FiRE: Scaling and Aligning Financial Foundation Models

Anonymous authors
Paper under double-blind review

Abstract

We introduce FiRE, a financial foundation model that combines generative pretraining with reward-guided post-training. Generative pretraining offers supervision directly from historical market trajectories, but scaling this paradigm exposes two limitations: high-capacity token spaces make categorical next-token prediction increasingly expensive, while sequence-modeling objectives are only indirectly aligned with the metrics that matter in downstream quantitative tasks. FiRE addresses the first challenge by encoding each candlestick bar as a 256-bit binary token, defining an implicit 2^{256} code space, and modeling next-token generation with conditional flow matching. This replaces vocabulary-sized categorical prediction with joint generation in the binary token space, enabling representation capacity to scale efficiently. For task-specific alignment, we introduce ReFLO (Rejected-Future Flow Optimization), which uses verifiable financial rewards to select low-scoring futures and constructs repulsive token-level regression targets, without differentiating through rewards or evaluating policy likelihoods. Extensive experiments demonstrate FiRE's superior performance across a range of quantitative finance tasks. FiRE improves return forecasting IC by 44% over the strongest baseline in the zero-shot setting, with the gain increasing to 51% after ReFLO post-training. It also achieves a 33% higher discriminative score than baselines for synthetic candlestick generation and delivers higher risk-adjusted returns in investment simulations.

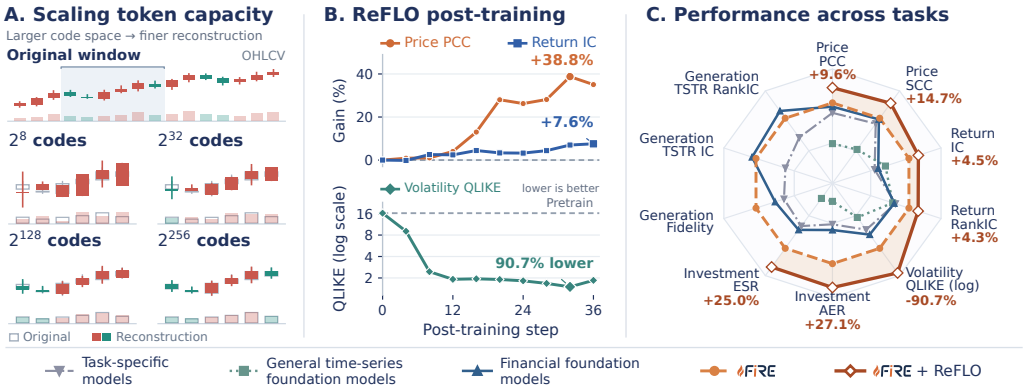


Figure 1: **Overview of FiRE.** (a) Scaling the binary token space improves market reconstruction. (b) ReFLO adapts the pretrained generator to task-specific rewards for price, return, and volatility forecasting. (c) Comprehensive performance of FiRE across several quantitative finance tasks; each baseline family shows its best result per metric, and all metrics are oriented so that higher is better.

1 Introduction

Foundation models learn transferable representations through large-scale pretraining, which are then adapted to downstream tasks (Bommasani et al., 2021). Time-series foundation models, such as Chronos (Ansari et al., 2024), TimesFM (Das et al., 2024) and Sundial (Liu et al., 2025c), bring this approach to temporal data, transferring learned dynamics across datasets and forecasting settings. In

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.

finance, the same history of prices and trading activity can inform different tasks, from forecasting returns to estimating volatility. This motivates financial foundation models that learn from market histories at scale and can subsequently be adapted to the objectives of individual applications.

Financial foundation models have developed along two main directions. One line models fine-grained market microstructure, including order flow and limit-order-book states, to learn short-horizon trading dynamics, such as TradeFM (Kawawa-Beaudan et al., 2026), MarS (Li et al., 2025) and M3 (Zhang et al., 2026). Because microstructure conventions and trading mechanisms vary substantially across exchanges and venues, these models are often developed for specific markets and short-horizon settings. A second line instead pretrains on more standardized market representations, like candlestick bars and other price-volume time series, as in LENS (Xu et al., 2025), FinCast (Zhu et al., 2025b) and Kronos (Shi et al., 2025). These standardized observations are naturally shared across assets, exchanges, and time granularities, making them well suited to broad pretraining and transfer. In this work, we focus on candlestick sequences, whose common OHLCVA structure defines a shared *market language* across heterogeneous financial markets.

A common paradigm in financial foundation modeling is *discretization–generation*: map market observations into tokens and model the resulting sequence autoregressively (Mai, 2024; Li et al., 2025; Shi et al., 2025; Zhang et al., 2026). Unlike training against a single task-specific target, generative pretraining uses the market trajectory itself as supervision, allowing the model to learn dependencies across variables and time (Shi et al., 2025). Learning from sequences before task specialization provides a basis for transfer across downstream objectives (Goswami et al., 2024). Kronos exemplifies this approach for candlesticks and further shows that scaling its vocabulary up to 2^{20} improves both reconstruction quality and downstream task performance (Shi et al., 2025). This empirical trend suggests that further vocabulary scaling is a promising direction for improving financial foundation models. Continuing along this route exposes two bottlenecks. **(1) Vocabulary scaling becomes increasingly costly.** Although the subtoken factorization used in Kronos makes training with a vocabulary at the 2^{20} scale practical, substantially larger vocabularies would require either a larger categorical prediction space or additional sequential prediction steps (Shi et al., 2025; Ai et al., 2026). **(2) Generative pretraining is not inherently aligned with downstream objectives.** It models market sequences rather than directly optimizing the metrics used by downstream tasks, such as correlation between predicted and realized returns, or volatility forecasting error. Consequently, improvements in the generative objective need not translate proportionally into improvements on a particular downstream task. Figure 2 shows that increasing the number of bits per market token continues to reduce reconstruction error, while downstream correlation metrics do not improve monotonically, motivating task-specific alignment alongside representation scaling.

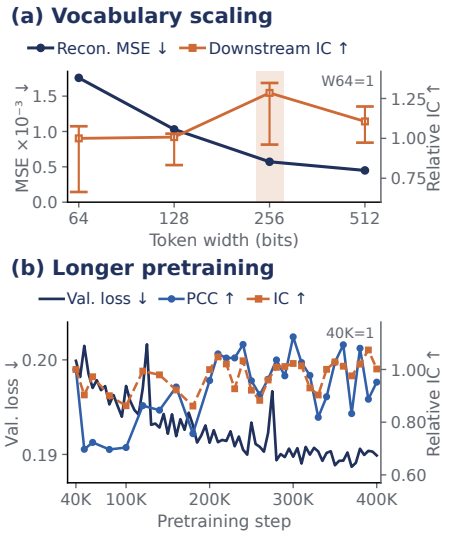


Figure 2: **Generative scaling and downstream performance.** (a) Increasing token width improves reconstruction, while downstream IC peaks at 256 bits. (b) Continued pretraining lowers validation loss without sustained gains in PCC or IC. In (a), downstream IC is normalized to the 64-bit setting and reports the three-seed median and range.

We introduce **FiRE**, a financial foundation model that combines generative pretraining with reward-guided post-training. For pretraining, we use lookup-free quantization (LFQ) (Yu et al., 2024) to encode each candlestick bar as a 256-bit token, defining an implicit 2^{256} code space. Drawing on autoregressive generation with continuous and binary tokens (Li et al., 2024; Fan et al., 2024; Ai et al., 2026), we couple a causal Transformer with a conditional flow-matching head (Lipman et al., 2022). At each market timestep, the Transformer encodes the preceding tokens, and the flow head generates all 256 bits of the next token jointly from noise, conditioned on this history. We then append the generated token to the history before generating the following one. Generation is therefore autoregressive across market timesteps, but requires neither sequential prediction within a token nor a vocabulary-sized output layer.

Recent work on reinforcement learning with verifiable rewards shows that penalizing incorrect generations alone can improve language-model reasoning, motivating adaptation through negative feedback (Zhu et al., 2025a). Building on this perspective, we introduce **ReFLO** (Rejected-Future Flow Optimization) for reward-guided adaptation of autoregressive market generators. ReFLO samples future trajectories from each market history and selects low-scoring continuations using verifiable financial rewards. Their generated tokens are re-noised and re-evaluated under their own prefixes, and detached regression targets move clean-token predictions away from the rejected endpoints. The update requires neither reward gradients nor policy likelihoods.

Our contributions are threefold:

- We introduce FiRE, a financial foundation model that combines 256-bit candlestick tokenization with autoregressive flow generation. By generating all bits of a token jointly, FiRE supports an implicit 2^{256} code space without a vocabulary-sized classifier or sequential subtoken prediction.
- We propose ReFLO, a reward-guided post-training method for autoregressive flow generators with verifiable financial rewards. It uses negative group-relative advantages to construct repulsive token-level update targets, enabling objective-specific adaptation without reward gradients or policy likelihoods.
- We conduct comprehensive evaluations across a broad range of quantitative finance tasks. FiRE outperforms both pretrained and task-specific baselines in financial forecasting and produces higher-fidelity synthetic candlestick sequences. ReFLO further improves predictive performance and risk-adjusted returns in investment simulations.

2 PRELIMINARIES

Autoregressive market generation. Let $\mathbf{x}_t = (O_t, H_t, L_t, C_t, V_t, A_t) \in \mathbb{R}^6$ represent the normalized OHLCVA attributes (open, high, low, and close prices, trading volume, and traded amount) at market time t. Given a history $\mathbf{x}_{1:T}$, our goal is to model the following H observations $\mathbf{x}_{T+1:T+H}$. A tokenizer $\mathcal{T}_\phi$ encodes the history as $\mathbf{b}_{1:T} = \mathcal{T}_\phi(\mathbf{x}_{1:T})$, where $\mathbf{b}_t \in \{0,1\}^B$ contains B bits. The tokenizer is causal: $\mathbf{b}_t$ depends only on observations up to time t. As in autoregressive token-based market models (Mai, 2024; Shi et al., 2025), we factorize the distribution of future tokens over time:

$$p_\theta(\mathbf{b}_{T+1:T+H} \mid \mathbf{b}_{1:T}) = \prod_{j=1}^{H} p_\theta(\mathbf{b}_{T+j} \mid \mathbf{b}_{1:T+j-1}). \tag{1}$$

Decoding the generated tokens yields future OHLCVA trajectories. The decoded OHLC values provide price-path forecasts; the relative change from the last observed close to the predicted terminal close gives a return forecast; applying a realized-volatility estimator to the decoded price path gives a volatility forecast. Appendix A.2 defines these readouts and their evaluation metrics. Existing token-based models use categorical prediction for the next token (Mai, 2024; Shi et al., 2025). FiRE instead models $p_\theta(\mathbf{b}_{t+1} \mid \mathbf{b}_{1:t})$ with a history-conditioned flow head.

Reward-guided adaptation. Reward-guided post-training adapts a pretrained generator to maximize expected task reward (Liu et al., 2025b; Zheng et al., 2026). In *reinforcement learning with verifiable rewards* (RLVR), a task-specific evaluation rule supplies this reward without requiring human preference annotations or a learned reward model. For autoregressive market generation, the next-token distribution $p_\theta(\mathbf{b}_{t+1} \mid \mathbf{b}_{1:t})$ serves as the policy, and a sampled future token sequence is a *rollout*. Predictions derived from decoded rollouts can be evaluated against realized market outcomes to obtain scalar rewards.

3 SCALING THE MARKET VOCABULARY

Kronos shows that expanding the vocabulary to 2^{20} improves reconstruction and forecasting (Shi et al., 2025). However, categorical prediction becomes difficult to scale further: a flat vocabulary of 2^B tokens requires $\mathcal{O}(2^B d)$ parameters for both the embedding table and output classifier. Subtoken factorization avoids this exponential growth by decomposing each token into smaller categorical

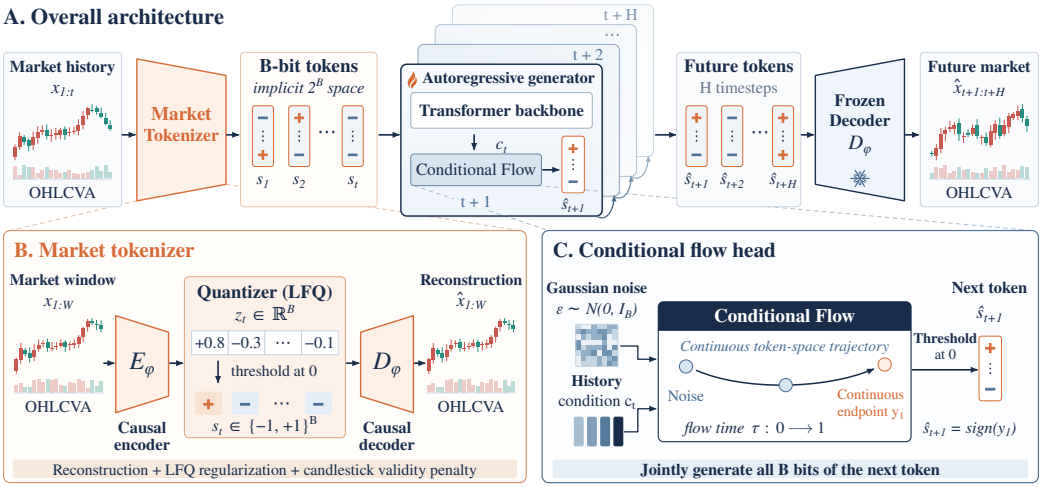


Figure 3: **FiRE generative architecture.** (a) A causal market tokenizer maps OHLCVA histories to B-bit tokens; an autoregressive Transformer–flow generator produces future tokens, which the frozen decoder maps back to market trajectories. (b) The tokenizer learns binary representations through lookup-free quantization and reconstruction. (c) Conditioned on the Transformer state, the flow head jointly generates all B bits of the next token from Gaussian noise.

predictions, but at the cost of additional within-token autoregressive steps. For example, representing a 2^{100} code space with 2^{10}-way subtokens requires ten sequential predictions per market timestep, increasing computation and exposing later predictions to errors from earlier ones.

To avoid vocabulary-sized prediction heads and additional within-token autoregressive steps, we adopt continuous-space next-token generation (Li et al., 2024) with conditional flow matching (Lipman et al., 2022). We retain the temporal autoregressive factorization in Eq. (1), while replacing categorical prediction with a history-conditioned flow head that generates all B bits jointly, following the binary-token generation approach of BitDance (Ai et al., 2026). With $B = 256$, this defines an implicit 2^{256} code space without enumerating its tokens. For fixed hidden widths, the parameter counts of the bit embedding and flow output projection grow linearly with B. The tokenizer is learned separately through reconstruction and held fixed during autoregressive pretraining. Figure 3 summarizes the generative architecture.

3.1 Market Tokenizer

Binary tokenization and reconstruction. The tokenizer consists of causal Transformer encoder and decoder networks connected by lookup-free quantization (LFQ) (Yu et al., 2024). Given an OHLCVA sequence, the encoder produces continuous latent vectors whose signs define binary tokens, while the decoder reconstructs the original observations from their quantized representations:

$$\begin{aligned} \mathbf{z}_{1:W} &= E_\phi(\mathbf{x}_{1:W}), & \mathbf{b}_{1:W} &= \mathbf{1}[\mathbf{z}_{1:W} \geq 0] \in \{0,1\}^{W\times B}, \\ \widehat{\mathbf{x}}_{1:W} &= D_\phi(\mathbf{q}_{1:W}), & \mathbf{q}_{1:W} &= (2\mathbf{b}_{1:W} - 1)/\sqrt{B}. \end{aligned} \tag{2}$$

With $B = 256$, the binary representation defines an implicit 2^{256} code space without an explicit codebook. We use a straight-through estimator to propagate reconstruction gradients through the binary quantization step (Appendix C.2).

Tokenizer objective. We train the tokenizer to preserve both numerical fidelity and market-relevant structure. The reconstruction term measures the mean squared error over the six OHLCVA fields:

$$\mathcal{L}_{\text{rec}} = \mathbb{E}_{\mathbf{x}_{1:W}}\left[\frac{1}{6W}\sum_{t=1}^{W}\|\widehat{\mathbf{x}}_t - \mathbf{x}_t\|_2^2\right], \tag{3}$$

$$\mathcal{L}_{\text{tok}} = \tfrac{1}{2}\big(\mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{LFQ}} + \mathcal{L}_{\text{legal}} + \lambda\mathcal{L}_{\text{dir}}\big). \tag{4}$$

In addition to LFQ regularization for code utilization, $\mathcal{L}_{\text{legal}}$ encourages valid OHLC geometry, while $\mathcal{L}_{\text{dir}}$ preserves directional information in candlestick bodies and adjacent close movements. Appendix C.3 provides the full definitions.

3.2 Autoregressive Flow Modeling

History conditioning. An L-layer causal Transformer encodes the token history using linearly projected bipolar tokens and calendar-time embeddings. The final-layer state at position t provides the history condition $\mathbf{c}_t = \mathbf{h}_t^{(L)}$, where $\mathbf{h}_t^{(L)}$ denotes the output of the final Transformer layer. Appendix C.4 details the backbone and calendar-time encoding.

Conditional flow matching. Conditioned on $\mathbf{c}_t$, a lightweight flow head models all B coordinates of the next token jointly, following BitDance (Ai et al., 2026). For training, we sample Gaussian noise $\boldsymbol{\epsilon} \sim \mathcal{N}(0, I_B)$ and a flow time $\tau \in [0, 1]$, then interpolate between the noise and the observed next token:

$$\mathbf{y}_\tau = (1-\tau)\boldsymbol{\epsilon} + \tau \mathbf{s}_{t+1}, \qquad \widehat{\mathbf{s}}_\theta = f_\theta(\mathbf{y}_\tau, \tau, \mathbf{c}_t). \tag{5}$$

The head predicts the clean endpoint $\widehat{\mathbf{s}}_\theta \in [-1, 1]^B$ rather than the flow velocity. We use the time-weighted endpoint formulation of flow matching (Lipman et al., 2022):

$$\mathcal{L}_{\text{FM}} = \mathbb{E}\left[\frac{\|\widehat{\mathbf{s}}_\theta - \mathbf{s}_{t+1}\|_2^2}{B a_\tau^2}\right], \tag{6}$$

where a_τ denotes $1 - \tau$ with a small positive floor for numerical stability. We optimize the backbone and flow head jointly while keeping the tokenizer fixed. Appendix C.5 gives the equivalent velocity formulation, auxiliary token-reconstruction losses, and training details.

