# LiFT: Neurosymbolic Conditioning For Portfolio Allocation

**Anonymous authors**
Paper under double-blind review

## Abstract

Deep learning portfolio allocators achieve strong in-sample performance but degrade in out-of-sample performance during regime shifts. To mitigate this, recent deep portfolio allocators add market knowledge and investment rules at three levels: the input, where regime labels are added to the features; the representation, where a conditioning signal modulates each stock's encoding; or the output, as constrained decision layers or differentiable logic penalties on portfolio limits. Each is demonstrated on a single architecture, so the levels cannot be compared, and no gain can be attributed to the underlying knowledge. Supplying all three to any encoder raises three challenges. A regime describes the current market state, not a stock, so it has no natural place in a stock's input series. Trading rules are discrete, giving no usable gradient, and their relative weights are undetermined. Risk rules cap holdings but never require a minimum, so holding only cash satisfies them all. To address all three, we present **LiFT**, a neurosymbolic framework that operates at the input, representation, and output level of any time-series encoder. At input, market-state features join the per-stock inputs, At the representation level, binary regime indicators modulate each stock's encoding via FiLM. At output, trading rules become differentiable penalties, with each rule's weight learned by dual ascent. A minimum-investment rule and a cash ceiling prevent the all-cash solution. Being encoder-agnostic, the three additions run unchanged on five encoders, two stock universes, and four walk-forward windows at matched capacity. Together they condition allocation on market state and penalize rule violations during training. LiFT improves out-of-sample Sharpe over the baseline by 50% on the DJIA, on all five encoders, and by 25% on the S&P 100, with Sortino and Calmar also improving. Separating the levels reveals which knowledge drives these gains.

## 1 Introduction

Portfolio allocation is a sequential decision problem over a fixed set of stocks and cash (Markowitz, 1952; Nystrup et al., 2020) that decides what fraction of capital to hold in each stock and cash. The portfolio is rebalanced at regular intervals, and every trade incurs fees and spreads. The fraction of the portfolio traded, its turnover, sets how high those costs are. A good allocator earns a high return for the risk it takes, net of those costs, measured here by the Sharpe ratio (Sharpe, 1994). Classically, those fractions came from three steps (Markowitz, 1952): estimate returns, estimate a covariance matrix, then optimize over both. A deep allocator replaces all three with a single network. It encodes a window of recent prices and volumes into one representation per stock. A softmax over those, plus a cash term, gives the weights, and the network is trained on the resulting Sharpe ratio rather than on any forecast.

Initial work built this representation with recurrent and convolutional encoders (Jiang et al., 2017; Zhang et al., 2020). Multivariate forecasting has since produced many encoders for that representation. Some treat each stock independently (Nie et al., 2023; Zeng et al., 2023). Others mix information across stocks with attention (Liu et al., 2024; Zhang & Yan, 2023) or a shared core (Han et al., 2024). TimeXer (Wang et al., 2024) and freTs (Shao et al., 2022) separate a stock's own history from series that describe the market. Each allocator pairs one such encoder with an allocation head (Jiang et al., 2017; Zhang et al., 2020; Cong et al., 2026). Recent allocators also provide the network with market knowledge in various forms. Regime labels mark each period, and market



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.

conditions are fed to the encoders as labels (Zhang et al., 2025). Market-guided gating computes a gate from market variables and uses it to scale the input features (Feng et al., 2024). Regime-robust objectives change the training loss so that performance is less sensitive to regime changes (Wood et al., 2026). Constraint indicators encode constraints as inputs, so the resulting weights can be explained (Mashayamombe et al., 2026).

These additions change where market knowledge enters, not the underlying estimation problem. An allocator fitted to historical data still loses much of its advantage out-of-sample. Consequently, Equal weighting, or $1/N$, fits nothing and is therefore hard to beat out-of-sample (DeMiguel et al., 2009). Deep allocators are no exception, and backtested gains often fail to survive out-of-sample (Bailey & Lopez de Prado, 2014). When gains are this fragile, it matters whether they come from the added market knowledge or from the particular architecture. Yet each of the systems above is built around one architecture and evaluated as a whole, so the two cannot be separated. Injecting market knowledge and trading rules into a time-series encoder is itself difficult, for three reasons. First, a regime is a property of the market, not of a stock, so it has no natural place among a stock's input series. Second, trading rules are discrete, so they give no usable gradient, and their relative weights are undetermined. Third, risk rules limit how much a portfolio holds but never require a minimum, so holding only cash satisfies all of them.

We present LiFT (Logic-informed FiLM Training), a neurosymbolic framework that answers all three. It attaches to any time-series encoder and operates on its input, representation, and output. Market-state features join the per-stock inputs. Binary regime indicators modulate each stock's representation through FiLM (Perez et al., 2018). Trading rules become differentiable penalties on the output weights (Badreddine et al., 2022; Hu et al., 2016; Garcez & Lamb, 2023). Each rule's weight is learned by dual ascent (Cotter et al., 2019; Fioretto et al., 2020). A minimum-investment rule and a gated cap exclude the all-cash solution. Each component can be turned off independently, making its contribution measurable. Our contributions are as follows.

- **An encoder-agnostic neurosymbolic framework.** LiFT adds domain knowledge at the input, representation, and output of any time-series encoder, leaving the encoder unchanged. Net of transaction costs, it raises out-of-sample Sharpe over a price-only baseline of the same size. The gain is about 50% on the DJIA and 25% on the S&P 100. All five encoders improve on the DJIA, and four of the five on the S&P 100, with Sortino and Calmar improving as well.
- **An attribution of the gain to each level.** Every configuration is trained at the same capacity, seeds, and windows, so the same 500 runs show which level does the work. Market-state inputs pay on both universes. The rules and the per-stock modulation add a further 20% on the DJIA, and nothing on the S&P 100. Shuffling the indicators in time leaves performance unchanged, so the gain does not depend on correctly detecting the regime.
- **The gap to equal weighting is a transaction-cost effect.** Rebalanced daily, as they are trained, no learned allocator in our study beats equal weighting net of costs. This is in line with DeMiguel et al. (2009). Holding each allocation for a month closes most of that gap, and LiFT's own advantage comes from lower turnover.

## 2 Related Work

**Classical allocation.** Mean–variance optimization estimates returns and covariances, then solves for weights (Markowitz, 1952). Shrinkage and Bayesian priors reduce the error in those estimates (Ledoit & Wolf, 2004; Black & Litterman, 1992). They remain noisy, however, which is why equal weighting is hard to beat out of sample (DeMiguel et al., 2009). A second line therefore drops return forecasts and allocates on risk alone. Hierarchical risk parity does so over a clustering of the assets (Lopez de Prado, 2016). Volatility-managed portfolios do so by cutting exposure when volatility is high (Moreira & Muir, 2017). We take three of these as reference points: equal weighting, inverse-volatility risk parity (Maillard et al., 2010) and hierarchical risk parity.

**Deep portfolio allocators.** Deep allocators replace that pipeline with a single network. Jiang et al. (2017) trained a convolutional policy network on portfolio return, and Zhang et al. (2020) optimized the Sharpe ratio end-to-end. Open reinforcement-learning libraries followed (Liu et al.,

2020), but none of these conditions on the state of the market. That matters because correlations rise and diversification weakens in bear markets (Ang & Bekaert, 2002; Bekaert et al., 2014). Recent allocators therefore add market knowledge, as cross-asset attention (Cong et al., 2026) or regime-labeled ensembles (Zhang et al., 2025). It also enters as a gate on the inputs (Feng et al., 2024) or as an objective robust across regimes (Wood et al., 2026). Outside allocation, Friday et al. (2026) use feature-wise modulation to predict price movements. Each of these systems is proposed and evaluated as a whole, on its own architecture and data, so its gains cannot be traced to any one part. Their reported results also differ in universe, rebalancing frequency and cost treatment, so we compare against baselines trained under our own protocol.

**Constraints and differentiable logic.** Domain knowledge can also be imposed on the output itself. Constrained output layers enforce structure exactly, through sparsemax, OptNet or convex optimization layers (Martins & Astudillo, 2016; Amos & Kolter, 2017; Agrawal et al., 2019). Decision-focused learning goes further and trains the model through the solver (Mandi et al., 2024). Both enforce constraints exactly, whereas differentiable logic makes them soft. Logic tensor networks assign truth values in $[0, 1]$ to first-order rules (Badreddine et al., 2022), so rule satisfaction becomes a differentiable function of the outputs. DeepProbLog does the same with probabilistic semantics (**?**), and Hu et al. (2016) instead distill rules from a rule-regularized teacher. Softening the rules leaves their weights open, and each can be a Lagrange multiplier updated by dual ascent (Cotter et al., 2019; Fioretto et al., 2020). In allocation, Mashayamombe et al. (2026) build constraint indicators into an LSTM to explain its weights. LiFT instead keeps a softmax head and places its rules in the training loss, so they shape training only. It also combines them with input- and representation-level knowledge, and measures what each adds over a model given the same inputs.

