2 verify initialization and data-order matching by hashing the loaded tensors and the data permutation.

REFERENCES

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework, 2019. URL https://arxiv.org/abs/1907.10902.

Torben G Andersen, Tim Bollerslev, Francis X Diebold, and Clara Vega. Real-time price discovery in global stock, bond and foreign exchange markets. *Journal of international Economics*, 73(2): 251–277, 2007.

Arseniy Andreyev and Pierfrancesco Beneventano. Edge of stochastic stability: Revisiting the edge of stability for sgd, 2025. URL https://arxiv.org/abs/2412.20553.

Elaine Angelino, Nicholas Larus-Stone, Daniel Alabi, Margo Seltzer, and Cynthia Rudin. Learning certifiably optimal rule lists for categorical data. *Journal of Machine Learning Research*, 18(234): 1–78, 2018. URL https://www.jmlr.org/papers/v18/17-716.html.

Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019. doi: 10.48550/arXiv.1907.02893. URL https://arxiv.org/abs/1907.02893.

Ole E. Barndorff-Nielsen and Neil Shephard. Power and bipower variation with stochastic volatility and jumps. *Journal of Financial Econometrics*, 2(1):1–37, 01 2004. ISSN 1479-8409. doi: 10.1093/jjfinec/nbh001. URL https://doi.org/10.1093/jjfinec/nbh001.

Pierfrancesco Beneventano. On the trajectories of sgd without replacement. *arXiv preprint arXiv:2312.16143*, 2023.

Pierfrancesco Beneventano, Andrea Pinto, and Tomaso Poggio. How neural networks learn the support is an implicit regularization effect of sgd, 2024. URL https://arxiv.org/abs/2406.11110.

Niklas Betz, Thomas L.A. Heil, and Franziska Peter. The best of both worlds? predicting realized volatility based on convolutional neural nets and coloured intraday return images. SSRN Scholarly Paper 4584108, SSRN, September 2023. URL https://ssrn.com/abstract=4584108. Available at SSRN.

Emily Black, Manish Raghavan, and Solon Barocas. Model multiplicity: Opportunities, concerns, and solutions. In *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '22, pp. 850–863, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450393522. doi: 10.1145/3531146.3533149. URL https://doi.org/10.1145/3531146.3533149.

Tim Bollerslev. Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*, 31(3):307–327, 1986. ISSN 0304-4076. doi: https://doi.org/10.1016/0304-4076(86)90063-1. URL https://www.sciencedirect.com/science/article/pii/0304407686900631.

Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri Chatterji, Annie Chen, Kathleen Creel, Jared Quincy Davis, Dora Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, Omar Khattab, Pang Wei Koh, Mark Krass, Ranjay Krishna, Rohith Kuditipudi, Ananya Kumar, Faisal Ladhak, Mina Lee, Tony Lee, Jure Leskovec, Isabelle Levent, Xiang Lisa Li, Xuechen Li, Tengyu Ma, Ali Malik, Christopher D. Manning, Suvir Mirchandani, Eric Mitchell, Zanele Munyikwa, Suraj Nair, Avanika Narayan, Deepak Narayanan, Ben Newman, Allen Nie, Juan Carlos Niebles, Hamed Nilforoshan, Julian Nyarko, Giray Ogut, Laurel Orr, Isabel Papadimitriou, Joon Sung Park, Chris Piech, Eva Portelance, Christopher Potts, Aditi Raghunathan, Rob Reich, Hongyu Ren, Frieda Rong, Yusuf Roohani, Camilo Ruiz, Jack Ryan, Christopher Ré, Dorsa Sadigh, Shiori Sagawa, Keshav Santhanam, Andy Shih, Krishnan Srinivasan, Alex Tamkin, Rohan Taori, Armin W. Thomas, Florian Tramèr, Rose E. Wang, William Wang, Bohan Wu, Jiajun Wu, Yuhuai Wu, Sang Michael Xie, Michihiro Yasunaga, Jiaxuan You, Matei Zaharia, Michael Zhang, Tianyi Zhang, Xikun Zhang, Yuhui Zhang, Lucia Zheng, Kaitlyn Zhou, and Percy Liang. On the opportunities and risks of foundation models, 2022. URL https://arxiv.org/abs/2108.07258.

Rafael R. Branco, Alexandre Rubesam, and Mauricio Zevallos. Forecasting realized volatility: Does anything beat linear models? *Journal of Empirical Finance*, 78:101524, 2024. ISSN 0927-5398. doi: https://doi.org/10.1016/j.jempfin.2024.101524. URL https://www.sciencedirect.com/science/article/pii/S0927539824000598.

Leo Breiman. Statistical modeling: The two cultures (with comments and a rejoinder by the author). *Statistical Science*, 16, 08 2001. doi: 10.1214/ss/1009213726.

Marc-Etienne Brunet, Ashton Anderson, and Richard Zemel. Implications of model indeterminacy for explanations of automated decisions. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 7810–7823. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/33201f38001dd381aba2c462051449ba-Paper-Conference.pdf.

Andrea Bucci. Realized volatility forecasting with neural networks. *Journal of Financial Econometrics*, 18(3):502–531, 06 2020.

Matias D Cattaneo and Boris Shigida. How memory in optimization algorithms implicitly modifies the loss. *arXiv preprint arXiv:2502.02132*, 2025.

Matias D Cattaneo and Boris Shigida. The effect of mini-batch noise on the implicit bias of adam. *arXiv preprint arXiv:2602.01642*, 2026.

Matias D Cattaneo, Jason M Klusowski, and Boris Shigida. On the implicit bias of adam. *arXiv preprint arXiv:2309.00079*, 2023.

Lei Chen and Joan Bruna. Beyond the edge of stability via two-step gradient updates. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 4330–4391. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/chen23b.html.

Luyang Chen, Markus Pelger, and Jason Zhu. Deep learning in asset pricing. *Management Science*, 70(2):714–750, 2024.

Lenaic Chizat, Edouard Oyallon, and Francis Bach. On lazy training in differentiable programming. *arXiv preprint arXiv:1812.07956*, 2019. doi: 10.48550/arXiv.1812.07956. URL https://arxiv.org/abs/1812.07956. Published in NeurIPS 2019.

Kim Christensen, Mathias Siggaard, and Bezirgen Veliyev. A machine learning approach to volatility forecasting*. *Journal of Financial Econometrics*, 21(5):1680–1727, 06 2022. ISSN 1479-8409. doi: 10.1093/jjfinec/nbac020. URL https://doi.org/10.1093/jjfinec/nbac020.

Jeremy M. Cohen, Behrooz Ghorbani, Shankar Krishnan, Naman Agarwal, Sourabh Medapati, Michal Badura, Daniel Suo, David Cardoze, Zachary Nado, George E. Dahl, and Justin Gilmer. Adaptive Gradient Methods at the Edge of Stability, July 2022a. URL http://arxiv.org/abs/2207.14484. arXiv:2207.14484 [cs].

Jeremy M. Cohen, Simran Kaur, Yuanzhi Li, J. Zico Kolter, and Ameet Talwalkar. Gradient descent on neural networks typically occurs at the edge of stability, 2022b. URL https://arxiv.org/abs/2103.00065.

Jeremy M. Cohen, Alex Damian, Ameet Talwalkar, Zico Kolter, and Jason D. Lee. Understanding Optimization in Deep Learning with Central Flows, October 2024. URL http://arxiv.org/abs/2410.24206. arXiv:2410.24206.

Fulvio Corsi. A simple approximate long-memory model of realized volatility. *Journal of Financial Econometrics*, 7(2):174–196, 02 2009. ISSN 1479-8409. doi: 10.1093/jjfinec/nbp001. URL https://doi.org/10.1093/jjfinec/nbp001.

Alex Damian, Eshaan Nichani, and Jason D. Lee. Self-Stabilization: The Implicit Bias of Gradient Descent at the Edge of Stability, April 2023. URL http://arxiv.org/abs/2209.15594. arXiv:2209.15594 [cs, math, stat].

Alexander D'Amour, Katherine Heller, Dan Moldovan, Ben Adlam, Babak Alipanahi, Alex Beutel, Christina Chen, Jonathan Deaton, Jacob Eisenstein, Matthew D. Hoffman, Farhad Hormozdiari, Neil Houlsby, Shaobo Hou, Ghassen Jerfel, Alan Karthikesalingam, Mario Lucic, Yian Ma, Cory McLean, Diana Mincu, Akinori Mitani, Andrea Montanari, Zachary Nado, Vivek Natarajan, Christopher Nielson, Thomas F. Osborne, Rajiv Raman, Kim Ramasamy, Rory Sayres, Jessica Schrouff, Martin Seneviratne, Shannon Sequeira, Harini Suresh, Victor Veitch, Max Vladymyrov, Xuezhi Wang, Kellie Webster, Steve Yadlowsky, Taedong Yun, Xiaohua Zhai, and D. Sculley. Underspecification presents challenges for credibility in modern machine learning, 2020. URL https://arxiv.org/abs/2011.03395.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting, 2024. URL https://arxiv.org/abs/2310.10688.

Marcos López de Prado. Backtesting on synthetic data. In *Advances in Financial Machine Learning*. Wiley, Hoboken, NJ, 2018. Chapter 13.

Steffen Dereich and Arnulf Jentzen. Convergence rates for the adam optimizer. *arXiv preprint arXiv:2407.21078*, 2024.

Gustavo Di-Giorgi, Rodrigo Salas, Rodrigo Avaria, Cristian Ubal, Harvey Rosas, and Romina Torres. Volatility forecasting using deep recurrent neural networks as garch models. *Computational Statistics*, 40(6):3229–3255, 2025. ISSN 1613-9658. doi: 10.1007/s00180-023-01349-1. URL https://doi.org/10.1007/s00180-023-01349-1.

Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1019–1028. PMLR, 2017. URL https://proceedings.mlr.press/v70/dinh17b.html.

R.Glen Donaldson and Mark Kamstra. An artificial neural network-garch model for international stock return volatility. *Journal of Empirical Finance*, 4(1):17–46, 1997. ISSN 0927-5398. doi: https://doi.org/10.1016/S0927-5398(96)00011-4. URL https://www.sciencedirect.com/science/article/pii/S0927539896000114.

Valeria D'Amato, Susanna Levantesi, and Gabriella Piscopo. Deep learning in predicting cryptocurrency volatility. *Physica A: Statistical Mechanics and its Applications*, 596:127158, 2022. ISSN 0378-4371. doi: https://doi.org/10.1016/j.physa.2022.127158. URL https://www.sciencedirect.com/science/article/pii/S0378437122001704.

Robert F Engle. Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation. *Econometrica: Journal of the econometric society*, pp. 987–1007, 1982.

Aaron Fisher, Cynthia Rudin, and Francesca Dominici. All models are wrong, but many are useful: Learning a variable's importance by studying an entire class of prediction models simultaneously. *Journal of Machine Learning Research*, 20(177):1–81, 2019.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations*, 2021. doi: 10.48550/arXiv.2010.01412. URL https://openreview.net/forum?id=6Tm1mposlrM. arXiv:2010.01412.

Mark B. Garman and Michael J. Klass. On the estimation of security price volatilities from historical data. *The Journal of Business*, 53(1):67–78, 1980. ISSN 00219398, 15375374. URL http://www.jstor.org/stable/2352358.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A. Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2:665–673, 2020. doi: 10.1038/s42256-020-00257-z. URL https://arxiv.org/abs/2004.07780. arXiv:2004.07780.

Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. Moment: A family of open time-series foundation models, 2024. URL https://arxiv.org/abs/2402.03885.

Shihao Gu, Bryan Kelly, and Dacheng Xiu. Empirical asset pricing via machine learning. *The Review of Financial Studies*, 33(5):2223–2273, 2020.

Suriya Gunasekar, Blake E. Woodworth, Srinadh Bhojanapalli, Behnam Neyshabur, and Nathan Srebro. Implicit regularization in matrix factorization. In *Advances in Neural Information Processing Systems*, volume 30, pp. 6151–6159, 2017. URL https://papers.nips.cc/paper/7195-implicit-regularization-in-matrix-factorization. arXiv:1705.09280.

Suriya Gunasekar, Jason Lee, Daniel Soudry, and Nathan Srebro. Implicit bias of gradient descent on linear convolutional networks. In *Advances in Neural Information Processing Systems*, volume 31, pp. 9461–9471, 2018a. URL https://papers.nips.cc/paper/8156-implicit-bias-of-gradient-descent-on-linear-convolutional-networks. arXiv:1806.00468.

Suriya Gunasekar, Jason Lee, Daniel Soudry, and Nathan Srebro. Characterizing implicit bias in terms of optimization geometry. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1832–1841. PMLR, 10–15 Jul 2018b. URL https://proceedings.mlr.press/v80/gunasekar18a.html.

James B Heaton, Nick G Polson, and Jan Hendrik Witte. Deep learning for finance: deep portfolios. *Applied Stochastic Models in Business and Industry*, 33(1):3–12, 2017.

Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural Computation*, 9(1):1–42, 1997a. doi: 10.1162/neco.1997.9.1.1. URL https://doi.org/10.1162/neco.1997.9.1.1.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997b.

Robert C. Holte. Very simple classification rules perform well on most commonly used datasets. *Machine Learning*, 11(1):63–90, 1993. doi: 10.1023/A:1022631118932. URL https://link.springer.com/article/10.1023/A:1022631118932.

Yifan Hu, Yuante Li, Peiyuan Liu, Yuxia Zhu, Naiqi Li, Tao Dai, Shu tao Xia, Dawei Cheng, and Changjun Jiang. Fintsb: A comprehensive and practical benchmark for financial time series forecasting, 2025. URL https://arxiv.org/abs/2502.18834.

Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features, 2019. URL https://arxiv.org/abs/1905.02175.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems*, volume 31, pp. 8571–8580, 2018. URL https://papers.nips.cc/paper/8076-neural-tangen-kernel-convergence-and-generalization-in-neural-networks. arXiv:1806.07572.

Stanislaw Jastrzebski, Devansh Arpit, Oliver Astrand, Giancarlo B Kerg, Huan Wang, Caiming Xiong, Richard Socher, Kyunghyun Cho, and Krzysztof J Geras. Catastrophic fisher explosion: Early phase fisher matrix impacts generalization. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 4772–4784. PMLR, 2021. URL https://proceedings.mlr.press/v139/jastrzebski21a.html.

Stanisław Jastrzębski, Zachary Kenton, Nicolas Ballas, Asja Fischer, Yoshua Bengio, and Amos Storkey. On the Relation Between the Sharpest Directions of DNN Loss and the SGD Step Length, December 2019. URL http://arxiv.org/abs/1807.05031. arXiv:1807.05031 [stat].

Stanisław Jastrzębski, Maciej Szymczak, Stanislav Fort, Devansh Arpit, Jacek Tabor, Kyunghyun Cho, and Krzysztof Geras. The Break-Even Point on Optimization Trajectories of Deep Neural Networks. *arXiv:2002.09572 [cs, stat]*, February 2020. URL http://arxiv.org/abs/2002.09572. arXiv:2002.09572.

Ziwei Ji and Matus Telgarsky. The implicit bias of gradient descent on nonseparable data. In Alina Beygelzimer and Daniel Hsu (eds.), *Proceedings of the Thirty-Second Conference on Learning Theory*, volume 99 of *Proceedings of Machine Learning Research*, pp. 1772–1798. PMLR, 25–28 Jun 2019. URL https://proceedings.mlr.press/v99/ji19a.html.

Keller Jordan, Yuchen Jin, Vlado Boza, Jiacheng You, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. Muon: An optimizer for hidden layers in neural networks, 2024. URL https://kellerjordan.github.io/posts/muon/.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima, 2017. URL https://arxiv.org/abs/1609.04836.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. URL https://arxiv.org/abs/1412.6980.

Dinesh Kumar and M. Thenmozhi. Forecasting stock market volatility using convolutional neural networks. *Expert Systems with Applications*, 166:114134, 2021. ISSN 0957-4174. doi: 10.1016/j.eswa.2020.114134. URL https://doi.org/10.1016/j.eswa.2020.114134.

Frederik Kunstner, Alan Milligan, Robin Yadav, Mark Schmidt, and Alberto Bietti. Heavy-tailed class imbalance and why adam outperforms gradient descent on language models. *Advances in Neural Information Processing Systems*, 37:30106–30148, 2024.

Akira Kurosawa. Rashomon. Film, 1950. RKO Radio Pictures.

Andrew Kyle Lampinen, Stephanie C. Y. Chan, and Katherine Hermann. Learned feature representations are biased by complexity, learning order, position, and more, 2024. URL https://arxiv.org/abs/2405.05847.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.

Jaehoon Lee, Lechao Xiao, Samuel S. Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. *arXiv preprint arXiv:1902.06720*, 2019. doi: 10.48550/arXiv.1902.06720. URL https://arxiv.org/abs/1902.06720. Published in NeurIPS 2019.

Radmir Mishelevich Leushuis and Nicolai Petkov. Advances in forecasting realized volatility: a review of methodologies. *Financial Innovation*, 12(1):14, 2026. ISSN 2199-4730. doi: 10.1186/s40854-025-00809-5. URL https://doi.org/10.1186/s40854-025-00809-5.

Yuxuan Liang, Haomin Wen, Yuqi Nie, Yushan Jiang, Ming Jin, Dongjin Song, Shirui Pan, and Qingsong Wen. Foundation models for time series analysis: A tutorial and survey. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '24, pp. 6555–6565. ACM, August 2024. doi: 10.1145/3637528.3671451. URL http://dx.doi.org/10.1145/3637528.3671451.

Bryan Lim and Stefan Zohren. Time-series forecasting with deep learning: a survey. *Philosophical transactions of the royal society a: mathematical, physical and engineering sciences*, 379(2194), 2021.

Jiachang Liu, Chudi Zhong, Boxuan Li, Margo Seltzer, and Cynthia Rudin. Fasterrisk: Fast and accurate interpretable risk scores. In *Advances in Neural Information Processing Systems*, 2022a. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/710344425903 1cc58051f8c9a4868533-Paper-Conference.pdf.

Jiachang Liu, Chudi Zhong, Margo Seltzer, and Cynthia Rudin. Fast sparse classification for generalized linear and additive models. In *Proceedings of the 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pp. 9304–9333. PMLR, 2022b. URL https://proceedings.mlr.press/v151/liu22f.html.

Andrew W. Lo and Manish Singh. Deep-learning models for forecasting financial risk premia and their interpretations. *Quantitative Finance*, 23(6):917–929, 2023. doi: 10.1080/14697688.2023.2203844. URL https://doi.org/10.1080/14697688.2023.2203844.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. doi: 10.48550/arXiv.1711.05101. URL https://openreview.net/forum?id=Bkg6RiCqY7. arXiv:1711.05101.

Yin Lou, Rich Caruana, Johannes Gehrke, and Giles Hooker. Accurate intelligible models with pairwise interactions. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 623–631. Association for Computing Machinery, 2013. doi: 10.1145/2487575.2487579. URL https://doi.org/10.1145/2487575.2487579.

Scott Lundberg and Su-In Lee. A unified approach to interpreting model predictions, 2017. URL https://arxiv.org/abs/1705.07874.

Kaifeng Lyu and Jian Li. Gradient descent maximizes the margin of homogeneous neural networks. In *International Conference on Learning Representations*, 2020. doi: 10.48550/arXiv.1906.05890. URL https://openreview.net/forum?id=SJeLIgBKPS. arXiv:1906.05890.

Chao Ma, Lei Wu, et al. A qualitative study of the dynamic behavior for adaptive gradient algorithms. In *Mathematical and scientific machine learning*, pp. 671–692. PMLR, 2022.

Wenfeng Ma, Yuxuan Hong, and Yuping Song. On stock volatility forecasting under mixed-frequency data based on hybrid rr-midas and cnn-lstm models. *Mathematics*, 12(10):1538, 2024. ISSN 2227-7390. doi: 10.3390/math12101538. URL https://www.mdpi.com/2227-7390/12/10/1538.

Stephan Mandt, Matthew D. Hoffman, and David M. Blei. Stochastic gradient descent as approximate bayesian inference. *Journal of Machine Learning Research*, 18(134):1–35, 2017. URL https://jmlr.org/papers/v18/17-214.html.

Charles T. Marx, Flavio du Pin Calmon, and Berk Ustun. Predictive multiplicity in classification, 2020. URL https://arxiv.org/abs/1909.06677.

Peter McCullagh and John A. Nelder. *Generalized Linear Models*. Chapman and Hall/CRC, 2 edition, 1989.

Duncan McElfresh, Sujay Khandagale, Jonathan Valverde, Vishak C. Prasad, Benjamin Feuer, Chinmay Hegde, Ganesh Ramakrishnan, Micah Goldblum, and Colin White. When do neural nets outperform boosted trees on tabular data? In *Advances in Neural Information Processing Systems*, 2023. doi: 10.5555/3666122.3669459. URL https://arxiv.org/abs/2305.02997.

Hayden McTavish, Chudi Zhong, Reto Achermann, Ilias Karimalis, Jacques Chen, Cynthia Rudin, and Margo Seltzer. Fast sparse decision tree optimization via reference ensembles. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 9604–9613, 2022. URL https://ojs.aaai.org/index.php/AAAI/article/view/21194.

Thomas Moreau, Ievgen Redko, Romain Tavenard, Ambroise Odonnat, Vasilii Feofanov, and Valentina Zantedeschi. Recent advances in time series foundation models: Have we reached the "bert moment"? NeurIPS 2025 Workshop, 2025. URL https://berts-workshop.github.io/.

Dean C. Mountain and Cheng Hsiao. A combined structural and flexible functional approach for modeling energy substitution. *Journal of the American Statistical Association*, 84(405):76–87, 1989. doi: 10.1080/01621459.1989.10478740.

Vardan Papyan, X. Y. Han, and David L. Donoho. Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of Sciences*, 2020. doi: 10.1073/pnas.2015509117. URL https://arxiv.org/abs/2008.08186. arXiv:2008.08186.

Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5301–5310. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/rahaman19a.html.