3.3 Autoregressive Sampling and Decoding

At each future timestep, we hold the history condition $\mathbf{c}_t$ fixed and integrate the flow from $\mathbf{y}_0 \sim \mathcal{N}(0, I_B)$:

$$\frac{\mathrm{d}\mathbf{y}_\tau}{\mathrm{d}\tau} = \frac{f_\theta(\mathbf{y}_\tau, \tau, \mathbf{c}_t) - \mathbf{y}_\tau}{a_\tau}, \qquad \widehat{\mathbf{b}}_{t+1} = \mathbf{1}[\mathbf{y}_{\tau=1} \geq 0]. \tag{7}$$

Thresholding the final continuous state yields the next binary token. We append this token to the history, recompute the condition, and repeat until the desired horizon is reached. The frozen decoder converts the generated tokens into an OHLCVA trajectory.

Independent rollouts yield an ensemble of decoded futures for Monte Carlo forecasting (Shi et al., 2025). We derive task-specific forecasts, such as stock returns and realized volatility, from this ensemble. We optionally average K independently sampled continuous endpoints under the same history condition before thresholding. This within-token averaging produces a single next token; it is distinct from aggregation across complete trajectories. Appendix C.6 specifies the numerical solver and both aggregation procedures.

4 ReFLO: Aligning Autoregressive Flow Generation

Rejected-Future Flow Optimization (ReFLO) aligns FiRE's autoregressive flow generator with verifiable financial rewards. Its conditional flow head does not expose the explicit token probabilities used in GRPO's likelihood-ratio objective (Shao et al., 2024). ReFLO uses negative group-relative advantages to construct token-level targets that steer clean predictions away from the endpoints of below-average continuations.

4.1 Reward-Based Rejection Selection

For each tokenized history $\mathcal{H}_i = \mathbf{b}_{i,1:T}$, the current generator independently samples G continuations, which are decoded into candidate trajectories $\widehat{\mathbf{Y}}_i^{(g)}$. A task-specific reward compares each candidate with the realized future $\mathbf{Y}_i$ to obtain a score R_{ig}. For price-path adaptation, we use the equal-weight mean of temporal PCC and SCC, computed along the forecast horizon for each OHLC field (Appendix A.2).

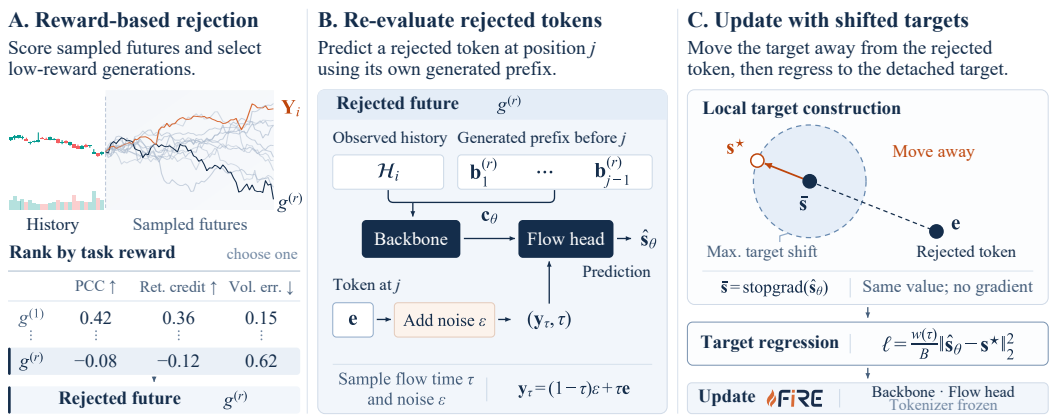


Figure 4: **ReFLO: Rejected-Future Flow Optimization.** (a) Candidate futures are scored by a task-specific reward, which is used to identify rejection signals from low-reward generations. (b) Tokens from rejected futures are re-noised and re-evaluated under their own generated prefixes. (c) A detached target is shifted away from each rejected token, and used to update the model.

We standardize rewards within each history to obtain group-relative advantages:

$$\overline{R}_i = \frac{1}{G}\sum_{g=1}^{G} R_{ig}, \qquad \sigma_i^2 = \frac{1}{G}\sum_{g=1}^{G}(R_{ig}-\overline{R}_i)^2, \qquad A_{ig} = \frac{R_{ig}-\overline{R}_i}{\sigma_i+\epsilon}, \tag{8}$$

where $\epsilon > 0$ ensures numerical stability. ReFLO uses the negative part of the advantage as its rejection signal, $\omega_{ig}^- = \min(A_{ig}, 0)$. Continuations scoring below the group mean receive repulsive updates weighted by the magnitude of their negative advantages. Nonnegative advantages yield zero signal. Groups with nonfinite rewards or $\sigma_i < \epsilon$ are inactive.

4.2 Negative-Advantage Flow Updates

Re-evaluating generated tokens. Following forward-process adaptation (Zheng et al., 2026; Choi et al., 2026), we add fresh noise to generated tokens and recompute their clean predictions. For candidate g of history i, let $\mathbf{e} = \mathbf{s}_{i,j}^{(g)}$ be its bipolar token at future position j. The backbone encodes the observed history and this candidate's generated prefix as $\mathbf{c}_{igj,\theta} = h_\theta(\mathcal{H}_i, \mathbf{b}_{i,<j}^{(g)})$. With flow time τ and Gaussian noise $\boldsymbol{\epsilon} \sim \mathcal{N}(0, I_B)$, we form

$$\mathbf{y}_\tau = (1-\tau)\boldsymbol{\epsilon} + \tau\mathbf{e}, \qquad \widehat{\mathbf{s}}_\theta = f_\theta(\mathbf{y}_\tau, \tau, \mathbf{c}_{igj,\theta}). \tag{9}$$

Each such evaluation is a *token probe*. The sampled tokens remain fixed, while gradients pass through the recomputed history condition and flow head. The rollout solver is not replayed.

Constructing repulsive targets. Each probe inherits its continuation's signal ω_{ig}^-. Let $\overline{\mathbf{s}} = \mathrm{sg}(\widehat{\mathbf{s}}_\theta)$, where sg denotes stop-gradient. Suppressing candidate indices, we construct

$$\mathbf{d} = \mathrm{Clip}_{\mathrm{RMS},c}\left[\eta\omega^-(\mathbf{e}-\overline{\mathbf{s}})\right], \qquad \mathbf{s}^\star = \mathrm{sg}(\overline{\mathbf{s}}+\mathbf{d}). \tag{10}$$

For a token from a negative-advantage continuation, $\omega^- < 0$, so the displacement points away from its generated endpoint. The step size $\eta > 0$ scales this displacement, and radial clipping bounds its RMS by c while preserving direction. The realized future determines the advantage, while the generated token and current prediction define the repulsive target.

Learning objective. We fit the detached target with a positive, time-weighted squared loss:

$$\ell_{\mathrm{ReFLO}}(\theta) = \frac{w(\tau)}{B}\,\|\widehat{\mathbf{s}}_\theta - \mathbf{s}^\star\|_2^2, \qquad w(\tau) > 0. \tag{11}$$

At target construction, the gradient with respect to the clean prediction is $-2w(\tau)\mathbf{d}/B$, giving a local repulsive update for negative signals and zero gradient for zero signals. Stop-gradient fixes the target during differentiation and preserves the intended update direction.

4.3 POST-TRAINING PROCEDURE

ReFLO alternates between collecting continuations and updating the backbone and flow head. The objective $\mathcal{L}_{\mathrm{ReFLO}}$ averages Eq. (11) over histories, all G candidates, and sampled token positions. Candidates with zero signal remain in the denominator. We average gradients equally across market–frequency tasks and control each update by bounding the change in clean-token predictions on a fixed set of inputs. The tokenizer remains frozen.

ReFLO requires neither reward gradients nor policy likelihoods. Unlike Flow-GRPO (Liu et al., 2025b), it retains ODE sampling without introducing stochastic denoising transitions for likelihood-ratio optimization. Reward evaluation is used only during post-training; inference uses the adapted generator without a scorer. Algorithm 1 summarizes the procedure. Appendix F specifies probe sampling, time weighting, reward functions, and optimization settings.

Algorithm 1: ReFLO: reward-guided post-training

Require: Pretrained generator θ, frozen tokenizer, reward evaluator R.

for each post-training iteration **do**
  Initialize $\overline{\mathbf{g}} \leftarrow 0$.
  for each market–frequency task in batch **do**
    Sample histories $\mathcal{H}_i$, realized futures $\mathbf{Y}_i$.
    Generate and decode G continuations per history (no gradients).
    Evaluate R; standardize within each history; set $\omega_{ig}^{-} = \min(A_{ig}, 0)$.
    Sample token positions, flow times, and Gaussian noises.
    Recompute clean predictions under each candidate's generated prefix.
    Form detached repulsive targets (Eq. 10).
    Average probe losses over all candidates; accumulate $\overline{\mathbf{g}}$ with equal task weights.
  end for
  Clip $\overline{\mathbf{g}}$ and form an SGD proposal.
  Backtrack to bound prediction changes on fixed inputs; accept or restore θ.
end for

5 EXPERIMENTS

We compare FiRE with established time-series models across representative financial tasks, evaluating its pretrained performance and the gains from reward-guided post-training.

5.1 EXPERIMENTAL SETUP

Following the evaluation framework of Kronos (Shi et al., 2025), we compare FiRE with supervised time-series models, general-purpose and financial foundation models, and dedicated time-series generators. FiRE uses a 40.46M-parameter predictor and a frozen 256-bit tokenizer. The benchmark spans equities, cryptocurrency, and foreign exchange at multiple frequencies. Price prediction is evaluated using Pearson (PCC) and Spearman (SCC) correlations between predicted and realized price paths over the forecast horizon. We use IC/RankIC for return prediction and QLIKE (Patton, 2011) for volatility prediction. Further details appear in Appendices A.2 and H–J.

5.2 MAIN RESULTS

Financial forecasting. Pretrained FiRE outperforms all baselines on the five forecasting metrics (Table 1). Return IC improves by 44.3% over Moirai 2.0, the strongest baseline on this metric, while volatility QLIKE is 59.0% lower than Kronos Base.

Reward-guided adaptation. ReFLO further improves all five forecasting metrics. Relative to pretraining, task-specific adaptation increases PCC by 9.6% and return IC by 4.5%, and reduces volatility QLIKE from 16.23 to 1.50 (a 90.8% reduction).

Investment simulation. To assess the economic value of the forecasts, we simulate a long-only CSI1000 portfolio in Qlib (Yang et al., 2020). Pretrained FiRE achieves a net annualized excess return (AER) of 13.77%, compared with 8.01% for the strongest baseline. Adapting FiRE with a cross-sectional ranking reward raises AER to 17.50% and the excess Sharpe ratio (ESR) from 1.20 to 1.50 (Table 1), extending the benefits of ReFLO to portfolio performance.

Synthetic candlestick generation. Following TimeGAN (Yoon et al., 2019) and Kronos (Shi et al., 2025), we evaluate synthetic data in terms of diversity, fidelity, and usefulness.

Table 1: **Forecasting and investment results.** PCC and SCC denote Pearson and Spearman correlations between predicted and realized price paths, respectively. Bold and underlining indicate the best and second-best scores.

Model	Price series		Returns		Volatility	Investment	
	PCC ↑	SCC ↑	IC ↑	RankIC ↑	QLIKE ↓	AER ↑	ESR ↑
Trained from scratch							
DLinear	0.0385	0.0367	0.0303	0.0457	57.82	−8.17	−0.82
iTransformer	0.0384	0.0378	0.0026	0.0292	55.03	7.03	0.79
FACT	0.0430	0.0412	0.0201	0.0351	63.13	−6.27	−0.74
SEER	0.0409	0.0395	0.0408	0.0646	79.82	−19.93	−1.96
TimeFilter	0.0394	0.0377	0.0287	0.0465	65.54	−4.27	−0.43
DUET	0.0403	0.0389	0.0598	0.0703	78.66	−12.22	−1.21
SRSNet	0.0324	0.0308	0.0433	0.0619	54.60	−5.95	−0.55
General-purpose pretrained models							
Chronos-2	0.0242	0.0234	0.0418	0.0511	345.10	−12.12	−1.32
TimesFM 3.0	0.0163	0.0148	0.0250	0.0410	122.55	−12.61	−1.05
TiRex-2	0.0203	0.0193	0.0678	0.0596	864.05	−17.89	−1.93
Toto 2.0 (2.5B)	0.0212	0.0200	0.0610	0.0515	120.79	−25.50	−2.94
Sundial (128M)	0.0167	0.0142	0.0710	0.0672	498.58	3.10	0.28
Moirai 2.0 (Small)	0.0178	0.0170	0.0752	0.0602	302.67	2.71	0.23
Financial pretrained models							
FinCast	0.0243	0.0234	0.0606	0.0555	1119.43	8.01	0.86
Kronos Base	0.0468	0.0442	0.0650	0.0689	39.63	7.36	0.86
FiRE	<u>0.0491</u>	<u>0.0450</u>	<u>0.1085</u>	<u>0.0853</u>	<u>16.23</u>	<u>13.77</u>	<u>1.20</u>
FiRE + ReFLO	**0.0538**	**0.0516**	**0.1134**	**0.0890**	**1.50**	**17.50**	**1.50**

AER: annualized net excess return (%); ESR: excess Sharpe ratio.

To examine diversity, we compare real and synthetic distributions using t-SNE and kernel density estimation (Figure 5 and Appendix J). FiRE's samples show substantial overlap with real data in the projections and reproduce the main features of the density profiles. We quantify fidelity using a discriminative score that measures how difficult it is to distinguish synthetic sequences from real data. Usefulness is evaluated through train-on-synthetic, test-on-real (TSTR) forecasting, with IC and RankIC measured on real test data. As shown in Table 2, FiRE achieves the best discriminative score and TSTR performance close to that of Kronos, outperforming TimeVAE (Desai et al., 2021), TimeGAN, and Diffusion-TS (Yuan & Qiao, 2024) on all three quantitative metrics.

Table 2: **Synthetic candlestick generation.** Fidelity is measured by the discriminative score (closer to 0.5 is better), and utility by TSTR IC and RankIC (higher is better).

Model	Fidelity	Usefulness (TSTR)	
	Disc.	IC ↑	RankIC ↑
TimeVAE	0.04065	0.01129	0.01144
TimeGAN	0.14681	0.01396	0.01327
Diffusion-TS	0.24942	0.01521	0.01527
Kronos Base	<u>0.29508</u>	**0.02505**	**0.02256**
FiRE	**0.39333**	<u>0.02472</u>	<u>0.02173</u>

5.3 Ablation Studies

Tokenizer reconstruction and downstream forecasting. Increasing the number of bits per LFQ token reduces reconstruction error, but downstream forecasting peaks at 256 bits (Table 3). Composite combines PCC, SCC, IC, and RankIC (Appendix A.2). Removing auxiliary losses raises MSE by 42% and lowers Composite; BSQ-256 has roughly ten times the reconstruction MSE of LFQ-256. These results support our choice of LFQ-256. Full metrics and protocols appear in Appendix D.7.

Prediction head and sampling. With 16-step Euler sampling, endpoint prediction gives the highest PCC and SCC at $K = 3$, while velocity prediction gives the highest RankIC (Table 5). Increasing K improves all four metrics for endpoint flow but has little effect on the BCE head, which models bits

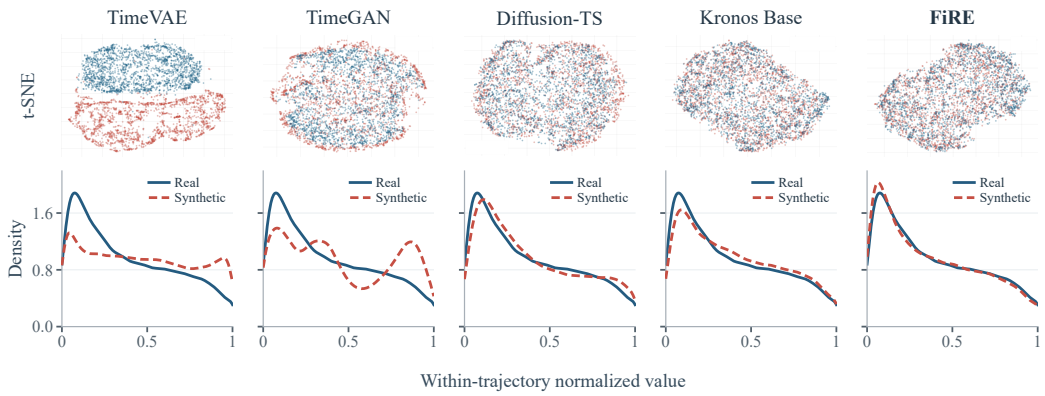


Figure 5: **Real (blue) and synthetic (red) Shanghai Stock Exchange, 15-minute candlesticks.** Top: t-SNE projections. Bottom: kernel density estimates of within-trajectory normalized values.

Table 3: **Tokenizer design ablations.** Effects of the number of bits per token, auxiliary losses, and quantizer choice on reconstruction and forecasting.

Configuration	MSE (10^{-3}) ↓	Composite ↑
LFQ-64	1.759	0.0397
LFQ-128	1.030	0.0432
LFQ-256	0.571	**0.0582**
LFQ-512	**0.449**	0.0491
LFQ-256 w/o aux.	0.813	0.0536
BSQ-256	5.750	–

Table 4: **Post-training signal ablation.** Price-forecasting performance of negative, positive, and paired reward signals relative to the pretrained model.

Signal	PCC ↑	SCC ↑
Pretrained	0.03383	0.03180
ReFLO	**0.04919**	**0.04787**
Positive	0.03218	0.03019
Paired	0.03788	0.03561

Table 5: **Prediction-head and sampling ablations.** Price and return forecasting across heads and samplers; K is the number of samples per token.

K	PCC ↑	SCC ↑	IC ↑	RankIC ↑
Endpoint flow: Euler (16)				
1	0.0205	0.0189	0.0707	0.0563
3	0.0322	0.0301	0.0922	0.0673
10	0.0428	0.0397	0.1032	0.0799
100	0.0491	0.0450	**0.1085**	**0.0853**
Endpoint flow: Consistency-style (8)				
1	0.0323	0.0305	0.0915	0.0659
3	0.0415	0.0388	0.1015	0.0778
10	0.0470	0.0439	0.1064	0.0838
100	**0.0509**	**0.0468**	0.1044	0.0847
Bitwise BCE: Bernoulli				
1	0.0273	0.0250	0.0717	0.0630
3	0.0287	0.0265	0.0729	0.0664
10	0.0294	0.0264	0.0726	0.0681
100	0.0295	0.0261	0.0737	0.0682
Direct velocity: Euler (16)				
3	0.0013	0.0010	0.0867	0.0715

independently using binary cross-entropy. Consistency-style sampling yields higher PCC and SCC than Euler sampling at each K; Appendix C.6 details both samplers.