## 3 Problem Statement

Let $\mathcal{U} = \{1, \ldots, N\}$ be a fixed universe of stocks. On trading day $t$, the allocator observes, for each stock $i$, the last $T = 60$ days of $F = 16$ features, $\mathbf{x}_{i,t} \in \mathbb{R}^{T \times F}$, and $M = 6$ binary market-regime indicators $\mathbf{l}_t \in \{0, 1\}^M$ computed from daily market data available at the close of day $t$. It outputs weights $\mathbf{w}_t \in \Delta^{N+1}$ over the $N$ stocks and cash, held over day $t + 1$. The portfolio return net of transaction costs is

$$r^{\text{net}}_{t+1} = \mathbf{w}_t^\top \mathbf{r}_{t+1} - c \, \|\mathbf{w}_t - \mathbf{w}_{t-1}\|_1, \tag{1}$$

where $\mathbf{r}_{t+1} \in \mathbb{R}^{N+1}$ holds the stocks' returns and a zero return for cash, and $c = 20$ basis points.

Performance on a test window is measured by the annualized Sharpe ratio of $r^{\text{net}}$ (Sharpe, 1994), together with the Sortino (Sortino & Price, 1994)Calmar (Young, 1991) ratios, maximum drawdown, volatility, turnover, and average cash.

**Problem.** A learned allocator $f_\theta : (\mathbf{x}_{\cdot,t}, \mathbf{l}_t) \mapsto \mathbf{w}_t$ is trained on days before a cut-off, selected on a validation split immediately before it, and evaluated on the test period after it.

Its generalization gap is

$$G(f_\theta) = S\big(f_\theta; \mathcal{D}_{\text{val}}\big) - S\big(f_\theta; \mathcal{D}_{\text{test}}\big), \tag{2}$$

where $S$ is the Sharpe ratio on a dataset.

We seek allocators that keep $G$ small across market regimes without giving up validation performance. The difficulty is that a test period can fall in a regime that is rare or absent in the training data.

Three properties of the standard set-up make the gap hard to close by training alone.

1. *Market conditions are not in the input.* The encoder reads each stock's own history, so market-wide volatility or the index's directions reaches it only through the trace it leaves in individual prices.
2. *The representation does not depend on the regime.* The encoder applies the same mapping to a stock's history in a calm market and in a crisis.
3. *Constraints are absent from the objective.* The model is trained on a differentiable in-batch estimate of the Sharpe ratio, which says nothing about which portfolios are admissible.

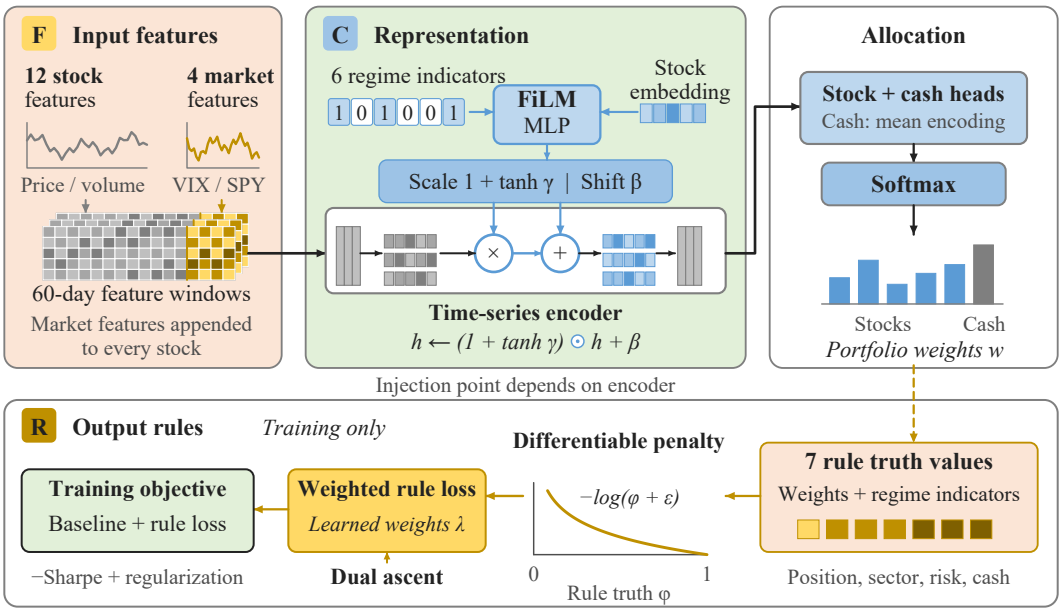


Figure 1: **Overview of LiFT.** Market knowledge enters an unchanged time-series allocator at three levels, each switchable independently. **(F) Input:** four market-state features are appended to every stock's price and volume features. **(C) Representation:** six binary regime indicators and a learned stock embedding drive a FiLM layer that scales and shifts each stock's vector, applied where the encoder first holds one vector per stock (Appendix A). **(R) Output, training only:** seven rules on position, sector, risk and cash are scored as truth values from the weights and indicators, turned into penalties weighted by multipliers learned by dual ascent, and added to the negative-Sharpe objective. Test-time weights come straight from the softmax.

> The limits a domain expert would impose on position size, sector exposure, and idle cash are logical statements with no gradient, so they never enter the loss. Without them, price-only baselines hold about 20% cash on average and more than 40% in the 2023–24 rising market.

LiFT addresses each property at one level; we ask which, once addressed, improves test performance and by how much.

## 4 Method

LiFT adds market knowledge to a baseline encoder at three levels: the input, the per-stock encoding, and the training loss. Each level can be switched off independently. Figure 1 shows where each one enters.

### 4.1 Baseline encoder

An encoder maps the feature window $\mathbf{x}_{i,t}$ of stock $i$ to a per-stock encoding $\mathbf{h}_{i,t} \in \mathbb{R}^d$. Encoders differ in how they mix information across stocks and across time; LiFT requires only that they produce one encoding per stock. A shared linear head produces a logit, and a second head gives a cash logit from the mean encoding $\bar{\mathbf{h}}_t$. A softmax gives the portfolio weights:

$$z_i = \mathbf{v}^\top \mathbf{h}_{i,t} + b, \qquad z_{N+1} = \mathbf{v}_c^\top \bar{\mathbf{h}}_t + b_c, \qquad \mathbf{w}_t = \mathrm{softmax}(z_1, \ldots, z_{N+1}). \tag{3}$$

The model is trained to maximize the Sharpe ratio of the portfolio return over each batch. An $\ell_2$ penalty on the weights and an $\ell_1$ penalty on turnover are added:

$$\mathcal{L}_0 = -\frac{\hat{\mu}(r)}{\hat{\sigma}(r)} + \lambda_2 \|\mathbf{w}_t\|_2 + \lambda_f \|\mathbf{w}_t - \mathbf{w}_{t-1}\|_1, \tag{4}$$

where $\hat{\mu}$ and $\hat{\sigma}$ are the mean and standard deviation of the portfolio return in the batch. We call this the *baseline*. It sees only the twelve per-stock features. Apart from TimeXer's exogenous input, nothing in its input describes the market as a whole.

### 4.2 Input level: market-state features

The encoder receives the market state directly. On every day of the window, four market-state features are appended to each stock's twelve price and volume features:

$$\mathbf{m}_t = \Big( \tfrac{\mathrm{VIX}_t}{20} - 1,\ \ \Delta_{21}\mathrm{VIX}_t,\ \ \tfrac{P_t}{\mathrm{SMA}_{50}(P)_t} - 1,\ \ \tfrac{P_t}{\mathrm{SMA}_{200}(P)_t} - 1 \Big), \tag{5}$$

where $\mathrm{VIX}_t$ is the Volatility Index, a measure of expected market volatility, and $\Delta_{21}$ is it21-day s relative chanys. $P_t$ is the S&P 500 index level, as measured by the SPY exchange-traded fund, and $\mathrm{SMA}_k$ is its $k$-day simple moving average. The same $\mathbf{m}_t$ is shared by all stocks, so the encoder can learn how each stock's allocation should depend on market conditions.