Eghbal Rahimikia and Ser-Huang Poon. Machine learning for realised volatility forecasting. Workingpaper, Social Science Research Network, United Kingdom, October 2020.

Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. *arXiv preprint arXiv:1904.09237*, 2019.

Rafael Reisenhofer, Xandro Bayer, and Nikolaus Hautsch. Harnet: A convolutional neural network for realized volatility forecasting, 2022. URL https://arxiv.org/abs/2205.07719.

Cynthia Rudin, Chaofan Chen, Zhi Chen, Haiyang Huang, Lesia Semenova, and Chudi Zhong. Interpretable machine learning: Fundamental principles and 10 grand challenges. *Statistic Surveys*, 16:1–85, 2022.

Cynthia Rudin, Chudi Zhong, Lesia Semenova, Margo Seltzer, Ronald Parr, Jiachang Liu, Srikar Katta, Jon Donnelly, Harry Chen, and Zachery Boner. Amazing things come from having many good models, 2024. URL https://arxiv.org/abs/2407.04846.

Marius Saether, Amir Kolic, Tomaso Poggio, and Pierfrancesco Beneventano. Does weight decay enhance training stability?, 2026. URL https://arxiv.org/abs/2605.16622.

Shiori Sagawa, Aditi Raghunathan, Pang Wei Koh, and Percy Liang. An investigation of why overparameterization exacerbates spurious correlations. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 8346–8356. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/sagawa20a.html.

Donald J. Schuirmann. A comparison of the two one-sided tests procedure and the power approach for assessing the equivalence of average bioavailability. *Journal of Pharmacokinetics and Biopharmaceutics*, 15:657–680, 1987. doi: 10.1007/BF01068419.

Lesia Semenova, Cynthia Rudin, and Ronald Parr. On the existence of simpler machine learning models. In *2022 ACM Conference on Fairness Accountability and Transparency*, FAccT '22, pp. 1827–1858. ACM, June 2022. doi: 10.1145/3531146.3533232. URL http://dx.doi.org/10.1145/3531146.3533232.

Lesia Semenova, Harry Chen, Ronald Parr, and Cynthia Rudin. A path to simpler models starts with noise. In *Advances in Neural Information Processing Systems*, volume 36, pp. 3362–3401, 2023.

Harshay Shah, Kaustav Tamuly, Aditi Raghunathan, Prateek Jain, and Praneeth Netrapalli. The pitfalls of simplicity bias in neural networks, 2020. URL https://arxiv.org/abs/2006.07710.

Samuel L Smith, Benoit Dherin, David GT Barrett, and Soham De. On the origin of implicit regularization in stochastic gradient descent. *arXiv preprint arXiv:2101.12176*, 2021.

Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro. The implicit bias of gradient descent on separable data. *Journal of Machine Learning Research*, 19(70):1–57, 2018. URL https://jmlr.org/papers/v19/18-188.html.

Hugo Gobato Souto and Amir Moradi. Forecasting realized volatility through financial turbulence and neural networks. *Economics and Business Review*, 9(2):133–159, April 2023. doi: 10.18559/ebr.2023.2.737. URL https://ideas.repec.org/a/vrs/ecobur/v9y2023i2p133-159n8.html.

Allan Timmermann. Forecast combinations. In Graham Elliott, Clive W. J. Granger, and Allan Timmermann (eds.), *Handbook of Economic Forecasting*, volume 1, chapter 4, pp. 135–196. Elsevier, 2006. doi: 10.1016/S1574-0706(05)01004-9. URL https://doi.org/10.1016/S1574-0706(05)01004-9.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Dimitrios I. Vortelinos. Forecasting realized volatility: Har against principal components combining, neural networks and garch. *Research in International Business and Finance*, 39:824–839, 2017. ISSN 0275-5319. doi: https://doi.org/10.1016/j.ribaf.2015.01.004. URL https://www.sciencedirect.com/science/article/pii/S0275531915000057. Special Issue articles on Finance Reconsidered edited by Dr. Thomas Lagoarde-Segot and Dr. Bernard Paranque, Special issue articles on Recent trends and challenges in financial and commodity markets Edited by Prof. Fredj Jawadi and Prof. Benoît Sevi; Special Issue articles on Recent Topics in Banking and Finance: New Findings and Implications Edited by Prof. Fredj Jawadi and Prof. Wael Louhichi.

Ashia C Wilson, Rebecca Roelofs, Mitchell Stern, Nati Srebro, and Benjamin Recht. The marginal value of adaptive gradient methods in machine learning. *Advances in neural information processing systems*, 30, 2017.

Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified training of universal time series forecasting transformers, 2024. URL https://arxiv.org/abs/2402.02592.

Rui Xin, Chudi Zhong, Zhi Chen, Takuya Takagi, Margo Seltzer, and Cynthia Rudin. Exploring the whole rashomon set of sparse decision trees. In *Advances in Neural Information Processing Systems*, volume 35, pp. 14071–14084, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/5afaa8b4dd18eb1eed055d2d821b58ae-Abstract-Conference.html.

Jinsung Yoon, Daniel Jarrett, and Mihaela van der Schaar. Time-series generative adversarial networks. In *Advances in Neural Information Processing Systems*, volume 32, pp. 5508–5518, 2019. doi: 10.5555/3454287.3454781. URL https://papers.neurips.cc/paper_files/paper/2019/hash/c9efe5f26cd17ba6216bbe2a7d26d490-Abstract.html.

Chao Zhang, Yihuang Zhang, Mihai Cucuringu, and Zhongmin Qian. Volatility forecasting with machine learning and intraday commonality, 2023. URL https://arxiv.org/abs/2202.08962.

Yushun Zhang, Congliang Chen, Tian Ding, Ziniu Li, Ruoyu Sun, and Zhiquan Luo. Why transformers need adam: A hessian perspective. *Advances in neural information processing systems*, 37: 131786–131823, 2024.

Rosie Zhao, Depen Morwani, David Brandfonbrener, Nikhil Vyas, and Sham Kakade. Deconstructing what makes a good optimizer for language models. *arXiv preprint arXiv:2407.07972*, 2024.

Chudi Zhong, Zhi Chen, Jiachang Liu, Margo Seltzer, and Cynthia Rudin. Exploring and interacting with the set of good sparse generalized additive models. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2303.16047.

Tian Zhou, PeiSong Niu, Xue Wang, Liang Sun, and Rong Jin. One fits all:power general time series analysis by pretrained lm, 2023. URL https://arxiv.org/abs/2302.11939.

C. Q. Zhu, M. Tian, Lesia Semenova, Jiachang Liu, J. Xu, J. Scarpa, and Cynthia Rudin. Fast and interpretable mortality risk scores for critical care patients. *arXiv preprint arXiv:2311.13015*, 2023. URL https://arxiv.org/abs/2311.13015.

Liu Ziyin, Hongchao Li, and Masahito Ueda. Noise balance and stationary distribution of stochastic gradient descent. *Physical Review E*, 111(6):065303, 2025.

Difan Zou, Yuan Cao, Yuanzhi Li, and Quanquan Gu. Understanding the generalization of adam in learning neural networks with proper regularization, 2021. URL https://arxiv.org/abs/2108.11371.

APPENDIX

A CHOOSING A DATASET

As a Gedankenexperiment, we postulate the ideal characteristics a dataset should possess:

- **Reliability and provenance.** Data are free of obvious errors, typos, and measurement artifacts; missing values are minimal, explicitly flagged, and documented. Units, definitions, and formats are consistent across series and over the full time span. The origin and transformation pipeline are transparent and traceable to reputable sources.
- **Temporal and cross-sectional breadth.** Time series are sufficiently long to span multiple economic regimes and market cycles (e.g., bull and bear markets, recessions, high/low inflation). The cross-section is large (e.g., hundreds of securities rather than a handful), and each timestamp includes a rich feature set beyond prices alone.
- **Diversity with low average correlation.** The universe covers heterogeneous exposures to known risk factors (e.g., value *vs* growth, small- *vs* large-cap, high *vs* low volatility) and a broad range of industries (technology, healthcare, energy, financials, etc.), reducing concentration risk and cross-series redundancy.
- **Modeling readiness.** The provided variables, or standard transformations (e.g., returns), are reasonably stationary, or the dataset includes guidance to achieve stationarity. The predictive task is clearly defined, with one or more precomputed target variables. Standardized training/validation/test splits and a baseline training protocol are supplied for reproducibility and fair comparison.

Synthetic Data. de Prado (2018); Yoon et al. (2019) propose using synthetic time series, respectively for backtesting/robustness and for training/evaluation via TSTR. While such data are useful for stress testing and mitigating overfitting, a benchmark's purpose is to assess performance against real market outcomes. Hence, we contend that synthetic data are not an appropriate benchmarking substrate, for the following reasons:

1. **Benchmarking aims at reality.** The primary purpose of a benchmark is to assess a model's performance against real market outcomes. Synthetic data are, by construction, a simplified and incomplete model of that reality.
2. **Model-implied worlds.** A synthetic generator is calibrated to statistical regularities observed in historical data. It cannot, by definition, reproduce dynamics that are absent from (or misspecified in) its data-generating assumptions.
3. **Validity is conditional on the generator.** Superior performance on synthetic data merely demonstrates skill at exploiting the rules of the generator, not the market. This invites specification-led overfitting.
4. **Structural breaks are not authentically captured.** While one can program regime shifts by changing generator parameters, that does not replicate *unforeseen* changes in market logic (i.e., true structural breaks). For instance, the post-2008 era of quantitative easing altered cross-asset correlations and the risk-free rate regime in ways not anticipated by pre-crisis models.

In short, optimizing to a synthetic world risks tailoring models to the generator's quirks rather than to markets, undermining external validity as a benchmark.

The Closest Real Dataset to the Ideal Dataset. We propose a dataset constructed from the Center for Research in Security Prices (CRSP), focusing on daily observations of S&P 500 constituents. The construction aligns with the ideal characteristics outlined above:

- **Reliability and provenance.** The dataset is sourced exclusively from CRSP, the institutional standard for academic financial research in the United States, whose data undergo rigorous cleaning and validation. Our construction—joining the CRSP Daily Stock File (DSF) with the S&P 500 constituents list (`msp500list`)—is explicit and reproducible. By

using historical membership windows, it correctly handles index composition changes and therefore avoids survivorship bias.

- **Temporal and cross-sectional breadth.** The sample spans from January 2000 to December 2024, covering multiple recessions (2001, 2008, 2020), the dot-com boom and bust, and a secular decline followed by a rise in interest rates. The cross-section comprises all securities during their tenure in the S&P 500 (roughly 500 at any time). Each observation includes not only the close but also open, high, and low prices, trading volume, and the adjustment factors needed to reflect corporate actions.
- **Diversity with low average correlation.** The S&P 500 is a diversified benchmark spanning all major GICS sectors of the U.S. economy. Its inclusion criteria yield leading, highly liquid firms with heterogeneous exposures (growth *vs* value, differing risk profiles) and a large-cap tilt. Tracking constituent histories captures the evolving sectoral and factor composition of the U.S. market.

B Data Diagnostics and Descriptive Checks

Our working panel contains 3,155,303 daily observations drawn from CRSP for S&P 500 constituents between January 2000 and December 2024. Each row corresponds to a `PERMNO–date` pair and includes returns and standard OHLCV fields plus CRSP adjustment factors (Table 3). Figures 6–8 summarize sectoral coverage, firm size, and cross-sectional dependence.

Table 3: Core variables in the analysis dataset

Variable	Description
`PERMNO`	CRSP permanent security identifier
`date`	Trading day (NYSE calendar)
`ret`	Daily return (CRSP; split/dividend adjusted)
`open, high, low, close`	Daily O/H/L/C prices
`vol`	Daily share volume
`cfacpr`	Price adjustment factor for corporate actions

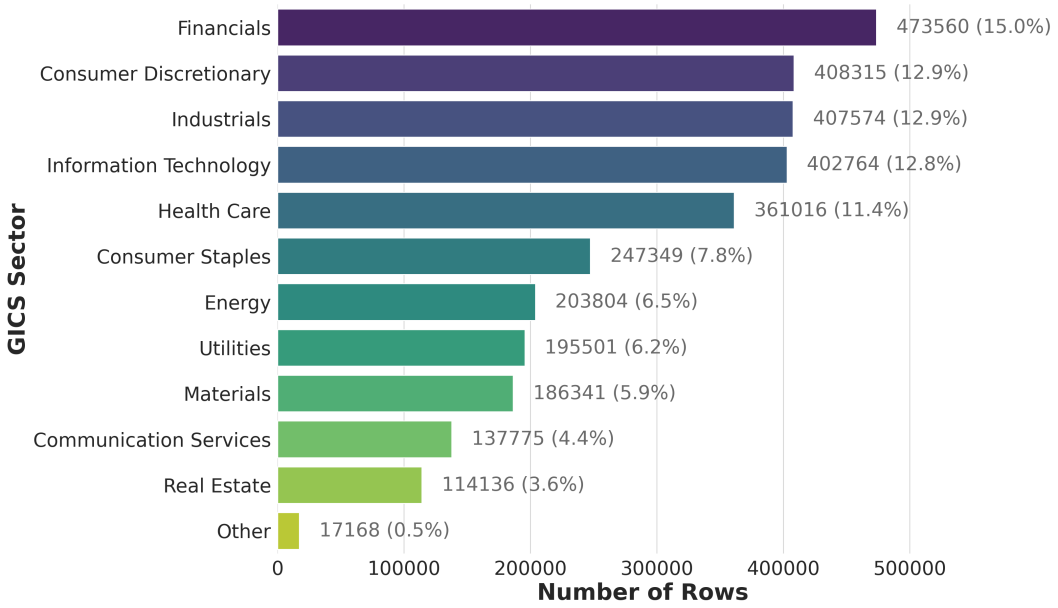


Figure 6: Distribution of observation counts by GICS sector.

The sample is balanced: Financials, Consumer Discretionary, Industrials, Information Technology, and Health Care account for most rows, with no sector too concentrated, and all sectors well represented.

B.1 Firm Size: Market Capitalization

As expected for S&P 500 constituents, capitalization levels are high and, within year, approximately normal on the log scale, with a right tail of very large firms in the latest years (Magnificent 7 effect). The panels show a stable center with modest cyclical shifts across regimes.

B.2 Cross-Sectional Dependence

Correlations are moderate and time-varying: higher in crisis/high-volatility periods (e.g., 2008, 2011, 2020) and lower in calmer years, consistent with standard co-movement dynamics—supporting models with time-varying dependence.

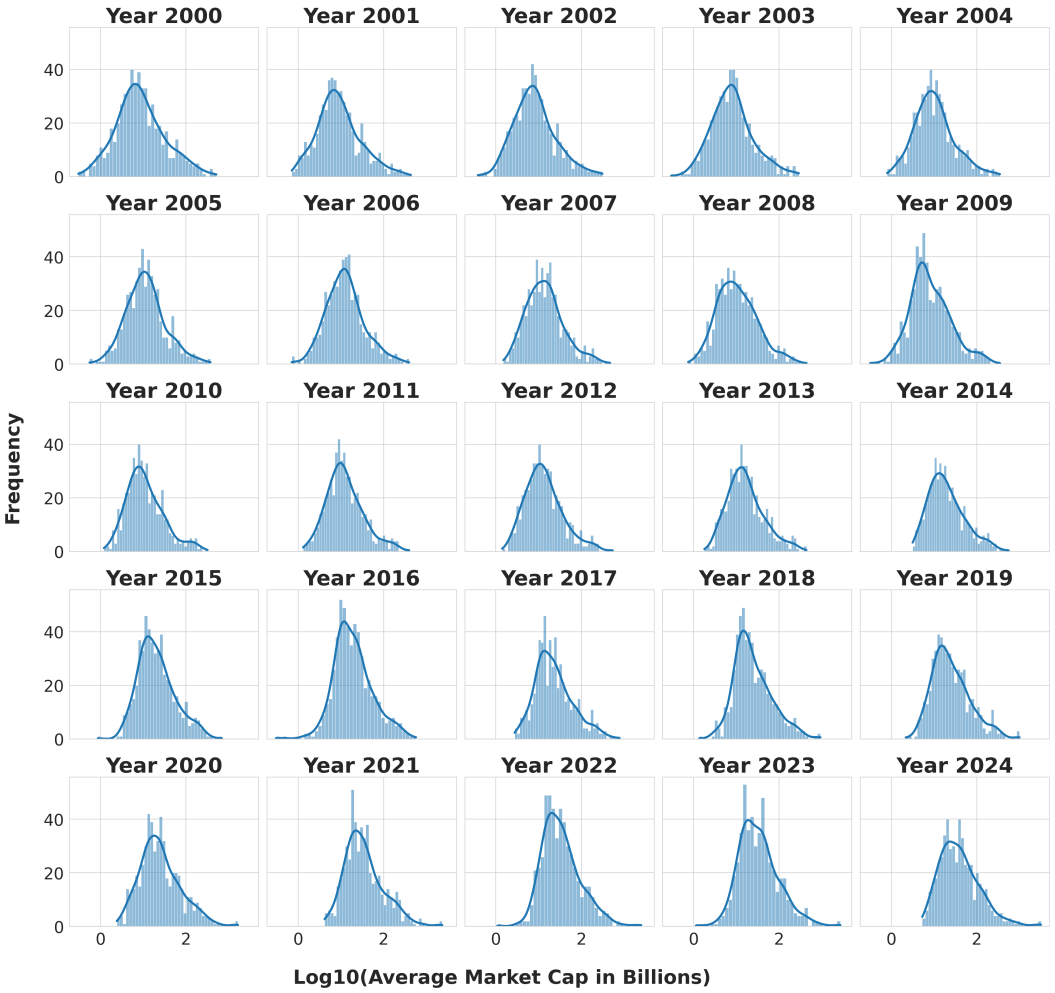


Figure 7: Yearly $\log_{10}$ distribution of average market capitalization per stock.

B.3 Reproducibility Notes

All series are pulled directly from CRSP using the WRDS Python API, using the following query:

```
SELECT
      s.permno,
      s.date,
      s.ret,
      s.prc AS close,
      s.vol,
      s.openprc AS open,
      s.askhi AS high,
      s.bidlo AS low,
      s.cfacpr
FROM crsp.dsf AS s
JOIN crsp.msp500list AS m
  ON s.permno = m.permno
WHERE s.date BETWEEN GREATEST(m.start, DATE '2000-01-01')
                 AND LEAST(m.ending, DATE '2024-12-31')
ORDER BY s.permno, s.date;
```

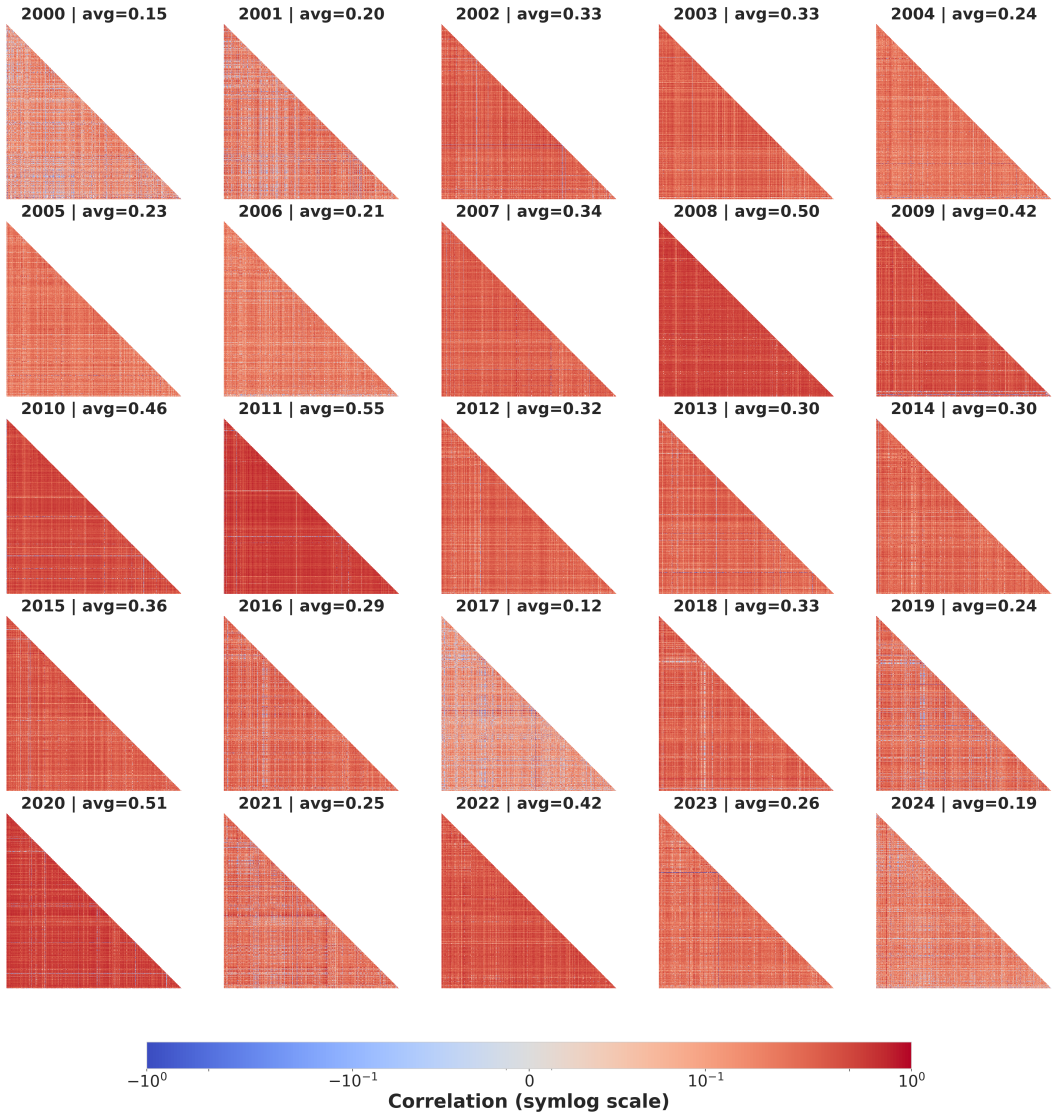


Figure 8: Correlation matrices of daily returns by year.