Comparison of post-training signals. Negative-only feedback (ReFLO) achieves the highest PCC and SCC (Table 4). All variants start from the same pretrained checkpoint and use equal-weight PCC/SCC rewards. Comparing each variant's best evaluated mean PCC/SCC, ReFLO improves the score by 47.9%, versus 12.0% for combined positive–negative feedback (Paired). Positive-only feedback underperforms the pretrained baseline. Appendix F provides the protocol.

6 Conclusion

We introduced FiRE, a financial foundation model that combines high-capacity binary market representations with autoregressive flow generation. Joint generation of each token's bits avoids vocabulary-sized prediction heads, while ReFLO adapts the pretrained generator to downstream objectives using verifiable rewards. FiRE outperforms the evaluated baselines in forecasting and investment simulation and generates realistic synthetic data with competitive predictive utility. ReFLO yields further gains across distinct financial objectives, demonstrating the benefit of combining scalable generative pretraining with reward-guided adaptation.

AI USE STATEMENT

We used generative AI tools to aid and polish manuscript writing, including language editing and translation, and to support literature search and retrieval. We also used these tools for limited code editing related to figure preparation and visualization. We did not use generative AI to generate synthetic datasets or to formulate or prove mathematical claims; other required-disclosure tasks are not applicable to this work. All AI-assisted materials were carefully reviewed and verified by the authors. Literature suggestions and references were checked against the original sources, and AI-assisted code was manually reviewed for correctness. The authors take full responsibility for the final content of this work, including its text, code, figures, references, and claims.

REFERENCES

Yuang Ai, Jiaming Han, Shaobin Zhuang, Weijia Mao, Xuefeng Hu, Ziyan Yang, Zhenheng Yang, Yali Wang, Huaibo Huang, Xiangyu Yue, and Hao Chen. BitDance: Scaling autoregressive generative models with binary tokens. *arXiv preprint arXiv:2602.14041*, 2026. URL https://arxiv.org/abs/2602.14041.

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. In *International Conference on Learning Representations Workshop*, 2017.

Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, Jasper Zschiegner, Danielle C. Maddix, Hao Wang, Michael W. Mahoney, Kari Torkkola, Andrew Gordon Wilson, Michael Bohlke-Schneider, and Yuyang Wang. Chronos: Learning the language of time series. *arXiv preprint arXiv:2403.07815*, 2024. URL https://arxiv.org/abs/2403.07815.

Abdul Fatir Ansari, Oleksandr Shchur, Jaris Küken, Andreas Auer, Boran Han, Pedro Mercado, Syama Sundar Rangapuram, Huibin Shen, Lorenzo Stella, Xiyuan Zhang, Mononito Goswami, Shubham Kapoor, Danielle C. Maddix, Pablo Guerron, Tony Hu, Junming Yin, Nick Erickson, Prateek Mutalik Desai, Hao Wang, Huzefa Rangwala, George Karypis, Yuyang Wang, and Michael Bohlke-Schneider. Chronos-2: From univariate to universal forecasting. *arXiv preprint arXiv:2510.15821*, 2025. URL https://arxiv.org/abs/2510.15821.

Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219, 2022. doi: 10.1162/coli_a_00422. URL https://aclanthology.org/2022.cl-1.7/.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013. URL https://arxiv.org/abs/1308.3432.

Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2305.13301.

Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021. URL https://arxiv.org/abs/2108.07258.

Kyle Chickering, Wei-An Lin, Swayam Bhanded, Dan Saunders, Akshat Tripathi, Jiaming Song, Shyamal Buch, and Xinchen Yan. Abra: Scaling diffusion image training. *arXiv preprint arXiv:2608.17286*, 2026. URL https://arxiv.org/abs/2608.17286.

Jaemoo Choi, Wei Guo, Yuchen Zhu, Arash Vahdat, Molei Tao, Julius Berner, and Yongxin Chen. Scaling reinforcement learning for diffusion models via velocity matching. *arXiv preprint arXiv:2608.23664*, 2026. URL https://arxiv.org/abs/2608.23664.

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

Abhyuday Desai, Cynthia Freeman, Zuhui Wang, and Ian Beaver. TimeVAE: A variational auto-encoder for multivariate time series generation. *arXiv preprint arXiv:2111.08095*, 2021. URL https://arxiv.org/abs/2111.08095.

Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12873–12883, 2021. URL https://arxiv.org/abs/2012.09841.

Lijie Fan, Tianhong Li, Siyang Qin, Yuanzhen Li, Chen Sun, Michael Rubinstein, Deqing Sun, Kaiming He, and Yonglong Tian. Fluid: Scaling autoregressive text-to-image generative models with continuous tokens. *arXiv preprint arXiv:2410.13863*, 2024. URL https://arxiv.org/abs/2410.13863.

Google Research. TimesFM 3.0 model card. Hugging Face model repository, 2026. URL https://huggingface.co/google/timesfm-3.0-pytorch.

Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. MOMENT: A family of open time-series foundation models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 16115–16152. PMLR, 2024. URL https://proceedings.mlr.press/v235/goswami24a.html.

Arthur E. Hoerl and Robert W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970. doi: 10.1080/00401706.1970.10488634.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022. URL https://arxiv.org/abs/2203.15556.

Yifan Hu, Guibin Zhang, Peiyuan Liu, Disen Lan, Naiqi Li, Dawei Cheng, Tao Dai, Shu-Tao Xia, and Shirui Pan. TimeFilter: Patch-specific spatial-temporal graph filtration for time series forecasting. In *Proceedings of the International Conference on Machine Learning*, 2025. URL https://proceedings.mlr.press/v267/hu25ac.html.

Ian T. Jolliffe. A note on the use of principal components in regression. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 31(3):300–303, 1982. doi: 10.2307/2348005.

Maxime Kawawa-Beaudan, Srijan Sood, Kassiani Papasotiriou, Daniel Borrajo, and Manuela Veloso. TradeFM: A generative foundation model for trade-flow and market microstructure. *arXiv preprint arXiv:2602.23784*, 2026. URL https://arxiv.org/abs/2602.23784.

Emaad Khwaja, Chris Lettieri, Gerald Woo, Eden Belouadah, Marc Cenac, Guillaume Jarry, Enguerrand Paquin, Xunyi Zhao, Viktoriya Zhukov, Othmane Abou-Amal, Chenghao Liu, Ameet Talwalkar, and David Asker. Toto 2.0: Time series forecasting enters the scaling era. *arXiv preprint arXiv:2605.20119*, 2026. URL https://arxiv.org/abs/2605.20119.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3519–3529. PMLR, 2019. URL https://proceedings.mlr.press/v97/kornblith19a.html.

Junjie Li, Yang Liu, Weiqing Liu, Shikai Fang, Lewen Wang, Chang Xu, and Jiang Bian. MarS: A financial market simulation engine powered by generative foundation model. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2409.07486.

Tianhong Li, Yonglong Tian, He Li, Mingyang Deng, and Kaiming He. Autoregressive image generation without vector quantization. In *Advances in Neural Information Processing Systems*, 2024. URL https://arxiv.org/abs/2406.11838.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022. URL https://arxiv.org/abs/2210.02747.

Chenghao Liu, Taha Aksu, Juncheng Liu, Xu Liu, Hanshu Yan, Quang Pham, Doyen Sahoo, Caiming Xiong, Silvio Savarese, and Junnan Li. Moirai 2.0: When less is more for time series forecasting. *arXiv preprint arXiv:2511.11698*, 2025a. URL https://arxiv.org/abs/2511.11698.

Jie Liu, Gongye Liu, Jiajun Liang, Yangguang Li, Jiaheng Liu, Xintao Wang, Pengfei Wan, Di Zhang, and Wanli Ouyang. Flow-GRPO: Training flow matching models via online RL. *arXiv preprint arXiv:2505.05470*, 2025b. URL https://arxiv.org/abs/2505.05470.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. iTransformer: Inverted transformers are effective for time series forecasting. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2310.06625.

Yong Liu, Guo Qin, Zhiyuan Shi, Zhi Chen, Caiyin Yang, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Sundial: A family of highly capable time series foundation models. *arXiv preprint arXiv:2502.00816*, 2025c. URL https://arxiv.org/abs/2502.00816.

Dat Mai. StockGPT: A GenAI model for stock prediction and trading. *arXiv preprint arXiv:2404.05101*, 2024. URL https://arxiv.org/abs/2404.05101.

NextStep Team, Chunrui Han, Guopeng Li, Jingwei Wu, Quan Sun, Yan Cai, Yuang Peng, Zheng Ge, Deyu Zhou, Haomiao Tang, Hongyu Zhou, Kenkun Liu, Ailin Huang, Bin Wang, Changxin Miao, Deshan Sun, En Yu, Fukun Yin, Gang Yu, Hao Nie, Haoran Lv, Hanpeng Hu, Jia Wang, Jian Zhou, Jianjian Sun, Kaijun Tan, Kang An, Kangheng Lin, Liang Zhao, Mei Chen, Peng Xing, Rui Wang, Shiyu Liu, Shutao Xia, Tianhao You, Wei Ji, Xianfang Zeng, Xin Han, Xuelin Zhang, Yana Wei, Yanming Xu, Yimin Jiang, Yingming Wang, Yu Zhou, Yucheng Han, Ziyang Meng, Binxing Jiao, Daxin Jiang, Xiangyu Zhang, and Yibo Zhu. NextStep-1: Toward autoregressive image generation with continuous tokens at scale. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2508.10711.

Andrew J. Patton. Volatility forecast comparison using imperfect volatility proxies. *Journal of Econometrics*, 160(1):246–256, 2011. URL https://public.econ.duke.edu/~ap172/Patton_robust_JoE_forthcoming.pdf.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

Patrick Podest, Marco Pichler, Elias Bürger, Levente Zólyomi, Bernhard Voggenberger, Wilhelm Berghammer, Daniel Klotz, Sebastian Böck, Günter Klambauer, and Sepp Hochreiter. TiRex-2: Generalizing TiRex to multivariate data and streaming. *arXiv preprint arXiv:2607.01204*, 2026. URL https://arxiv.org/abs/2607.01204.

Xiangfei Qiu, Xingjian Wu, Yan Lin, Chenjuan Guo, Jilin Hu, and Bin Yang. DUET: Dual clustering enhanced multivariate time series forecasting. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1185–1196, 2025. doi: 10.1145/3690624.3709325. URL https://arxiv.org/abs/2412.10859.

Xiangfei Qiu, Xvyuan Liu, Tianen Shen, Xingjian Wu, Hanyin Cheng, Bin Yang, and Jilin Hu. SEER: Transformer-based robust time series forecasting via automated patch enhancement and replacement. *arXiv preprint arXiv:2602.00589*, 2026. URL https://arxiv.org/abs/2602.00589.

Olivier Roy and Martin Vetterli. The effective rank: A measure of effective dimensionality. In *15th European Signal Processing Conference*, pp. 606–610, 2007.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. URL https://arxiv.org/abs/2402.03300.

Noam Shazeer. GLU variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020. URL https://arxiv.org/abs/2002.05202.

Yu Shi, Zongliang Fu, Shuo Chen, Bohan Zhao, Wei Xu, Changshui Zhang, and Jian Li. Kronos: A foundation model for the language of financial markets. *arXiv preprint arXiv:2508.02739*, 2025. URL https://arxiv.org/abs/2508.02739.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 32211–32252. PMLR, 2023. URL https://proceedings.mlr.press/v202/song23a.html.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. RoFormer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*, 2021. URL https://arxiv.org/abs/2104.09864.

Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *arXiv preprint arXiv:2006.10739*, 2020. URL https://arxiv.org/abs/2006.10739.

Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://arxiv.org/abs/1711.00937.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008. URL https://jmlr.org/papers/v9/vandermaaten08a.html.

Bram Wallace, Meihua Dang, Rafael Rafailov, Linqi Zhou, Aaron Lou, Senthil Purushwalkam, Stefano Ermon, Caiming Xiong, Shafiq Joty, and Nikhil Naik. Diffusion model alignment using direct preference optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8228–8238, 2024. URL https://arxiv.org/abs/2311.12908.

Huiqiang Wang, Jieming Shi, and Qing Li. FACT: Fine-grained across-variable convolution for multivariate time-series forecasting. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=j3gNYqrHtl.

Xingjian Wu, Xiangfei Qiu, Hanyin Cheng, Zhengyu Li, Jilin Hu, Chenjuan Guo, and Bin Yang. Enhancing time series forecasting through selective representation spaces: A patch perspective. In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2510.14510.

Yuanjian Xu, Jianing Hao, Anxian Liu, Zhenzhuo Li, Shichang Meng, Shuai Yuan, and Guang Zhang. LENS: Large pre-trained transformer for exploring financial time series regularities. In *Proceedings of the 6th ACM International Conference on AI in Finance*, pp. 771–778. Association for Computing Machinery, 2025. doi: 10.1145/3768292.3770349. URL https://doi.org/10.1145/3768292.3770349.

Shuchen Xue, Chongjian Ge, Shilong Zhang, Yichen Li, and Zhi-Ming Ma. Advantage weighted matching: Aligning RL with pretraining in diffusion models. In *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2509.25050.

Dennis Yang and Qiang Zhang. Drift-independent volatility estimation based on high, low, open, and close prices. *The Journal of Business*, 73(3), 2000. doi: 10.1086/209650.

Xiao Yang, Weiqing Liu, Dong Zhou, Jiang Bian, and Tie-Yan Liu. Qlib: An AI-oriented quantitative investment platform. *arXiv preprint arXiv:2009.11189*, 2020. URL https://arxiv.org/abs/2009.11189.

Jinsung Yoon, Daniel Jarrett, and Mihaela van der Schaar. Time-series generative adversarial networks. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/c9efe5f26cd17ba6216bbe2a7d26d490-Abstract.html.

Lijun Yu, José Lezama, Nitesh B. Gundavarapu, Luca Versari, Kihyuk Sohn, David Minnen, Yong Cheng, Vighnesh Birodkar, Agrim Gupta, Xiuye Gu, Alexander G. Hauptmann, Boqing Gong, Ming-Hsuan Yang, Irfan Essa, David A. Ross, and Lu Jiang. Language model beats diffusion—tokenizer is key to visual generation. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2310.05737.

Xinyu Yuan and Yan Qiao. Diffusion-TS: Interpretable diffusion for general time series generation. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=4h1apFjO99.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023. URL https://arxiv.org/abs/2205.13504.

Biao Zhang and Rico Sennrich. Root mean square layer normalization. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://arxiv.org/abs/1910.07467.

Yanzhi Zhang, Yu Ma, Yilin Cheng, Jian Li, and Yitong Duan. M3: A state-event generative foundation model for market microstructure dynamics. *arXiv preprint arXiv:2608.19227*, 2026. URL https://arxiv.org/abs/2608.19227.

Yue Zhao, Yuanjun Xiong, and Philipp Krähenbühl. Image and video tokenization with binary spherical quantization. *arXiv preprint arXiv:2406.07548*, 2024. URL https://arxiv.org/abs/2406.07548.

Kaiwen Zheng, Huayu Chen, Haotian Ye, Haoxiang Wang, Qinsheng Zhang, Kai Jiang, Hang Su, Stefano Ermon, Jun Zhu, and Ming-Yu Liu. DiffusionNFT: Online diffusion reinforcement with forward process. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2509.16117.

Lei Zhu, Fangyun Wei, Yanye Lu, and Dong Chen. Scaling the codebook size of VQ-GAN to 100,000 with a utilization rate of 99%. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://arxiv.org/abs/2406.11837.

Xinyu Zhu, Mengzhou Xia, Zhepei Wei, Wei-Lin Chen, Danqi Chen, and Yu Meng. The surprising effectiveness of negative reinforcement in LLM reasoning. In *Advances in Neural Information Processing Systems*, 2025a. URL https://arxiv.org/abs/2506.01347.

Zhuohang Zhu, Haodong Chen, Qiang Qu, and Vera Chung. FinCast: A foundation model for financial time-series forecasting. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pp. 4539–4549. Association for Computing Machinery, 2025b. doi: 10.1145/3746252.3761261. URL https://doi.org/10.1145/3746252.3761261.

APPENDIX CONTENTS

Page

A Data and Evaluation . . . 16
A.1 Pretraining Data . . . 16
A.2 Evaluation Metrics . . . 16

B Related Work . . . 19
B.1 Financial Foundation Models . . . 19
B.2 High-Capacity Tokenization and Generative Autoregression . . . 19
B.3 Post-Training Diffusion and Flow Models . . . 19

C Representation and Generative Pretraining . . . 20
C.1 Data Normalization and Training Sequences . . . 20
C.2 Tokenizer Architecture and Quantization . . . 20
C.3 Tokenizer Objective . . . 20
C.4 History Conditioning . . . 21
C.5 Flow Head and Training Objective . . . 21
C.6 Inference Sampling and Aggregation . . . 22

D Analysis of the Market Tokenizer . . . 24
D.1 Evaluation Protocol . . . 24
D.2 Covariance Structure and Bit Utilization . . . 24
D.3 Market Attributes in the Principal Components . . . 24
D.4 Linear Readouts beyond the Leading Components . . . 26
D.5 Decoder Sensitivity to Code Perturbations . . . 27
D.6 Additional Analysis Details . . . 29
D.7 Tokenizer Design Ablations . . . 29

E Predictor Representations and Forecasting . . . 31
E.1 Predictive Readouts across Layers and Checkpoints . . . 31
E.2 Historical Structure in Hidden States . . . 32
E.3 Forecasting Heads and Training Variants . . . 35
E.4 Summary . . . 36
E.5 Experimental Protocols . . . 36

F ReFLO: Rewards and Optimization . . . 38
F.1 Group-Relative Advantages and Token Probes . . . 38
F.2 Repulsive Regression and Its Local Gradient . . . 38
F.3 Financial Reward Functions . . . 39
F.4 Optimization and Training Configuration . . . 40

G Scaling the Flow Predictor . . . 41
G.1 Model Family and Evaluation . . . 41
G.2 Joint Scaling with Parameters and Training Tokens . . . 41

H Forecasting Benchmark and Evaluation . . . 43
H.1 Benchmark . . . 43
H.2 Models Trained from Scratch . . . 43
H.3 Pretrained Baselines . . . 44
H.4 FiRE and ReFLO . . . 45

I Qlib Backtesting and Economic Evaluation . . . 46
I.1 Market Data and Signal Construction . . . 46
I.2 Ranking-Reward Post-Training . . . 47
I.3 Portfolio Construction and Return Accounting . . . 47
I.4 Backtest Results . . . 48

J Synthetic Candlestick Generation: Additional Visualizations . . . 49
J.1 Generation and Evaluation Protocol . . . 49
J.2 Distributional Comparisons . . . 49

K Case Studies: Reconstruction and Forecasting Visualization . . . 52
K.1 Tokenizer Reconstruction . . . 52
K.2 Forecasting Examples . . . 54

A Data and Evaluation

A.1 Pretraining Data

FiRE's pretraining corpus is built on the cleaned OHLCVA bar data used by Kronos (Shi et al., 2025). We follow the same data-cleaning pipeline and summarize below the market coverage and temporal split used in this work.