### 4.3 Representation level: regime-conditioned modulation

The second level lets market conditions change the per-stock encoding itself. The same price history can then be read differently in a volatile market than in a calm one. Six binary regime indicators $\mathbf{l}_t \in \{0,1\}^6$ are fixed thresholds on daily market data, set before any model was trained: (a) VIX above 25; (b) the S&P 500 index below its 50-day simple moving average; (c) VIX up more than 15% over five days; (d) the index below its 200-day simple moving average; (e) the index's 14-day Relative Strength Index (RSI), a momentum measure, above 70; and (f) more than 60% of the universe's stocks below their own 50-day averages.

The indicators drive feature-wise linear modulation (FiLM) (Perez et al., 2018) of each stock's encoding. A two-layer network $g_\psi$ maps the indicators and a learned stock embedding $\mathbf{s}_i \in \mathbb{R}^{16}$ to a stock-specific scale and shift:

$$(\boldsymbol{\gamma}_{i,t}, \boldsymbol{\beta}_{i,t}) = g_\psi\big([\mathbf{l}_t; \mathbf{s}_i]\big), \qquad \mathbf{h}_{i,t} \leftarrow \mathbf{h}_{i,t} \odot \big(1 + \tanh \boldsymbol{\gamma}_{i,t}\big) + \boldsymbol{\beta}_{i,t}. \tag{6}$$

The stock embedding lets the same regime raise one stock's encoding and lower another's. We also evaluate a shared variant, whose scale and shift depend on $\mathbf{l}_t$ alone and apply identically to every stock, as an ablation.

The modulation is applied at the first point where an encoder holds one vector per stock. A per-stock scale and shift is defined only there. SOFTS and iTransformer embed each stock as a single token at the input. The modulation is applied before their layers that mix information across stocks. PatchTST and TimeXer encode each stock's patches separately. They form one vector per stock only after that stage, so the modulation is applied there. Crossformer forms one vector per stock between its cross-time and cross-stock attention stages. The modulation is applied between them (Appendix A). TimeXer's native exogenous input also receives the indicators, in every configuration including the baselines.

### 4.4 Output level: rules as a differentiable penalty

The first two levels provide the model with information; neither indicates which portfolios are admissible. The third level defines knowledge as seven rules governing the portfolio weights and indicators (Table 1). The rules use the semantics of logic tensor networks (Badreddine et al., 2022). Each rule $a$ has a truth value $\phi_a(\mathbf{w}, \mathbf{l}) \in [0,1]$ that is differentiable in $\mathbf{w}$. It contributes the penalty $-\log(\phi_a + \epsilon)$.

Rules 1 and 4 reduce risk in stressed markets, and rules 3 and 5 limit concentration in every regime. Rules 6 and 7 act on cash. Rule 6 prevents the degenerate solution of satisfying rules 1, 2, and 4 by holding cash every day. Rule 7 caps cash at 30% except when high volatility and a negative trend coincide. Implications use the Łukasiewicz form $\mathrm{clamp}(1 - A + B, 0, 1)$. Here, $A$ is the truth value of the condition, and $B$ is that of the consequence. Caps use one minus the hinge excess over the

Table 1: The seven rules, with initial weights and weights learned by dual ascent (mean over runs). High VIX and negative trend are indicators (a) and (b).

| # | Rule | Form | Initial | Learned |
|---|---|---|---|---|
| 1 | High VIX → low total stock weight | implication | 1 | 1.11 |
| 2 | High VIX → high cash or high weight in defensive stocks | implication | 1 | 1.02 |
| 3 | No single stock above 15% | cap | 5 | 5.12 |
| 4 | Negative trend → low weight in non-defensive stocks | implication | 1 | 1.96 |
| 5 | No sector above 40% | cap | 1 | 0.88 |
| 6 | Not high VIX → cash not high | implication | 1 | 2.49 |
| 7 | Not (high VIX and negative trend) → cash at most 30% | gated cap | 0 | 0.74 |

cap, summed over stocks or sectors. This assigns a gradient to each violating position and keeps each truth value bounded away from zero (Appendix F). The rule loss is

$$\mathcal{L}_{\text{rules}} = \rho_t \sum_{a=1}^{5} \lambda_a \big(-\log(\phi_a + \epsilon)\big) + \sum_{a \in \{6,7\}} \lambda_a \big(-\log(\phi_a + \epsilon)\big), \qquad \rho_t = 1 - \tfrac{1}{2}\,\bar{l}_{\text{HighVIX}}, \quad (7)$$

where $\bar{l}_{\text{HighVIX}}$ is the share of high-VIX days in the batch. The factor $\rho_t$ relaxes rules 1–5 when the market is already stressed; It does not apply to rules 6 and 7, which apply in calm markets. The full objective is $\mathcal{L} = \mathcal{L}_0 + \lambda_R\,\mathcal{L}_{\text{rules}}$ with $\lambda_R = 5 \times 10^{-3}$. The rules act only during training. At test time, the weights come from the softmax head with no projection or clipping. Any effect of the rules is therefore an effect on what the encoder learned.

**Rule weights.** The weights $\lambda_a$ are learned rather than fixed. Following constrained deep learning (Cotter et al., 2019; Fioretto et al., 2020), we treat each as a Lagrange multiplier. The constraint is that the truth value of rule $a$, averaged over the batch, is at least 0.95. We update it by projected dual ascent after every training step:

$$\lambda_a \leftarrow \text{clip}\big(\lambda_a + \eta\,\bar{v}_a,\ 0,\ \lambda_{\max}\big), \qquad \bar{v}_a \leftarrow 0.9\,\bar{v}_a + 0.1\,(0.95 - \phi_a), \quad (8)$$

with step size $\eta = 0.01$ and $\lambda_{\max} = 10$, starting from the initial values in Table 1. A weight rises when its rule falls short of the target and falls otherwise. The balance between the rules is therefore set during training rather than by hand. Each run learns its own weights, reported per encoder in Appendix F.

## 5 Experimental Setup

**Data and protocol.** We use daily OHLCV data from Yahoo Finance, January 2010 to February 2025, for the current 30 constituents of the Dow Jones Industrial Average (DJIA) and 99 of the S&P 100. Membership is as of 2025; survivorship bias is identical across configurations, so it affects levels, not contrasts. Each stock carries twelve price and volume features and four market-state features from SPY and the VIX, and six binary indicators threshold those series together with within-universe breadth (Appendix I). Features use only data available at the close of day $t$ and are z-scored from training rows alone; weights chosen on day $t$ earn the $t$ to $t + 1$ return. Four test windows share dates across universes and span July 2019 to December 2024: W0 the COVID-19 crash and recovery, W1 a rate-driven bear market, W2 recovery, and W3 a bull market (Appendix E). Each trains on all data from 2010 up to its test period, with the last 15% of training days held out for early stopping, so no test day is in sample. Every encoder has the same width and depth, $d = 128$ with three layers, held identical across arms, and no hyperparameter is tuned (Appendix A). Training stops early on validation Sharpe net of the same 20-basis-point cost as the test metric (Appendix H). Portfolios rebalance daily, and we then measure what that costs.