C Implementation and Experimental Setup

We employ the Garman–Klass estimator to compute a daily realized-variance proxy from OHLC data. We use the original chronological walk-forward construction with train/validation/test ratios 3:1:1. The evaluation includes 2,290,406 merged fitting observations and 564,795 test observations after the original filtering. To ensure numerical stability and mitigate the impact of extreme outliers (some exceeding 300 standard deviations), we apply the original per-asset clipping and log-standardization pipeline. During tuning, its fitting data are training-only; for the retained merged fits, preprocessing is refitted on train+validation. Here "train" in the formulas denotes the permitted fitting pool:

$$x \leftarrow \min\big(\max(x,\, q^{\text{train}}_{0.1\%}),\, q^{\text{train}}_{99.9\%}\big).$$

Subsequently, the values are annualized (scaled by 252) and standardized via z-score normalization using the mean and standard deviation of the training set:

$$x \leftarrow \frac{x - \mu_{\text{train}}}{\sigma_{\text{train}}}.$$

The final input vector comprises the processed log-variances $\mathbf{x}_t = (\widehat{\sigma}^2_{t-L+1}, \ldots, \widehat{\sigma}^2_t)$ used to predict the next-day log-variance y_t. We use $L = 100$.

C.1 Model Architectures

We evaluate four distinct deep learning architectures. All models share a consistent fine-tuning protocol focused on the learning rate and weight decay.

- **LSTM:** We utilize a stacked LSTM with 2 layers of 256 hidden units each. We employ the hidden state of the last time step for prediction (*readout: last*). To preserve signal integrity, we disable dropout and bidirectional processing.
- **CNN:** The convolutional network consists of three 1D-convolution blocks with channel sizes $[64, 128, 256]$, a kernel size of 8, and padding of 1. We use ReLU activation and Adaptive Average Pooling ($k = 16$). The output is flattened and passed through an MLP head with sizes $[512, 256]$.
- **MLP:** A pure Multilayer Perceptron baseline consisting of four fully connected layers with sizes $[512, 256, 256, 128]$ and ReLU activations.
- **Transformer:** We use an encoder-only Transformer with $L = 2$ layers, model dimension $d_{\text{model}} = 128$, and $h = 8$ attention heads. Inputs are projected to d_{model} and combined with learned positional embeddings. The sequence representation is aggregated via mean pooling (*readout: mean*) and passed to an MLP prediction head with sizes $[128, 64]$.

C.2 Linear Baselines

In addition to deep learning models, we evaluate three linear baselines.

- **OLS:** We implement Ordinary Least Squares regression using `statsmodels`, computing the weights directly via the closed-form analytical solution.
- **LASSO:** We employ Least Absolute Shrinkage and Selection Operator regression via `scikit-learn`. The penalization parameter is tuned via grid search over $\alpha \in \{0.001, 0.01, 0.025, 0.05, 0.1, 0.25\}$. We select the α that minimizes the validation loss. Consistent with our general protocol, the final model is refitted on the combined training and validation sets using the optimal α.
- **HAR:** We implement the Heterogeneous Autoregressive model. The model takes the form

$$\hat{\sigma}^2_{t+1} \;=\; \beta_0 \;+\; \sum_{h \in \mathcal{H}} \beta_h \, \overline{\sigma^2_{t,h}}, \qquad \overline{\sigma^2_{t,h}} \;=\; \frac{1}{h} \sum_{i=0}^{h-1} \sigma^2_{t-i}. \tag{7}$$

  We perform a grid search over the horizon set $\mathcal{H}$, comparing the canonical specification $\{1, 5, 22\}$ (daily, weekly, monthly) of Corsi (2009) with the extended specification $\{5, 22, 66\}$ (weekly, monthly, quarterly), and select the latter as it achieves lower validation loss in our setting. Coefficients are estimated by Ordinary Least Squares on the training set and the model is refitted on the combined training and validation sets prior to test-set evaluation.

C.3 Training, Selection, and Frozen Cohorts

Hyperparameters are selected by validation loss, after which models are fitted on the combined training and validation data. We distinguish the representative predictor used for single-model diagnostics from the 13 repeated fits used to measure initialization variability. Reported training losses are therefore train+validation fitting losses. Table 4 gives the selected learning rates, weight decay and batch sizes; the accompanying reproducibility records identify the exact checkpoints and fitting budgets.

Optimizer construction and regularization. SGD uses the recorded momentum (zero for the selected SGD references). Adam is `torch.optim.Adam`, including `fused=True`; Muon updates two-dimensional parameters and uses a companion Adam optimizer for remaining parameters.

Table 4: Selected optimizer settings. The reproducibility records retain full precision.

Configuration	Learning rate η	Weight decay	Batch size
MLP–Adam	0.00019243323	0.01	512
MLP–Muon	0.00019243323	0.91	512
MLP–SGD	0.050134518	0.0018197008	512
CNN–Adam	0.00019243323	0.001	512
CNN–Muon	2.0145131×10^{-5}	0.0002978566	512
CNN–SGD	0.029985065	0.0034651588	512
LSTM–Adam	0.00019243323	0.003	512
LSTM–Muon	0.00019243323	0.42	512
LSTM–SGD	0.091933565	0.0031556215	512
Transformer–Adam	1.0719864×10^{-5}	0	512
Transformer–Muon	1.7073967×10^{-5}	0.086617615	512
Transformer–SGD	0.0030821083	0.00014953811	512

One-dimensional parameters and names ending in `bias` are exempt from decay. All other parameters receive the recorded optimizer-specific decay. Adam is not replaced by AdamW; optimizer state and group-specific settings are preserved. Full parameter-group assignments are provided in the reproducibility records.

Reproducibility records. Table 5 reports fitting and test errors for the representative predictors, together with seed means, standard deviations, and t-based 95% intervals for mean error. Every designated run is retained, including loss outliers. These intervals describe variation across fits conditional on the observed data, not uncertainty across financial histories.

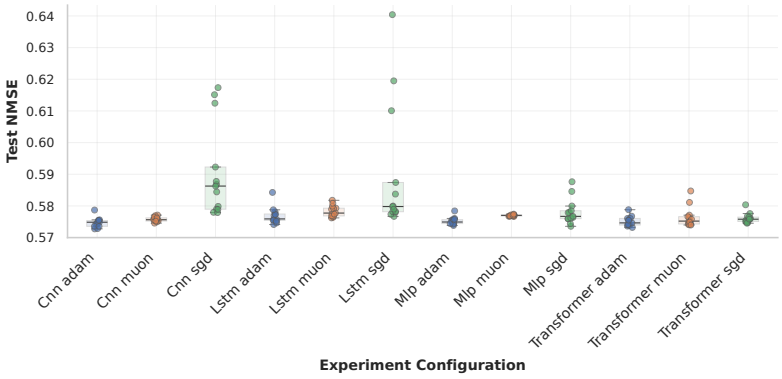


Figure 9: **Repeated-fit error distributions.** All 13 runs per configuration are retained. CNN–SGD and LSTM–SGD exhibit greater run-to-run variability than their adaptive-optimizer counterparts.

C.4 Compute Resources

Training and evaluation use a GPU cluster, including NVIDIA A100 and H100 GPUs. Detailed hardware and runtime records accompany the reproducibility materials.

C.5 Numerical Settings for Curvature Measurements

The measurements in Appendix F use the recorded native edge-of-stochastic-stability implementation. LOBPCG estimates one leading eigenvalue from 4,096 sampled observations, with at most 100 iterations, relative tolerance 0.005, and the existing eigenvector cache. The directional-curvature diagnostic averages individual mini-batch Rayleigh quotients over at most 1,000 estimates, with minimum-estimate setting 20 and stopping tolerance 0.005. Actual estimate counts and Monte Carlo error summaries are retained in the reproducibility records; these describe diagnostic sampling variability, not variation across training seeds.

Table 5: Representative fitting and test errors, with repeated-fit uncertainty. Fit MSE uses merged train+validation data. Seed summaries use the same 13 runs as Table 1; SD measures run-to-run variation, whereas the 95% interval concerns the mean NMSE.

Configuration	Fit MSE	Test MSE	Test NMSE	Seed mean ± SD	95% CI
MLP–Adam	0.4459	0.3927	0.5742	0.5752 ± 0.0012	[0.5745, 0.5759]
MLP–Muon	0.4501	0.3951	0.5778	0.5770 ± 0.0002	[0.5769, 0.5771]
MLP–SGD	0.4390	0.3943	0.5766	0.5781 ± 0.0040	[0.5757, 0.5805]
CNN–Adam	0.4428	0.3924	0.5738	0.5747 ± 0.0015	[0.5738, 0.5757]
CNN–Muon	0.4330	0.3980	0.5820	0.5758 ± 0.0007	[0.5753, 0.5762]
CNN–SGD	0.4539	0.4004	0.5855	0.5905 ± 0.0147	[0.5816, 0.5993]
LSTM–Adam	0.4459	0.3938	0.5758	0.5769 ± 0.0026	[0.5753, 0.5784]
LSTM–Muon	0.4387	0.3971	0.5807	0.5783 ± 0.0018	[0.5773, 0.5794]
LSTM–SGD	0.4630	0.4081	0.5967	0.5899 ± 0.0203	[0.5776, 0.6021]
Tr.–Adam	0.4424	0.3930	0.5747	0.5751 ± 0.0016	[0.5742, 0.5760]
Tr.–Muon	0.4434	0.3941	0.5763	0.5764 ± 0.0032	[0.5745, 0.5783]
Tr.–SGD	0.4454	0.3933	0.5751	0.5761 ± 0.0015	[0.5751, 0.5770]

The diagnostic batches contain 512 observations for control and LR/10 and 1,024 for batch×2. Measurement uses evaluation mode, including disabled Transformer dropout, and the data-MSE objective without weight decay. The continuation uses the saved optimizer state, recorded parameter groups, one reused shuffle and the final partial batch. Source records retain full-precision learning rates, weight-decay settings and measurement cadence. Faint curves show individual curvature measurements and dark curves show half-epoch medians. Loss panels report epoch final-fit and test MSE.

The displayed reference is c/η, with $c = 2$ for SGD, the gradient-descent stability threshold, and $c = 38$ for Adam, the adaptive threshold of Cohen et al. (2022a). We plot the same $38/\eta$ line for Muon purely as a visual reference: no corresponding threshold has been established for it. Labels round this value to one decimal place. These reference labels do not rescale the raw-curvature axes. The adaptive stability analysis of Cohen et al. (2022a) concerns preconditioned curvature, so displaying the reference alongside a raw Hessian does not constitute that stability test. The distinction between the literature batch-sharpness definition and the recorded diagnostic is given in Appendix F.2.

D Robustness of the Leaderboard Tie

For each pair let $\delta = \mu_B - \mu_A$ be the difference in mean test NMSE across the two 13-run cohorts. Welch's two-sided test addresses $H_0 : \delta = 0$. We apply Holm correction to all 66 comparisons as one family. There are 35 raw rejections and seven corrected rejections at 0.05; none of the corrected rejections involves SGD. Larger variability can reduce power to detect a mean difference, so nonrejection must not be interpreted as equivalence.

TOST instead tests $H_0 : \delta \le -\Delta$ or $\delta \ge \Delta$ against $H_1 : -\Delta < \delta < \Delta$ (Schuirmann, 1987). With Welch standard error $s = \sqrt{s_A^2/n_A + s_B^2/n_B}$ and Welch–Satterthwaite degrees of freedom, the two one-sided statistics are $(\widehat{\delta} + \Delta)/s$ and $(\widehat{\delta} - \Delta)/s$. The pair-level p-value is the larger of the two one-sided p-values. Holm correction is then applied across 66 pair-level tests separately at each margin.

The five margins, $0.001, 0.002, 0.003, 0.005, 0.010$, are exploratory sensitivity values. The five families are not jointly corrected; selecting a successful margin after inspection is not a prespecified practical-equivalence conclusion. At these margins the corrected counts are respectively 0, 1, 10, 31, and 45. MLP Adam–Muon illustrates the distinction: equality is rejected, while equivalence passes at 0.005 but not 0.003 after correction. These tests follow the independent-cohort Welch analysis; they quantify run variability conditional on the observed test set, not sampling uncertainty over dependent dates/assets.

Table 6: All 66 Welch tests. Difference is mean(B) minus mean(A). Tr. denotes Transformer.

A	B	$\widehat{\delta}$	Raw p	Holm p
CNN–Adam	LSTM–Muon	+0.00362	0.0000	0.0007
LSTM–Muon	MLP–Adam	-0.00312	0.0000	0.0020
LSTM–Muon	Tr.–Adam	-0.00325	0.0000	0.0031
CNN–Muon	MLP–Muon	+0.00122	0.0001	0.0037
MLP–Adam	MLP–Muon	+0.00176	0.0001	0.0090
CNN–Muon	LSTM–Muon	+0.00258	0.0002	0.0109
CNN–Adam	MLP–Muon	+0.00225	0.0002	0.0112
MLP–Muon	Tr.–Adam	-0.00189	0.0009	0.0538
LSTM–Muon	Tr.–SGD	-0.00227	0.0020	0.1168
CNN–Adam	CNN–SGD	+0.01574	0.0022	0.1276
CNN–SGD	Tr.–Adam	-0.01537	0.0026	0.1477
CNN–SGD	MLP–Adam	-0.01524	0.0028	0.1537
CNN–Muon	CNN–SGD	+0.01470	0.0036	0.1923
CNN–SGD	Tr.–SGD	-0.01439	0.0041	0.2184
CNN–SGD	Tr.–Muon	-0.01409	0.0048	0.2516
CNN–SGD	LSTM–Adam	-0.01360	0.0060	0.3057
CNN–SGD	MLP–Muon	-0.01348	0.0062	0.3093
CNN–Adam	MLP–SGD	+0.00339	0.0112	0.5477
CNN–SGD	MLP–SGD	-0.01234	0.0112	0.5477
CNN–SGD	LSTM–Muon	-0.01212	0.0117	0.5486
LSTM–Muon	MLP–Muon	-0.00136	0.0172	0.7915
CNN–Adam	LSTM–Adam	+0.00214	0.0196	0.8832
CNN–Adam	LSTM–SGD	+0.01514	0.0199	0.8832
MLP–SGD	Tr.–Adam	-0.00303	0.0211	0.9057
LSTM–SGD	Tr.–Adam	-0.01477	0.0224	0.9398
LSTM–SGD	MLP–Adam	-0.01464	0.0233	0.9550
MLP–Adam	MLP–SGD	+0.00290	0.0240	0.9595
LSTM–SGD	MLP–Muon	-0.01288	0.0411	1.0000
LSTM–Muon	Tr.–Muon	-0.00197	0.0650	1.0000
LSTM–SGD	Tr.–Muon	-0.01348	0.0348	1.0000
LSTM–SGD	MLP–SGD	-0.01174	0.0616	1.0000
CNN–Adam	CNN–Muon	+0.00103	0.0433	1.0000
MLP–Adam	Tr.–Muon	+0.00116	0.2335	1.0000
MLP–Adam	Tr.–SGD	+0.00085	0.1266	1.0000
MLP–Muon	MLP–SGD	+0.00114	0.3206	1.0000
MLP–Muon	Tr.–Muon	-0.00060	0.5052	1.0000
MLP–Muon	Tr.–SGD	-0.00090	0.0579	1.0000
MLP–SGD	Tr.–Muon	-0.00174	0.2275	1.0000
MLP–SGD	Tr.–SGD	-0.00204	0.1029	1.0000
Tr.–Adam	Tr.–Muon	+0.00129	0.2038	1.0000
LSTM–SGD	Tr.–SGD	-0.01379	0.0309	1.0000
MLP–Adam	Tr.–Adam	-0.00013	0.8116	1.0000
LSTM–Adam	MLP–Adam	-0.00164	0.0543	1.0000
LSTM–Muon	LSTM–SGD	+0.01152	0.0639	1.0000
CNN–Adam	MLP–Adam	+0.00049	0.3660	1.0000
CNN–Adam	Tr.–Adam	+0.00036	0.5547	1.0000
CNN–Adam	Tr.–Muon	+0.00165	0.1073	1.0000
CNN–Adam	Tr.–SGD	+0.00135	0.0355	1.0000
CNN–Muon	LSTM–Adam	+0.00111	0.1647	1.0000
CNN–Muon	LSTM–SGD	+0.01410	0.0277	1.0000
CNN–Muon	MLP–Adam	-0.00054	0.1764	1.0000
CNN–Muon	MLP–SGD	+0.00236	0.0551	1.0000
CNN–Muon	Tr.–Adam	-0.00067	0.1805	1.0000
LSTM–Muon	MLP–SGD	-0.00022	0.8544	1.0000
CNN–Muon	Tr.–Muon	+0.00062	0.5033	1.0000
CNN–SGD	LSTM–SGD	-0.00060	0.9317	1.0000
LSTM–Adam	LSTM–Muon	+0.00148	0.1063	1.0000
LSTM–Adam	LSTM–SGD	+0.01300	0.0403	1.0000
Tr.–Adam	Tr.–SGD	+0.00098	0.1194	1.0000
LSTM–Adam	MLP–Muon	+0.00011	0.8778	1.0000

A	B	$\widehat{\delta}$	Raw p	Holm p
LSTM–Adam	MLP–SGD	+0.00125	0.3518	1.0000
LSTM–Adam	Tr.–Adam	-0.00178	0.0487	1.0000
LSTM–Adam	Tr.–Muon	-0.00049	0.6717	1.0000
LSTM–Adam	Tr.–SGD	-0.00079	0.3597	1.0000
CNN–Muon	Tr.–SGD	+0.00032	0.5171	1.0000
Tr.–Muon	Tr.–SGD	-0.00030	0.7595	1.0000

Table 7: All 66 TOST comparisons at the five exploratory margins: Holm-adjusted pair-level p-values. Values below .05 support equivalence for that margin. Unadjusted values and mean differences are retained in the complete accompanying CSV.

A	B	.001	.002	.003	.005	.010
CNN–Adam	CNN–Muon	1.0000	1.0000	0.0185	0.0000	$< 10^{-6}$
CNN–Adam	CNN–SGD	1.0000	1.0000	1.0000	1.0000	1.0000
CNN–Adam	LSTM–Adam	1.0000	1.0000	1.0000	0.0512	$< 10^{-6}$
CNN–Adam	LSTM–Muon	1.0000	1.0000	1.0000	0.5592	$< 10^{-6}$
CNN–Adam	LSTM–SGD	1.0000	1.0000	1.0000	1.0000	1.0000
CNN–Adam	MLP–Adam	1.0000	0.3150	0.0035	$< 10^{-6}$	$< 10^{-6}$
CNN–Adam	MLP–Muon	1.0000	1.0000	1.0000	0.0007	$< 10^{-6}$
CNN–Adam	MLP–SGD	1.0000	1.0000	1.0000	1.0000	0.0005
CNN–Adam	Tr.–Adam	1.0000	0.3895	0.0067	0.0000	$< 10^{-6}$
CNN–Adam	Tr.–Muon	1.0000	1.0000	1.0000	0.0514	0.0000
CNN–Adam	Tr.–SGD	1.0000	1.0000	0.3054	0.0001	$< 10^{-6}$
CNN–Muon	CNN–SGD	1.0000	1.0000	1.0000	1.0000	1.0000
CNN–Muon	LSTM–Adam	1.0000	1.0000	0.5846	0.0035	$< 10^{-6}$
CNN–Muon	LSTM–Muon	1.0000	1.0000	1.0000	0.0075	$< 10^{-6}$
CNN–Muon	LSTM–SGD	1.0000	1.0000	1.0000	1.0000	1.0000
CNN–Muon	MLP–Adam	1.0000	0.0365	0.0001	$< 10^{-6}$	$< 10^{-6}$
CNN–Muon	MLP–Muon	1.0000	0.0859	0.0000	$< 10^{-6}$	$< 10^{-6}$
CNN–Muon	MLP–SGD	1.0000	1.0000	1.0000	0.4514	0.0002
CNN–Muon	Tr.–Adam	1.0000	0.3895	0.0044	0.0000	$< 10^{-6}$
CNN–Muon	Tr.–Muon	1.0000	1.0000	0.4705	0.0064	0.0000
CNN–Muon	Tr.–SGD	1.0000	0.0795	0.0009	$< 10^{-6}$	$< 10^{-6}$
CNN–SGD	LSTM–Adam	1.0000	1.0000	1.0000	1.0000	1.0000
CNN–SGD	LSTM–Muon	1.0000	1.0000	1.0000	1.0000	1.0000
CNN–SGD	LSTM–SGD	1.0000	1.0000	1.0000	1.0000	1.0000
CNN–SGD	MLP–Adam	1.0000	1.0000	1.0000	1.0000	1.0000
CNN–SGD	MLP–Muon	1.0000	1.0000	1.0000	1.0000	1.0000
CNN–SGD	MLP–SGD	1.0000	1.0000	1.0000	1.0000	1.0000
CNN–SGD	Tr.–Adam	1.0000	1.0000	1.0000	1.0000	1.0000
CNN–SGD	Tr.–Muon	1.0000	1.0000	1.0000	1.0000	1.0000
CNN–SGD	Tr.–SGD	1.0000	1.0000	1.0000	1.0000	1.0000
LSTM–Adam	LSTM–Muon	1.0000	1.0000	1.0000	0.0124	$< 10^{-6}$
LSTM–Adam	LSTM–SGD	1.0000	1.0000	1.0000	1.0000	1.0000
LSTM–Adam	MLP–Adam	1.0000	1.0000	1.0000	0.0124	$< 10^{-6}$
LSTM–Adam	MLP–Muon	1.0000	0.6874	0.0510	0.0005	$< 10^{-6}$
LSTM–Adam	MLP–SGD	1.0000	1.0000	1.0000	0.1464	0.0000
LSTM–Adam	Tr.–Adam	1.0000	1.0000	1.0000	0.0204	$< 10^{-6}$
LSTM–Adam	Tr.–Muon	1.0000	1.0000	0.8150	0.0124	$< 10^{-6}$
LSTM–Adam	Tr.–SGD	1.0000	1.0000	0.4130	0.0018	$< 10^{-6}$
LSTM–Muon	LSTM–SGD	1.0000	1.0000	1.0000	1.0000	1.0000
LSTM–Muon	MLP–Adam	1.0000	1.0000	1.0000	0.0728	$< 10^{-6}$
LSTM–Muon	MLP–Muon	1.0000	1.0000	0.1634	0.0002	$< 10^{-6}$
LSTM–Muon	MLP–SGD	1.0000	1.0000	0.7739	0.0197	0.0000
LSTM–Muon	Tr.–Adam	1.0000	1.0000	1.0000	0.1992	$< 10^{-6}$
LSTM–Muon	Tr.–Muon	1.0000	1.0000	1.0000	0.1087	0.0000
LSTM–Muon	Tr.–SGD	1.0000	1.0000	1.0000	0.0075	$< 10^{-6}$
LSTM–SGD	MLP–Adam	1.0000	1.0000	1.0000	1.0000	1.0000