A.1.1 Pretraining Corpus

The corpus consists of OHLCVA bar sequences, where each bar records the open, high, low, and close prices, trading volume, and traded amount. It spans four major asset classes: equities and ETFs, equity indices, foreign exchange, and cryptocurrencies. The equity and ETF universe covers 46 exchanges across 40 countries and regions, while index data are organized separately by country or region. FiRE is pretrained across twelve bar frequencies: 1 min, 5 min, 10 min, 15 min, 20 min, 30 min, 40 min, 60 min, 2 h, 4 h, daily, and weekly. The available history varies across markets and frequencies.

The tokenizer and predictor are trained on the same data universe and the same set of twelve bar frequencies, but apply different eligibility criteria for context length.

Table A.1: Overview of the FiRE pretraining corpus.

Item	Coverage
Asset classes	Equities and ETFs, equity indices, foreign exchange, and cryptocurrencies
Market coverage	Equities and ETFs from 46 exchanges across 40 countries and regions; indices organized separately by country or region
Bar frequencies	1 min, 5 min, 10 min, 15 min, 20 min, 30 min, 40 min, 60 min, 2 h, 4 h, daily, and weekly
Pretraining cutoff	February 29, 2024

A.1.2 Data Processing

Data cleaning follows Kronos (Shi et al., 2025, Appendix B, Algorithm 1 and Table 4), including missing-value handling and the removal of low-quality segments. FiRE is trained directly on the resulting cleaned, contiguous sequences, without additional quality filtering. Window normalization and activity dropout are described in Appendix C.1.

A.1.3 Temporal Splits

The shared pretraining corpus ends on February 29, 2024. For predictor training, we use March 1–June 30, 2024 for validation and July 1–December 31, 2024 for the standard prediction test. Training, validation, and test targets are therefore strictly ordered in time. Validation samples may draw on earlier observations only as historical context for the required lookback window.

The frozen 62-task benchmark in Section 5.2 uses a separate, fixed evaluation split, whose coverage and construction are described in Appendix H.1. Metric definitions are provided in Appendix A.2, while the backtesting period and protocol are described separately in Appendix I.

A.2 Evaluation Metrics

Tokenizer reconstruction and bit utilization. For each normalized validation bar t, let $\mathbf{x}_t$ and $\widehat{\mathbf{x}}_t$ denote the original and reconstructed OHLCVA. Reconstruction error is measured by the six-field MSE, $\|\widehat{\mathbf{x}}_t - \mathbf{x}_t\|_2^2/6$, averaged over validation bars. Strict candle validity is the fraction of reconstructed bars satisfying $\widehat{H}_t \geq \max(\widehat{O}_t, \widehat{C}_t)$ and $\widehat{L}_t \leq \min(\widehat{O}_t, \widehat{C}_t)$ before any validity-enforcing output projection.

For a B-bit token, let p_j denote the empirical fraction of 131,072 sampled validation tokens for which bit j is positive, and define $m_j = \min(p_j, 1 - p_j)$. Mean bit entropy is

$$\frac{1}{B}\sum_{j=1}^{B}\left[-p_j \log_2 p_j - (1-p_j)\log_2(1-p_j)\right], \tag{A.1}$$

with $0 \log_2 0 = 0$. We count bits with $m_j \leq 0.01$ as dead and those with $m_j \leq 0.05$ as near-constant. These metrics characterize marginal bit utilization across tokens.

To assess the downstream utility of each tokenizer, we additionally report a composite score over the four forecasting correlations defined below:

$$\text{Composite} = 0.35\,\text{PCC} + 0.15\,\text{SCC} + 0.35\,\text{IC} + 0.15\,\text{RankIC}. \tag{A.2}$$

The tokenizer ablation uses one generated trajectory per forecast window, 16-step ODE sampling, and $K = 3$ endpoint samples per token (Appendix C.6).

For the forecasting metrics below, let $a \in \mathcal{A}$ index one of the 62 market–frequency settings and i index its forecast windows. We denote the context and forecast lengths by T_a and H_a, respectively. For OHLC field f, let $\mathbf{y}_{i,f} = (Y_{i,1,f}, \ldots, Y_{i,H_a,f})$ and $\widehat{\mathbf{y}}_{i,f}$ denote the realized and predicted future paths in market units. When considering a fixed setting a, we omit the subscript a for readability and write $C_{i,0}$ for the last observed close.

Price series prediction. For nonconstant vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$, Pearson and Spearman correlation are

$$\rho(\mathbf{u}, \mathbf{v}) = \frac{\sum_{j=1}^{n}(u_j - \bar{u})(v_j - \bar{v})}{\sqrt{\sum_{j=1}^{n}(u_j - \bar{u})^2 \sum_{j=1}^{n}(v_j - \bar{v})^2}}, \qquad \rho_{\text{S}}(\mathbf{u}, \mathbf{v}) = \rho(\text{rank}(\mathbf{u}), \text{rank}(\mathbf{v}))\,, \tag{A.3}$$

where ties receive their average rank. Let $\mathcal{F}_i$ be the OHLC fields whose realized future paths are nonconstant, and let $\mathcal{W}_a$ contain the windows with $\mathcal{F}_i \neq \emptyset$. Temporal PCC is

$$\text{PCC}_a = \frac{1}{|\mathcal{W}_a|}\sum_{i \in \mathcal{W}_a}\frac{1}{|\mathcal{F}_i|}\sum_{f \in \mathcal{F}_i}\rho(\widehat{\mathbf{y}}_{i,f}, \mathbf{y}_{i,f}); \tag{A.4}$$

SCC uses ρ_{S} in place of ρ. Thus, price correlations are computed along the forecast horizon and then averaged over fields and windows. The set of evaluated fields depends only on the target. A constant prediction on an evaluated field yields an undefined correlation, which is retained as undefined in the setting-level score and the overall mean.

Return prediction. For each prediction window i, we compute the realized and predicted returns over horizon H_a relative to the observed close at the start of the window:

$$r_i = \frac{C_{i,H_a}}{C_{i,0}} - 1, \qquad \widehat{r}_i = \frac{\widehat{C}_{i,H_a}}{C_{i,0}} - 1. \tag{A.5}$$

Let $\mathbf{r}_a$ and $\widehat{\mathbf{r}}_a$ denote the realized and predicted returns across all evaluation windows. We report

$$\text{IC}_a = \rho(\widehat{\mathbf{r}}_a, \mathbf{r}_a), \qquad \text{RankIC}_a = \rho_{\text{S}}(\widehat{\mathbf{r}}_a, \mathbf{r}_a). \tag{A.6}$$

Volatility prediction. We estimate horizon variance from an OHLC path using the Yang–Zhang construction (Yang & Zhang, 2000). For a horizon of $h = H_a$ bars, write O_j, H_j, L_j, C_j for the four prices and define

$$\begin{aligned} o_j &= \log(O_j/C_{j-1}), \qquad c_j = \log(C_j/O_j), \\ s_o^2 &= \frac{1}{h}\sum_{j=1}^{h}(o_j - \bar{o})^2, \qquad s_c^2 = \frac{1}{h}\sum_{j=1}^{h}(c_j - \bar{c})^2, \\ s_{\text{RS}}^2 &= \frac{1}{h}\sum_{j=1}^{h}\left[\log(H_j/O_j)\log(H_j/C_j) + \log(L_j/O_j)\log(L_j/C_j)\right]. \end{aligned} \tag{A.7}$$

The first opening return uses the last observed close C_0. The resulting variance is

$$v = h \max\{s_o^2 + k_h s_c^2 + (1 - k_h) s_{\mathrm{RS}}^2, 0\}, \qquad k_h = \frac{0.34}{1.34 + (h+1)/(h-1)}. \tag{A.8}$$

The evaluated implementation uses population moments with denominator h and reports horizon variance without annualization. Applying the same estimator to predicted and realized paths gives $\widehat{v}_i$ and v_i.

We use the nonnegative form of QLIKE (Patton, 2011), with variance floors and ratio clipping for numerical stability:

$$q_i = \mathrm{clip}\left(\frac{\max(v_i, \epsilon_v)}{\max(\widehat{v}_i, \epsilon_v)}, 10^{-4}, 10^{4}\right), \qquad \ell_{\mathrm{QLIKE},i} = q_i - \log q_i - 1, \qquad \epsilon_v = 10^{-8}. \tag{A.9}$$

For positive variances without these safeguards, this loss differs from $\log \widehat{v}_i + v_i/\widehat{v}_i$ by the target-dependent constant $-\log v_i - 1$. Lower values indicate better variance forecasts.

Aggregation and uncertainty. For Table 1, QLIKE is averaged over windows within each setting. All five forecasting metrics are averaged over settings with equal weight: $\bar{m} = |\mathcal{A}|^{-1} \sum_{a \in \mathcal{A}} m_a$. The investment metrics in the same table instead aggregate daily portfolio returns, as defined in Appendix I.3.

B Related Work

B.1 Financial Foundation Models

Market pretraining spans different levels of temporal aggregation. At the event level, MarS learns order generation for interactive simulation (Li et al., 2025), while M3 jointly models order events and evolving limit-order-book states (Zhang et al., 2026). TradeFM uses scale-invariant trade representations to support cross-asset and cross-region generalization (Kawawa-Beaudan et al., 2026), showing that transfer is also possible at the microstructure level. For aggregated sequences, StockGPT models daily returns autoregressively (Mai, 2024); LENS and FinCast develop financial pretraining architectures for forecasting under noise and distribution shifts (Xu et al., 2025; Zhu et al., 2025b). Kronos instead learns multivariate candlestick tokens and a hierarchical autoregressive prior, supporting several quantitative tasks through a common market generator (Shi et al., 2025). FiRE extends this candlestick-based approach by coupling direction-aware, high-capacity market representations with flow-based generation and objective-specific post-training. This brings representation scaling and task adaptation into a common financial modeling framework.

B.2 High-Capacity Tokenization and Generative Autoregression

Learned tokenization separates representation learning from sequence modeling. VQ-VAE and VQGAN establish discrete latent vocabularies for autoregressive generation (van den Oord et al., 2017; Esser et al., 2021). Later work expands representation capacity through binary quantization without an explicit codebook (Yu et al., 2024; Zhao et al., 2024) or improved utilization of large learned codebooks (Zhu et al., 2024). A related line separates autoregressive dependency modeling from categorical output prediction. MAR models per-token distributions through diffusion (Li et al., 2024); Fluid studies how token representation and generation order affect scaling behavior (Fan et al., 2024); and NextStep-1 combines a large autoregressive backbone with a lightweight flow head (NextStep Team et al., 2026). BitDance provides the closest architectural precedent, pairing high-capacity binary tokens with continuous-space diffusion rather than a softmax over token identities (Ai et al., 2026). In time series, Sundial uses flow matching to predict the distribution of continuous-valued future patches without discrete tokenization (Liu et al., 2025c). These studies distinguish two design choices: whether the representation is discrete, and how its conditional distribution is generated. FiRE retains discrete binary market tokens but models their conditional distribution through flow matching. This design lets us study how representation capacity and financial price structure affect downstream quantitative performance within a common generative framework.

B.3 Post-Training Diffusion and Flow Models

Reward-based post-training connects generative models to objectives not directly optimized during pretraining. Denoising Diffusion Policy Optimization (DDPO) treats denoising as a sequential decision process and applies policy gradients to terminal rewards (Black et al., 2024). Diffusion-DPO instead learns from pairwise preferences using an objective derived from the diffusion evidence lower bound (Wallace et al., 2024). For flow models, Flow-GRPO introduces stochastic denoising transitions through an ODE-to-SDE conversion and uses group-relative policy optimization (Liu et al., 2025b). More closely related to our update mechanism, Advantage Weighted Matching reweights score- or flow-matching losses by sample advantages (Xue et al., 2026), while DiffusionNFT contrasts positive and negative generations to construct forward-process flow-matching supervision (Zheng et al., 2026). ReFLO adapts autoregressive market generators through repulsive token-level regression, using negative group-relative advantages derived from verifiable financial rewards.

C MARKET REPRESENTATION AND GENERATIVE PRETRAINING

C.1 DATA NORMALIZATION AND TRAINING SEQUENCES

We standardize the four OHLC prices using a shared mean and standard deviation, preserving their relative scale within each candlestick. Trading volume and traded amount are standardized separately, and normalized inputs are clipped to $[-8, 8]$. During tokenizer training and autoregressive pretraining, these statistics are computed over the sampled training window. At inference, we estimate normalization statistics from the observed history and transform generated observations back to the original scale.

The maximum training input length is 512 bars for 5–40-minute data, 360 for 60-minute, 2-hour, and 4-hour data, 120 for daily data, and 32 for weekly data. We jointly mask trading volume and traded amount across an entire sequence with probability 0.05, encouraging the model to predict from prices alone when these fields are unavailable.

C.2 TOKENIZER ARCHITECTURE AND QUANTIZATION

The tokenizer maps each candlestick to one binary token using causal Transformer encoder and decoder networks. Both use rotary position embeddings (Su et al., 2021), pre-RMSNorm (Zhang & Sennrich, 2019), and SwiGLU feed-forward layers (Shazeer, 2020). Table C.1 summarizes the architecture.

Table C.1: Market tokenizer architecture.

Component	Configuration
Encoder / decoder depth	4 / 4 causal Transformer layers
Hidden dimension / token width	256 / 256 bits
Attention heads	4
Feed-forward dimension	512
Input / latent / output projections	$6 \to 256$, 256, $256 \to 6$
Quantization	Lookup-free quantization
Trainable parameters	5,390,086

Following LFQ (Yu et al., 2024), the signs of the encoder output define $\mathbf{b}_t = \mathbf{1}[\mathbf{z}_t \geq 0]$ and its bipolar representation $\mathbf{s}_t = 2\mathbf{b}_t - 1$. The decoder receives $\mathbf{q}_t = \mathbf{s}_t/\sqrt{B}$. We train through quantization with a straight-through estimator (Bengio et al., 2013):

$$\widetilde{\mathbf{s}}_t = \mathbf{z}_t + \mathrm{sg}(\mathbf{s}_t - \mathbf{z}_t), \qquad \mathbf{q}_t = \widetilde{\mathbf{s}}_t/\sqrt{B}. \tag{C.1}$$

This retains binary values in the forward pass while allowing reconstruction gradients to reach the encoder.

C.3 TOKENIZER OBJECTIVE

The regularizers in Eq. (4) encourage code utilization and preserve candlestick structure. Expectations below denote minibatch averages over tokens.

Code utilization. The LFQ objective combines commitment, marginal bit balance, and grouped entropy:

$$\mathcal{L}_{\mathrm{LFQ}} = 0.05\mathcal{L}_{\mathrm{commit}} + 0.01\mathcal{L}_{\mathrm{balance}} + 0.0005\mathcal{L}_{\mathrm{group}}. \tag{C.2}$$

Commitment keeps the encoder output close to its quantized value, while marginal balance encourages each bit to be active on approximately half of the tokens:

$$\begin{aligned} \mathcal{L}_{\mathrm{commit}} &= \mathbb{E}\left[\frac{1}{B}\|\mathbf{z} - \mathrm{sg}(\mathbf{s})\|_2^2\right], \\ \mathcal{L}_{\mathrm{balance}} &= \frac{1}{B}\sum_{i=1}^{B}(\overline{p}_i - \tfrac{1}{2})^2, \qquad \overline{p}_i = \mathbb{E}[\mathrm{sigmoid}(z_i)]. \end{aligned} \tag{C.3}$$

For the entropy term, we partition the bits into four-bit groups and normalize the encoder output as $\mathbf{z}^{\mathrm{ent}} = \sqrt{B}\,\mathbf{z}/\|\mathbf{z}\|_2$. Each group g defines a soft distribution π_g over its 16 sign patterns $\mathbf{e}_k \in \{-1,+1\}^4$:

$$\pi_g(k \mid \mathbf{z}) = \mathrm{softmax}_k\big((\mathbf{z}_g^{\mathrm{ent}})^\top \mathbf{e}_k\big)\,, \qquad \overline{\pi}_g = \mathbb{E}[\pi_g(\cdot \mid \mathbf{z})],$$
$$\mathcal{L}_{\mathrm{group}} = 1 + \frac{1}{\log 16}\,\mathrm{mean}_g\left(\mathbb{E}[\mathsf{H}(\pi_g(\cdot \mid \mathbf{z}))] - \mathsf{H}(\overline{\pi}_g)\right), \tag{C.4}$$

where H is Shannon entropy. This favors confident assignments for individual tokens and diverse assignments across the minibatch (Yu et al., 2024), while avoiding enumeration of the full 2^B code space.

OHLC ordering. We penalize a reconstructed high below the candle body and a reconstructed low above it:

$$\mathcal{L}_{\mathrm{legal}} = \frac{1}{2}\mathbb{E}\Big[\big[\max(\widehat{O},\widehat{C}) - \widehat{H}\big]_+^2 + \big[\widehat{L} - \min(\widehat{O},\widehat{C})\big]_+^2\Big], \tag{C.5}$$

where $[a]_+ = \max(a, 0)$.

Price direction. We also preserve the direction of candlestick bodies, $C_t - O_t$, and adjacent close changes, $C_t - C_{t-1}$. For either price change d and its reconstruction $\widehat{d}$, we use the hinge penalty

$$\ell_{\mathrm{dir}}(d, \widehat{d}) = \big[\min(|d|, \mu) - \mathrm{sign}(d)\widehat{d}\big]_+, \qquad \mu = 10^{-3}. \tag{C.6}$$

The margin is bounded by the observed change, so small movements do not require exaggerated reconstructions. We average the penalty over nonzero changes of each type and give the two averages equal weight to obtain $\mathcal{L}_{\mathrm{dir}}$. Its weight in Eq. (4) is fixed at $\lambda = 0.05$.

C.4 History Conditioning

The backbone has eight causal Transformer layers with hidden dimension $d = 512$, eight attention heads, and feed-forward dimension 2048.