**Ablation design.** Three binary factors, the inputs (F), conditioning (C) and rules (R), give eight configurations, each trained for 25 seeds, four windows and five encoders: 500 runs per configuration and universe. The *baseline* uses no level; LiFT without a qualifier is F+C+R. Because the inputs

Table 2: Sharpe ratio by encoder (top) and all metrics (bottom), mean over 100 runs per encoder and 500 runs per universe. Return, volatility, drawdown, and cash in %. Bold: best in row. †: LiFT differs from the baseline (both intervals); ‡: from LiFT (F) (run interval).

| **DJIA** | Baseline | LiFT (F) | LiFT |
|---|---|---|---|
| *Sharpe ratio by encoder* | | | |
| SOFTS | 0.299 | 0.353 | **0.382** |
| iTransformer | −0.011 | 0.001 | **0.046** |
| PatchTST | 0.281 | 0.407 | **0.449** |
| TimeXer | 0.204 | 0.216 | **0.394** |
| Crossformer | 0.298 | **0.356** | 0.338 |
| *Mean over encoders* | | | |
| Sharpe ↑ | 0.214 | 0.267 | **0.322**†‡ |
| Sortino ↑ | 0.348 | 0.425 | **0.499**†‡ |
| Calmar ↑ | 0.297 | 0.383 | **0.410**† |
| Return ↑ | 3.76 | 4.26 | **4.95**†‡ |
| Volatility ↓ | **16.77** | 17.11 | 17.15 |
| Drawdown ↓ | 19.21 | 19.30 | **19.04** |
| Turnover ↓ | 0.134 | 0.134 | **0.123**†‡ |
| Cash | 20.2 | 21.1 | 20.6 |

| **S&P 100** | Baseline | LiFT (F) | LiFT |
|---|---|---|---|
| *Sharpe ratio by encoder* | | | |
| SOFTS | 0.523 | 0.563 | **0.580** |
| iTransformer | 0.125 | **0.197** | 0.195 |
| PatchTST | 0.261 | **0.475** | 0.400 |
| TimeXer | 0.213 | **0.368** | 0.318 |
| Crossformer | 0.373 | **0.378** | 0.370 |
| *Mean over encoders* | | | |
| Sharpe ↑ | 0.299 | **0.396** | 0.373† |
| Sortino ↑ | 0.471 | **0.606** | 0.576† |
| Calmar ↑ | 0.313 | **0.397** | 0.385† |
| Return ↑ | 6.03 | **6.58** | 5.91 |
| Volatility ↓ | **17.10** | 18.35 | 18.70† |
| Drawdown ↓ | **20.05** | 21.63 | 22.18† |
| Turnover ↓ | 0.104 | **0.097** | 0.104 |
| Cash | 19.4 | 10.4 | 10.7† |

are the level any encoder can adopt on its own, C and R are measured against LiFT (F); against the baseline they would be credited with gains the inputs produce. Configurations with conditioning but no rules use the shared modulation while LiFT uses the stock-specific one, so contrasts changing both are marked. The shuffled-indicator control permutes indicators in time, separately in training and in each test window, keeping each indicator's frequency but not its day, with parameters identical to LiFT.

**Statistical testing.** Every effect is a paired difference on the same seed, window and encoder, reported as the mean with two 95% bootstrap intervals: a *run* interval resampling pairs, and an *encoder* interval resampling encoders. An effect is *reliable* only when both exclude zero. Runs and windows count equally, and we report means over all runs rather than the best (Appendices B and C).

# 6 Results

## 6.1 LiFT against the baselines

LiFT raises the baseline Sharpe ratio by 50% on the DJIA (0.214 to 0.322) and by 25% on the S&P 100 (0.299 to 0.373), with both intervals clear of zero (Table 2). It improves all five encoders on the DJIA and four of five on the S&P 100, where Crossformer is unchanged; separated by encoder, the interval clears zero in three of five and four of five respectively (Table 5). It also narrows the generalization gap $G$ of Equation equation 2, from 0.233 for the baseline and 0.241 for LiFT (F) to 0.180 on the DJIA, and from 0.157 to 0.106 on the S&P 100, where LiFT (F) narrows it further still (0.071; Table 6). On the DJIA, LiFT raises annual return from 3.8% to 5.0% and trades 8% less, with volatility and drawdown unchanged. On the S&P 100 it cuts cash from 19.4% to 10.7%, which raises volatility from 17.1% to 18.7% and maximum drawdown from 20.0% to 22.2%, and the gain concentrates in the 2023–24 bull market. Even so, Sortino and Calmar rise by 22% and 23%. Per-encoder behavior across all measures is in Figure 3.

## 6.2 Which level does the work

LiFT's input level is the only one that helps on both universes (Table 3): LiFT (F) beats the baseline by 0.053 on the DJIA and by 0.097 on the S&P 100, which there exceeds the whole gain of LiFT (+0.074). LiFT (F+C), which uses the shared modulation, is neutral on the DJIA and lowers the S&P 100 Sharpe ratio by 0.058, raising cash from 10.4% to 15.8%. On the DJIA the rules help

Table 3: Change in Sharpe ratio between configurations, paired over 500 matched runs. †: both intervals exclude zero; *: run interval only.

| Contrast | DJIA | S&P 100 |
|---|---|---|
| LiFT (F) − baseline | $+\mathbf{0.053}^{\dagger}$ | $+\mathbf{0.097}^{\dagger}$ |
| LiFT (R) − baseline | $+\mathbf{0.039}^{\dagger}$ | $-0.021$ |
| LiFT (F+C) − LiFT (F) | $-0.022$ | $-0.058^{\dagger}$ |
| LiFT (F+R) − LiFT (F) | $+0.024$ | $-0.003$ |
| LiFT − LiFT (F) | $+\mathbf{0.055}^{*}$ | $-0.023$ |
| LiFT, shuffled − LiFT | $-0.008$ | $+0.009$ |

Table 4: Annual return (%) by test window. Bold: best within each universe.

| | W0 | W1 | W2 | W3 |
|---|---|---|---|---|
| *DJIA* | | | | |
| Baseline | 13.3 | −16.7 | 13.3 | 5.2 |
| LiFT (F) | 15.1 | −17.7 | 12.4 | **7.4** |
| LiFT | **15.4** | **−16.3** | **13.5** | 7.2 |
| *S&P 100* | | | | |
| Baseline | 19.1 | **−14.6** | 14.0 | 5.7 |
| LiFT (F) | **20.4** | −17.2 | **14.3** | **8.8** |
| LiFT | 20.0 | −19.2 | 14.1 | 8.7 |

with or without the inputs: LiFT (R) beats the baseline by 0.039 with both intervals clear, and the rules with per-stock modulation add a further 0.055 to LiFT (F) through higher returns and lower turnover. That last effect holds across runs but not across encoders, resting on TimeXer alone while Crossformer moves the other way (Table 5). Across eight measures and five encoders, LiFT improves on LiFT (F) in 10 of the 40 encoder–measure pairs on the DJIA and is worse in none. On the S&P 100 the rules and the modulation add nothing, changing the Sharpe ratio by less than 0.001 in 65% of paired runs. Shuffling the indicators in time leaves LiFT unchanged on both universes, so the gain does not depend on detecting the market regime. It comes from what the rules and the modulation do on almost every day, limiting single-stock, sector, and idle-cash weight.

### 6.3 Gain differs across encoders

The gain of LiFT (F) over the baseline differs across encoders on the S&P 100 (Cochran's $Q = 19.5$, $p = 0.001$), and it follows how much of the market each encoder already sees. It is largest for PatchTST ($+0.214$) and TimeXer ($+0.154$), which encode each stock's series separately; smaller for iTransformer ($+0.073$) and SOFTS ($+0.040$), which mix across stocks; and absent for Crossformer ($+0.005$), which attends over both time and stocks. An encoder without a cross-stock pathway sees the market only through the inputs (Appendix A). On the DJIA, PatchTST again gains most ($+0.127$), but the differences are not significant ($p = 0.39$). The effects of the rules and of conditioning do not differ significantly across encoders on either universe (Appendix B).

### 6.4 Market regimes

We compare windows by annual return, which, unlike the Sharpe ratio, ranks losses correctly when returns are negative. On the DJIA, LiFT earns the highest return in three of four windows (Table 4). It loses least in the 2021–22 bear market, 16.3% against 17.7% for LiFT (F), with a lower drawdown and 10% less trading, and in the 2023–24 bull market it earns slightly less than LiFT (F) with a lower drawdown. On the S&P 100, LiFT (F)'s gains come from the 2023–24 bull market, where it cuts the baseline's 46% cash to 24% and raises returns from 5.7% to 8.8%. The same lower cash costs money in the 2021–22 bear market, $-17.2\%$ against $-14.6\%$, and LiFT does not improve on LiFT (F) in either window.