A	B	.001	.002	.003	.005	.010
LSTM–SGD	MLP–Muon	1.0000	1.0000	1.0000	1.0000	1.0000
LSTM–SGD	MLP–SGD	1.0000	1.0000	1.0000	1.0000	1.0000
LSTM–SGD	Tr.–Adam	1.0000	1.0000	1.0000	1.0000	1.0000
LSTM–SGD	Tr.–Muon	1.0000	1.0000	1.0000	1.0000	1.0000
LSTM–SGD	Tr.–SGD	1.0000	1.0000	1.0000	1.0000	1.0000
MLP–Adam	MLP–Muon	1.0000	1.0000	0.0671	0.0000	$< 10^{-6}$
MLP–Adam	MLP–SGD	1.0000	1.0000	1.0000	1.0000	0.0003
MLP–Adam	Tr.–Adam	1.0000	0.0725	0.0008	$< 10^{-6}$	$< 10^{-6}$
MLP–Adam	Tr.–Muon	1.0000	1.0000	1.0000	0.0171	0.0000
MLP–Adam	Tr.–SGD	1.0000	1.0000	0.0177	0.0000	$< 10^{-6}$
MLP–Muon	MLP–SGD	1.0000	1.0000	1.0000	0.0708	0.0000
MLP–Muon	Tr.–Adam	1.0000	1.0000	0.5846	0.0003	$< 10^{-6}$
MLP–Muon	Tr.–Muon	1.0000	1.0000	0.4383	0.0066	0.0000
MLP–Muon	Tr.–SGD	1.0000	0.7499	0.0109	0.0000	$< 10^{-6}$
MLP–SGD	Tr.–Adam	1.0000	1.0000	1.0000	1.0000	0.0003
MLP–SGD	Tr.–Muon	1.0000	1.0000	1.0000	0.4000	0.0001
MLP–SGD	Tr.–SGD	1.0000	1.0000	1.0000	0.3330	0.0001
Tr.–Adam	Tr.–Muon	1.0000	1.0000	1.0000	0.0242	0.0000
Tr.–Adam	Tr.–SGD	1.0000	1.0000	0.0803	0.0000	$< 10^{-6}$
Tr.–Muon	Tr.–SGD	1.0000	1.0000	0.3287	0.0035	$< 10^{-6}$

E Functional Divergence Analysis

E.1 Distances, Coverage, and Uncertainty

Both D^2 and D in Eq. (4) are saved for all $\binom{168}{2} = 14{,}028$ model pairs, with representatives and repeated runs (13 seeds) identified separately. Matrices below summarize individual RMS distances computed on the test set. Seed-matrix diagonals exclude self-pairs; off-diagonals include all 13×13 cross-configuration pairs.

For optimizer cohorts A and B, define within-cohort means W_A, W_B over distinct fitted runs and cross-cohort mean C over all pairs. We display all three separately, together with the ratio R of Eq. (6) and the stricter $R^{\max} = C/\max(W_A, W_B)$.

To estimate uncertainty, delete a fitted run, not an individual pairwise distance: many distances share the same checkpoint. Where seeds were shared, delete the shared seed block from both cohorts; partial overlaps retain shared blocks and singleton blocks. For independent cohorts, add the two delete-one jackknife variance contributions and use Welch–Satterthwaite degrees of freedom. Shared-block intervals use $K - 1$ degrees of freedom, with K the number of seed blocks. Ratio intervals are formed on the log scale. These pointwise intervals are exploratory, conditional on this test panel, and not corrected across comparisons.

Table 8: Within-architecture repeated-fit distances. R divides C by $(W_A + W_B)/2$; $R^{\max}$ divides it by $\max(W_A, W_B)$ and is therefore the stricter comparison. Optimizer-specific fitting budgets are part of each recorded configuration. Full component intervals are in the accompanying table.

Cohorts A / B	W_A	W_B	C	R [95% interval]	$R^{\max}$ [95% interval]
MLP–Adam / Muon	0.0400	0.0119	0.0539	2.08 [1.81, 2.39]	1.35 [1.13, 1.61]
MLP–Adam / SGD	0.0400	0.0749	0.0764	1.33 [1.23, 1.44]	1.02 [0.87, 1.20]
MLP–Muon / SGD	0.0119	0.0749	0.0843	1.94 [1.70, 2.22]	1.12 [0.94, 1.34]
CNN–Adam / Muon	0.0475	0.0364	0.0580	1.38 [1.29, 1.48]	1.22 [1.05, 1.42]
CNN–Adam / SGD	0.0475	0.1032	0.1015	1.35 [1.14, 1.59]	0.98 [0.80, 1.22]
CNN–Muon / SGD	0.0364	0.1032	0.1067	1.53 [1.27, 1.84]	1.03 [0.84, 1.27]
LSTM–Adam / Muon	0.0477	0.0519	0.0693	1.39 [1.24, 1.56]	1.33 [1.18, 1.51]
LSTM–Adam / SGD	0.0477	0.1073	0.0899	1.16 [0.92, 1.46]	0.84 [0.60, 1.18]
LSTM–Muon / SGD	0.0519	0.1073	0.1007	1.26 [1.08, 1.47]	0.94 [0.67, 1.32]
Tr.–Adam / Muon	0.0461	0.0505	0.0545	1.13 [1.04, 1.22]	1.08 [0.89, 1.31]
Tr.–Adam / SGD	0.0461	0.0478	0.0622	1.33 [1.18, 1.49]	1.30 [1.09, 1.56]
Tr.–Muon / SGD	0.0505	0.0478	0.0567	1.15 [1.06, 1.26]	1.12 [0.99, 1.27]

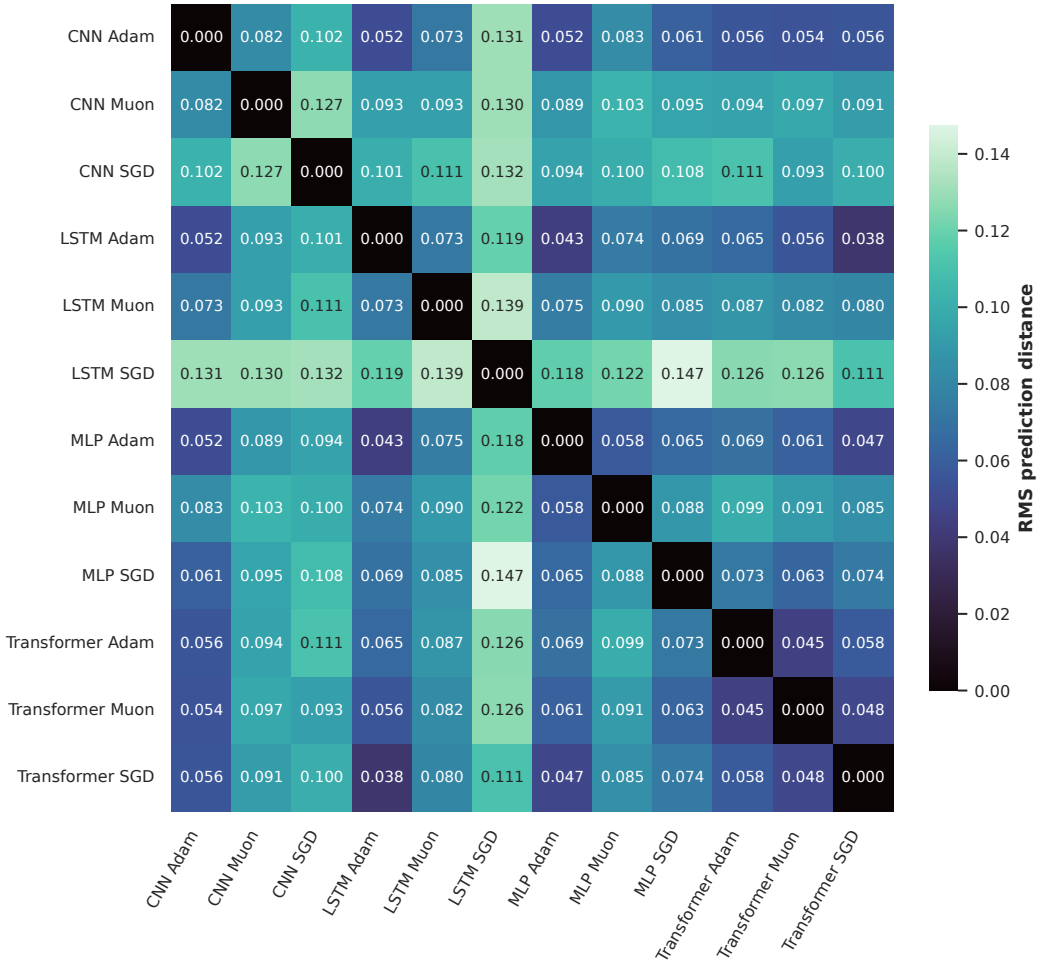


Figure 10: Representative RMS prediction distances on all test observations. Diagonal self-distances are zero.

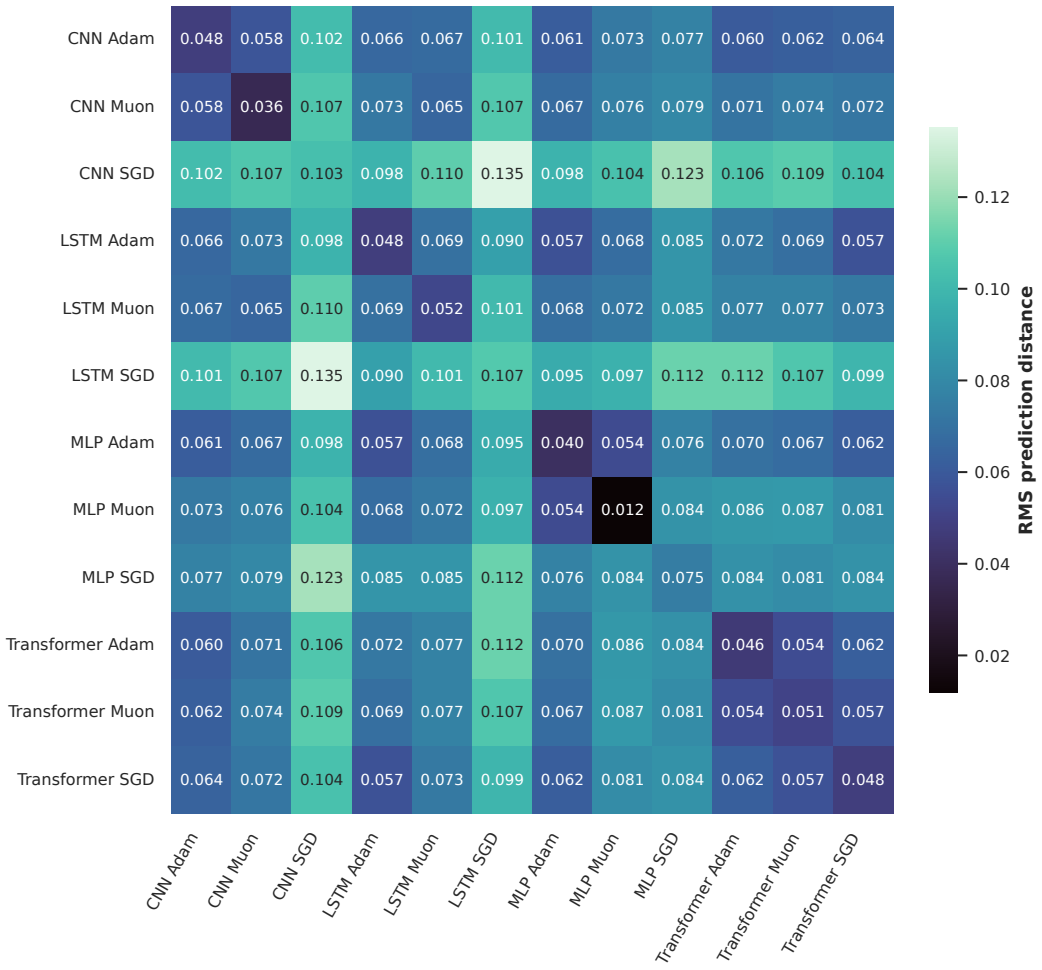


Figure 11: Mean individual RMS distances among repeated-fit cohorts. Diagonals compare different runs of the same configuration; nonzero diagonal values are expected.

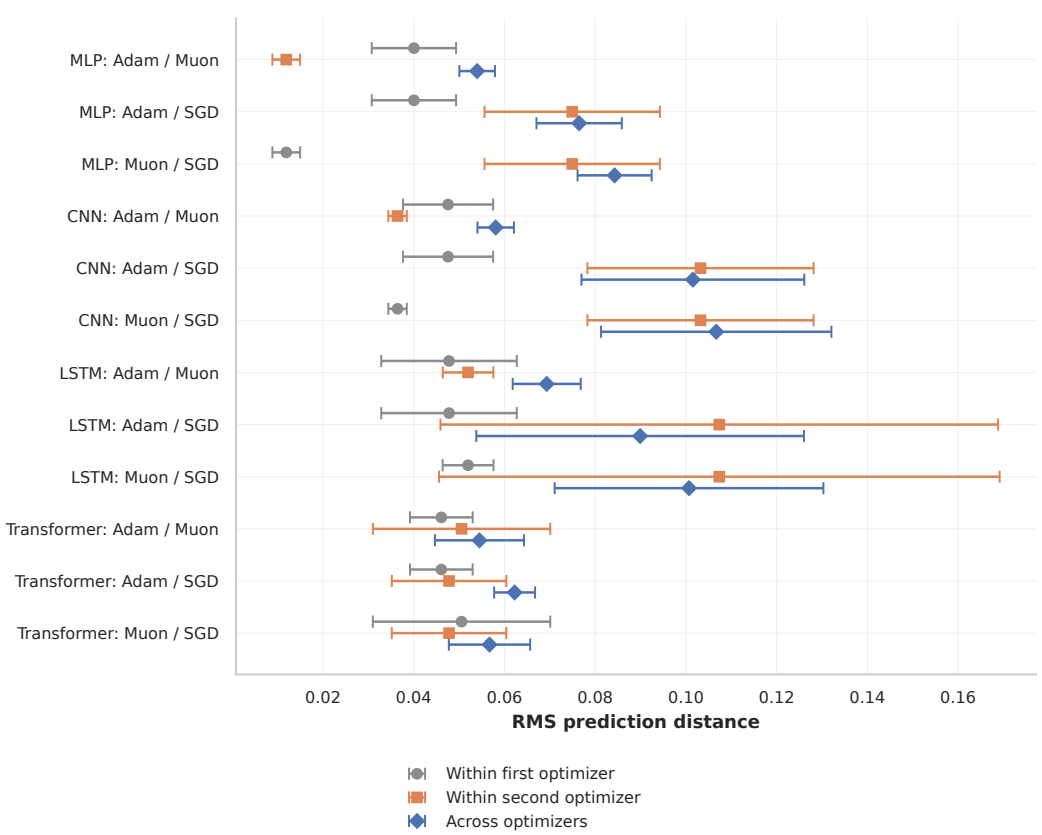
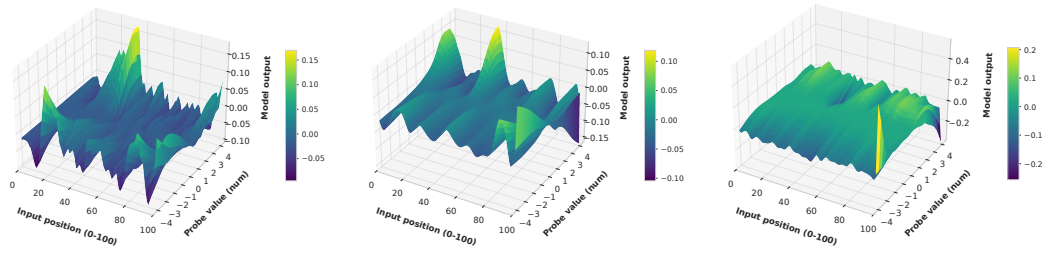


Figure 12: **The components behind functional separation.** Both within-optimizer levels and the across-optimizer level are shown separately for all within-architecture comparisons. Bars are pointwise 95% seed-level jackknife intervals. Across-optimizer disagreement need not exceed both within-optimizer levels.



(a) CNN (b) LSTM (c) Transformer

Figure 13: Muon minus Adam response surfaces for the selected configurations, on the common position/probe grid. Colorbars give standardized output differences.

E.2 Paired Initialization and Data Order

We repeat each of the twelve architecture–optimizer configurations from three independently sampled initializations, pairing the initial weights across optimizers within each architecture. All 36 runs completed their intended budgets. Hashes of the loaded initial tensors and data permutation verify the matching. The data order is fixed across runs and reused across epochs; batch size is 512, including the final partial batch. Transformer dropout uses a paired random-number seed. All runs minimize MSE on the same merged training and validation observations; the loss function is not changed between optimizers. Test loss does not select a checkpoint.

Learning rates, weight decay and parameter groups match Table 4 and remain optimizer-specific, so this experiment controls initialization and data order, not regularization or compute. Budgets are 50 epochs except the Transformer (25) and CNN–SGD (7). The LSTM–SGD run uses learning rate 0.0984626751 and decay 0.0197855670, a separate configuration from the seven-epoch LSTM–SGD reference of Table 1. These runs neither replace nor augment the 13-run cohorts.

For each optimizer pair within an architecture, we compare functions with the same initialization:

$$C_{AB}^{\text{paired}} = \frac{1}{3}\sum_{s=1}^{3} D(f_{A,s}, f_{B,s}). \tag{8}$$

Within-optimizer distances average the three distinct initialization pairs. This paired quantity differs from C_{AB} in Eq. (5), which averages all cross-cohort pairs. Predictions use the same 564,795 test observations and common standardized log-variance scale. Figure 14 shows individual distances and their means; shared-model pairs are not independent observations, and three initializations do not provide a precise population uncertainty estimate.

Forecasting error and prediction disagreement become directly comparable once both are expressed as root-mean-square quantities on the common standardized log-variance scale. Writing $\mathrm{RMSE} = \sqrt{\mathrm{NMSE} \cdot \mathrm{Var}(y)}$ with $\mathrm{Var}(y) = 0.6839$ on the test panel, every fitted model here has $\mathrm{RMSE} \approx 0.627$. Table 9 places the two quantities side by side: $|\Delta\mathrm{RMSE}|$ is how far apart two models' error magnitudes are, and D is how far apart their predictions are, in the same units.

The separation is systematic. Across all twelve comparisons the paired cross-optimizer distance exceeds *both* within-optimizer distances, not merely their average, by factors of 1.0 to 7.8; eleven of the twelve exceed them by at least 1.3. The same strict comparison holds for nine of twelve seed-cohort pairs in Table 8 on point estimates, and for four once their seed-level intervals are required to exclude one; holding initialization and data order fixed is what sharpens it. Two models that start from identical weights, see identical data in identical order, and differ only in optimizer end up further apart than two runs of the same optimizer started from different weights.

The magnitudes make the same point on the error scale. Predictions differ by 5.0–16.7% of the typical forecast error, while the models' error magnitudes differ by 0.003–0.62% of it: D exceeds $|\Delta\mathrm{RMSE}|$ by factors of 21 to 2859. MLP SGD–Adam is the extreme case, with error magnitudes matching to 2×10^{-5} while predictions sit 0.059 apart. Equivalently, the two fitted models' residuals correlate at $\rho = 0.986$–0.999, and the residual $1 - \rho$ is what the loss cannot see. Initialization noise does not account for this: it is the smaller effect in every comparison.