Calendar-time encoding. We supplement each market token with the calendar time of its observation. The timestamp is decomposed into five components: minute, hour, weekday, day of month, and month. Each component is mapped to sine and cosine features at multiple frequencies (Tancik et al., 2020). For a scalar calendar value a, the feature map takes the form

$$\boldsymbol{\gamma}(a) = \big[\sin(\omega_1 a), \cos(\omega_1 a), \ldots, \sin(\omega_m a), \cos(\omega_m a)\big]^\top, \tag{C.7}$$

where $\omega_1, \ldots, \omega_m$ denote its encoding frequencies. Lower frequencies vary more gradually with the calendar value, while higher frequencies resolve finer variations. The Fourier features of all five calendar components are linearly projected to obtain a calendar-time embedding. We sum this embedding with the separately projected token embedding and the positional embedding to form the Transformer input:

$$\mathbf{h}_t^{(0)} = \mathrm{Linear}_{\mathrm{tok}}(\mathbf{s}_t) + \mathrm{Linear}_{\mathrm{time}}(\mathbf{f}_t) + \mathbf{p}_t, \tag{C.8}$$

where $\mathbf{f}_t$ collects the Fourier features of the five calendar components and $\mathbf{p}_t$ denotes the positional embedding at position t.

C.5 Flow Head and Training Objective

Endpoint prediction. The flow head concatenates the noisy token, the history condition, and a 1024-dimensional embedding of flow time. A four-layer MLP with hidden width 1024 and SiLU activations produces the bounded endpoint estimate

$$f_\theta(\mathbf{y}_\tau, \tau, \mathbf{c}_t) = 2\,\mathrm{sigmoid}(F_\theta(\mathbf{y}_\tau, \tau, \mathbf{c}_t)) - 1. \tag{C.9}$$

Flow matching. We sample $\tau = \mathrm{sigmoid}(\xi)$ with $\xi \sim \mathcal{N}(-0.8, 0.8^2)$ and use the interpolation in Eq. (5). With $a_\tau = \max(1-\tau, 0.05)$, the velocity-regression form of the objective (Lipman et al., 2022) is

$$\begin{aligned} \mathbf{v}_\tau^{\mathrm{tar}} &= \frac{\mathbf{s} - \mathbf{y}_\tau}{a_\tau}, \qquad \widehat{\mathbf{v}}_\theta = \frac{\widehat{\mathbf{s}}_\theta - \mathbf{y}_\tau}{a_\tau}, \\ \ell_v &= \frac{1}{B}\|\widehat{\mathbf{v}}_\theta - \mathbf{v}_\tau^{\mathrm{tar}}\|_2^2 = \frac{1}{Ba_\tau^2}\|\widehat{\mathbf{s}}_\theta - \mathbf{s}\|_2^2. \end{aligned} \tag{C.10}$$

This recovers the time-weighted endpoint loss in Eq. (6).

C.6 Inference Sampling and Aggregation

The predictor generates future tokens autoregressively. Inference uses two sampling budgets: K endpoint draws for each token and N complete future trajectories. The endpoint solver determines how each continuous draw is produced.

Aggregation and decoding. At future position j, the predictor computes the history condition $\mathbf{c}_j$ and samples K continuous endpoints independently. Their mean is thresholded to produce the next binary token:

$$\overline{\mathbf{y}}_j = \frac{1}{K}\sum_{k=1}^{K} \mathbf{y}_{\mathrm{end},j}^{(k)}, \qquad \widehat{\mathbf{b}}_j = \mathbf{1}[\overline{\mathbf{y}}_j \geq 0]. \tag{C.11}$$

The generated token is appended to the history before computing $\mathbf{c}_{j+1}$. Thus, K controls averaging before quantization, and its effect propagates through subsequent autoregressive steps. Averaging continuous endpoints is distinct from majority voting over thresholded bits.

Each of the N trajectories starts from the same observed history and uses independent noise, conditioning subsequent predictions on its own generated tokens. After generating L future tokens, the decoder takes their bipolar representations $\widehat{\mathbf{q}}_j = (2\widehat{\mathbf{b}}_j - 1)/\sqrt{B}$ and produces normalized OHLCVA sequences. Only the generated token block is decoded. The historical normalization statistics then restore the outputs to market units (Appendix C.1). Writing this decoding and restoration map as $\mathcal{D}$, the mean-path forecast is

$$\widehat{\mathbf{X}}^{(n)} = \mathcal{D}(\widehat{\mathbf{b}}_{1:L}^{(n)}), \qquad \widehat{\mathbf{X}} = \frac{1}{N}\sum_{n=1}^{N} \widehat{\mathbf{X}}^{(n)}. \tag{C.12}$$

Here K averages endpoints within each autoregressive step, whereas N averages decoded trajectories after generation. A nonlinear quantity such as realized volatility can be computed from the mean path or averaged across individual paths; these estimators generally differ. The task-specific evaluation protocol determines the aggregation used.

ODE sampling. For one endpoint draw at position j, initialize $\mathbf{y}_0 \sim \mathcal{N}(0, I_B)$ and integrate the learned velocity field using Euler steps, following the flow-matching formulation (Lipman et al., 2022). Let S denote the number of solver steps, $\Delta = 1/S$, and $\widetilde{\tau}_s = \min(s\Delta, 0.95)$. The update is

$$\mathbf{y}_{s+1} = \mathbf{y}_s + \Delta\,\frac{f_\theta(\mathbf{y}_s, \widetilde{\tau}_s, \mathbf{c}_j) - \mathbf{y}_s}{\max(1-\widetilde{\tau}_s, 0.05)}, \qquad s = 0, \ldots, S-1. \tag{C.13}$$

The solver returns $\mathbf{y}_{\mathrm{end}} = \mathbf{y}_S$. Given the initial noise and history condition, all updates are deterministic. We use $S = 16$, requiring 16 endpoint-network evaluations per draw.

Consistency-style sampling. Following the alternating prediction and noise-injection pattern of multistep consistency sampling (Song et al., 2023), we apply the steps below to our flow-matching endpoint network using the linear interpolation from training. Starting from $\mathbf{y}_0 \sim \mathcal{N}(0, I_B)$, let $\tau_s = s/S$. At each step, predict

$$\widehat{\mathbf{s}}_s = f_\theta(\mathbf{y}_s, \tau_s, \mathbf{c}_j), \qquad s = 0, \ldots, S-1. \tag{C.14}$$

For $s < S-1$, combine this prediction with fresh Gaussian noise to obtain the next input:

$$\mathbf{y}_{s+1} = \tau_{s+1}\widehat{\mathbf{s}}_s + (1-\tau_{s+1})\boldsymbol{\epsilon}_s, \qquad \boldsymbol{\epsilon}_s \sim \mathcal{N}(0, I_B). \tag{C.15}$$

The last step returns $\mathbf{y}_{\mathrm{end}} = \widehat{\mathbf{s}}_{S-1}$ directly. We use $S = 8$ endpoint evaluations.

The forecasting comparisons and synthetic-generation experiments specify their sampling budgets and postprocessing in Appendices H.4 and J.1, respectively.

D Analysis of the Market Tokenizer

We analyze the market tokenizer through its covariance structure, linear readouts of market attributes, and perturbations of the decoder inputs. The results distinguish variance concentration from information loss: although a few principal components explain most token variance, additional components improve attribute recovery and reconstruction with the frozen decoder.

D.1 Evaluation Protocol

We use the frozen 256-bit tokenizer and analyze the scaled bipolar codes $\mathbf{q}_t = \mathbf{s}_t/\sqrt{B}$ from Eq. (2). Principal component analysis (PCA) and linear readouts are fitted on a training panel and evaluated on a separate validation panel (Table D.1). Inputs follow the reconstruction-window normalization in Appendix C.1. Estimation details appear in Appendix D.6.

Table D.1: **Analysis panels and bit utilization.** Entropy is the mean marginal entropy per bit. A bit is classified as near-constant or dead when its less frequent value occurs in fewer than 5% or 1% of sampled positions, respectively.

Panel	Token Counts	Entropy (bits)	Near-constant	Dead
Training	8,192	0.859	2/256	0/256
Validation	14,432	0.865	2/256	0/256

D.2 Covariance Structure and Bit Utilization

Token variance is concentrated in a small number of principal components. The first component explains 52.6% of the variance, the first four explain 74.3%, and the first sixteen explain 89.8%. Retaining 90% and 99% of the variance requires 17 and 88 components, respectively (Figure D.1).

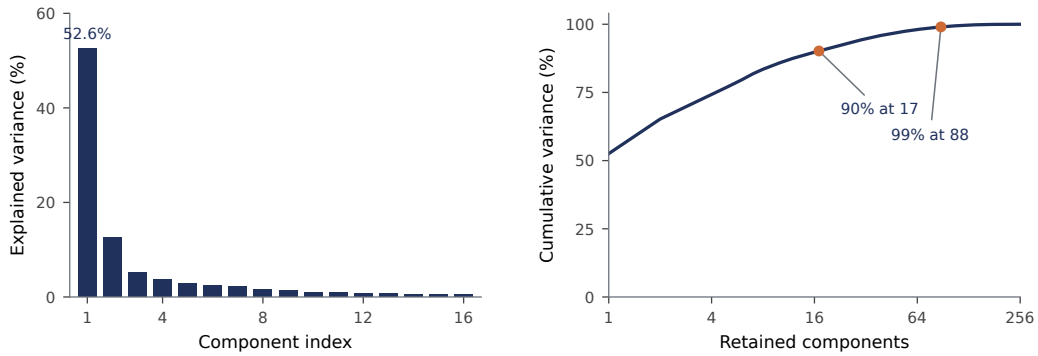


Figure D.1: **Covariance spectrum of the token representation.** Left: explained variance of the first 16 principal components. Right: cumulative explained variance across all 256 components, with the 90% and 99% thresholds marked. Both panels use the training panel.

Let $\lambda_1 \geq \cdots \geq \lambda_B \geq 0$ be the covariance eigenvalues and $\pi_i = \lambda_i / \sum_{j=1}^{B} \lambda_j$. The entropy-based effective rank of the covariance matrix (Roy & Vetterli, 2007) and its participation ratio summarize this concentration:

$$\mathrm{ER} = \exp\left(-\sum_{i=1}^{B} \pi_i \log \pi_i\right) \approx 9.0, \qquad \mathrm{PR} = \frac{1}{\sum_{i=1}^{B} \pi_i^2} \approx 3.3. \tag{D.1}$$

This spectral concentration coexists with variation in nearly every bit. On the validation panel, mean marginal entropy is 0.865 bits, only two bits are near-constant, and none meets the dead-bit threshold. The code therefore varies across many coordinates, but those variations are strongly correlated.

D.3 Market Attributes in the Principal Components

We project validation tokens onto the training-panel eigenvectors to obtain scores $\mathbf{v}_i^\top(\mathbf{q}_t - \boldsymbol{\mu})$, where $\boldsymbol{\mu}$ is the training-panel mean. Figure D.2 reports Spearman correlations between the first 16 scores and

15 descriptors of price level, trading activity, candlestick shape, and recent price history. Eigenvector signs are arbitrary, so the signs of these correlations depend on the orientation of the fitted basis.

PC1 is strongly associated with normalized close ($\rho = -0.946$). Activity is associated with PC3, which correlates with volume (-0.887) and traded amount (-0.845), and with PC6, which correlates with absolute normalized volume (-0.821). Candlestick shape appears in later components: PC9 correlates with the signed body $C_t - O_t$ (0.574), while PC7 correlates with both the body (0.464) and the range $H_t - L_t$ (0.398). Associations with the history descriptors are weaker: among the first 16 components, the largest absolute correlation is between PC1 and the 16-bar close change (-0.307).



Figure D.2: **Associations between principal components and market attributes.** Spearman correlations between the first 16 component scores and 15 descriptors on the validation panel. The PCA basis is fitted on the training panel, and rows are grouped by descriptor family.

Illustrative candlesticks. Figure D.3 shows individual observed candles from low, middle, and high score groups for PC1, PC9, and PC7. Each example is the bar closest to its group's median close, body, or range, respectively. Close decreases across the PC1 examples; the PC9 examples move from a negative to a positive body; and the high-PC7 example has a larger high–low range. Centering the shape examples at their own opening prices makes these differences easier to compare. The examples illustrate these associations in observed validation bars.

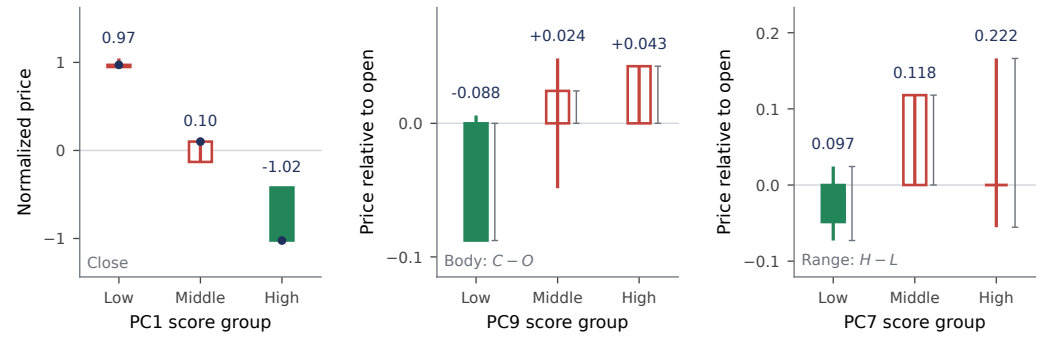


Figure D.3: **Observed candles across component-score groups.** Left: close (PC1); middle: body $C - O$ (PC9); right: range $H - L$ (PC7). Each candle is an observed bar nearest its group's median descriptor value. Numbers give descriptor values. Red hollow candles indicate $C \geq O$; green filled candles indicate $C < O$.

Non-monotonic associations. PC2 has little rank correlation with signed close ($\rho = -0.108$), but a strong correlation with its magnitude (-0.932). Its scores form a roughly inverted V as a function of normalized close (Figure D.4). Regressions fitted on the training panel quantify this relationship: a linear fit from signed close to PC2 yields validation $R^2 = -0.004$, a linear fit from $|C_t|$ yields 0.722, and a piecewise-linear fit from signed close yields 0.845. The weak rank correlation with signed close therefore masks a systematic non-monotonic association.

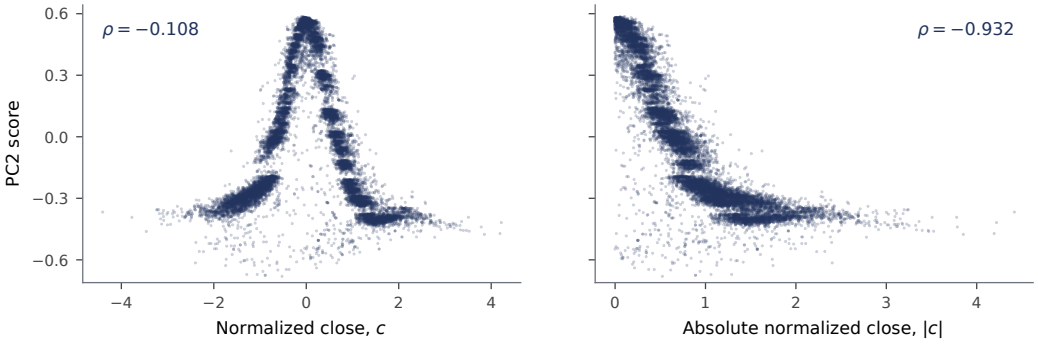


Figure D.4: **PC2 tracks the magnitude of normalized close.** PC2 scores of all validation tokens against normalized close (left) and its absolute value (right), on a shared vertical axis. Each point is one token; ρ denotes Spearman correlation.

D.4 Linear Readouts beyond the Leading Components

PCA ranks directions by their variance, not by their relevance to a regression target (Jolliffe, 1982). We therefore assess the attributes accessible to linear readouts of the frozen representation (Alain & Bengio, 2017). For each descriptor, we fit a ridge regression from the leading k component scores and report validation R^2. The primary analysis retains the original score scales and selects the regularization coefficient using only the training panel (Appendix D.6).

Mean R^2 across the 15 descriptors rises from 0.350 at $k = 8$ to 0.489 at $k = 16$, 0.576 at $k = 32$, 0.661 at $k = 128$, and 0.667 with all 256 components (Figure D.5). Components beyond the first sixteen account for only about 10% of the variance, yet substantially improve linear recovery of the measured attributes.

Table D.2 shows substantial differences across descriptor families. With all 256 components, close and volume are recovered with $R^2 = 0.984$ and 0.979, respectively. Candlestick shape is less uniformly accessible: R^2 reaches 0.879 for range and 0.813 for the signed body, but only 0.53–0.71 for wick lengths and body magnitude. History descriptors are harder to recover from a single token,

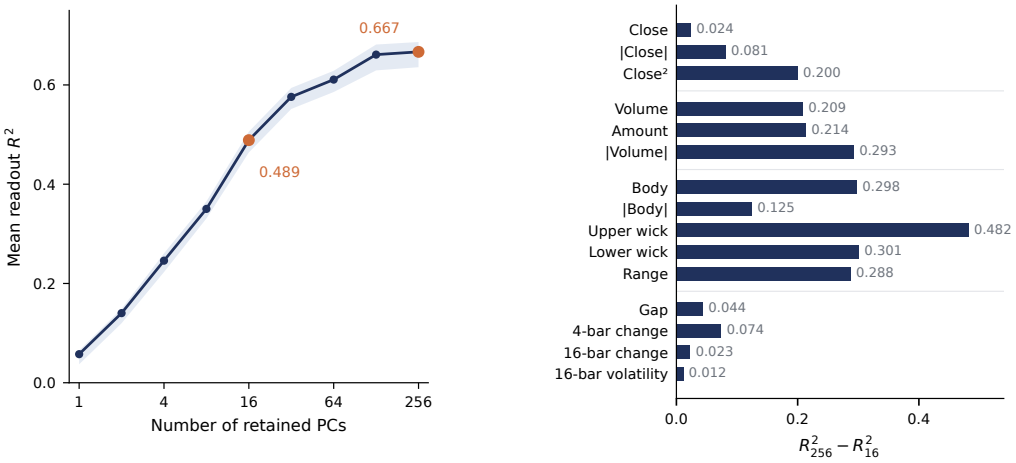


Figure D.5: **Linear readout performance as more components are retained.** Left: mean validation R^2 across 15 descriptors for ridge regressions on the leading k scores; shading denotes a bootstrap interval. Right: per-descriptor gains in R^2 from 16 to 256 components.

with R^2 of about 0.2 for the 4- and 16-bar close changes, 0.283 for trailing volatility, and 0.073 for the opening gap. These results characterize linear accessibility from one token, rather than the total historical information in the token sequence.

Table D.2: **Linear recovery of market attributes from a single token.** Validation R^2 for all 15 descriptors using the primary readout specification.