### 6.5 Transaction costs and equal weighting

Rebalanced daily, as trained, no learned configuration beats equal weighting net of costs (Figure 2, left edge). Each allocation is held for a fixed period and allowed to drift, with costs charged only on rebalancing days, and every allocator uses the same accounting (Appendix D). On the DJIA, LiFT beats equal weighting over a two-week holding period and LiFT (F) over a monthly one, reaching 0.690 and 0.677 against 0.618; on the S&P 100 the gap shrinks from $-0.42$ to $-0.07$. LiFT's advantage over LiFT (F) shrinks with it, from $+0.120$ daily to $+0.013$ monthly, so its gain comes from lower turnover and matters most where trading is frequent. Rebalanced daily, equal weighting and inverse-volatility risk parity are the strongest classical allocators, and LiFT beats minimum variance, mean–variance and cross-sectional momentum on both universes (Table 7, Appendix J). That ordering does not survive a longer holding period: on the DJIA LiFT is the strongest allocator

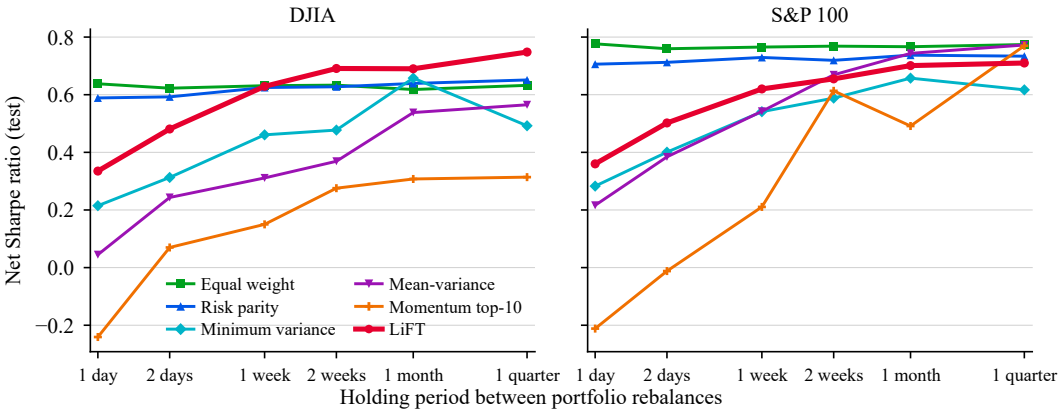


Figure 2: Net Sharpe ratio by holding period between portfolio rebalances, mean over the four test windows. Each allocation is held for the full period and drifts with realized returns, so turnover is charged only on rebalancing days, at 20 basis points. The left edge is the daily rebalancing the models were trained for.

of any kind from two weeks onward, while on the S&P 100 it is overtaken by mean–variance and momentum by one quarter.

## 7 Discussion

Market-state inputs improve both universes, making LiFT (F) the reference for assessing conditioning and rules. Shuffling indicators produces no detectable performance change, so the control does not establish a benefit from correct regime timing. On the S&P 100, rules change Sharpe by less than 0.001 in 65% of paired runs, versus 19% on the DJIA. Early stopping selects epoch 3 or earlier in about two-thirds of runs, limiting the steps available for rule penalties to act (Appendix H). DJIA turnover is highest in 2021–22: 0.21 for LiFT (F), versus 0.10–0.11 elsewhere; LiFT reduces it by 10%. Apparent associations between larger gains and weaker references disappear on held-out seeds.

**Limitations.** Gains beyond LiFT (F) are established only on the DJIA and rest on one encoder. On the S&P 100, LiFT's advantage over some classical allocators does not persist at quarterly rebalancing. Current index membership introduces survivorship bias. TimeXer's baselines receive regime indicators. Comparing configurations with shared modulation and no rules against LiFT changes both rules and modulation. A five-seed subset reverses the input effect's sign, motivating use of all 25 seeds per contrast (Appendix E).

## 8 Conclusion

We presented LiFT, a neurosymbolic framework that adds market knowledge to any time-series encoder, as input features, as stock-specific modulation, and as differentiable rules with weights learned by dual ascent. Across five encoders and two stock universes, in matched runs, LiFT raises the Sharpe ratio of a price-only baseline by 25–50%, with Sortino and Calmar improving alongside. The rules and the per-stock modulation add to the DJIA through higher returns and less trading, do not depend on the timing of the regime signals, and conditioning alone never helps. Longer training, per-rule diagnostics, learned market state, and pre-trained encoders are natural next steps.

### AI use statement

In this work, we used generative AI tools to draft and edit the manuscript text, to write analysis scripts, and to run experiments that aggregate run reports into the tables reported here. We have not used generative AI tools for generating research ideas or producing results. All AI-drafted text

was revised by the authors, and every number in the paper was recomputed from the run reports by Chetan Choudhary with scripts that Chetan Choudhary reviewed. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

### Ethics statement

This work uses public daily price data for large-capitalization US equities and involves no human subjects or personal data. The results are a controlled comparison between allocator configurations and are not investment advice. None of the learned allocators beats equal weighting net of costs under daily rebalancing. Both universes use current index membership, which introduces survivorship bias that affects every arm equally, as stated in Setup.

### Reproducibility statement

Section 5 provides the data sources, window calendar, encoder width and depth, optimizer settings, and loss coefficients. Appendix I defines the features and regime indicators, while Appendix I provides the ticker lists and data-processing details. Every reported effect is a paired difference over identified (seed, window, encoder) cells, and the bootstrap procedure is described in Section 5. The training harness, run submission scripts, per-run reports, and analysis scripts that produce every table will be released as supplementary material at `https://anonymous.4open.science/r/LiFT-release-1431/`

## References


Akshay Agrawal, Brandon Amos, Shane Barratt, Stephen Boyd, Steven Diamond, and J. Zico Kolter. Differentiable convex optimization layers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Brandon Amos and J. Zico Kolter. OptNet: Differentiable optimization as a layer in neural networks. In *International Conference on Machine Learning (ICML)*, 2017.

Andrew Ang and Geert Bekaert. Regime switches in interest rates. *Journal of Business and Economic Statistics*, 20(2):163–182, 2002.

Samy Badreddine, Artur d'Avila Garcez, Luciano Serafini, and Michael Spranger. Logic tensor networks. *Artificial Intelligence*, 303:103649, 2022.

David H. Bailey and Marcos Lopez de Prado. The deflated sharpe ratio: Correcting for selection bias, backtest overfitting, and non-normality. *Journal of Portfolio Management*, 40(5):94–107, 2014.

Geert Bekaert, Michael Ehrmann, Marcel Fratzscher, and Arnaud Mehl. The global crisis and equity market contagion. *Journal of Finance*, 69(6):2597–2649, 2014.

Fischer Black and Robert Litterman. Global portfolio optimization. *Financial Analysts Journal*, 48 (5):28–43, 1992.

Lin William Cong, Ke Tang, and Jingyuan Wang. AlphaPortfolio: Goal-oriented investment management through deep reinforcement learning. Working Paper 35195, National Bureau of Economic Research, May 2026.

Andrew Cotter, Heinrich Jiang, Maya Gupta, Serena Wang, Taman Narayan, Seungil You, and Karthik Sridharan. Optimization with non-differentiable constraints with applications to fairness, recall, churn, and other goals. *Journal of Machine Learning Research*, 20(172):1–59, 2019.

Victor DeMiguel, Lorenzo Garlappi, and Raman Uppal. Optimal versus naive diversification: How inefficient is the 1/n portfolio strategy? *Review of Financial Studies*, 22(5):1915–1953, 2009.

Tong Feng, Chaoyue Chen, Yeshen Zhao, Hao Pan, Pengjun Jiang, Ning Ding, and Minlie Huang. MASTER: Market-guided stock transformer for stock price forecasting. In *AAAI Conference on Artificial Intelligence*, 2024.

Ferdinando Fioretto, Pascal Van Hentenryck, Terrence W. K. Mak, Cuong Tran, Federico Baldo, and Michele Lombardi. Lagrangian duality for constrained deep learning. In *Machine Learning and Knowledge Discovery in Databases: Applied Data Science and Demo Track (ECML PKDD)*, volume 12461 of *Lecture Notes in Computer Science*, pp. 118–135. Springer, 2020. doi: 10.1007/978-3-030-67670-4_8.

Ibanga Kpereobong Friday, Sarada Prasanna Pati, Debahuti Mishra, and Sashikala Mishra. A meta-adaptive framework combining gated attention mechanisms and feature-wise modulation for multi-horizon stock price movement prediction. *IEEE Access*, 14:26581–26603, 2026. doi: 10.1109/ACCESS.2026.3663386.

Artur d'Avila Garcez and Luis C. Lamb. Neurosymbolic AI: The 3rd wave. *Artificial Intelligence Review*, 56(11):12387–12406, 2023.

Lu Han, Han-Jia Ye, and De-Chuan Zhan. SOFTS: Efficient multivariate time series forecasting with series-core fusion. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard Hovy, and Eric P. Xing. Harnessing deep neural networks with logic rules. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pp. 2410–2420, 2016.

Zhengyao Jiang, Dixing Xu, and Jinjun Liang. A deep reinforcement learning framework for the financial portfolio management problem, 2017.