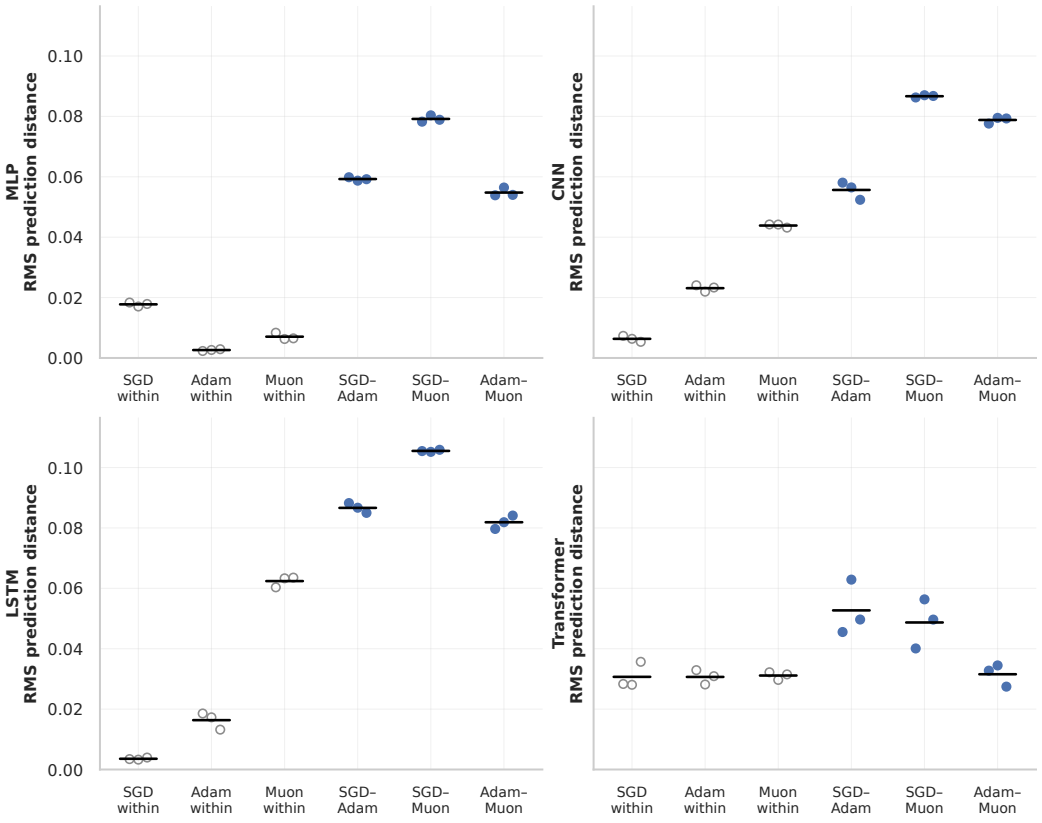


Figure 14: **Functional differences under paired initialization.** Within each architecture, the first three groups compare different initializations under the same optimizer configuration (open points); the last three compare optimizers from identical initial weights (filled points). Each group contains three distances; black bars mark their means, not confidence intervals. Data order is common, while tuned optimizer settings and budgets are retained.

Table 9: **Loss gap versus functional gap, on one scale.** Paired-initialization runs; all quantities are RMS in standardized log-variance units, where the typical forecast error is $\mathrm{RMSE} \approx 0.627$. $|\Delta\mathrm{RMSE}|$ compares the two models' accuracy; D compares their predictions; $W^{\max}$ is the larger of the two within-optimizer distances from different initializations, i.e. the initialization-noise baseline.

Architecture	Pair	$\lvert\Delta\mathrm{RMSE}\rvert$	D	$W^{\max}$	$D/\lvert\Delta\mathrm{RMSE}\rvert$	$D/W^{\max}$
MLP	SGD/Adam	0.00002	0.0593	0.0178	2859	3.34
MLP	SGD/Muon	0.00155	0.0792	0.0178	51	4.46
MLP	Adam/Muon	0.00153	0.0548	0.0070	36	7.77
CNN	SGD/Adam	0.00102	0.0556	0.0231	55	2.41
CNN	SGD/Muon	0.00282	0.0867	0.0439	31	1.98
CNN	Adam/Muon	0.00384	0.0788	0.0439	21	1.80
LSTM	SGD/Adam	0.00390	0.0866	0.0163	22	5.30
LSTM	SGD/Muon	0.00038	0.1055	0.0624	278	1.69
LSTM	Adam/Muon	0.00352	0.0819	0.0624	23	1.31
Transf.	SGD/Adam	0.00047	0.0527	0.0307	112	1.72
Transf.	SGD/Muon	0.00071	0.0487	0.0311	68	1.56
Transf.	Adam/Muon	0.00025	0.0315	0.0311	128	1.01

Table 10: Final fit MSE across the three paired initializations (mean ± sample SD). Test errors are summarized in Table 9.

Architecture	SGD	Adam	Muon
MLP	0.438831 ± 0.000100	0.447133 ± 0.000042	0.450316 ± 0.000340
CNN	0.446792 ± 0.000118	0.444018 ± 0.000220	0.433788 ± 0.000042
LSTM	0.454815 ± 0.000095	0.446676 ± 0.000270	0.439183 ± 0.000537
Transformer	0.444912 ± 0.000413	0.442442 ± 0.000323	0.443362 ± 0.000176

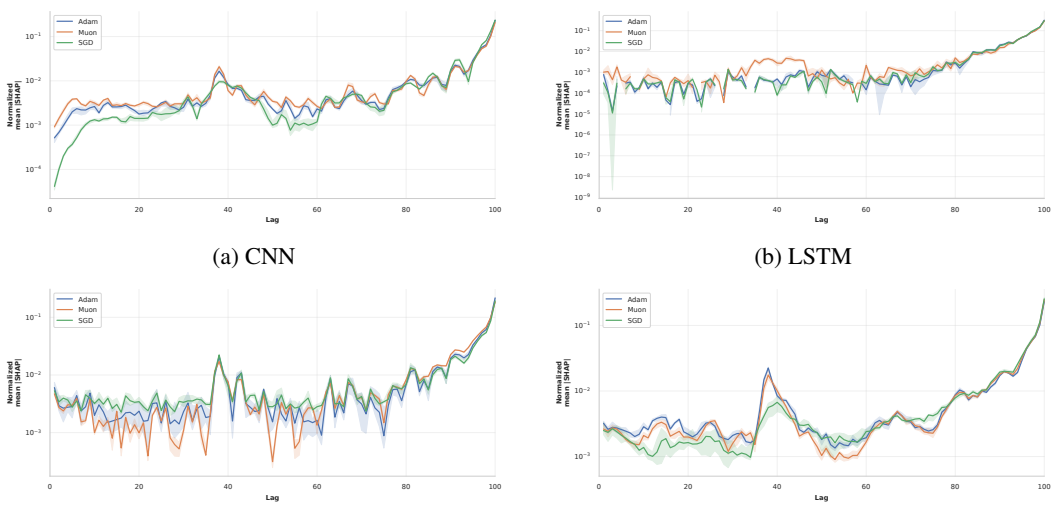
E.3 Attribution Estimators and Checks

All attribution comparisons use correctly merged-scaled inputs and fixed background/explanation samples shared within the comparison. We report each run's normalized mean absolute attribution, then take the median and IQR across the 13 runs at each lag. This preserves the distinction between seed spread and a confidence interval. A logarithmic vertical axis exposes smaller lag contributions. The sampling-based SHAP estimates can include exact zeros. Zero-valued medians and quartile bounds cannot be represented on this axis and are masked, producing gaps in the curves or shaded bands. These gaps are display artifacts, not missing observations; the stored attribution values are unchanged.

For MLP/CNN, the retained DeepSHAP protocol uses 300 background and 800 explanation observations for repeated fits, and 1,000/2,000 for representatives. LSTM uses KernelSHAP with 25 background centroids and 150 explanation observations for repeated fits, and 15/100 for representatives (250 coalition samples in the representative protocol). Settings and reconstruction checks are recorded per artifact. Transformer uses KernelSHAP throughout, with no mixture of DeepSHAP and KernelSHAP within its comparisons. The replacement protocol uses 15 weighted merged-training background centroids, 200 shared reproducibly sampled test observations, all 100 features, 2,048 coalition samples, and no feature-selection penalty.

Transformer representative estimates were repeated with an independent coalition-sampling seed and at 4,096 samples. The diagnostic targets were normalized-profile L1 difference at most 0.05 and at least eight shared top-ten features. All three passed at 2,048: the repeat/budget L1 differences were 0.0132/0.0340 (Adam), 0.0086/0.0146 (Muon), and 0.0026/0.0031 (SGD), with all ten top features shared. The same accepted budget is applied to each optimizer's complete 14-checkpoint group (one representative plus 13 repeats).

The earlier Transformer DeepSHAP evaluations failed prediction-reconstruction checks and emitted an unsupported-layer warning. This concerns the attribution decomposition, not the correctness of model predictions. KernelSHAP reconstruction is checked, but its near-exact reconstruction is partly enforced by the estimator and is insufficient on its own; the repeat/budget checks assess profile stability. These finite-sample checks support use of the displayed summaries without establishing a unique explanation under correlated lagged inputs.



(a) CNN (b) LSTM

(c) MLP (d) Transformer

Figure 15: **Optimizer-Driven Feature Attribution (SHAP).** Normalized mean |SHAP| across input lags over 13 runs. Solid lines show medians; shaded bands show the IQR (seed spread). The vertical axis is logarithmic. Positions run from 1 (oldest input) to 100 (most recent input).

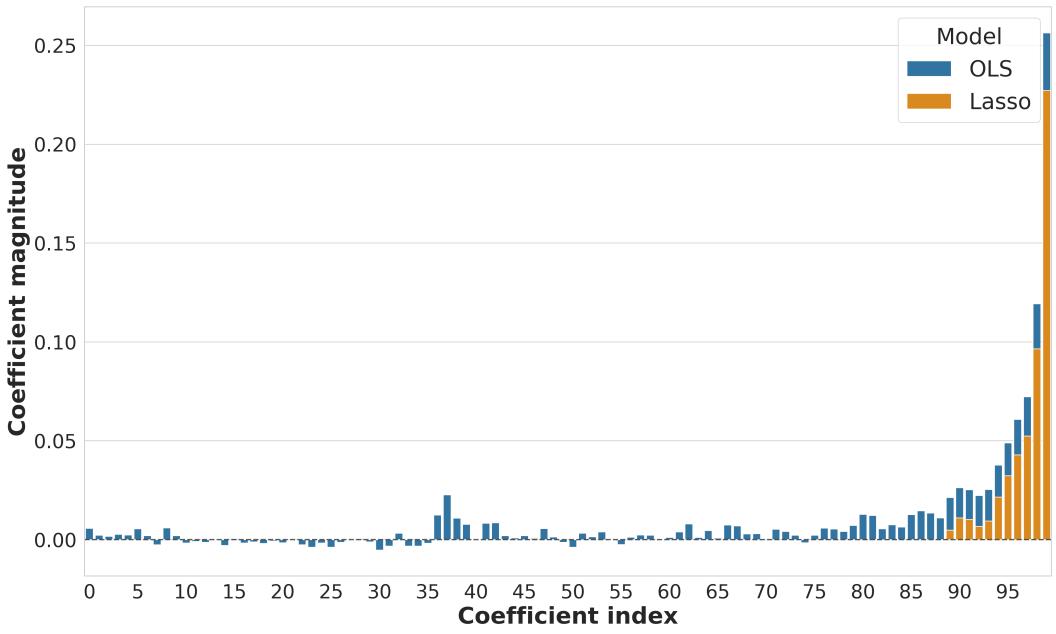
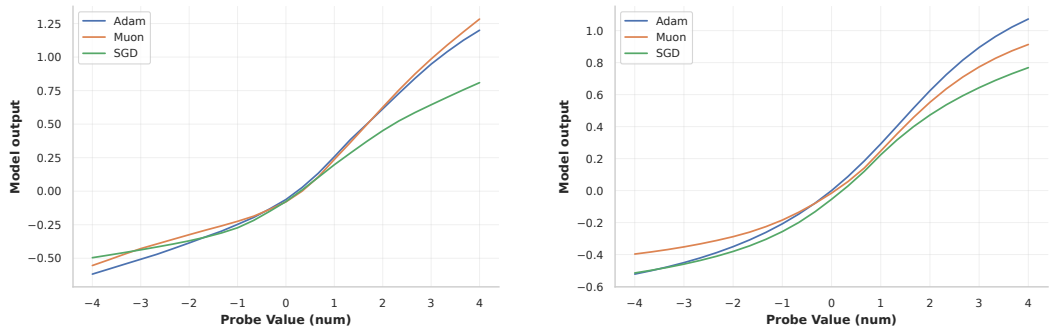
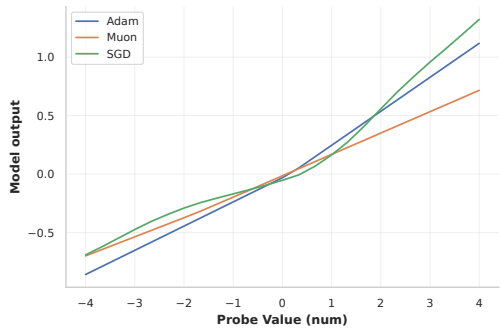


Figure 16: **Linear Baselines.** Coefficients of OLS and LASSO regressions for 100 lagged squared volatility terms (σ^2_{t-99} through σ^2_t). Note the sparsity of LASSO versus the dense allocation of OLS.

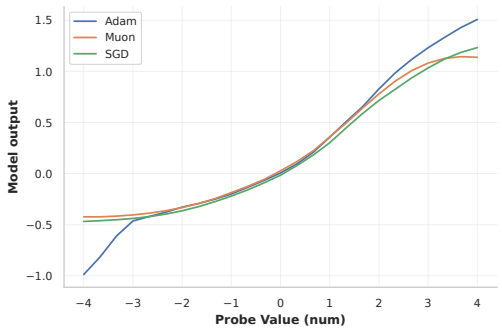


(a) **CNN** Impulse Response

(b) **LSTM** Impulse Response

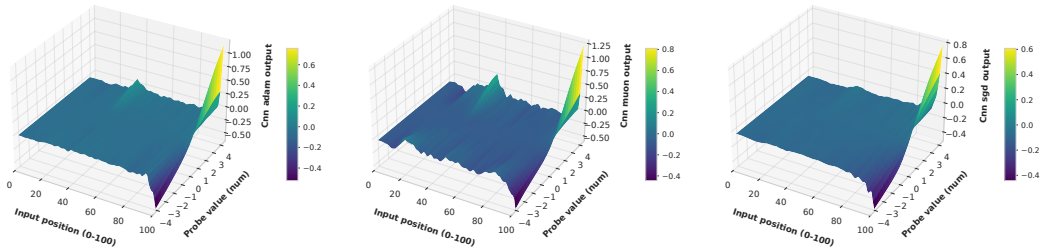


(c) **MLP** Impulse Response



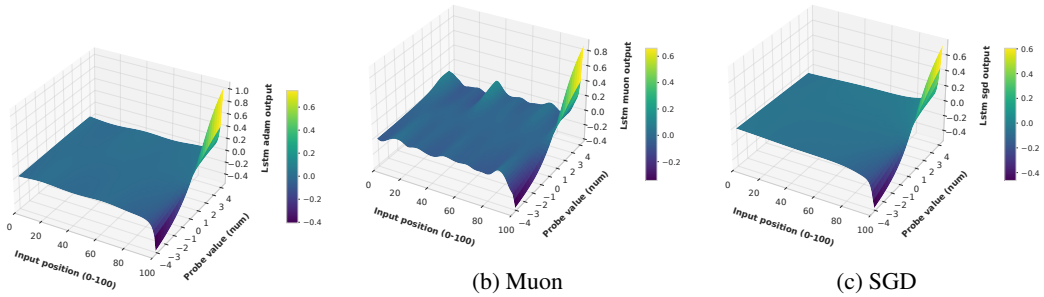
(d) **Transformer** Impulse Response

Figure 17: **Functional Divergence (Impulse Responses).** Comparison of model sensitivity at $t-1$ across optimizers.



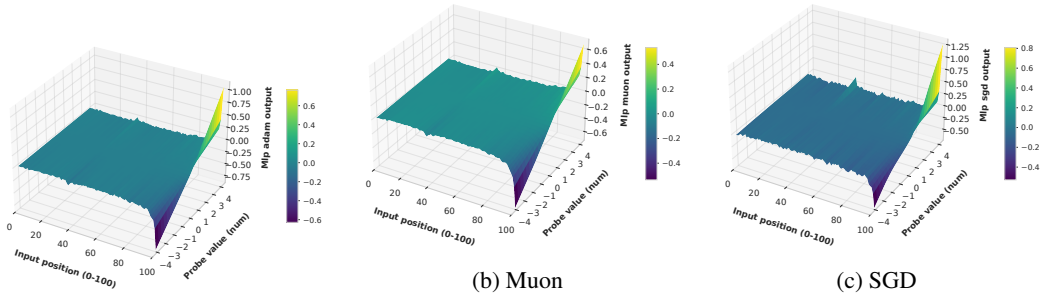
(a) Adam (b) Muon (c) SGD

Figure 18: **CNN response surfaces.** Left to right: Adam, Muon, SGD.



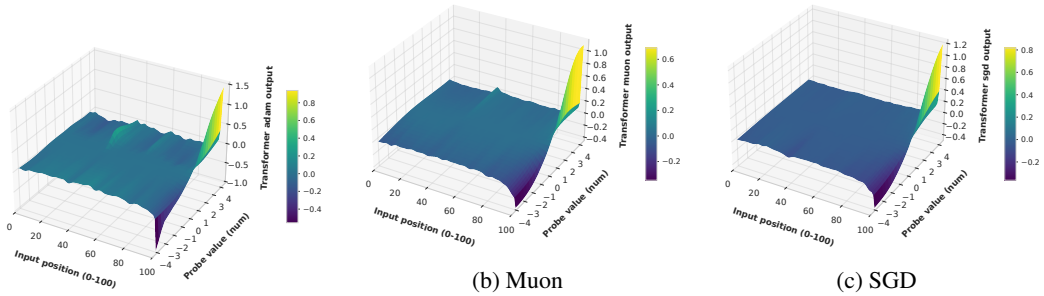
(a) Adam (b) Muon (c) SGD

Figure 19: **LSTM response surfaces.** Left to right: Adam, Muon, SGD.



(a) Adam (b) Muon (c) SGD

Figure 20: **MLP response surfaces.** Left to right: Adam, Muon, SGD.



(a) Adam (b) Muon (c) SGD

Figure 21: **Transformer response surfaces.** Left to right: Adam, Muon, SGD.

F Search for a Mechanism

F.1 Stability Constraints and Function Selection

How do training procedures with similar forecasting errors select different functions? The stability viewpoint offers a candidate mechanism: the region reached by training depends on the constraints imposed by its update dynamics. Self-stabilization can therefore contribute to function selection, rather than merely determine the speed of approach to a solution (Damian et al., 2023).

In full-batch gradient descent, the edge of stability describes continued training near the local quadratic boundary $2/\eta$, with short-time oscillations and nonlinear regulation of curvature (Cohen et al., 2022b). Central flow formalizes a related constraint on the time-averaged trajectory (Cohen et al., 2024); stochastic training additionally couples curvature to mini-batch gradients (Andreyev & Beneventano, 2025). These results motivate a direct experiment: change the dynamics at a fitted solution and ask whether sharpening and further fitting resume.

Around a fixed positive-definite quadratic minimum, reducing the learning rate contracts the stationary fluctuations of SGD without changing the Hessian (Mandt et al., 2017). If, instead, the apparent endpoint is maintained by a dynamical curvature constraint, smaller steps can permit continued motion into sharper regions. *Reducing the learning rate releases renewed progressive sharpening, indicating that the preceding curvature plateau reflects a constraint of the training dynamics rather than merely fluctuations around a fixed local quadratic minimum.* This is the interpretation tested below: the interventions support a dynamical explanation, but do not prove the absence of a local minimum, since contraction within a nonquadratic neighborhood can also change curvature.

The numerical height of the plateau is not the sole criterion. Weight decay can alter progressive sharpening and sustain subcritical curvature levels in regularized full-batch dynamics (Saether et al., 2026). This supplies a reason to examine the response to interventions rather than infer unconstrained convergence from $\lambda_{\max} < 2/\eta$ alone. Our minibatch experiments test that response; they do not identify weight decay as its unique cause.

F.2 Measurements and Intervention Design

We record the leading data-loss Hessian eigenvalue $\lambda_{\max}$ and directional mini-batch curvature. For mini-batch gradient g_B and Hessian H_B, the plotted diagnostic is

$$\mathrm{BS}_{\mathrm{diag}} = \mathbb{E}_B\left[\frac{g_B^\top H_B g_B}{\|g_B\|^2}\right]. \tag{9}$$

It averages Rayleigh quotients; the literature definition instead uses $\mathbb{E}_B[g_B^\top H_B g_B]/\mathbb{E}_B[\|g_B\|^2]$, weighting batches by gradient norm (Andreyev & Beneventano, 2025). The two need not coincide.

Each intervention starts from common fitted weights and saved optimizer state: unchanged control, LR/10, or batch$\times 2$. Decay settings are preserved. We compare equal additional data passes, so batch doubling halves the update count and can change accumulated regularization. Curvature is measured in evaluation mode, excluding decay from the Hessian. Diagnostic batch size is 512 for control and LR/10, and 1,024 for batch$\times 2$; the latter changes both training and the measured statistic. Appendix C.5 records numerical settings.

Training panels are separate realizations, not the histories leading to the continuation checkpoints. CNN–SGD and LSTM–SGD use the historical diagnostic settings. Transformer–SGD training uses $\eta = 0.0028016$, while its continuation uses the selected configuration with $\eta = 0.0030821$; these rows are contextual comparisons, not one trajectory. The remaining pairs use the corresponding predictive settings. Recorded horizons appear on the axes.

F.3 Training and Intervention Evidence

SGD: releasing the curvature constraint. MLP provides the clearest illustration (Figure 22). Training sharpens before settling into a fluctuating range. From the fitted solution, LR/10 releases a sustained increase in both curvature measures, accompanied by further loss reduction. Over the final fifth of the common 20-epoch window, median $\lambda_{\max}$ is approximately 29 under control, 85 under

LR/10 and 34 under batch×2; final-fit MSE at epoch 20 is 0.4406, 0.4320 and 0.4371, respectively. The intervention thus exposes further fitting beyond the preceding plateau.

The CNN diagnostic shows a stronger response: LR/10 raises both curvature measures and approaches a higher level, whereas control remains near its previous range (Figure 23). The late-window eigenvalue is about 7.6 times control. Transformer–SGD shows a weaker response: its continuation eigenvalue rises modestly under LR/10 while control declines, and batch sharpness remains noisy (Figure 25). Late-window eigenvalues are approximately 178, 219 and 209 for control, LR/10 and batch×2. Its separately initialized training trace decreases rather than progressively sharpens.

The LSTM–SGD diagnostic also shows renewed sharpening after LR/10 (Figure 24). Over the common eight-epoch window, its late-window eigenvalue is about 2.8 times control. The LR/10 arm ends earlier than the other arms; this supports a response comparison rather than a shared convergence claim.