Family	Descriptor	$k = 128$	$k = 256$
Price level	Normalized close	0.984	0.984
	|Close|	0.923	0.926
	Close^2	0.862	0.871
Activity	Volume	0.976	0.979
	Amount	0.967	0.971
	|Volume|	0.953	0.956
Shape	Range	0.867	0.879
	Body	0.809	0.813
	Lower wick	0.601	0.619
	Upper wick	0.696	0.710
	|Body|	0.525	0.532
Trailing history	16-bar volatility	0.283	0.283
	16-bar close change	0.200	0.199
	4-bar close change	0.200	0.205
	Gap	0.071	0.073

D.5 Decoder Sensitivity to Code Perturbations

Linear readouts measure accessible attributes; decoder perturbations test whether reconstruction depends on directions with relatively little variance. We first report reconstruction from codes on the full validation panel, then evaluate perturbations on the 64-window subset.

Reconstruction result. Table D.3 summarizes reconstruction on validation set. OHLC MSE is 0.000777, and 98.78% of reconstructed bars satisfy OHLC ordering. The latter is encouraged by the soft penalty in Appendix C.3, rather than imposed as a hard constraint. Three-way directional agreement is 77.8% for the candle body and 85.2% for adjacent-close changes.

Table D.3: **Reconstruction result.** Mean squared errors are measured in normalized coordinates. Directional agreement treats flat bars as a third class.

Error		Structure	
Six-field MSE	0.000696	OHLC validity	98.78%
OHLC MSE	0.000777	Body direction	77.8%
Close MSE	0.000639	Adjacent-close direction	85.2%

Removing lower-variance components. For the leading eigenvectors $\mathbf{V}_k = [\mathbf{v}_1, \ldots, \mathbf{v}_k]$, we compare continuous projection and re-binarization:

$$\mathbf{q}_{t,\text{cont}}^{(k)} = \boldsymbol{\mu} + \mathbf{V}_k \mathbf{V}_k^\top (\mathbf{q}_t - \boldsymbol{\mu}),$$
$$\mathbf{q}_{t,\text{bin}}^{(k)} = \frac{2\mathbf{1}[\mathbf{q}_{t,\text{cont}}^{(k)} \geq 0] - 1}{\sqrt{B}}. \tag{D.2}$$

A matched control randomly flips the same number of bits per token as re-binarization. All three interventions modify the entire token sequence, including preceding decoder context, and are evaluated on the same 64 windows. Figure D.6 reports OHLC MSE relative to reconstruction from unmodified codes on this subset.

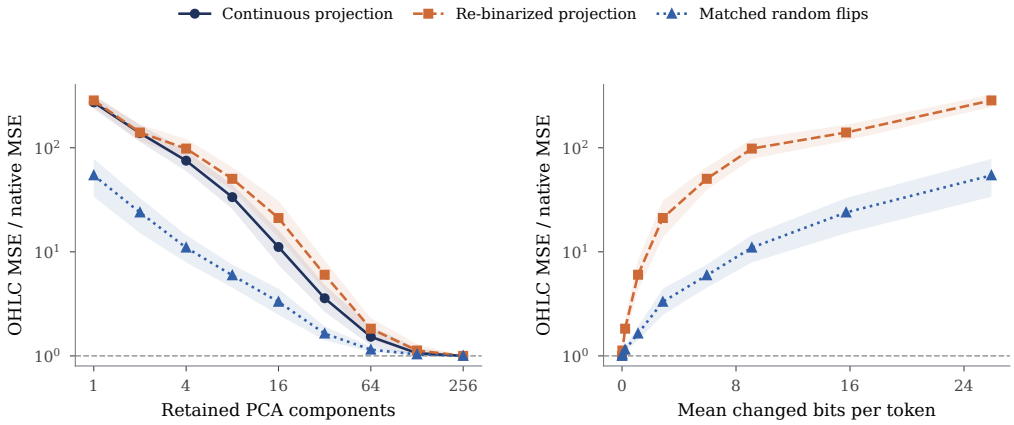


Figure D.6: **Decoder sensitivity to code perturbations.** Left: relative OHLC MSE for continuous projection, re-binarization, and random flips matched to the per-token Hamming distance, against the number of retained components. Right: the two binary interventions against the mean number of changed bits per token. The random control averages 16 repetitions; shading denotes paired source-file bootstrap intervals. The dashed line marks reconstruction from unmodified codes.

At $k = 8$, re-binarization changes 6.0 bits per token on average. Continuous projection, re-binarization, and matched random flips yield 33.5, 50.3, and 5.9 times the baseline error, respectively. At $k = 32$, the ratios are 3.6, 6.0, and 1.6; at $k = 128$, all are within about 13% of baseline.

Re-binarization does not restore the fidelity lost through projection: returning to binary-valued coordinates is insufficient to recover the original reconstruction. The gap between re-binarization and matched random flips also shows that the pattern of bit changes matters beyond their number.

OHLC validity versus reconstruction fidelity. Under random flips of 32 and 64 bits, OHLC MSE reaches about 24 and 144 times baseline, yet 92.9% and 93.4% of bars still satisfy $\widehat{H}_t \geq \max(\widehat{O}_t, \widehat{C}_t)$ and $\widehat{L}_t \leq \min(\widehat{O}_t, \widehat{C}_t)$, compared with 98.8% for unmodified codes (Figure D.7). Three-way body-direction agreement declines from 74.8% at baseline to 65.5%, 61.0%, and 53.5% at 16, 32, and 64 flips.

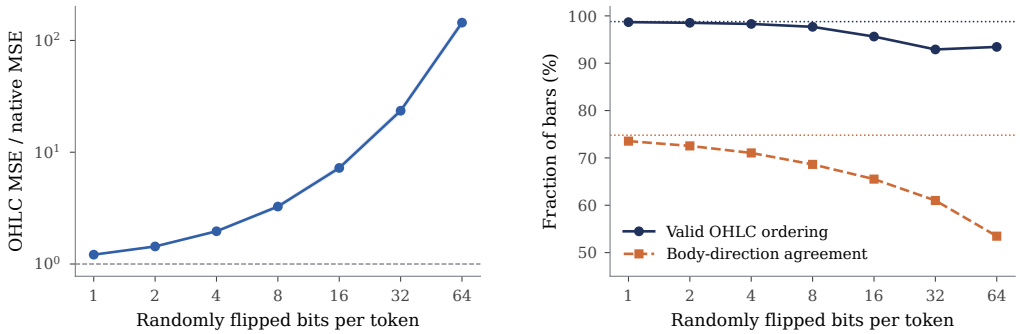


Figure D.7: **Numerical fidelity, OHLC validity, and direction under random flips.** Left: relative OHLC MSE on the eval set. Right: fraction of bars with valid OHLC ordering and fraction whose three-way body direction matches the observed bar.

D.6 Additional Analysis Details

Market descriptors. The 15 descriptors comprise price level (C_t, $|C_t|$, and C_t^2); activity (V_t, A_t, and $|V_t|$); candlestick shape (body $C_t - O_t$, its magnitude, upper wick $H_t - \max(O_t, C_t)$, lower wick $\min(O_t, C_t) - L_t$, and range $H_t - L_t$); and recent history (gap $O_t - C_{t-1}$, close changes $C_t - C_{t-4}$ and $C_t - C_{t-16}$, and the population standard deviation of the 16 adjacent close differences ending at t). All descriptors are computed in the normalized coordinates used for reconstruction.

Linear readouts. For $k \in \{1, 2, 4, 8, 16, 32, 64, 128, 256\}$, let $\mathbf{X}_k$ contain the centered training-panel scores and $\widetilde{\mathbf{Y}}$ contain targets standardized using training means and variances. With n training positions, ridge regression (Hoerl & Kennard, 1970) fits

$$\mathbf{W}_k = \arg\min_{\mathbf{W}} \left\{ \frac{1}{n} \|\mathbf{X}_k \mathbf{W} - \widetilde{\mathbf{Y}}\|_F^2 + \alpha_k \|\mathbf{W}\|_F^2 \right\}. \tag{D.3}$$

For each k, a single coefficient α_k is selected for all descriptors from $\{10^{-4}, 10^{-3}, \ldots, 10^{2}\}$ by mean R^2 on an 80/20 split of the training panel. We then refit on all training positions.

D.7 Tokenizer Design Ablations

Table D.4 extends the reconstruction and forecasting comparison in Table 3 with bit utilization and candlestick validity. MSE measures reconstruction of normalized OHLCVA inputs by each tokenizer. Composite measures the corresponding downstream predictor's forecasting performance, combining PCC, SCC, IC, and RankIC with the weights in Appendix A.2. The same appendix defines bit entropy, dead and near-constant bits, and exact candlestick validity.

Table D.4: **Complete tokenizer design ablations.** Effects of token width, auxiliary losses, and quantizer choice on reconstruction, bit utilization, candlestick validity, and forecasting.

	Reconstruction	Bit utilization			Valid candles	Forecasting
Tokenizer	MSE ↓	Entropy (bits) ↑	Dead ↓	Near-const. ↓	Exact (%) ↑	Composite ↑
LFQ-64	1.759	**0.896**	0	2	98.24	0.0397
LFQ-128	1.030	0.793	0	**1**	97.35	0.0432
LFQ-256	0.571	0.828	0	7	96.32	**0.0582**
LFQ-512	**0.449**	0.607	9	102	93.71	0.0491
LFQ-256 w/o aux.	0.813	0.572	5	56	92.72	0.0536
BSQ-256	5.750	0.171	167	199	**99.22**	–

MSE is in units of 10^{-3}. "w/o aux." removes bit-balance, grouped-entropy, and OHLC-ordering losses. Composite uses ODE16, $K = 3$. Metric definitions are in Appendix A.2; shading, bold, and underlining follow Table 1.

Increasing LFQ width from 256 to 512 bits reduces reconstruction MSE, but lowers forecasting performance, mean bit entropy, and candle validity. Removing the auxiliary losses from LFQ-256 increases the number of dead bits from 0 to 5 and near-constant bits from 7 to 56. BSQ-256 has the highest candle validity, but also 167 dead bits and roughly ten times the reconstruction MSE of LFQ-256.

E PREDICTOR REPRESENTATIONS AND FORECASTING

Return probes, flow-head retraining, and historical-descriptor probes examine how predictive information in the representation evolves during pretraining. Probes measure recoverable information (Alain & Bengio, 2017), which need not be the information actually used by the generator (Belinkov, 2022). We therefore distinguish readout performance from end-to-end autoregressive forecasting.

Evaluation setting. We use the 95K-update reference checkpoint unless stated otherwise and average scores equally over 62 market–frequency tasks. PCC and SCC are Pearson and Spearman correlations between predicted and realized price paths; IC and RankIC are pooled Pearson and Spearman correlations between predicted and realized terminal returns (Appendix A.2). Forecasts use Gaussian initialization and a 16-step ODE sampler, with $K = 3$ endpoint samples averaged in continuous space before token thresholding. Probe details appear in Appendix E.5.

E.1 PREDICTIVE READOUTS ACROSS LAYERS AND CHECKPOINTS

Return probes and native forecasts follow different training trends. We fit ridge probes to the final history-position state at the input embedding, all eight Transformer layers, and the final RMSNorm output. Figure E.1 compares checkpoints at 40K, 95K, 140K, and 200K updates and requested horizons $H \in \{1, 8, 32, 96\}$. The best readout layer depends on the horizon. Later layers are generally stronger at short horizons, whereas the embedding gives the highest probe IC at requested $H = 32$ and $H = 96$. At $H = 8$, layer-7 probe IC increases from 0.088 at 95K to 0.094 at 140K and 0.100 at 200K, while native forecast IC decreases from 0.092 at 95K to 0.087 at 200K. Thus, the decline in native forecasting is not accompanied by a uniform decline in linearly recoverable return information.

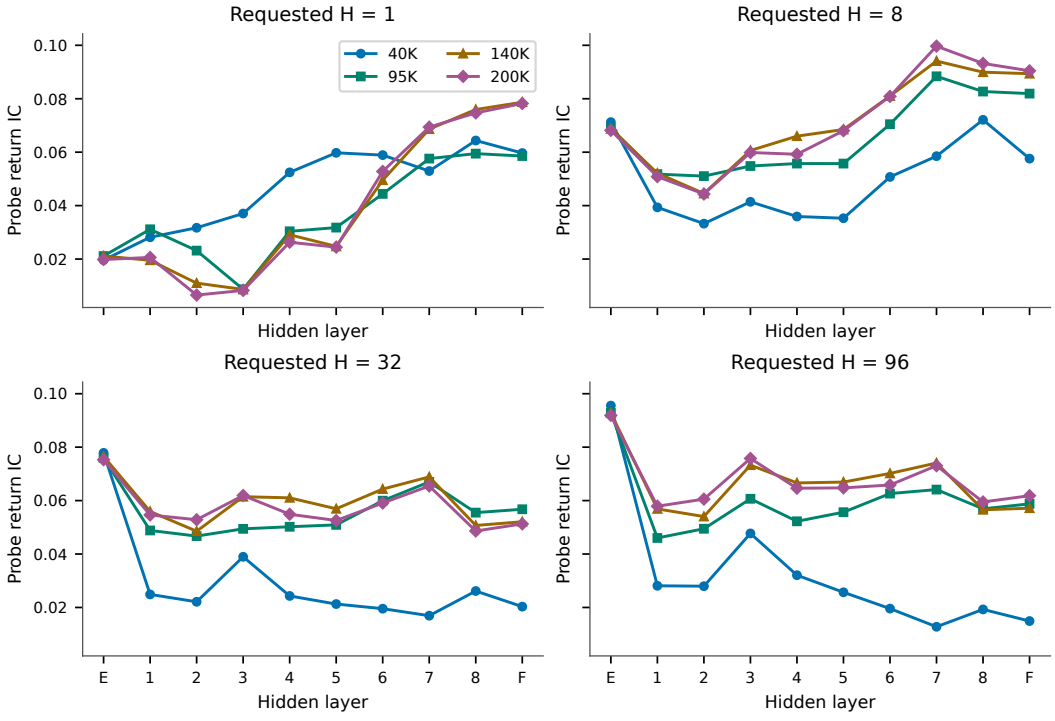


Figure E.1: **Return-probe IC by layer, checkpoint, and requested horizon.** E denotes the input embedding and F the final RMSNorm output. Each point is an equal-weight mean over 62 tasks. The effective horizon is $\min(H, H_a)$ for task a; consequently, the $H = 96$ panel uses each task's terminal return, not a common 96-bar target. The task-horizon distribution is given in Appendix E.5.

Later backbones support better readouts under a fixed retraining budget. We replace the flow head with a newly initialized bounded MLP and train it for 5K updates while freezing the rest of the model. All four backbones use the same head initialization, data order, noise, and optimization settings. The retrained head's forecast IC increases from 0.061 on the 95K backbone to 0.076 on the 200K backbone (Figure E.2). Useful predictive information therefore remains accessible to a newly trained head at later checkpoints. However, all four retrained heads remain below their native counterparts: the experiment shows an improvement under a fixed readout-training budget, not a recovery of native forecasting performance through head replacement.

Representation similarity and readout transfer measure different properties. Linear centered kernel alignment (CKA) (Kornblith et al., 2019) between the final states at 95K and later checkpoints remains high: 0.994 at 140K and 0.993 at 200K. Yet a terminal-return probe fitted at 95K performs worse at those checkpoints than a probe refitted there (Figure E.2). This is compatible with CKA's invariance to orthogonal changes of feature coordinates: similar geometry does not imply that a fixed set of regression coefficients will transfer.

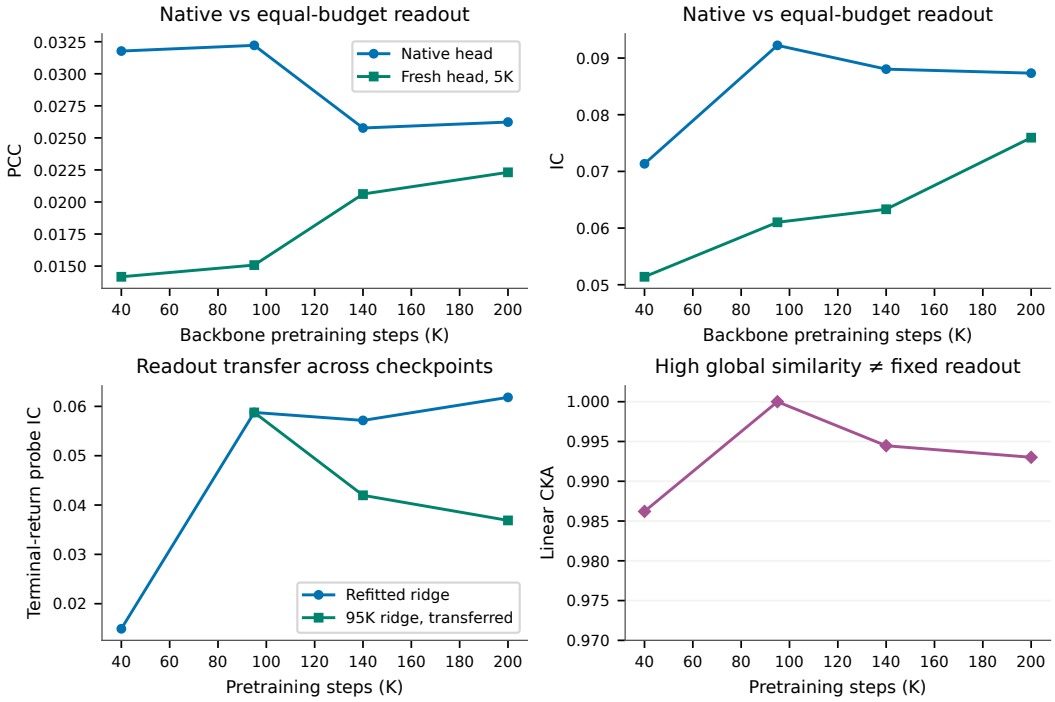


Figure E.2: **Forecasting performance, readout transfer, and representation similarity across checkpoints.** Top: complete forecasts from the native flow head and a new head trained for 5K updates on the same frozen backbone. Bottom left: terminal-return probes refitted at each checkpoint versus the fixed 95K probe. Bottom right: linear CKA of final states relative to 95K, computed on the same 1,024 windows. Connected points are checkpoints, not independent runs.