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

Xiao-Yang Liu, Hongyang Yang, Qian Chen, Runjia Zhang, Liuqing Yang, Bowen Xiao, and Christina Dan Wang. FinRL: A deep reinforcement learning library for automated stock trading in quantitative finance. In *NeurIPS Workshop on Deep Reinforcement Learning*, 2020.

Yong Liu, Tengge Hu, Haoran Zhang, Ling Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. iTransformer: Inverted transformers are effective for time series forecasting. In *International Conference on Learning Representations (ICLR)*, 2024.

Marcos Lopez de Prado. Building diversified portfolios that outperform out of sample. *Journal of Portfolio Management*, 42(4):59–69, 2016.

Sébastien Maillard, Thierry Roncalli, and Jérôme Teïletche. The properties of equally weighted risk contribution portfolios. *The Journal of Portfolio Management*, 36(4):60–70, 2010. doi: 10.3905/jpm.2010.36.4.060.

Jayanta Mandi, James Kotary, Senne Berden, Maxime Mulamba, Victor Bucarey, Tias Guns, and Ferdinando Fioretto. Decision-focused learning: Foundations, state of the art, benchmark and future opportunities. *Journal of Artificial Intelligence Research*, 80:1623–1701, 2024.

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

André F. T. Martins and Ramón Fernandez Astudillo. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *International Conference on Machine Learning (ICML)*, pp. 1614–1623, 2016.

Elias Mashayamombe, Charis Harley, and Thulane Paepae. A methodological framework for intrinsic explainability in portfolio allocation: Constraint-aware portfolio reasoning network. *International Journal of Financial Studies*, 14(9):229, 2026. doi: 10.3390/ijfs14090229.

Alan Moreira and Tyler Muir. Volatility-managed portfolios. *The Journal of Finance*, 72(4):1611–1644, 2017.

Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *International Conference on Learning Representations (ICLR)*, 2023.

Peter Nystrup, Stephen Boyd, Erik Lindström, and Henrik Madsen. Multi-period portfolio selection with drawdown control. *Annals of Operations Research*, 282:245–271, 2020.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual reasoning with a general conditioning layer. In *AAAI Conference on Artificial Intelligence*, 2018.

Zezhi Shao, Zhao Zhang, Fei Wei, Fei Wang, Yongjun Xu, and Xin Cao. Spatial-temporal identity: A simple yet effective baseline for multivariate time series forecasting. In *ACM International Conference on Information and Knowledge Management (CIKM)*, 2022.

William F. Sharpe. The sharpe ratio. *The Journal of Portfolio Management*, 21(1):49–58, 1994.

Frank A. Sortino and Lee N. Price. Performance measurement in a downside risk framework. *The Journal of Investing*, 3(3):59–64, 1994.

Yuxuan Wang, Haixu Wu, Jiaxiang Dong, Yong Liu, Mingsheng Long, and Jianmin Wang. TimeXer: Empowering transformers for time series forecasting with exogenous variables. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Kieran Wood, Stephen J. Roberts, and Stefan Zohren. DeePM: Regime-robust deep learning for systematic macro portfolio management, 2026.

Terry W. Young. Calmar ratio: A smoother tool. *Futures*, 20(1):40, 1991.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *AAAI Conference on Artificial Intelligence*, 2023.

Yiyao Zhang, Diksha Goel, Hussain Ahmad, and Claudia Szabo. RegimeFolio: A regime aware ML system for sectoral portfolio optimization in dynamic markets. *IEEE Access*, 13:184722–184744, 2025. doi: 10.1109/ACCESS.2025.3624822.

Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *International Conference on Learning Representations (ICLR)*, 2023.

Zihao Zhang, Stefan Zohren, and Stephen Roberts. Deep learning for portfolio optimization. *The Journal of Financial Data Science*, 2(4):8–20, 2020.

## A Encoders and injection points

Every encoder maps a window of $T = 60$ days and $F = 16$ features per stock to one vector per stock, from which a per-stock logit head and a pooled cash head produce the $N+1$ portfolio weights. The encoders differ in how that per-stock vector is built and in whether anything is shared across stocks, which is where the conditioning is applied and why its effect varies by encoder.

**SOFTS.** Each stock's $(T, F)$ history is flattened and projected by a shared linear layer to one embedding per stock. A STAR block then averages those embeddings into a single global core, concatenates the core back to each stock and projects the pair, so cross-stock information passes through one market-level summary rather than through attention. FiLM is applied to the per-stock embeddings before the first STAR block, so the modulation is visible to every later mixing step. Cost is linear in $N$.

**iTransformer.** Each stock's full $(T, F)$ history is flattened to a single vector and projected to one token, so the sequence fed to the encoder has $N$ tokens, one per stock, and self-attention runs across stocks rather than across time. FiLM is applied to those token embeddings before the encoder stack, so the modulation enters ahead of all cross-asset attention.

**PatchTST.** Each stock's series is cut into overlapping patches of length 8 at stride 4, each patch is projected and given a positional embedding, and self-attention runs over patches within a stock. Stocks are processed independently and never exchange information. The patch tokens are mean-pooled into one embedding per stock, and FiLM is applied to that pooled embedding, so the modulation enters after the encoder rather than before it.

**TimeXer.** The endogenous path patches each stock's series as in PatchTST and adds one learnable global token per stock, with self-attention over the patch tokens and that global token. The six regime indicators form the exogenous path, each becoming a variate token, and the global token cross-attends to them. The indicators therefore reach the representation through the architecture itself, and FiLM, when enabled, adds a second and separate path on top. This has a consequence for the contrasts: TimeXer reads the indicators through cross-attention whenever they are supplied, independent of whether FiLM is switched on, so its conditioning-off arms are not blind to the market state in the way the other encoders' are.

**Crossformer.** Each stock's series is divided into segments of length 6, and the first stage runs self-attention over segments within a stock. The second stage mixes across stocks through ten learned router tokens, which first aggregate from all stocks and then redistribute, giving cross-stock attention at cost linear in $N$. FiLM is applied to the per-stock segment embeddings between the two stages, so it follows cross-time attention and precedes cross-stock mixing. Crossformer is the only encoder with an explicit pathway over both time and stocks.

**Capacity and conditioning size.** All encoders use $d = 128$ with three layers and batch size 64. Capacity is held identical across every arm within an encoder, which is what makes each contrast matched; it may differ between encoders, and no encoder is ever compared against another. Stock-specific FiLM concatenates the six indicators with a learned 16 dimensional stock embedding before producing the scale and shift, which is why the conditioning pathway grows slightly with $N$. The conditioning pathway is 36K parameters on the DJIA and 38K on the S&P 100, under 6% of any model.

The three groupings this produces are what Section 6.3 reports against. PatchTST and TimeXer build each stock's representation in isolation, so the market reaches them only through the inputs or, for TimeXer, the exogenous channel. SOFTS and iTransformer mix across stocks, through a global core and through attention respectively. Crossformer mixes over both time and stocks, and gains least from the market-state inputs.

## B PER-ENCODER RESULTS, ALL MEASURES

Table 5 gives the two contrasts that carry the paper's claims, per encoder and with run intervals. Figure 3 extends those two contrasts to eight measures. Table 6 gives the validation-to-test gap per encoder and arm.

The two contrasts behave differently once the encoders are separated. LiFT improves on the baseline by an interval clear of zero in three of five encoders on the DJIA and four of five on the S&P 100. Crossformer is the only encoder whose interval includes zero on both universes, and iTransformer's DJIA interval also includes it. The gain over LiFT (F) is clear of zero for one encoder only, TimeXer on the DJIA, and for none on the S&P 100. The aggregate $+0.055$ in Table 3 therefore rests on a single architecture, and Crossformer moves the other way at $-0.018$. We report the aggregate because the runs support it, and this table because the encoders do not.

Cumulative return, downside deviation and the share of high-cash days are omitted from Figure 3 as close transforms of annualized return, the Sortino denominator and average cash respectively; they are recorded in the per-run outputs.

## C PER-WINDOW RESULTS PER ENCODER

[The window-by-encoder tables for both universes and every mode, the Cochran's $Q$ tests, the Benjamini–Hochberg-adjusted cell tests, and the split-half check of Section 7.]