Together, the SGD interventions support stability-constrained region selection, most clearly for MLP and CNN. Renewed sharpening, stabilization at a higher level and inverse-learning-rate scaling are distinct observations: the first does not establish the other two. The weaker batch response may be consistent with the large-batch regime discussed by Andreyev & Beneventano (2025, Section 5.2), but these experiments do not locate its crossover.

Reading the figures. Each figure pairs training (top) and interventions (bottom); columns show $\lambda_{\max}$, batch sharpness and loss. Training curvature is green for SGD, blue for Adam and orange for Muon. Training-loss traces are grey for minibatch measurements, blue for bin medians and red for final-fit MSE. In continuation rows, grey denotes control, red LR/10 and dashed blue batch×2; dotted loss lines show final fit and solid lines show test MSE. Faint/dark curvature traces show measurements/half-epoch medians, and vertical dotted lines mark the common completed horizon. The axes or legends report $2/\eta$ for SGD and the adaptive threshold $38/\eta$ of Cohen et al. (2022a) for Adam; raw curvature is not normalized. The same line is drawn for Muon purely as a reference, no threshold having been established for it.

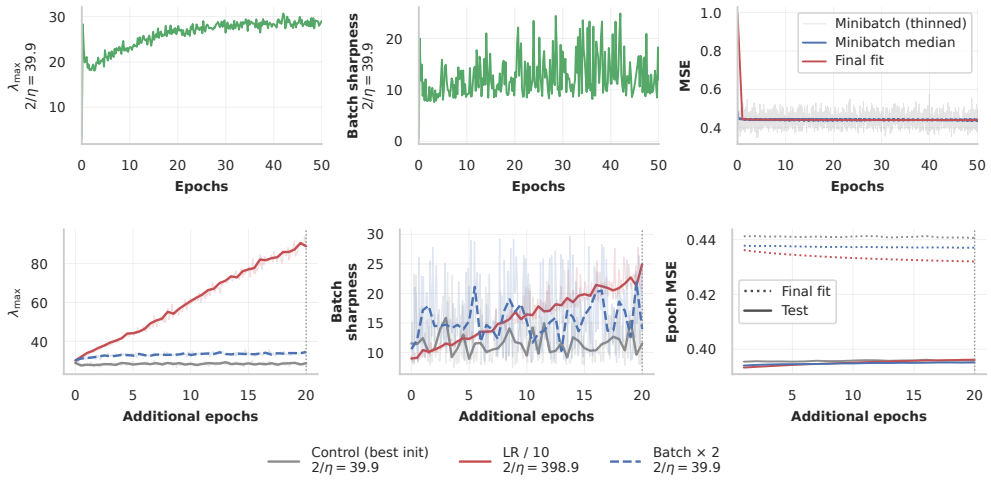


Figure 22: **MLP–SGD.** Training: sharpening then plateau. LR/10: renewed sharpening in both measures and lower final-fit loss; batch doubling gives a smaller response.

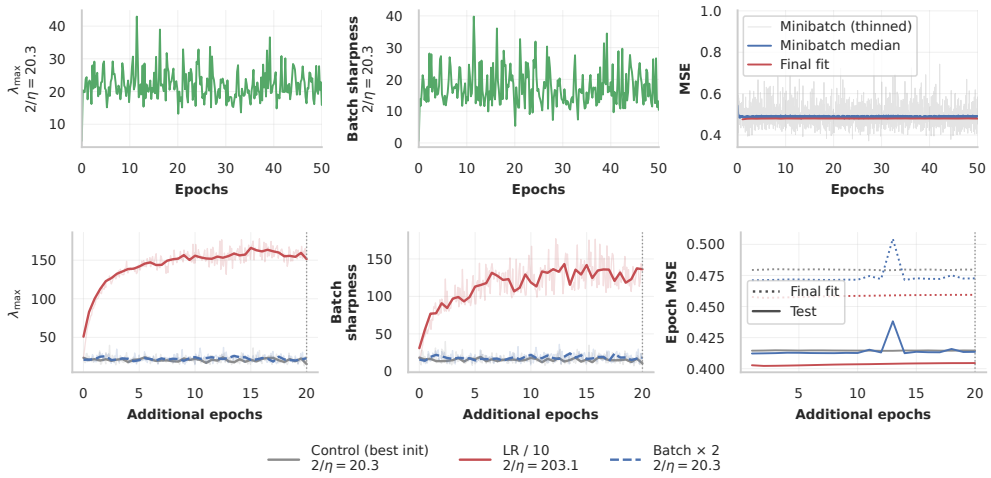


Figure 23: **CNN–SGD.** Training: fluctuating curvature plateau. LR/10: strong renewed sharpening and a higher level in both measures; batch doubling shows a transient loss spike.

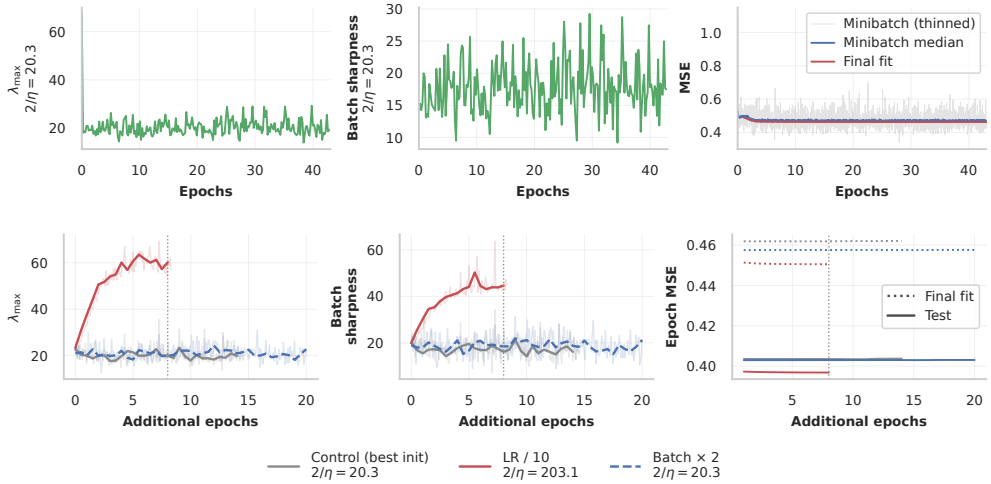


Figure 24: **LSTM–SGD.** Training: sharpening toward a fluctuating level. LR/10: renewed sharpening in both measures over the common eight-epoch window; the intervention arms have different completed horizons. Training above; interventions below.

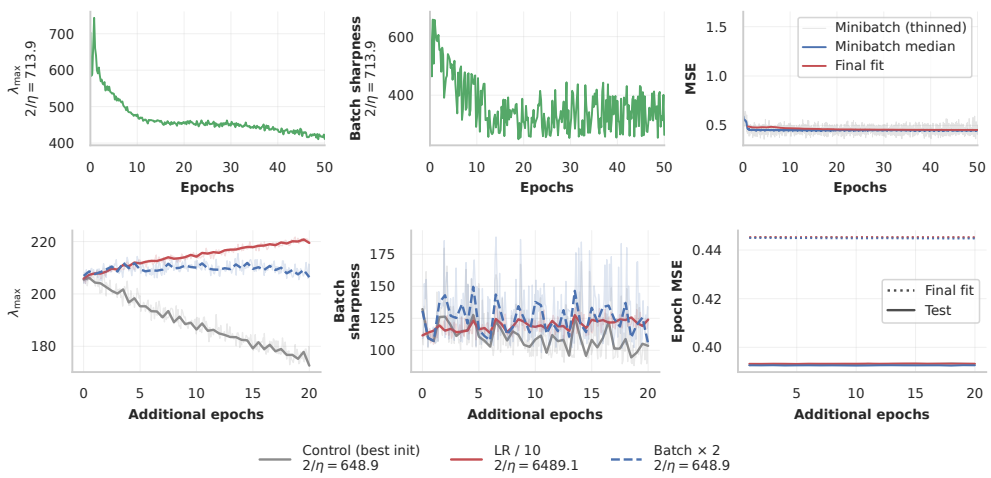


Figure 25: **Transformer–SGD.** Training: declining curvature in the historical configuration. Selected-configuration continuation: modest eigenvalue increase under LR/10, with noisy batch sharpness; the two rows are not a continuous run.

Adam: modest or persistent curvature differences. MLP–Adam continues sharpening during training, but the continuation arms show little separation. CNN–Adam also sharpens during training; LR/10 produces a modestly higher continuation eigenvalue, with much less separation in batch sharpness. LSTM–Adam shows upward responses in both curvature measures over its shorter common window (Figure 28). Transformer–Adam behaves differently: training curvature declines, and LR/10 largely maintains its starting eigenvalue while control decreases. Remaining above control here is not renewed progressive sharpening. Adaptive stability depends on effective update geometry, including optimizer state (Cohen et al., 2022a); the raw Hessian and batch diagnostic do not measure that complete geometry. We therefore describe Adam and Muon empirically rather than apply the SGD mechanism unchanged.

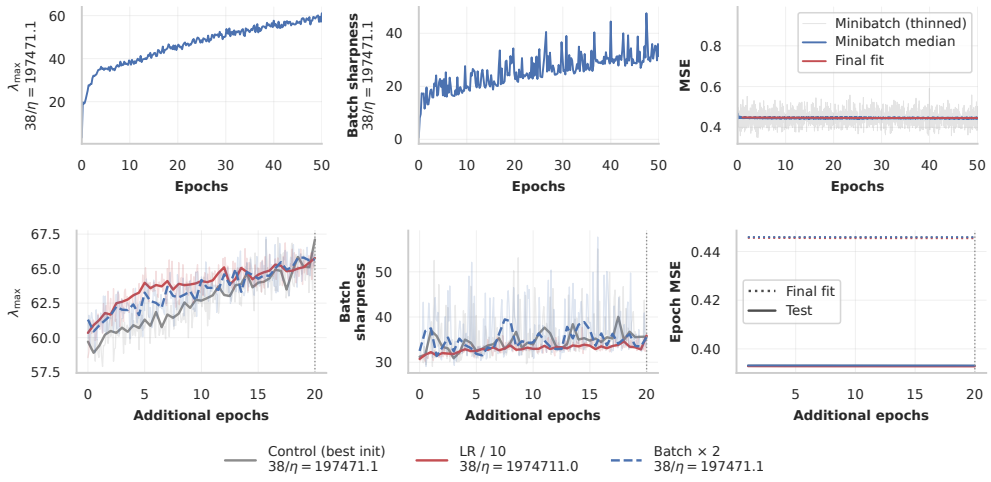


Figure 26: **MLP–Adam.** Training: continued sharpening. Interventions: weak curvature separation and little visible loss separation.

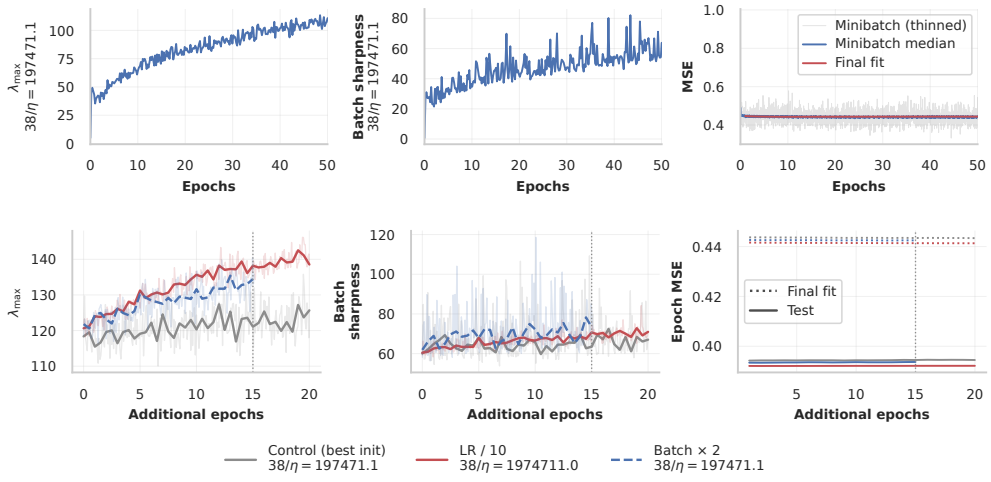


Figure 27: **CNN–Adam.** Training: continued sharpening. LR/10: modest eigenvalue separation from control, with much weaker separation in batch sharpness.

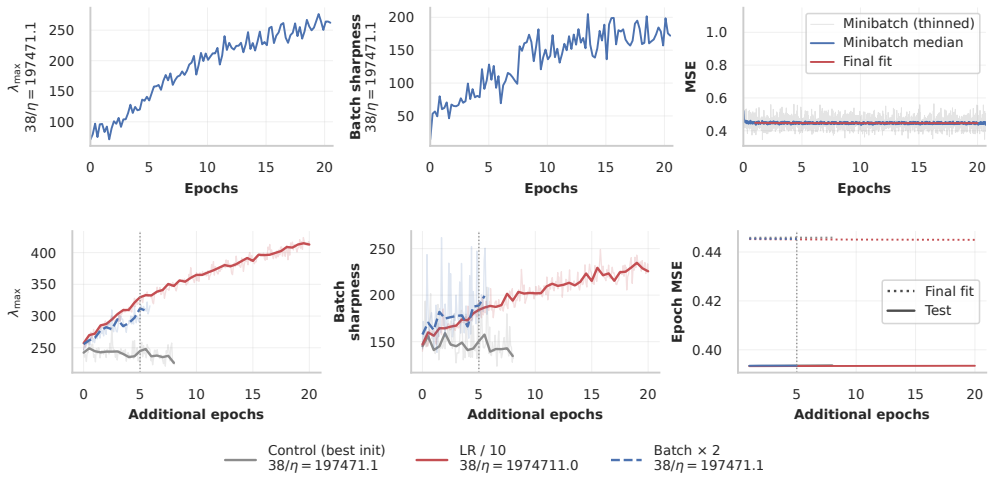


Figure 28: **LSTM–Adam.** Training: continued sharpening. LR/10 raises both curvature measures relative to control over the shorter common observation window; the arms have different completed horizons. Training above; interventions below.

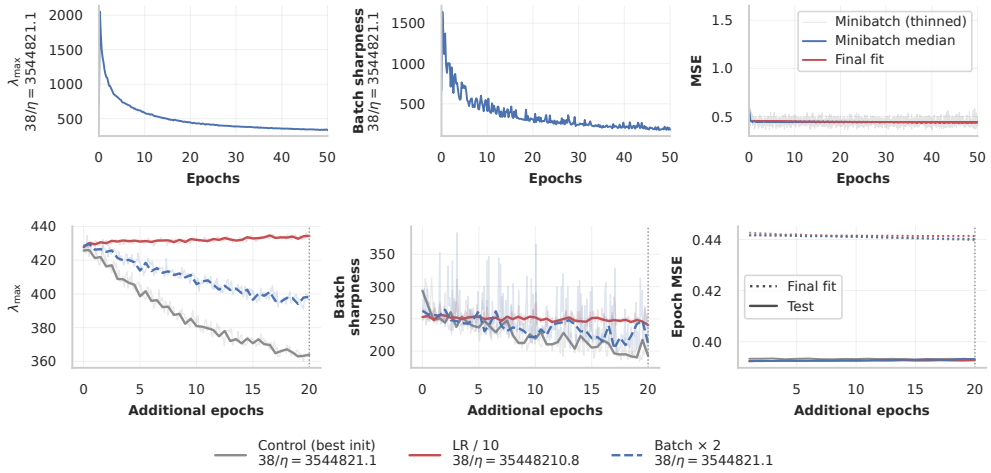


Figure 29: **Transformer–Adam.** Training: declining curvature. LR/10 largely maintains the eigenvalue while control declines; batch sharpness is noisy and generally declines. This is not renewed sharpening.

Muon: heterogeneous responses. MLP–Muon sharpens during training, yet LR/10 ends with lower curvature than control. CNN–Muon shows continued training sharpening, while LR/10 slows subsequent growth in both curvature measures relative to control. LSTM–Muon instead shows an upward LR/10 response in both measures over its shorter recorded window (Figure 32). Transformer–Muon training curvature declines; LR/10 approximately maintains its eigenvalue and batch sharpness as control decreases. These responses differ from the strongest SGD examples. The stability mechanism for Muon is less understood, and the plotted reference does not turn these raw-curvature observations into a stability test.

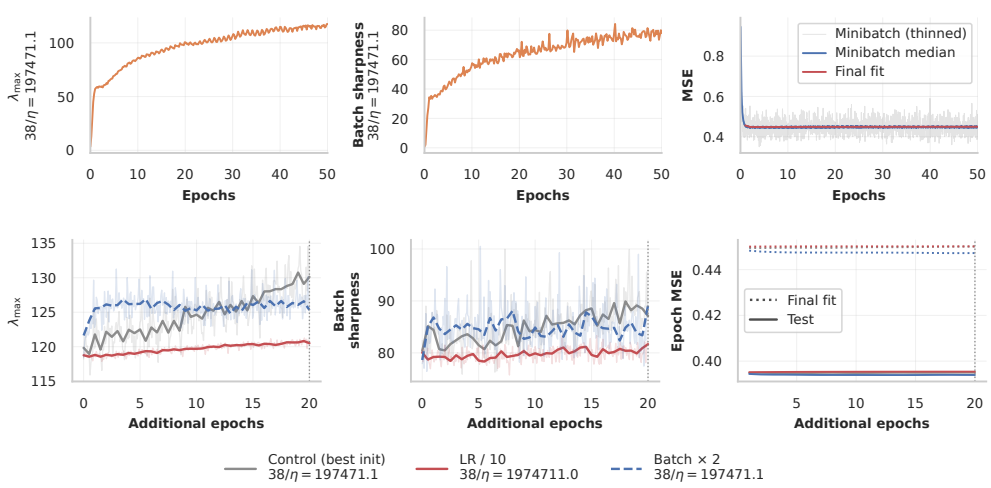


Figure 30: **MLP–Muon.** Training: continued sharpening. LR/10 produces lower late-window eigenvalue and batch sharpness than control.

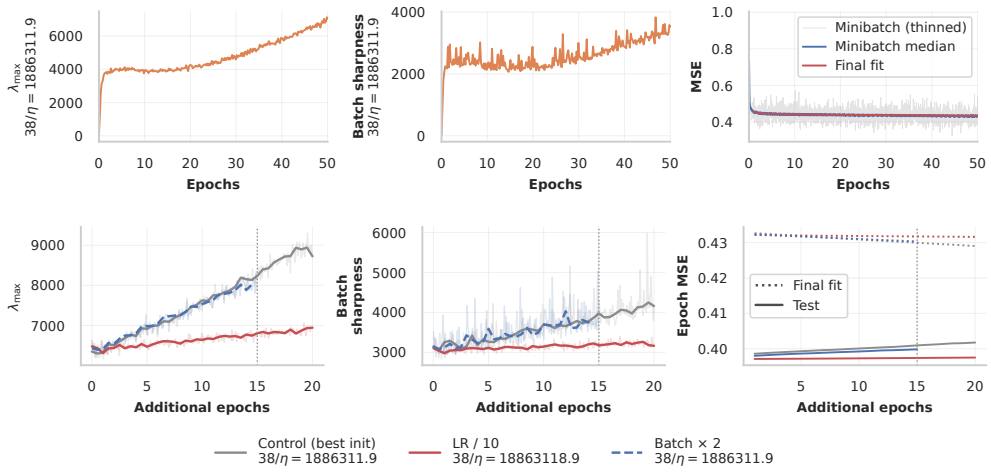


Figure 31: **CNN–Muon.** Training: continued sharpening. LR/10 slows further growth in both curvature measures relative to control.

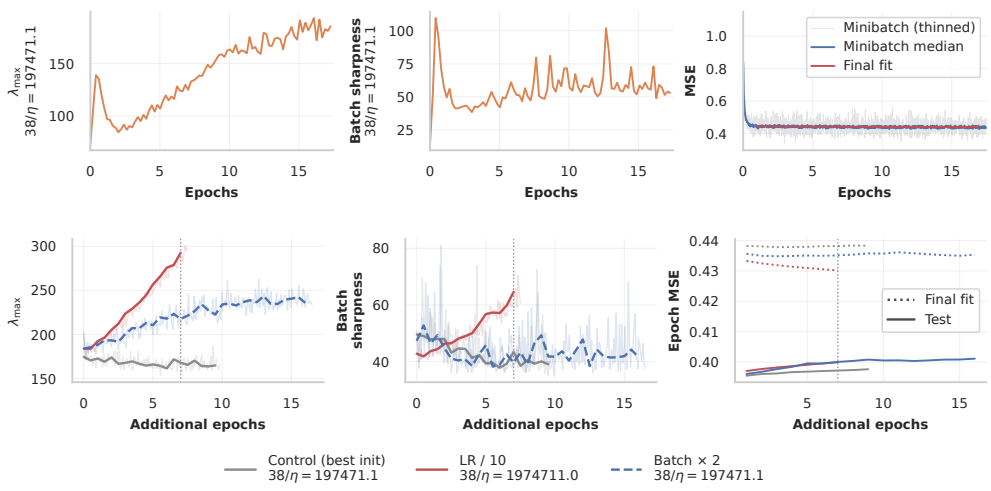


Figure 32: **LSTM–Muon.** Training: continued sharpening. LR/10 raises both curvature measures over its recorded window, unlike the MLP and CNN responses; the arms have different completed horizons. Training above; interventions below.