E.2 Historical Structure in Hidden States

Sequence processing improves the recovery of historical descriptors. We probe 35 descriptors spanning trend, momentum, reversals, oscillation, directional runs, volatility, candlestick geometry, and serial dependence. These descriptors summarize price behavior within the observed history. Probe quality is measured by within-task Spearman correlation ρ_S, averaged over tasks. At 95K, the final state improves on the embedding for full-window slope (0.880 versus 0.687) and close-increment autocorrelation (0.511 versus 0.139). Figure E.3 shows the layerwise pattern for six descriptor families. Historical readouts remain strong at the 200K and 600K checkpoints; for example, slope correlation reaches 0.891 and 0.902, respectively.

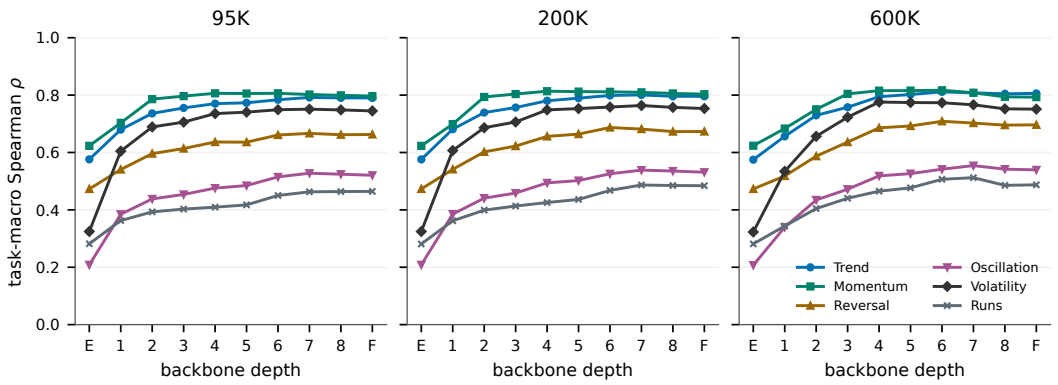


Figure E.3: **Historical-descriptor probes across layers.** Each curve averages within-task Spearman correlations over tasks and then over the descriptors in one family. Six of the eight families are shown. E and F denote the embedding and final RMSNorm output.

Descriptor readouts improve over static controls. A strong descriptor probe may exploit readily available endpoint features rather than sequence structure. We therefore compare it with a static baseline comprising eight features of the terminal bar and its position within the history, expanded to 44 linear and quadratic terms. We also fit a conditional readout after regressing both the hidden state and the descriptor on these controls. Its held-out R^2 gain over the static prediction, denoted ΔR^2, measures the additional predictive value of this residual readout; the construction is defined in Appendix E.5.

Table E.1 shows gains for slope, momentum, autocorrelation, and relative volatility. The gains are not uniform across descriptors: longest upward and downward runs have $\Delta R^2 = -0.614$ and -0.387, respectively. Figure E.4 summarizes the family-level results. The representations support several historical summaries beyond the chosen controls, but this regularized readout does not improve every aspect of path structure.

Table E.1: **Historical-descriptor readouts at 95K.** The middle columns compare separate static and final-state ridge probes using task-mean Spearman correlation. The last column reports the conditional readout's task-mean R^2 gain over its static baseline; it is not the difference between the two correlation columns.

Descriptor	Static ρ_S	Hidden ρ_S	Conditional ΔR^2
Full-window slope	0.742	0.880	0.240
Quarter-window momentum	0.758	0.820	0.079
Close-increment autocorrelation	0.133	0.511	0.250
Relative volatility	0.404	0.861	0.583

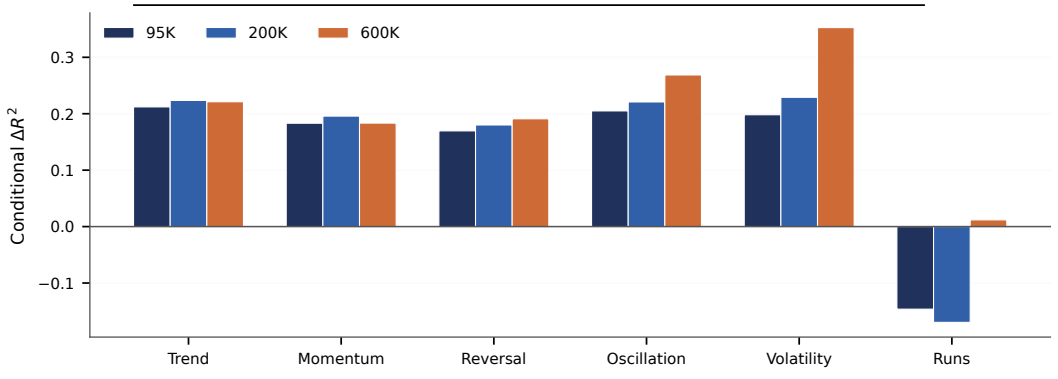


Figure E.4: **Incremental descriptor prediction after controlling for static features.** Bars average held-out ΔR^2 across tasks and descriptors within each displayed family. Negative values mean that adding the residual hidden-state readout worsens prediction.

Different descriptors rely on different principal subspaces. We refit probes in selected PCA subspaces to measure the predictive content of each subspace (Jolliffe, 1982). For the 95K final state, trend-family correlation is 0.790 in the full 512-dimensional space, 0.613 in the first four components, and 0.706 in the first 32. Removing the first four components reduces it to 0.438. The same removal has little effect on candlestick descriptors (0.603 to 0.600), but lowers volatility-family correlation from 0.745 to 0.616. The leading components are therefore important for these trend and volatility readouts, without accounting for all recoverable historical structure.

Path readouts capture shape but miss intermediate detail. Probes of 16 evenly spaced normalized close values recover the last historical coordinate almost perfectly ($\rho_S = 0.995$ at both 95K and 600K), but intermediate coordinates are substantially less accurate (Figure E.5). For five-way path-shape classification, balanced accuracy is 0.445 at 95K and 0.460 at 600K, compared with 0.394 for the static baseline. A complementary test uses 457 matched pairs of opposite-shape histories. Pair-ordering accuracy is 82.0% at 95K and 89.2% at 600K, versus 85.9% for the static baseline. Thus, the 95K representation improves overall shape classification but not this more controlled pairwise comparison. Figure E.7 illustrates the recovered paths and classification errors; these are probe reconstructions of observed histories, not generated future trajectories.

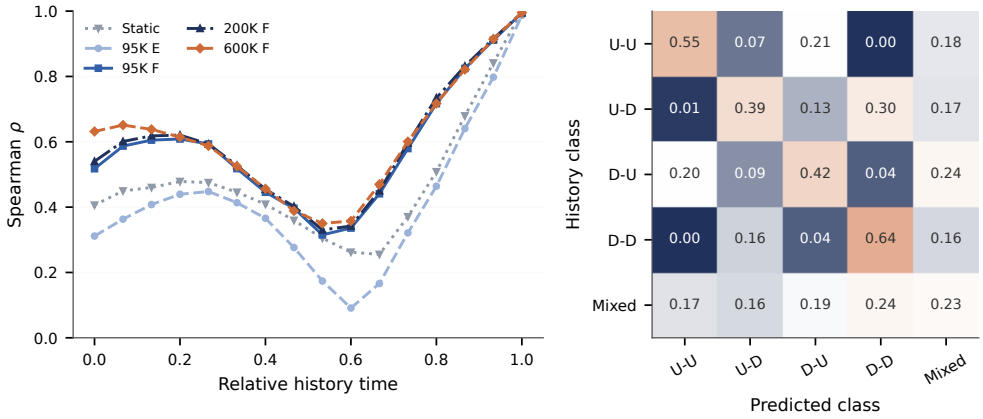


Figure E.5: **Historical-path readouts.** Left: task-mean Spearman correlation at 16 positions in the observed history. Right: row-normalized confusion matrix for the 95K final-state shape probe. U and D indicate positive and negative half-window slopes; Mixed denotes the low-slope category. Balanced accuracy is 0.445, compared with 0.394 for the static baseline.

Descriptor readouts transfer across market groups. We fit probes on ten of eleven market/exchange groups and evaluate on the remaining group, repeating this for every group. At 95K, slope and relative-volatility correlations are 0.875 and 0.851, close to the corresponding results under the standard probe split (0.880 and 0.861). At 600K, the corresponding correlations are 0.899 and 0.852 (Figure E.6).

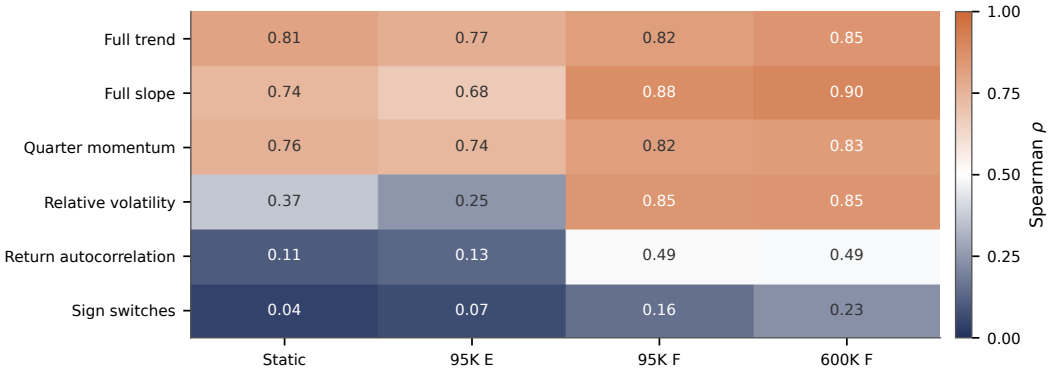


Figure E.6: **Leave-one-market-group-out descriptor prediction.** Entries are equal-weight task means of held-out Spearman correlations for static features, 95K embedding and final states, and 600K final states.

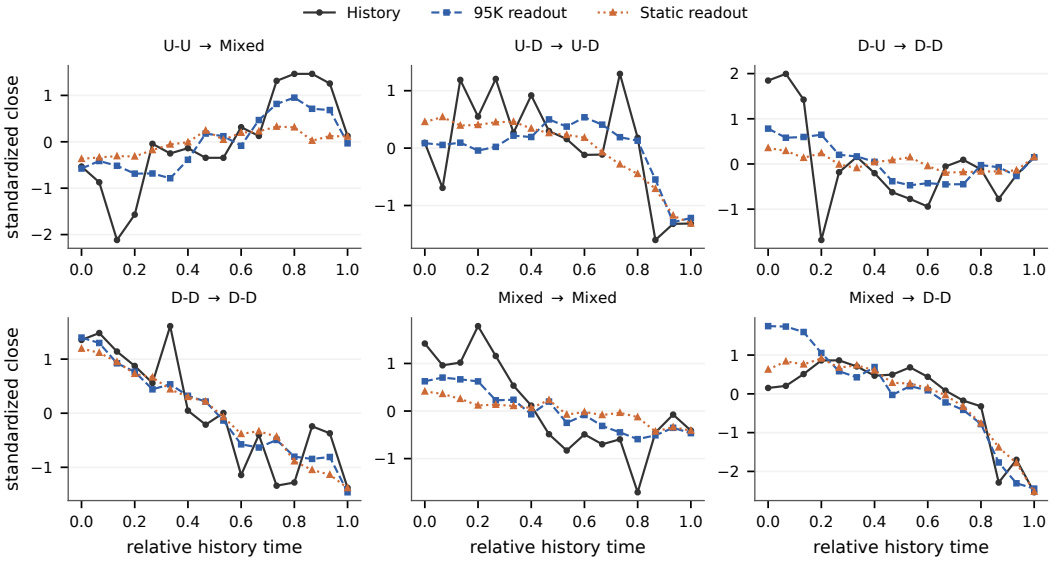


Figure E.7: **Historical paths reconstructed by ridge probes at 16 sampled positions.** Black curves show observed histories; blue and orange curves show 95K final-state and static-feature readouts. The first five cases are selected by within-class path representativeness; the sixth is the 95K probe's most confident classification error. Panel titles give the true and predicted shape classes. Lines interpolate probe outputs; they are not outputs of the market decoder.

E.3 Forecasting Heads and Training Variants

Endpoint, bitwise, and velocity prediction. We compare the flow head with a bitwise Bernoulli head trained by binary cross-entropy (BCE) and a head that predicts velocity directly. Each model is trained for 100K updates and evaluated at its minimum-validation-loss checkpoint. The BCE sampler takes a majority vote over three Bernoulli draws per bit. The flow models instead average three continuous endpoints before thresholding.

Flow matching learns a vector field along a noise-to-data path (Lipman et al., 2022). The clean-endpoint head converts its token estimate to a velocity using a_τ as defined in Appendix C.5:

$$\widehat{\mathbf{v}}_\theta^{\mathrm{end}}(\mathbf{y}_\tau, \tau, \mathbf{c}_t) = \frac{f_\theta(\mathbf{y}_\tau, \tau, \mathbf{c}_t) - \mathbf{y}_\tau}{a_\tau}. \tag{E.1}$$

The direct-velocity head outputs the vector field without this endpoint parameterization. Table E.2 shows that endpoint prediction exceeds the BCE head on all four reported correlations. Direct velocity

prediction retains return IC but has much lower price correlations; it achieves the highest RankIC. These runs favor endpoint prediction for price-path forecasting, not uniformly across all metrics.

Table E.2: **Forecasting-head comparison using minimum-validation-loss checkpoints selected from 100K-update runs.** All scores are equal-weight means over 62 tasks. Bold marks the largest value in each column.

Head	PCC	SCC	IC	RankIC
Endpoint flow	**0.0322**	**0.0301**	**0.0922**	0.0673
Bitwise BCE	0.0287	0.0265	0.0729	0.0664
Direct velocity	0.0013	0.0010	0.0867	**0.0715**

Other configuration comparisons. We also compare the reference with attention over backbone layers, auxiliary return regression at horizons $\{1, 8, H_a\}$, and a flow head with adaptive layer normalization (AdaLN; Peebles & Xie, 2023). None exceeds the reference on PCC or IC (Table E.3).

Table E.3: **Other configuration comparisons.** Each run is trained for 100K updates and evaluated at its minimum-validation-loss checkpoint.

Configuration	PCC	IC
Reference flow	0.0322	0.0922
Cross-layer attention	0.0257	0.0848
Auxiliary return supervision	0.0218	0.0805
AdaLN flow head	0.0159	0.0660

E.4 Summary

Forecast IC can decline while selected return probes and newly trained heads improve. The same hidden states retain historical structure beyond static controls, although the advantage depends on the descriptor and readout task. These findings motivate evaluating the forecasting head and rollout procedure alongside representation quality.

E.5 Experimental Protocols

Data and reporting. Each task contributes 64 windows for probe fitting and 64 for evaluation, giving 3,968 windows in each split. The two sets are disjoint. Probe scores are computed within tasks and then averaged equally over the 62 tasks.

Return probes and head retraining. Return probes use ridge regression (Hoerl & Kennard, 1970) with $\alpha = 10$. For task a, window i, and requested horizon H, the target is

$$r_{a,i}^{\mathrm{probe}}(H) = \log\left(\frac{C_{a,i,h_a(H)}}{C_{a,i,0}}\right), \qquad h_a(H) = \min(H, H_a), \tag{E.2}$$

where $C_{a,i,0}$ is the last observed close and subsequent closes are in market units. This log-return probe target is distinct from the simple-return target used by the full-forecast IC. The task horizons H_a are 12, 18, 24, 32, 48, and 96 bars, with 29, 2, 11, 2, 9, and 9 tasks, respectively. Fresh-head experiments train only the bounded-MLP flow head for 5K updates, with batch size 128, learning rate 8×10^{-4}, and weight decay 8×10^{-6}. The tokenizer, input and calendar-time embeddings, backbone, final normalization, and auxiliary modules remain frozen. Initialization, data order, and noise are shared across backbones.

Historical probes and controls. Descriptor probes share ridge coefficients across tasks after fitting-set task-conditional centering. We select $\alpha \in \{1, 10, 100, 1000\}$ using a 48/16 internal split and standardized MSE. The eight descriptor families cover full-window trend; recent momentum; turning points; crossing counts and oscillation amplitude; directional run lengths; price-increment variation; OHLC range and body statistics; and autocorrelation and acceleration. The eight static features are normalized terminal OHLC, the final close's position within the historical range, the last close increment, candle body, and high–low range. Including all second-order monomials gives 44 features.

The standalone static probe tunes its regularization, whereas the static ridge in the conditional analysis uses $\alpha = 100$.

For a descriptor d_i, hidden state $\mathbf{h}_i$, and static features $\mathbf{q}_i$, let $g(\mathbf{q}_i)$ predict the descriptor and $m(\mathbf{q}_i)$ predict the hidden state from the controls. A residual ridge u predicts $d_i - g(\mathbf{q}_i)$ from $\mathbf{h}_i - m(\mathbf{q}_i)$. The resulting predictor and reported gain are

$$\widehat{d}_i^{\,\mathrm{cond}} = g(\mathbf{q}_i) + u(\mathbf{h}_i - m(\mathbf{q}_i))\,, \tag{E.3}$$

$$\Delta R^2 = \frac{1}{|\mathcal{A}|} \sum_{a \in \mathcal{A}} \left[R_a^2(\widehat{\mathbf{d}}_a^{\,\mathrm{cond}}) - R_a^2(\widehat{\mathbf{d}}_a^{\,\mathrm{static}}) \right], \tag{E.4}$$

where $\widehat{d}_i^{\,\mathrm{static}} = g(\mathbf{q}_i)$, $\mathcal{A}$ is the set of 62 tasks, and R_a^2 is computed on task a's evaluation windows against the observed descriptor values. All residualization maps are fitted on the fitting split. Family-level scores average the resulting descriptor scores. A within-task label-permutation control over three seeds gives mean $\rho_{\mathrm{S}} \approx -0.013$.

Subspaces and market transfer. PCA is fitted on the 3,968 fitting windows after mean centering, without coordinate-wise standardization. We compare the full 512-dimensional state with components 1–4, components 1–32, and components 5–512. Market-transfer probes are fitted on ten market groups and evaluated on the remaining group in each of eleven splits; predictions are aggregated by task.

Path readouts and matched pairs. Ridge probes predict 16 normalized close coordinates evenly spaced over the historical window. The five shape categories—up–up, up–down, down–up, down–down, and mixed—use the two half-window slopes, with slope-magnitude threshold 0.35 for the mixed category. Case studies select a path at the median distance to its class-mean path for each true class, plus the 95K probe's most confident error. For the pairwise test, greedy within-task matching pairs up–down and down–up histories without window reuse, controlling for final-close z-score, close position within the historical range, net trend, and relative roughness. The maximum standardized root-mean-square matching distance is one. Pair ordering compares the probes' up–down versus down–up class-score margins. Accuracy is averaged within tasks and then across tasks; it is distinct from single-window balanced accuracy.