## D HOLDING PERIOD AND TRANSACTION COSTS

Figure 2 reports six holding periods for every allocator. Each allocation is held for the full period and drifts with realized returns, so turnover is charged only on rebalancing days. The main results

Table 5: Change in out-of-sample Sharpe ratio per encoder, as paired differences over matched runs. Runs are paired on window and seed, 100 per cell. Brackets give a 95% bootstrap interval over those runs. Bold: the interval excludes zero.

| | DJIA | | S&P 100 | |
|---|---|---|---|---|
| Encoder | LiFT − baseline | LiFT − LiFT (F) | LiFT − baseline | LiFT − LiFT (F) |
| SOFTS | **+0.083** [+0.044, +0.123] | +0.029 [−0.026, +0.087] | **+0.057** [+0.033, +0.080] | +0.017 [−0.020, +0.055] |
| iTransformer | +0.057 [−0.006, +0.119] | +0.045 [−0.013, +0.107] | **+0.070** [+0.010, +0.125] | −0.002 [−0.065, +0.057] |
| PatchTST | **+0.168** [+0.074, +0.272] | +0.041 [−0.018, +0.101] | **+0.139** [+0.037, +0.246] | −0.074 [−0.155, +0.008] |
| TimeXer | **+0.190** [+0.092, +0.284] | **+0.178** [+0.079, +0.277] | **+0.105** [+0.011, +0.201] | −0.049 [−0.130, +0.029] |
| Crossformer | +0.040 [−0.057, +0.133] | −0.018 [−0.115, +0.077] | −0.003 [−0.093, +0.082] | −0.008 [−0.102, +0.081] |

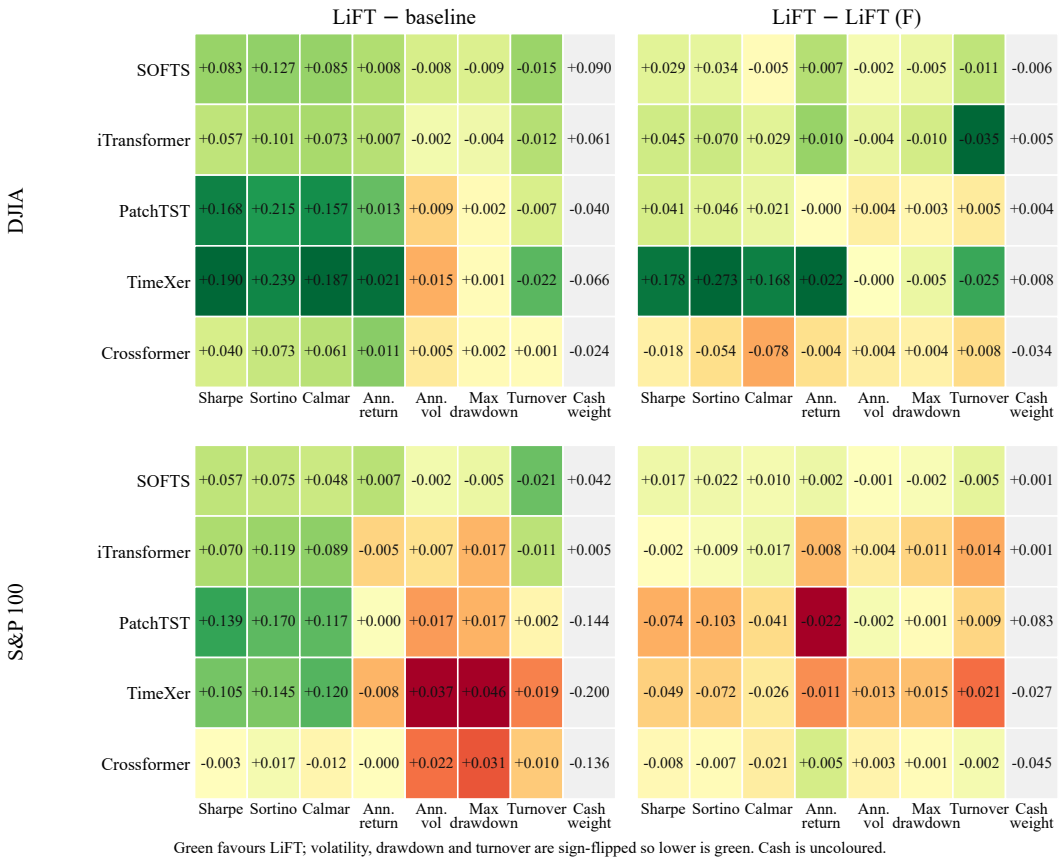


Figure 3: Change in each measure from LiFT, by encoder, in the measure's own units. Left: against the baseline. Right: against LiFT (F). Green marks a favorable change, so volatility, drawdown and turnover are sign-flipped before coloring while the printed number keeps its sign. Color is scaled within each column, since a 0.05 move in Sharpe and in turnover are not comparable. Average cash is uncolored, being an exposure rather than a performance measure. Of the 140 colored cells, 56 have a run interval clear of zero, and the two panels against LiFT (F) hold only 11 of those.

table charges the full weight change every day, so the one-day column here is not identical to it; the comparison across holding periods is internally consistent, which is what the question needs.

Low-turnover allocators are nearly flat in the holding period: equal weight spans 0.02 on the DJIA at a daily turnover of 0.006. Every high-turnover method improves sharply, because at 20 basis points a daily rebalance is a tax proportional to turnover. Momentum on the DJIA turns over 0.288 per

Table 6: Share of validation Sharpe lost out-of-sample, 100 runs per cell. The validation score is the early-stopping maximum and is optimistically biased; the gap is the drop from the score at which each model was selected.

| | DJIA | | | S&P 100 | | |
|---|---|---|---|---|---|---|
| Encoder | Baseline | LiFT (F) | LiFT | Baseline | LiFT (F) | LiFT |
| SOFTS | 21% | 27% | 21% | −11% | −3% | −4% |
| iTransformer | 107% | 100% | 82% | −4% | 17% | 33% |
| PatchTST | 49% | 33% | 31% | 58% | 9% | 31% |
| TimeXer | 68% | 68% | 34% | 64% | 33% | 37% |
| Crossformer | 42% | 31% | 35% | 22% | 21% | 20% |
| Mean gap (Sharpe) | 0.233 | 0.241 | 0.180 | 0.157 | 0.071 | 0.106 |

day and pays 14.5% a year in costs, which is the whole of its $-0.241$. The identity of the strongest classical allocator therefore changes with the holding period: equal weight at one day, minimum variance at one month on the DJIA. Minimum variance is the one non-monotone curve, peaking at one month and falling at one quarter, consistent with its 60-day covariance estimate going stale when held that long.

Table 7: Net Sharpe ratio by holding period between portfolio rebalances, mean over the four test windows. Each allocation is held for the period and drifts with realized returns, so turnover is charged only on rebalancing days, at 20 basis points. Bold: best in column. The one-day column is the frequency the learned models were trained for.

| Allocator | 1 day | 2 days | 1 week | 2 weeks | 1 month | 1 quarter |
|---|---|---|---|---|---|---|
| *DJIA* | | | | | | |
| LiFT | 0.335 | 0.481 | 0.629 | **0.691** | **0.690** | **0.748** |
| Equal weight | **0.638** | **0.623** | **0.631** | 0.633 | 0.618 | 0.632 |
| Risk parity | 0.589 | 0.593 | 0.625 | 0.627 | 0.639 | 0.651 |
| Minimum variance | 0.215 | 0.313 | 0.461 | 0.477 | 0.658 | 0.492 |
| Mean–variance | 0.045 | 0.243 | 0.311 | 0.369 | 0.538 | 0.565 |
| Momentum top-10 | −0.241 | 0.070 | 0.150 | 0.276 | 0.308 | 0.314 |
| *S&P 100* | | | | | | |
| LiFT | 0.360 | 0.502 | 0.620 | 0.655 | 0.701 | 0.710 |
| Equal weight | **0.777** | **0.760** | **0.765** | **0.768** | **0.767** | **0.774** |
| Risk parity | 0.706 | 0.712 | 0.729 | 0.719 | 0.737 | 0.734 |
| Minimum variance | 0.283 | 0.401 | 0.541 | 0.588 | 0.657 | 0.617 |
| Mean–variance | 0.216 | 0.384 | 0.542 | 0.669 | 0.743 | 0.773 |
| Momentum top-10 | −0.212 | −0.012 | 0.211 | 0.613 | 0.491 | 0.770 |

## E AGGREGATION AND RUN-TO-RUN VARIANCE

Every cell averages 500 runs: five encoders, four test windows and 25 seeds. The four out-of-sample windows span 2019-07-01 to 2024-12-31 and contain 505, 252, 251 and 378 trading days, 1386 in total. Reported figures are unweighted means over the four windows, so each regime counts equally; this suits a claim about behavior across regimes, and it is not the same as the Sharpe ratio of the concatenated series. Weighting windows by day count raises every allocator by 0.09 to 0.14, because it demotes the 2021–22 window in which every allocator loses. Pooling the 1386 days instead gives lower values than the unweighted mean for the strongest allocators and higher ones for the weakest: equal weight on the DJIA is 0.638 unweighted, 0.739 day-weighted and 0.609 pooled. Conclusions are unchanged under all three; levels are not.