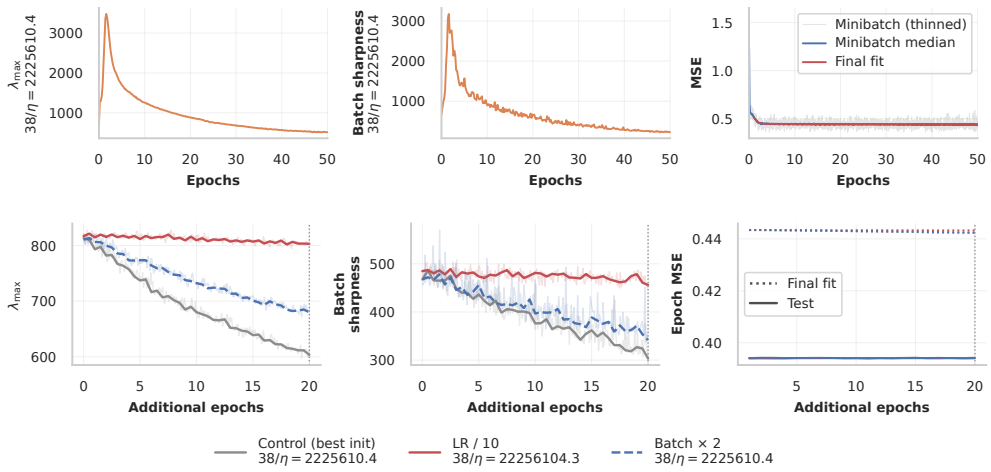


Figure 33: **Transformer–Muon.** Training: declining curvature. LR/10 approximately maintains both measures while control declines; the difference reflects persistence rather than renewed sharpening.

Movement in prediction space. The available MLP–SGD checkpoints also connect the interventions to function selection. At the shared 20-epoch endpoint, prediction distances from the starting solution are 0.04328 for control, 0.05101 for LR/10 and 0.03810 for batch$\times 2$. Relative to control, LR/10 moves predictions by 0.05547 RMS and batch$\times 2$ by 0.02433; the two intervention endpoints differ by 0.03909. These use the same test observations and distance definition as Eq. (4). The interventions therefore change the learned predictions as well as curvature; this does not isolate curvature as the unique cause of functional movement.

G VOLATILITY-MANAGED PORTFOLIOS

This appendix reports additional details for the downstream portfolio experiment used to illustrate how optimizer-induced functional differences in volatility forecasts translate into differences in *decision stability*. The analysis is not intended as a performance benchmark, but as a diagnostic of the implementability of metrically equivalent predictors.

G.1 CROSS-SECTIONAL IMPLEMENTATION

On each recorded trading day, assets are sorted into quintiles using predicted volatility. Within each quintile, portfolios are equal-weighted and reconstituted daily. The full-history diagnostics use 2,854,980 asset–date observations from 25 May 2000 to 24 December 2024 and the selected representative checkpoints. They include 2,290,185 fitting observations through 26 December 2019 and 564,795 held-out observations from 21 May 2020 onward. The full-history results therefore describe retrospective decision behavior, not a continuous out-of-sample trading backtest. Table 12 separately reports the held-out results.

Prediction rows are aligned by asset and observation identity, and the common merged-training transformation is inverted before portfolio construction. Portfolios are not averaged across training seeds. The original return/date convention is retained; executable signal timing is not independently established by this diagnostic.

G.2 TURNOVER AND PERFORMANCE METRICS

Portfolio turnover is computed as $\frac{1}{2}\sum_i |w_{i,t} - w_{i,t-1}|$ and averaged across successive recorded dates, where both weight vectors are equal-weight *target* allocations. This measures changes in portfolio membership and target weights; it does not adjust previous weights for intervening asset returns. Annualized return is the mean daily return multiplied by 252, annualized volatility is daily standard deviation multiplied by $\sqrt{252}$, and the reported gross Sharpe ratio is their ratio, without risk-free subtraction. Maximum drawdown is computed from cumulative gross returns. No transaction costs are deducted, and no confidence interval or equality test for Sharpe ratios is asserted.

G.3 SUMMARY RESULTS

Across neural predictors, gross Sharpe ratios range from 0.883 to 0.959 in Q1 and from 0.411 to 0.429 in Q5. Corresponding annualized volatilities are 12.92–13.00% and 29.54–29.90%. These are descriptive comparisons over the observed period, not statistical equivalence claims about portfolio returns.

Turnover varies more substantially: the maximum-to-minimum ratio among neural predictors is 1.41 in Q1 and 1.58 in Q5. CNN–SGD and LSTM–SGD trade less than their Adam and Muon counterparts, but MLP–Muon trades less than MLP–SGD, and Transformer turnover is relatively similar across optimizers. Thus the economic consequence is configuration-dependent decision multiplicity, rather than a uniform SGD advantage.

G.4 TIME-SERIES EVIDENCE: TURNOVER DYNAMICS

Figure 35 summarizes optimizer averages; Figures 36–38 retain the architecture-specific paths for Q1, Q3 and Q5. Rolling averages describe variation over time and should be read alongside the complete-period table. The disaggregated plots are essential because optimizer averaging hides the

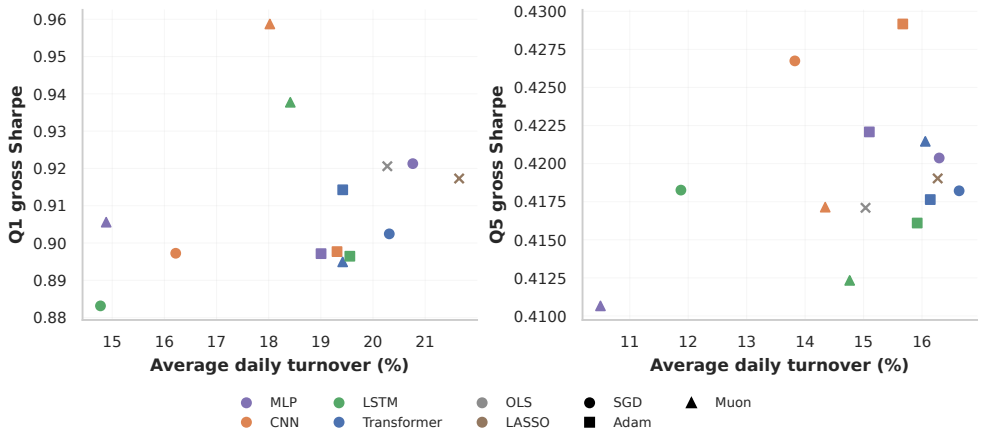


Figure 34: **Sharpe–Turnover Frontier.** Selected representative models over the available May 2000–December 2024 history, including fitting observations. Q1 (left) and Q5 (right) are the lowest- and highest-predicted-volatility quintiles. Gross Sharpe ratios occupy a narrow range while average daily target-weight turnover differs across fitted predictors.

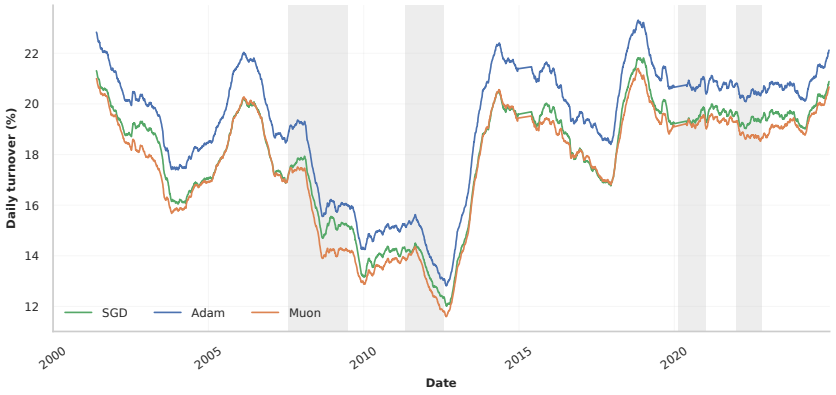


Figure 35: **Rolling target-weight turnover, Q1.** One-year rolling averages over the available 2000–2024 history, with neural architectures equally represented within each optimizer average. The average does not imply the same ordering in every architecture.

different MLP and CNN/LSTM orderings. No universal ordering or amplification during every stress period is implied. Grey bands mark the global financial crisis (August 2007–June 2009), European debt crisis (May 2011–July 2012), COVID period (15 February–December 2020), and inflation shock (January–October 2022). These are contextual stress windows, not formal recession dates. Each quintile has its own labeled vertical scale.

Together, these results show that similar forecasting errors and gross portfolio performance can coexist with materially different allocation changes. The financial experiment supplies a downstream illustration of the learned functions' differences, rather than a test of which optimizer is universally preferable.

Table 11: **Volatility-sorted portfolios over the full available history.** Selected representatives, May 2000–December 2024, including fitting observations. Q1 and Q5 denote lowest and highest predicted volatility. Return, volatility, drawdown and daily turnover are fractions.

Model	Ann. Ret.	Ann. Vol.	Sharpe	Max DD	Turnover
Q1: Low Volatility					
OLS	0.120	0.130	0.921	-0.397	0.203
LASSO	0.119	0.130	0.917	-0.402	0.217
CNN (SGD)	0.117	0.130	0.897	-0.406	0.162
CNN (Muon)	0.124	0.129	0.959	-0.389	0.180
CNN (Adam)	0.117	0.130	0.898	-0.398	0.193
LSTM (SGD)	0.114	0.129	0.883	-0.403	0.148
LSTM (Muon)	0.121	0.129	0.938	-0.398	0.184
LSTM (Adam)	0.116	0.130	0.896	-0.395	0.196
MLP (SGD)	0.120	0.130	0.921	-0.397	0.208
MLP (Muon)	0.117	0.129	0.906	-0.390	0.149
MLP (Adam)	0.117	0.130	0.897	-0.397	0.190
Transformer (SGD)	0.117	0.130	0.902	-0.395	0.203
Transformer (Muon)	0.116	0.130	0.895	-0.392	0.194
Transformer (Adam)	0.119	0.130	0.914	-0.396	0.194
Q5: High Volatility					
OLS	0.124	0.297	0.417	-0.718	0.150
LASSO	0.124	0.296	0.419	-0.717	0.163
CNN (SGD)	0.127	0.297	0.427	-0.701	0.138
CNN (Muon)	0.124	0.297	0.417	-0.684	0.143
CNN (Adam)	0.127	0.296	0.429	-0.703	0.157
LSTM (SGD)	0.125	0.298	0.418	-0.712	0.119
LSTM (Muon)	0.122	0.297	0.412	-0.706	0.148
LSTM (Adam)	0.123	0.296	0.416	-0.723	0.159
MLP (SGD)	0.124	0.296	0.420	-0.709	0.163
MLP (Muon)	0.123	0.299	0.411	-0.719	0.105
MLP (Adam)	0.125	0.297	0.422	-0.712	0.151
Transformer (SGD)	0.124	0.295	0.418	-0.717	0.166
Transformer (Muon)	0.125	0.295	0.421	-0.710	0.161
Transformer (Adam)	0.124	0.296	0.418	-0.716	0.161

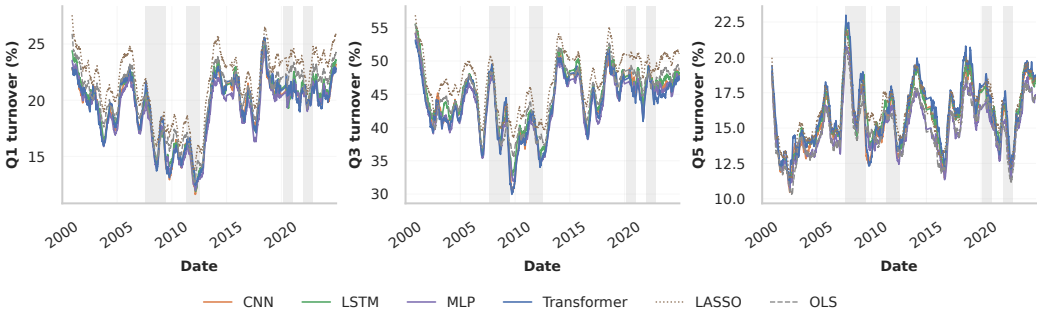


Figure 36: **Rolling turnover by volatility quintile (Adam).** Six-month rolling target-weight turnover over the available 2000–2024 history for Q1 (low volatility), Q3 and Q5 (high volatility), from left to right. Vertical scales differ across panels. Grey bands mark the market-stress windows defined above.

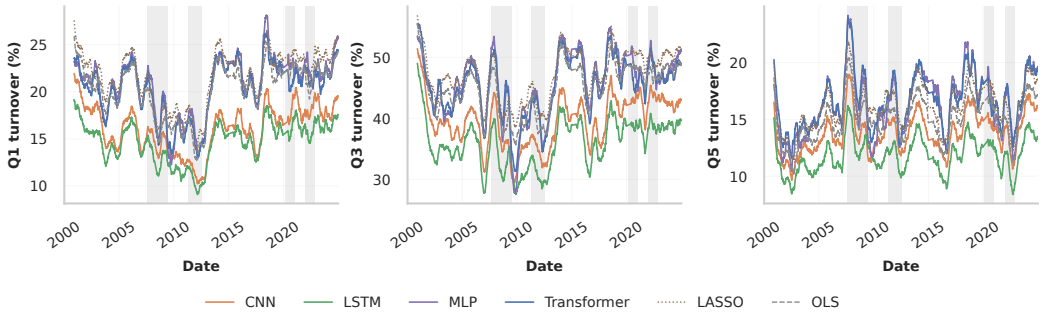


Figure 37: **Rolling turnover by volatility quintile (SGD).** Six-month rolling target-weight turnover over the available 2000–2024 history for Q1 (low volatility), Q3 and Q5 (high volatility), from left to right. Vertical scales differ across panels. Grey bands mark the market-stress windows defined above.

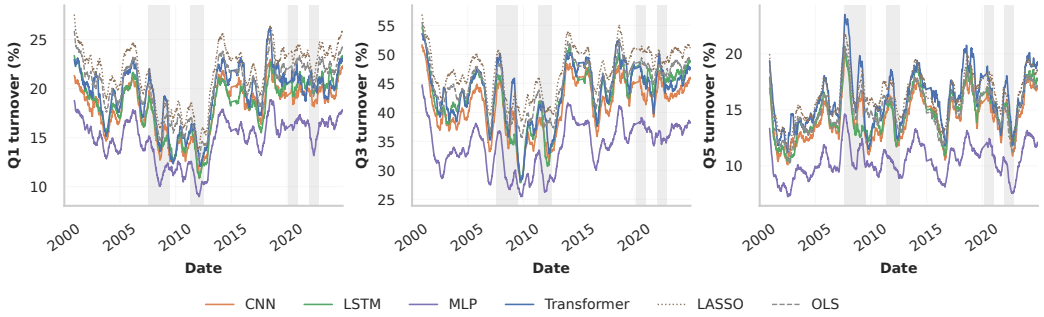


Figure 38: **Rolling turnover by volatility quintile (Muon).** Six-month rolling target-weight turnover over the available 2000–2024 history for Q1 (low volatility), Q3 and Q5 (high volatility), from left to right. Vertical scales differ across panels. Grey bands mark the market-stress windows defined above.

Table 12: **Held-out portfolio comparison, May 2020–December 2024.** The same representatives, restricted to test observations. Turnover is a daily fraction; Sharpe is gross.

	Q1: Low volatility		Q5: High volatility	
Model	Turnover	Sharpe	Turnover	Sharpe
OLS	0.223	1.142	0.153	0.859
LASSO	0.237	1.149	0.166	0.843
CNN (SGD)	0.184	1.158	0.144	0.823
CNN (Muon)	0.198	1.150	0.150	0.813
CNN (Adam)	0.210	1.122	0.161	0.853
LSTM (SGD)	0.164	1.116	0.121	0.851
LSTM (Muon)	0.209	1.182	0.157	0.841
LSTM (Adam)	0.212	1.152	0.163	0.834
MLP (SGD)	0.228	1.133	0.165	0.849
MLP (Muon)	0.164	1.144	0.107	0.850
MLP (Adam)	0.208	1.110	0.154	0.836
Transformer (SGD)	0.218	1.136	0.168	0.829
Transformer (Muon)	0.208	1.105	0.164	0.840
Transformer (Adam)	0.208	1.150	0.164	0.835

H Further Related Work

Our work relates to several strands of literature on underspecification, optimization-induced inductive bias, interpretability, and empirical benchmarking in financial machine learning. While each of these literatures documents important facets of modern learning systems, they are rarely studied jointly in low signal-to-noise time-series environments.

H.1 Underspecification and Predictive Multiplicity

When the available data are only *weakly informative* about the target relationship, empirical risk minimization is best viewed as searching a landscape with a *broad, shallow basin*: many parameter settings sit on (nearly) the same "loss contour," so the training objective does not single out a canonical predictor. This phenomenon has been formalized as *underspecification* (D'Amour et al., 2020): an ML *pipeline* (data, preprocessing, hypothesis class, optimizer, validation protocol) is underspecified when it can output many *distinct* predictors that are indistinguishable under in-distribution validation yet behave differently in deployment. Underspecification is closely related to Breiman's *Rashomon effect* (Breiman, 2001), named after Kurosawa's film in which multiple coherent narratives coexist without a single definitive truth (Kurosawa, 1950). The key methodological point is not merely that multiple models achieve similar error, but that *the data do not refute many qualitatively different functional explanations*.

Rashomon sets as a geometric object. Fix a function class $\mathcal{F}$ and loss ℓ. Writing $\widehat{R}_n(f) = \frac{1}{n}\sum_{i=1}^{n} \ell(f(x_i), y_i)$ and $\widehat{R}_n^\star = \inf_{f\in\mathcal{F}} \widehat{R}_n(f)$, the (empirical) Rashomon set at tolerance $\varepsilon \geq 0$ is

$$\mathcal{R}_\varepsilon(\mathcal{F}) \;=\; \Big\{ f \in \mathcal{F} \;:\; \widehat{R}_n(f) \leq \widehat{R}_n^\star + \varepsilon \Big\}.$$

In high-noise or intrinsically stochastic domains, $\mathcal{R}_\varepsilon(\mathcal{F})$ can be *large* even for small ε: recent results isolate regimes where noisy label generation provably induces large collections of near-optimal models (Semenova et al., 2023), and perspective work emphasizes that such multiplicity is common in high-stakes tabular settings (e.g., credit decisions, healthcare, criminal justice), precisely because the outcome mechanism is nondeterministic (Rudin et al., 2024). Moreover, a large Rashomon set is empirically and theoretically linked to the existence of *simple-yet-accurate* predictors (Semenova et al., 2022), contributing to the broader evidence that, on many tabular problems, high accuracy does not require black-box complexity (Holte, 1993; Lou et al., 2013; Angelino et al., 2018; McTavish et al., 2022; Liu et al., 2022b; McElfresh et al., 2023).

Near-optimal risk need not mean functional equivalence. A central complication is that proximity in scalar risk does not control pointwise agreement. Two predictors can both lie in $\mathcal{R}_\varepsilon(\mathcal{F})$ and yet diverge sharply on individual instances, which is precisely the operational content of *predictive multiplicity*. In classification, one natural disagreement functional is

$$\mathrm{Dis}(f, g) \;=\; \mathbb{P}_{X\sim\mathcal{D}_X}\big[f(X) \neq g(X)\big], \qquad f, g \in \mathcal{R}_\varepsilon(\mathcal{F}),$$

and analogues for regression can track sign flips or threshold exceedances (e.g., $\mathbb{P}[\,|f(X) - g(X)| > \tau\,]$), which are often more decision-relevant than MSE. Predictive multiplicity has been explicitly quantified in classification (Marx et al., 2020) and broadened to procedural/justificatory concerns in policy-facing pipelines (Black et al., 2022). Importantly, the underlying observation that "many models fit about equally well" appears in older statistical and econometric discussions of model specification and competing adequate descriptions (Mountain & Hsiao, 1989; McCullagh & Nelder, 1989); what is comparatively less emphasized there is the downstream implication Breiman highlighted: if multiple distinct predictors induce different narratives (feature attributions, mechanisms, causal stories), then *any single-model explanation is, without further structure, underdetermined by the data* (Breiman, 2001; Marx et al., 2020).

Why we need behavioral evaluation beyond a single score. These observations motivate evaluation protocols that treat a benchmark metric as *necessary but not sufficient*. Rather than interpreting a one-number gap (say, a tiny Δ in validation loss) as evidence of substantive superiority, one should ask how much *behavioral latitude* remains within $\mathcal{R}_\varepsilon(\mathcal{F})$:

- *Stability of decisions:* how large can $\mathrm{Dis}(f, g)$ be among near-optimal f, g? (Marx et al., 2020; Black et al., 2022)
- *Stability of boundary geometry:* how much can decision regions/threshold crossings move while preserving $\widehat{R}_n$? (Rudin et al., 2024)
- *Constraint feasibility:* does the Rashomon set contain models satisfying domain constraints (monotonicity, fairness, sparsity) *without* sacrificing accuracy? (Rudin et al., 2024)

Recent algorithms now make this viewpoint operational by explicitly representing or approximating Rashomon sets for nontrivial classes—including sparse decision trees, sparse GAMs, and risk scores—so that one can *query* the set for models that satisfy user preferences and examine variable importance *across all good models*, rather than conditioning on a single trained predictor (Xin et al., 2022; Zhong et al., 2023; Liu et al., 2022a; Zhu et al., 2023; Rudin et al., 2024).

Implications for financial forecasting. Financial prediction problems are prototypical "noisy-label" regimes: outcomes are affected by latent factors and stochastic shocks, so it is natural to expect large Rashomon sets even when the modeling class is heavily restricted (Rudin et al., 2024; Semenova et al., 2023). Yet evaluation in finance often reduces model selection to a single scalar score (e.g., a loss, a correlation, or a portfolio-level summary), effectively projecting a high-dimensional behavioral object $\mathcal{R}_\varepsilon(\mathcal{F})$ onto one axis. The econometrics literature often responds to model uncertainty by combining forecasts (Timmermann, 2006); in contrast, we focus on a different question: when we *must* select a single predictor, which *inductive biases* (regularization, architecture, optimization, data processing choices) decide *where* we land within $\mathcal{R}_\varepsilon(\mathcal{F})$, and how that choice governs the stability of instance-level predictions, decision thresholds, and downstream actions under underspecification.

H.2 Implicit Bias of Optimization Trajectories

Changing optimizer or hyperparameters: same objective, different solutions. A substantial literature documents that changing the optimizer (or its noise scale) can move learning toward different regions of parameter space with different curvature, geometry, and generalization behavior. Empirically, SGD-like methods are often associated with flatter solutions and better test performance than adaptive methods in high-signal domains (Wilson et al., 2017; Keskar et al., 2017), while adaptive optimizers can exhibit distinct "effective regularization" behavior unless properly regularized (notably via decoupled weight decay) (Loshchilov & Hutter, 2019; Zou et al., 2021). These observations connect to a longer thread linking curvature/sharpness to generalization (Hochreiter & Schmidhuber, 1997a; Keskar et al., 2017; Dinh et al., 2017; Jastrzebski et al., 2021; Foret et al., 2021), though the sharpness–generalization relationship is subtle and depends on parametrization and invariances (Dinh et al., 2017). From the present paper's perspective, the key point is methodological: much of the optimizer-comparison literature focuses on regimes where optimizer choice changes *test error*. In contrast, in low-signal settings (the regime of our experiments), optimizer choice can be consequential even when test losses are statistically indistinguishable, because the optimizer determines *which* near-optimal function is selected.

SGD *vs* Adam. A useful mental model is that the *loss* draws the landscape, but the *optimizer* chooses the vector field that actually drives the dynamics; in that sense, optimizing the same L with different methods can amount to (approximately) following gradient flow on different "effective" functionals.

- **Empirical signatures of the gap.** In language modeling, heavy-tailed (Zipf) class frequencies create a regime where GD/SGD makes little progress on rare classes, while Adam and sign-based methods remain comparatively insensitive (Kunstner et al., 2024). In transformers, the Hessian spectra vary dramatically across parameter blocks ("block heterogeneity"), so a single global learning rate makes SGD ill-suited, whereas Adam's coordinate-wise scaling mitigates this heterogeneity (Zhang et al., 2024). More broadly, adaptive methods can select qualitatively different solutions (and generalize differently) than SGD on problems with many minimizers (Wilson et al., 2017).
- **Modified-loss viewpoint for SGD.** For small but finite step size, the mean SGD iterate (under random shuffling) stays close to gradient flow on a *modified loss*—the original loss plus an implicit regularizer penalizing minibatch-gradient norms—with correction scale proportional to η/B (Smith

et al., 2021). In the more realistic without-replacement regime, SGD is locally equivalent to an *additional* step on a novel regularizer, which effectively regularizes the trace of the gradient-noise covariance (and often a weighted Fisher) along flat directions (Beneventano, 2023). Related symmetry-based analyses show that minibatch noise can select "noise-balanced" solutions and shape the stationary distribution of the stochastic gradient flow (Ziyin et al., 2025). At a high level, these results can be summarized as a first-order perturbation

$$\tilde{L}_{\mathrm{SGD}}(\theta) = L(\theta) + \frac{\eta}{B} R(\theta) + o(\eta/B), \qquad \dot{\theta} \approx -\nabla \tilde{L}_{\mathrm{SGD}}(\theta).$$

- **Adam differs already at leading order.** For Adam/RMSProp, backward-error analysis yields hyperparameter-dependent drift terms that can penalize a (perturbed) ℓ_1-type norm of ∇L or even *impede* its reduction (a typical regime), rather than inducing the same ℓ_2-type regularization as GD (Cattaneo et al., 2023; Cattaneo & Shigida, 2025; 2026). Moreover, taking $\eta \to 0$ while keeping momentum parameters fixed, Adam approaches a sign-gradient flow, indicating that the limiting vector field can differ from $-\nabla L$ already at *zeroth* order in η (Ma et al., 2022). This discrepancy is reflected in convergence theory: Adam can track an "Adam vector field" whose zeros generally differ from critical points of L (Dereich & Jentzen, 2024), and classic convex counterexamples show that Adam may even fail to converge to the minimizer (Reddi et al., 2019).

Classical implicit bias: minimum-norm and maximum-margin principles. Beyond comparisons between optimizers, a complementary line of work asks *which* predictor is selected by a given optimization trajectory when the empirical objective admits many global minimizers. For linearly separable classification with exponential-tail losses (e.g., logistic / cross-entropy), gradient descent can drive $\|w_t\| \to \infty$ while the *direction* $w_t/\|w_t\|$ converges to a maximum-margin separator, thereby making the "implicit regularizer" explicit (Soudry et al., 2018; Ji & Telgarsky, 2019; Lyu & Li, 2020; Gunasekar et al., 2018b). Concretely, in the linear case the limiting direction aligns with a solution of the hard-margin SVM problem,

$$\hat{w} \in \arg\min_{w} \ \|w\|_2^2 \quad \text{s.t.} \quad y_i \langle w, x_i \rangle \geq 1 \ \ \forall i,$$

even though the training objective contains no explicit norm penalty (Soudry et al., 2018; Ji & Telgarsky, 2019). In deep *homogeneous* models trained with exponential-tail losses, analogous margin-maximizing behavior emerges (in appropriate normalizations/limits), with the normalized margin increasing and converging to KKT points of a natural max-margin problem (Lyu & Li, 2020). A key nuance is that the induced notion of "simplicity" depends not only on the loss but also on the *parameterization* and the *optimization geometry*: reparameterizations realizing the same predictor class can yield different implicit regularizers (Gunasekar et al., 2018b). For instance, linear convolutional parameterizations bias toward sparsity in the frequency domain via an $\ell_{2/L}$-type penalty, in contrast to fully connected linear parameterizations whose bias is ℓ_2 max-margin (Gunasekar et al., 2018a); and factorized parameterizations in underdetermined quadratic objectives can bias toward low-complexity (e.g., low nuclear-norm) solutions (Gunasekar et al., 2017).

In regression and "lazy" (linearized) regimes, the same selection phenomenon appears as *minimum-norm interpolation* in an induced function space. In the infinite-width limit, training dynamics of wide networks are governed by kernel gradient descent with the neural tangent kernel (NTK) (Jacot et al., 2018; Lee et al., 2019; Chizat et al., 2019), so that (for square loss, and in the interpolating setting) gradient-based training selects the minimum-RKHS-norm interpolant:

$$f_\star \in \arg\min_{f \in \mathcal{H}_{\mathrm{NTK}}} \ \|f\|_{\mathcal{H}_{\mathrm{NTK}}} \quad \text{s.t.} \quad f(x_i) = y_i \ \ \forall i.$$

Taken together, these works support a unifying picture: when many interpolating solutions exist, the optimizer–parameterization pair often induces a geometry in which training selects a canonical low-complexity solution (max-margin for separable classification; minimum-norm in an induced hypothesis space for regression), despite the absence of any explicit complexity penalty (Gunasekar et al., 2018b; Jacot et al., 2018; Lee et al., 2019).

Biases along the trajectory: learning order, signal selection, and terminal geometry. Implicit bias is not only a statement about the final predictor, but also about *which components are learned first* and *which signals dominate* under the training dynamics. A prominent example is *spectral bias*: neural networks tend to fit low-frequency components before high-frequency components,

shaping intermediate functions along training and sometimes constraining what is practically learned under finite-time optimization (Rahaman et al., 2019). (Complementarily, NTK-based analyses make explicit how learning rates depend on the kernel spectrum, which can be interpreted as a frequency- or eigendirection-dependent ordering of fit (Jacot et al., 2018; Lee et al., 2019).) At the representation level, late-phase training can exhibit structured terminal geometry, including within-class collapse and simplex-like class arrangements ("neural collapse") (Papyan et al., 2020). Alongside these geometric phenomena, modern evidence emphasizes that standard ERM/SGD pipelines can preferentially exploit "simple" or "shortcut" features—highly predictive yet brittle signals—depending on the data/optimization geometry (Ilyas et al., 2019; Geirhos et al., 2020; Shah et al., 2020; Sagawa et al., 2020). This line also connects to methods designed to *avoid* such shortcut reliance by enforcing invariances across environments (e.g., invariant risk minimization) (Arjovsky et al., 2019). Most recently, theory suggests SGD-like dynamics can even *recover support* by suppressing irrelevant features, yielding sparse *effective* solutions without explicit sparsity regularization (Beneventano et al., 2024). Overall, these results reinforce that optimizer-induced preferences can manifest as differences in (i) *temporal learning order* (e.g., low-to-high frequency), (ii) *feature reliance* (robust *vs* spurious/shortcut), and (iii) *terminal representation geometry*, even when the scalar training loss is unchanged.

Edge of Stability. A growing literature connects optimizer hyperparameters to *curvature-regulated* training dynamics, framing optimization as a trajectory constrained by stability considerations rather than a purely "loss-minimizing" procedure. In full-batch (or sufficiently large-batch) training, this viewpoint typically decomposes learning into two regimes: an early phase of *progressive sharpening*—where the dominant curvature scale increases rapidly—and a later oscillatory phase where training hovers near the boundary of instability, i.e., the *Edge of Stability* (EoS).

Concretely, prior work documents that the leading Hessian eigenvalue $\lambda_{\max}$ often grows throughout early training (sometimes after a brief initial decrease), reflecting that the trajectory moves into progressively sharper regions of the landscape (Jastrzębski et al., 2019; 2020; Cohen et al., 2022b). This growth does not continue indefinitely: Jastrzębski et al. (2020) highlight a relatively abrupt *phase transition* ("break-even") that marks the end of the progressive sharpening regime. Importantly, this transition is typically attributed to *algorithmic* stability limits (i.e., the update rule becoming marginally unstable), rather than a static property of the loss surface. Consistent with that interpretation, the location of the transition depends on the optimization method and its hyperparameters even when the underlying task and model are held fixed (Jastrzębski et al., 2019; 2020; Cohen et al., 2022b). In the EoS regime itself, full-batch GD exhibits the canonical signature $\lambda_{\max} \approx 2/\eta$ (the quadratic stability threshold), with $\lambda_{\max}$ fluctuating around that boundary during training (Cohen et al., 2022b). Empirically, a substantial fraction of optimization—especially under MSE-like objectives—often occurs in this marginally stable regime, which can strongly influence the curvature (and hence geometry) of the final solution (Cohen et al., 2022b). Subsequent analyses explain why $\lambda_{\max}$ can slightly exceed $2/\eta$ in practice: higher-order nonlinearities shift the effective stability condition away from the quadratic approximation (Chen & Bruna, 2023). Mechanistically, Damian et al. (2023) propose a "self-stabilization" picture for GD at the edge, where third-order effects can act as a stabilizing force under empirically observed alignment conditions; however, extending this explanation cleanly to mini-batch training remains nontrivial.

Adaptive and Stochastic Edge of Stability. Recent work extends the "edge" framework to adaptive optimizers. Cohen et al. (2022a) show that full-batch Adam does not generally sit at an edge defined by the *raw* Hessian. Instead, it operates near an *Adaptive Edge of Stability* (AEoS): the stability boundary is governed by the spectrum of a *preconditioned* curvature matrix (e.g., the maximum eigenvalue of $P^{-1}H$, where P is Adam's second-moment preconditioner). In this view, the relevant instability threshold scales like $\lambda_{\max}(P^{-1}H) \approx c/\eta$ (with c depending on Adam's hyperparameters, e.g., β_1), while the unpreconditioned Hessian can continue to grow because the preconditioner dynamically rescales directions of high curvature. This produces a qualitative distinction from GD/SGD: non-adaptive methods are "pushed away" from high-curvature regions by step-size instability, whereas adaptive methods can traverse into higher raw-curvature regions while maintaining stability via preconditioning.

Finally, bringing the edge narrative to *mini-batch* SGD is delicate because oscillations and noise are not, by themselves, certificates of instability. Andreyev & Beneventano (2025) propose the *Edge of*

Stochastic Stability (EoSS), which reframes the edge in terms of computable *instability certificates* for the local quadratic model under stochastic gradients. A key empirical and conceptual contribution is to replace full-batch $\lambda_{\max}$ with an SGD-relevant curvature statistic—*Batch Sharpness*—that captures direction-aware curvature at the mini-batch level. Empirically, Batch Sharpness sharpens and then saturates near a stability boundary proportional to $2/\eta$, and threshold crossings coincide with catapult-like excursions and loss spikes under targeted (η, b) interventions. We use this line of work to motivate the matched-checkpoint learning-rate and batch-size interventions. Their raw-curvature responses support a stability-constraint interpretation in identified configurations; the measured statistic and stabilizing feedback need not coincide with the canonical EoSS criterion.

What is (still) poorly understood in tied-loss regimes. Most implicit-bias results are derived in regimes where either (i) the data are separable/high-signal and the optimizer selects a margin- or norm-optimal classifier (Soudry et al., 2018; Lyu & Li, 2020), or (ii) optimization choices drive measurable test-error differences (Wilson et al., 2017; Keskar et al., 2017). By contrast, the regime emphasized in this paper—*predictive equivalence with functional divergence*—asks for a different kind of characterization: not "which optimizer generalizes better," but "which function is selected when generalization is a tie". Our empirical findings can be viewed as evidence that, in low-signal time series, optimizer-dependent stability constraints and update geometry instantiate a consequential implicit prior over admissible predictors, making the choice of optimizer an integral part of the modeling assumption rather than a tool to achieve better performance.

H.3 Interpretability and SHAP

Interpretability and signal selection. A parallel literature shows that identical test error can conceal qualitatively different feature reliance and decision logic. Explanatory or interpretability multiplicity arises when near-optimal predictors admit conflicting post-hoc explanations for the same inputs (Brunet et al., 2022). Related work in vision demonstrates that models trained with standard empirical risk minimization may exploit shortcut or non-robust features that are predictive but semantically misaligned (Ilyas et al., 2019). Closest to our focus, studies of optimization bias show that learning dynamics can prioritize particular signals even when richer alternatives exist within the hypothesis space (Shah et al., 2020; Lampinen et al., 2024). These findings suggest that optimization pathways shape *which* signal is learned, not merely how well it fits the data. In the financial domain, Lo & Singh (2023) similarly emphasize that high predictive performance is insufficient; understanding the internal mechanics and "interpretability" of deep models is a prerequisite for their deployment in risk premia forecasting.

H.4 Neural Networks in Finance

Benchmarking in financial machine learning. Benchmark studies in finance report mixed evidence: deep models can help with rich covariates or cross-series pooling, but well-specified linear baselines remain competitive in low-SNR settings (Gu et al., 2020; Chen et al., 2024; Vortelinos, 2017; Branco et al., 2024). To demonstrate gains, many studies introduce additional features, alternative targets, or increasingly complex architectures (Heaton et al., 2017; Lim & Zohren, 2021; Christensen et al., 2022). While valuable, this focus leaves open a more fundamental question: when performance ties persist, what distinguishes the learned predictors themselves? Our work departs from the benchmark race by holding data and objectives fixed and instead interrogating the functional consequences of training choices.

Volatility forecast with classical ML. Volatility forecasting has a long history in econometrics. Classical models like ARCH/GARCH processes (Engle, 1982; Bollerslev, 1986) and their extensions treat volatility as a latent process and have achieved considerable success. The availability of high-frequency data has further refined these targets, allowing for non-parametric ex-post measurement of volatility via realized variance and bipower variation (Barndorff-Nielsen & Shephard, 2004), which now serve as standard ground-truth proxies. The Heterogeneous Autoregressive (HAR) model (Corsi, 2009) further improved daily volatility forecasts by incorporating multi-horizon averages (e.g., daily, weekly, monthly volatility) in a simple linear framework. These parsimonious models often perform robustly across datasets and remain strong benchmarks (Vortelinos, 2017; Branco et al., 2024).

With the rise of data mining, researchers began exploring machine learning (ML) algorithms for volatility prediction. Tree-based ensembles and kernel methods (e.g., random forests, boosting, SVMs) have been applied to capture nonlinear patterns in volatility dynamics. However, early studies found that when using only past returns or volatilities as features, complex ML models often provided little to no improvement over well-specified linear models (Branco et al., 2024; Vortelinos, 2017; Souto & Moradi, 2023; Ma et al., 2024). For example, Branco et al. (2024) find that no ML method consistently beats a HAR model when forecasting realized volatility out-of-sample, especially at longer horizons. These results echo other comparisons showing that traditional models can remain competitive in feature-constrained settings (Vortelinos, 2017; Branco et al., 2024). In practice, the performance gap narrows without additional data: adding macro-financial features or high-frequency inputs is often helpful for ML models to outperform simple baselines. This underscores that in low signal-to-noise regimes, many models achieve similar predictive accuracy, setting the stage for potential underspecification.

Deep Neural Networks for Volatility Forecasting. The application of deep learning to volatility forecasting has attracted significant attention in recent years (Leushuis & Petkov, 2026). Early neural network approaches, including multilayer perceptrons and recurrent networks, were explored as alternatives to GARCH. For instance, Donaldson & Kamstra (1997) proposed one of the first ANN-GARCH hybrid models and found that a neural network could approximate the volatility process similarly to GARCH. More recently, Di-Giorgi et al. (2025) formalize this connection by showing that deep recurrent neural networks can replicate the recursive structure of GARCH-type models while retaining the flexibility of data-driven learning. Deep architectures have also been deployed at scale (e.g., Bucci, 2020): LSTMs, in particular, have been a popular choice due to their ability to capture long-term dependencies in financial time series (Hochreiter & Schmidhuber, 1997b). An empirical study by Souto & Moradi (2023) reports that LSTM-based models can outperform HAR in some settings; improvements vary by horizon and features. Recent work has attempted to extend these architectures by incorporating attention mechanisms or hybridizing them with classical error-correction terms (Leushuis & Petkov, 2026; Kumar & Thenmozhi, 2021).

Beyond RNNs, researchers have explored CNNs and attention-based models (Reisenhofer et al., 2022; Betz et al., 2023). Temporal convolutional networks and Transformers have been evaluated on volatility prediction tasks, sometimes as components of larger frameworks. For instance, Zhang et al. (2023) leverage cross-asset intraday data and show that a simple feed-forward network pooling many stocks' volatilities outperforms linear and tree models, attributing the gain to the network's ability to capture latent "commonality" in volatility. Moreover, it is demonstrated that, given rich feature sets or additional signals (like macro variables, order flow), deep networks can significantly outperform traditional models (Rahimikia & Poon, 2020; Zhang et al., 2023). On the other hand, in purely univariate settings with only past realizations as inputs, even sophisticated networks often end up "tied" with linear models on standard error metrics (Gu et al., 2020; Chen et al., 2024; Bucci, 2020; Betz et al., 2023). This phenomenon is evident in many recent financial ML benchmarks, where increasing model complexity yields diminishing returns (Hu et al., 2025). As a result, many studies report that volatility forecasting is a tough low-signal task: simple and complex models often achieve statistically indistinguishable accuracy. This backdrop of frequent leaderboard ties motivates going beyond aggregate metrics to examine what models are actually learning.

Foundation Models and Recent Advances in Time-Series Forecasting. Inspired by the success of large-scale models in NLP and vision, researchers have begun developing foundation models for time series (Bommasani et al., 2022). These are extensive pre-trained models (often Transformer-based) that aim to capture universal time-series patterns and enable zero-shot or few-shot forecasting across diverse tasks. For example, Google's TimesFM is a pre-trained Transformer that demonstrated accurate zero-shot forecasts across many time series without task-specific training (Das et al., 2024). Similarly, Salesforce's MOIRAI and other large time-series models have been released, alongside open-source initiatives like MOMENT (Woo et al., 2024; Goswami et al., 2024). These models leverage massive datasets and novel architectures to approach time-series forecasting as a generalizable transfer-learning problem (Zhou et al., 2023). Despite their promise, current foundation time-series models still face the classic pitfalls seen in earlier studies. A recent survey by Liang et al. (2024) highlights that carefully tuned lightweight models can rival or even beat foundation models on many benchmarks, especially when the latter are not fine-tuned. Indeed, a NeurIPS 2025 workshop (Moreau et al., 2025) noted that time-series foundation models have yet to achieve a definitive "BERT

moment"—in other words, they do not strictly dominate dedicated models in all cases. In volatility forecasting specifically, even the most advanced deep models (e.g., Transformer encoders or hybrid LSTM-GARCH schemes) often yield performance on par with simpler methods (D'Amato et al., 2022; Vortelinos, 2017; Christensen et al., 2022). Moreover, these large models typically rely on adaptive optimizers (e.g., Adam) during training, treating the optimizer as a default setting rather than a design choice. Our work raises the point that even in such state-of-the-art systems, if multiple training runs or optimization strategies achieve similar loss, they could embed different inductive biases. As foundation models push predictive performance limits, understanding the role of the training procedure becomes crucial to ensure consistency and reliability of the learned patterns across deployments.

Positioning of Our Contribution. In light of the above, our work sits at the intersection of these threads. We focus on a domain—daily equity volatility forecasting—where deep learning models have struggled to decisively outperform parsimonious econometric models (Branco et al., 2024; Vortelinos, 2017; Christensen et al., 2022). Rather than introduce another architecture or exogenous feature, we interrogate the underspecification that arises in this low signal-to-noise regime (D'Amour et al., 2020). Prior literature has observed that many ML predictors are observationally equivalent in such settings, typically choosing between them based on minor differences in validation loss or aesthetic preferences. We extend this discussion by showing that when several models tie in predictive accuracy, the choice of optimizer can be a decisive factor in which functional form is learned. While others have noted the implicit biases of optimization in high-impact domains (Wilson et al., 2017; Zou et al., 2021), we demonstrate this effect in financial time series, where it has been less systematically studied. In doing so, we bridge the gap between benchmarking studies (emphasizing which model yields lower error) and interpretability or decision-focused analyses (emphasizing how the model's behavior matters for users). Our findings suggest that financial ML researchers should evaluate models not just by Sharpe ratios or MSE, but also by the stability and economic plausibility of the patterns they capture. By revealing that different architecture–optimizer pairs can encode substantively different volatility dynamics despite equal performance, we provide a new perspective on model selection: when performance is tied, selecting the model with the right inductive bias (e.g., smoother responses, longer memory, lower turnover) becomes paramount for downstream applications. This nuanced approach to evaluation complements existing work on interpretable ML (Rudin et al., 2022) in finance and calls for broader adoption of diagnostics that account for functional and decision-level differences, not just error metrics.