Run-to-run variance at fixed configuration is large enough to reverse a contrast. Daily return series were retained for seeds 0–4 only. On those five seeds the market-state input effect on the DJIA is $-0.080$, against $+0.053$ over all 25 seeds, a sign reversal from the seed subset alone. Any analysis

restricted to a five-seed subset is therefore unreliable at this effect size, and all reported contrasts use the full 25 seeds.

## F RULE-WEIGHT CALIBRATION BY DUAL ASCENT

Converged multipliers at dual learning rate 0.01 on the Dow 30, with the initial value in brackets: VIX safety 1.07–1.17 (1.00); concentration 4.92–5.38 (5.00); momentum 1.65–2.62 (1.00); sector 0.80–1.04 (1.00); cash efficiency 1.92–3.16 (1.00); equity floor 0.41–1.11 (0.00). At dual learning rates of 0.1 and 0.5, the cash-efficiency and equity-floor multipliers saturate the cap of 10; the calibration holds at the default rate only. This arm was trained in a separate batch on seeds 0–4; only its within-arm comparison to the matching LiFT runs is reported. [Out-of-sample contrast against hand-set weights per encoder and measure.]

**Learned rule weights.** Table 8 gives the rule weights learned by dual ascent for each encoder on the DJIA, as the mean and standard deviation over its runs. The learned weights depend on the dual step size: all reported results use $\eta = 0.01$.

Table 8: Rule weights learned by dual ascent on the DJIA, mean (standard deviation) over 20 runs for each encoder.

| Rule | Initial | SOFTS | iTransformer | PatchTST | TimeXer | Crossformer |
|---|---|---|---|---|---|---|
| 1 | 1 | 1.12 (0.08) | 1.11 (0.09) | 1.07 (0.06) | 1.07 (0.06) | 1.17 (0.15) |
| 2 | 1 | 1.05 (0.06) | 1.03 (0.07) | 1.00 (0.05) | 1.00 (0.06) | 1.02 (0.10) |
| 3 | 5 | 4.92 (0.04) | 5.38 (0.25) | 5.12 (0.19) | 5.03 (0.08) | 5.13 (0.17) |
| 4 | 1 | 1.65 (0.08) | 1.91 (0.22) | 1.86 (0.28) | 1.75 (0.15) | 2.62 (0.36) |
| 5 | 1 | 0.86 (0.03) | 1.04 (0.10) | 0.87 (0.04) | 0.85 (0.03) | 0.80 (0.05) |
| 6 | 1 | 2.37 (0.27) | 1.92 (0.27) | 2.42 (0.42) | 2.57 (0.43) | 3.16 (0.46) |
| 7 | 0 | 0.73 (0.15) | 0.41 (0.13) | 0.70 (0.19) | 0.79 (0.22) | 1.11 (0.22) |

**Two relaxations.** Implications (rules 1, 2, 4 and 6) use the Łukasiewicz form $\mathrm{clamp}(1 - A + B, 0, 1)$, where $A$ is the truth value of the condition and $B$ of the consequence; it degrades smoothly as $B$ is violated and equals one when $A$ is false. Caps (rules 3 and 5) apply to every stock or sector, where a t-norm conjunction behaves poorly: a product over 30 terms drives both truth value and gradient toward zero, and a minimum passes gradient to one position per step. We instead use one minus the summed hinge excess over the cap, which gives every violating position a gradient in the same step. Because the weights sum to one, the excess is at most $1 - \theta$ for a cap $\theta$, so $\phi_3 \geq 0.15$ and $\phi_5 \geq 0.40$ stay away from the singularity of $-\log$. One consequence of the implication form: a rule adds no loss and no gradient when its condition is off, so rules 1 and 2 are inactive whenever VIX is below 25, as on 99% of days in the 2023–24 test window. Appendix G evaluates graded indicators that keep these rules partly active.

## G GRADED INDICATORS PILOT

DJIA, 100 runs per arm, equity floor on, trained locally in a separate batch from the main campaign, so these arms compare only with each other. Mean out-of-sample Sharpe: LiFT (F) 0.231; LiFT with binary indicators 0.306; with graded indicators 0.316; and graded with a minimum conjunction 0.340. Graded against binary: the Sharpe ratio is not distinguishable; volatility is 16.6% against 17.3% (both intervals), and cash is 24.8% against 18.3%. The minimum and product conjunctions are not distinguishable. The logistic widths were fixed in advance: VIX 2.5 index points, trend 1%, five-day VIX change 2.5%, 200-day trend 2%, RSI 2.5 points, breadth 2.5%. With binary indicators, the product, minimum, and Łukasiewicz conjunctions coincide in rule 7, so the choice of operator is meaningful only with graded indicators.

## H EARLY STOPPING

Training stops early in almost every run. Over 8498 runs of the main campaign, with the epoch chosen by maximum validation net Sharpe, the selected epoch is 1 in 42% of runs and 3 or earlier in 69%, with a median of 2 and a maximum of 89 out of a possible 150. The pattern is stronger on the S&P 100, where 55–73% of runs by arm select epoch 1, against 22–29% on the DJIA. It also varies by encoder: SOFTS selects epoch 1 in 60% of runs and epoch 3 or earlier in 92%, while TimeXer selects epoch 1 in 24% and epoch 3 or earlier in 51%. The rule loss therefore acts over very few steps, which bounds how much the rules can move an allocation and is the most likely reason they leave the S&P 100 models unchanged in most paired runs.

## I DATA PROVENANCE

Both universes use current index membership, taken as of the 2024 DJIA composition after Amazon replaced Walgreens, and the S&P 100 list filtered to the 99 names with continuous coverage. Membership is not point-in-time, so every result carries survivorship bias: the universes exclude firms that left the index or failed during the sample. Prices run from 2010-01-01 to 2025-03-01, from which the four test windows are carved.

Each stock is described by sixteen features per day. Twelve are per-stock and technical: one-, five- and twenty-one-day log returns, fourteen-day RSI, price relative to its fifty- and two-hundred-day moving averages, twenty-one-day realized volatility, MACD histogram, Bollinger %b, volume ratio, cross-sectional rank of the twenty-one-day return, and distance from the fifty-two-week high. Four are market state, identical for every stock on a given day and broadcast across the cross-section: VIX normalized as $\mathrm{VIX}/20 - 1$, the twenty-one-day change in VIX, and the index relative to its fifty- and two-hundred-day moving averages. The market series are VIX and SPY. These four are what LiFT (F) adds and what `--zero-macro-features` blanks.

The six regime indicators are binary and threshold the same market series: VIX above 25; the index below its fifty-day average; a five-day VIX change above 15%; the index below its two-hundred-day average; index fourteen-day RSI above 70; and more than 60% of constituents below their fifty-day average.

The model sees a sixty-day window and predicts the allocation held into the next trading day, so the target is the next-day return and no feature in a window uses information from the target day. Features are standardized with means and standard deviations computed from training rows only, recomputed per window, so no test statistic enters the normalization. Transaction costs are charged at 20 basis points per unit of turnover, where turnover is the sum of absolute weight changes, applied identically to every learned and classical allocator.

## J CLASSICAL ALLOCATOR IMPLEMENTATION

Five reference allocators, all long-only with weights summing to one and a cash column held at zero return. Risk parity, mean–variance and minimum variance estimate from a trailing sixty-day window of daily returns. Mean–variance and minimum variance use a Ledoit–Wolf shrinkage covariance and its closed forms, $w \propto \Sigma^{-1}\mu$ and $w \propto \Sigma^{-1}\mathbf{1}$, with negative weights clipped and the result renormalized; a window whose covariance fails to invert falls back to equal weight. Risk parity is inverse volatility, with no correlation term. Momentum equal-weights the ten stocks with the highest trailing twenty-day return, re-ranked daily. Equal weight holds $1/N$ with no cash.

All five rebalance daily, as the learned allocators do, and are charged the same 20 basis points per unit of turnover, so the friction model is identical across every arm in the paper. Window results are combined as unweighted means over the four test windows, whose lengths are 505, 252, 251 and 378 days; see Appendix E.