F REFLO: ADVANTAGES, REWARDS, AND OPTIMIZATION

ReFLO uses negative group-relative advantages to construct token-level targets that steer predictions away from low-reward generated endpoints. This appendix details advantage construction, token re-evaluation, the learning objective, financial rewards, and optimization settings.

F.1 GROUP-RELATIVE ADVANTAGES AND TOKEN PROBES

Candidate rewards. For a batch of P histories $\mathcal{H}_i$, the current generator independently samples G continuations per history. Decoding them with the frozen tokenizer yields candidate trajectories $\widehat{\mathbf{Y}}_i^{(g)}$ of length H. A reward evaluator R assigns a scalar score R_{ig} to each candidate by comparing $\widehat{\mathbf{Y}}_i^{(g)}$ with the realized future $\mathbf{Y}_i$. The evaluator may score trajectories directly or assign candidate-level credit from panel statistics. Scores are held fixed during optimization.

Negative-advantage weighting. We standardize rewards within each history to obtain group-relative advantages (Shao et al., 2024):

$$\overline{R}_i = \frac{1}{G}\sum_{g=1}^{G} R_{ig}, \qquad \sigma_i^2 = \frac{1}{G}\sum_{g=1}^{G}(R_{ig}-\overline{R}_i)^2, \qquad A_{ig} = \frac{R_{ig}-\overline{R}_i}{\sigma_i+\epsilon}, \tag{F.1}$$

Here $\overline{R}_i$ and σ_i are the group mean and population standard deviation, and $\epsilon > 0$ ensures numerical stability. ReFLO retains only the negative part:

$$\omega_{ig}^{-} = \min(A_{ig}, 0). \tag{F.2}$$

Candidates scoring below the group mean receive repulsive updates weighted by the magnitude of their negative advantages; candidates at or above the mean contribute zero gradient when the targets are constructed. Groups with nonfinite rewards or $\sigma_i < \epsilon$ are inactive.

Token probes. For each candidate, we sample Q future positions uniformly without replacement (Table F.1). At each position j, we independently draw $\tau \sim \mathcal{U}(0.1, 0.9)$ and $\boldsymbol{\epsilon} \sim \mathcal{N}(0, I_B)$, then form the noised latent and recompute its clean prediction under the candidate's own generated prefix:

$$\begin{aligned} \mathbf{y}_{igj,\tau} &= (1-\tau)\boldsymbol{\epsilon} + \tau \mathbf{e}_{igj}, \\ \widehat{\mathbf{s}}_{\theta,igj} &= f_\theta(\mathbf{y}_{igj,\tau}, \tau, h_\theta(\mathcal{H}_i, \mathbf{b}_{i,<j}^{(g)})). \end{aligned} \tag{F.3}$$

Here $\mathbf{e}_{igj} = \mathbf{s}_{i,j}^{(g)}$ is the candidate's bipolar endpoint at position j. Rollouts are collected without gradients; sampled tokens remain fixed while the backbone and flow head are re-evaluated. This forward-process approach, used in DiffusionNFT and RVM (Zheng et al., 2026; Choi et al., 2026), supplies predictor gradients without differentiating through sampling or the frozen tokenizer.

F.2 REPULSIVE REGRESSION AND ITS LOCAL GRADIENT

Bounded target displacement. The regression target in Eq. (10) displaces the detached prediction $\bar{\mathbf{s}}$ along $\omega^-(\mathbf{e} - \bar{\mathbf{s}})$. Define $\mathrm{RMS}(\mathbf{v}) = \|\mathbf{v}\|_2/\sqrt{B}$. The radial clipping operator is

$$\mathrm{Clip}_{\mathrm{RMS},c}(\mathbf{v}) = \frac{\mathbf{v}}{\max\{1, \mathrm{RMS}(\mathbf{v})/c\}}. \tag{F.4}$$

Clipping constrains each target's displacement from the current prediction while preserving direction. The target may lie outside $[-1, 1]^B$; the parameter update itself is controlled separately through gradient clipping and backtracking (Section F.4).

Time weighting and batch objective. Under the clean-endpoint parameterization of the linear flow path, squared velocity error scales as $(1-\tau)^{-2}$ relative to squared endpoint error (Lipman et al., 2022). We cap the weight at 25 and rescale it:

$$w(\tau) = \frac{\min\{(1-\tau)^{-2}, 25\}}{3.125}. \tag{F.5}$$

We average the weighted squared error over histories, candidates, token positions, and coordinates:

$$\mathcal{L}_{\mathrm{ReFLO}}(\theta) = \frac{1}{PGQB} \sum_{i=1}^{P} \sum_{g=1}^{G} \sum_{j \in \mathcal{J}_{ig}} w(\tau_{igj}) \left\| \widehat{\mathbf{s}}_{\theta,igj} - \mathbf{s}^{\star}_{igj} \right\|_2^2, \tag{F.6}$$

where $\mathcal{J}_{ig}$ indexes the Q probed positions for candidate (i, g). Zero-signal candidates remain in the averaging denominator.

Local repulsive gradient. At the parameters used to construct the target, $\widehat{\mathbf{s}}_\theta - \mathbf{s}^\star = -\mathbf{d}$, so the gradient of the per-probe loss is

$$\frac{\partial \ell_{\mathrm{ReFLO}}}{\partial \widehat{\mathbf{s}}_\theta} = -\frac{2w(\tau)}{B}\mathbf{d}, \qquad \nabla_\theta \ell_{\mathrm{ReFLO}} = -\frac{2w(\tau)}{B} J_\theta^\top \mathbf{d}, \tag{F.7}$$

where J_θ is the Jacobian of the endpoint prediction, including its recomputed condition. For negative signals, $\mathbf{d}$ points away from the generated endpoint, providing repulsive supervision. Without stop-gradient, the unclipped squared error reduces to $\eta^2(\omega^-)^2\|\widehat{\mathbf{s}}_\theta - \mathbf{e}\|_2^2$, whose minimization attracts predictions toward rejected endpoints.

F.3 Financial Reward Functions

ReFLO decouples the financial objective from the update mechanism. A task-specific evaluator assigns scalar rewards to generated continuations, either directly or through their contribution to a panel-level metric. Replacing this evaluator adapts the same pretrained predictor to a different objective while preserving the group-relative advantage construction and token-update rule. The evaluator need not be differentiable, and the update requires no policy likelihood evaluations. Below, we define rewards for price forecasting, return prediction, and volatility estimation.

Price-path reward. Let $\widehat{\mathbf{y}}_{i,f}^{(g)}$ and $\mathbf{y}_{i,f}$ denote predicted and realized paths for OHLC field f. For each candidate, we compute temporal Pearson and Spearman correlations (Appendix A.2) over fields with sufficient valid observations, then average:

$$\begin{aligned} R_{ig}^{\mathrm{PCC}} &= \frac{1}{|\mathcal{F}_{ig}|} \sum_{f \in \mathcal{F}_{ig}} \rho\Big(\widehat{\mathbf{y}}_{i,f}^{(g)}, \mathbf{y}_{i,f}\Big), \\ R_{ig}^{\mathrm{SCC}} &= \frac{1}{|\mathcal{F}_{ig}|} \sum_{f \in \mathcal{F}_{ig}} \rho\Big(\mathrm{rank}(\widehat{\mathbf{y}}_{i,f}^{(g)}), \mathrm{rank}(\mathbf{y}_{i,f})\Big), \\ R_{ig}^{\mathrm{price}} &= \frac{1}{2}(R_{ig}^{\mathrm{PCC}} + R_{ig}^{\mathrm{SCC}}). \end{aligned} \tag{F.8}$$

Here $\mathcal{F}_{ig}$ contains fields with at least two paired finite values and nonzero variance in both paths. Ranks use paired valid observations, with average ranks for ties. When no field is valid, both components are zero.

Return-prediction reward. Terminal returns $\widehat{r}_i^{(g)} = \widehat{C}_{i,H}^{(g)}/C_{i,0} - 1$ and $r_i = C_{i,H}/C_{i,0} - 1$ are compared across histories using panel-level correlations, where $C_{i,0}$ is the last observed close. We assign candidate credit through a marginal Pearson component and a rank component. Let $\chi(r) = \mathrm{clip}(r, -1, 1)$ and form panel anchors $\mathbf{a} = (a_1, \ldots, a_P)$ by averaging valid clipped predictions within each history. Let $\mathbf{r}_\chi = (\chi(r_1), \ldots, \chi(r_P))$. Replacing history i's anchor with candidate g's clipped return yields the marginal credit

$$\Delta_{ig} = P[\rho(\mathbf{a}^{(i,g)}, \mathbf{r}_\chi) - \rho(\mathbf{a}, \mathbf{r}_\chi)], \tag{F.9}$$

where $\mathbf{a}^{(i,g)}$ equals $\mathbf{a}$ with entry i replaced by $\chi(\widehat{r}_i^{(g)})$. Correlations use valid entries, and invalid candidates receive zero credit. The rank component uses average ranks over valid histories for each candidate index g:

$$R_{ig}^{\mathrm{rank}} = \widehat{z}_{ig} z_{ig}, \qquad R_{ig}^{\mathrm{ret}} = \frac{1}{2}[3\tanh(\Delta_{ig}/3) + R_{ig}^{\mathrm{rank}}]. \tag{F.10}$$

Here $\widehat{z}_{ig}$ and z_{ig} are average ranks of the unclipped predicted and realized returns, centered and scaled to unit population variance over the same valid histories. Invalid entries and degenerate rank vectors contribute zero. The bounded transformation limits the influence of large Pearson credits. Averaging R_{ig}^{rank} over valid histories within each candidate index recovers the panel's Spearman correlation.

Volatility reward. We compute predicted and realized horizon variances using the Yang–Zhang construction in Eq. (A.8) (Yang & Zhang, 2000) and compare them using the nonnegative QLIKE loss in Eq. (A.9) (Patton, 2011). Writing this loss for candidate (i, g) as $\ell_{\mathrm{QLIKE},ig}$, the reward is

$$R_{ig}^{\mathrm{vol}} = -\log(1 + \ell_{\mathrm{QLIKE},ig}) - 2\nu_{ig}, \tag{F.11}$$

where ν_{ig} is the larger of the invalid OHLC-entry fractions in the predicted and realized paths. An entry is invalid if it is nonfinite or nonpositive, measured before the price safeguards described in Appendix A.2. The logarithm compresses large QLIKE losses, while the second term penalizes invalid prices.

F.4 Optimization and Training Configuration

Training mixture and gradient aggregation. We sample N_{cyc} panels from each of the $|\mathcal{A}| = 62$ market–frequency settings per update. Gradients are computed at the same predictor parameters and averaged:

$$\overline{\mathbf{g}} = \frac{1}{N_{\mathrm{cyc}}|\mathcal{A}|} \sum_{m=1}^{N_{\mathrm{cyc}}} \sum_{a \in \mathcal{A}} \nabla_\theta \mathcal{L}_{\mathrm{ReFLO}}^{(a,m)}. \tag{F.12}$$

Each market–frequency setting receives equal weight in gradient aggregation. Return credits are computed within each panel before aggregation.

Update control. We clip the accumulated gradient norm and form an SGD update, whose scale is reduced by backtracking against a fixed anchor bank of predictor evaluations. This update-level control is distinct from the per-probe RMS cap in Eq. (F.4): the latter bounds a regression-target displacement, whereas the former controls the accepted predictor update.

Shared configuration. Post-training initializes from the pretrained predictor and retains its frozen tokenizer. Forecasting adaptation updates the full predictor; the investment experiment updates the backbone as specified in Appendix I.2. Table F.1 summarizes the candidate-generation and regression settings. Candidate collection uses one endpoint draw per autoregressive step ($K = 1$), thresholds it, and extends that candidate's own prefix. Thus, G counts complete continuations, K counts within-token endpoint draws, and Q counts training probes. These sampling budgets are distinct from the multi-path aggregation used for evaluation.

Table F.1: Candidate-generation and regression settings for ReFLO.

Component	Setting
Histories per panel; candidates per history	$P = 32$; $G = 16$
Market–frequency settings; panels per setting	$\lvert\mathcal{A}\rvert = 62$; $N_{\mathrm{cyc}} = 4$ per update
Context fraction; forecast-length cap	0.75; 96 tokens
Candidate sampler	Gaussian source; 16 deterministic Euler steps
Within-token endpoint draws	$K = 1$
Advantage weighting	Group standardization; $\omega_{ig}^{-} = \min(A_{ig}, 0)$
Probes per candidate	$Q = \min(32, H)$
Probe times; corruption noise	$\mathcal{U}[0.1, 0.9]$; $\mathcal{N}(0, I_B)$
Target step; displacement cap	$\eta = 0.05$; $c = 0.2$
Time weight	$w(\tau) = \min\{(1-\tau)^{-2}, 25\}/3.125$

G SCALING THE FLOW PREDICTOR

We study how the flow predictor's validation loss scales with model size and the number of training tokens. Across six predictor sizes and five training-token budgets, larger models and longer training consistently reduce loss. A joint power-law fit captures both trends over the measured range.

G.1 MODEL FAMILY AND EVALUATION

We evaluate six predictors ranging from 9.51M to 327.39M parameters (Table G.1). Each combines a causal Transformer backbone with a four-layer MLP flow head that predicts bounded endpoints. All models use rotary position embeddings, SwiGLU feed-forward layers, Fourier time embeddings, and no dropout. The parameter count P includes all trainable predictor modules and excludes the shared frozen tokenizer.

Each model is evaluated after 1B, 3B, 10B, 15B, and 20B training tokens. We denote the cumulative number of input tokens processed by D, counting repeated observations each time they are used. Each occurrence of a candlestick token counts once, irrespective of its bit width or the number of flow-time draws.

Table G.1: **Predictor configurations.** All models share a frozen 256-bit tokenizer. The flow head has four layers, and the attention-head dimension is 64 throughout. Parameter counts exclude the tokenizer.

Parameters (M)	Backbone width	Backbone layers	Attention heads	FFN dimension	Flow-head width
9.51	320	4	5	1280	640
20.61	384	7	6	1536	768
40.46	512	8	8	2048	1024
82.53	640	11	10	2560	1280
161.08	832	13	13	3328	1664
327.39	1024	18	16	4096	2048

Validation metric. We measure the mean squared error between the model's velocity field and the target velocity of the linear interpolation path (Lipman et al., 2022). For an observed next token $\mathbf{s}_{t+1}$ and its history condition $\mathbf{c}_t$, we construct $\mathbf{y}_\tau$ and predict $\widehat{\mathbf{s}}_\theta$ as in Eq. (5), with $\boldsymbol{\epsilon} \sim \mathcal{N}(0, I_B)$ and $\tau \sim \mathcal{U}(0,1)$. Using a_τ from Appendix C.5, the predicted velocity is

$$\widehat{\mathbf{v}}_\theta = \frac{\widehat{\mathbf{s}}_\theta - \mathbf{y}_\tau}{a_\tau}, \tag{G.1}$$

and the validation loss is

$$\mathcal{L}_{\text{val}} = \mathbb{E}_{(\mathbf{s}_{t+1}, \mathbf{c}_t), \tau, \boldsymbol{\epsilon}} \left[\frac{1}{B} \left\| \widehat{\mathbf{v}}_\theta - (\mathbf{s}_{t+1} - \boldsymbol{\epsilon}) \right\|_2^2 \right]. \tag{G.2}$$

G.2 JOINT SCALING WITH PARAMETERS AND TRAINING TOKENS

Recent work reports predictable loss scaling for flow-matching image generators (Chickering et al., 2026). To characterize the dependence on model size and training tokens in our setting, we fit the additive power-law form of Hoffmann et al. (2022):

$$\widehat{\mathcal{L}}_{\text{val}}(P, D) = E + A_P \left(\frac{P}{10^7} \right)^{-\alpha} + A_D \left(\frac{D}{10^9} \right)^{-\beta}. \tag{G.3}$$

Here E is a fitted offset, and the two power-law terms describe the dependence on parameter count and cumulative training tokens. We estimate all five coefficients jointly by nonlinear least squares on the validation losses, with equal weight assigned to each model–budget pair. The fitted relation is

$$\widehat{\mathcal{L}}_{\text{val}}(P, D) = 0.11555 + 0.02445 \left(\frac{P}{10^7} \right)^{-0.451} + 0.25061 \left(\frac{D}{10^9} \right)^{-1.022}. \tag{G.4}$$

The in-sample root mean squared error is 5.27×10^{-4}. Figure G.1 shows the observations and the corresponding slices of this joint fit.

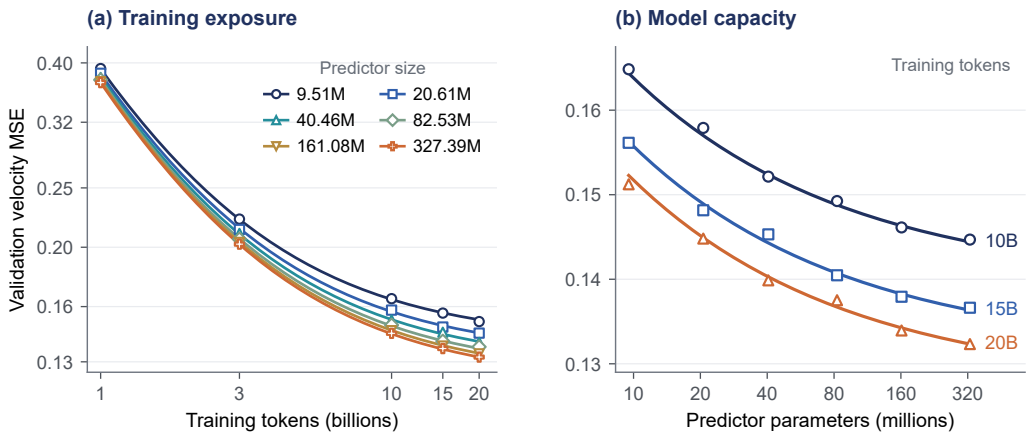


Figure G.1: **Validation loss decreases with model size and training tokens.** Markers show measured velocity MSE under the evaluation protocol; curves are slices of the joint fit in Eq. (G.4). (a) Loss versus training tokens for all six predictors, on log–log axes. (b) Loss versus parameter count at 10B, 15B, and 20B training tokens, with a logarithmic parameter axis and a linear loss axis.

Training tokens. Every predictor improves across the evaluated token budgets, with the largest absolute reductions occurring early in training. For the 327.39M model, loss falls from 0.37141 at 1B tokens to 0.20233 at 3B tokens and 0.13233 at 20B tokens. The fitted training-token term scales as $D^{-1.02}$, describing continued improvement with diminishing absolute gains.

Model size. Larger predictors achieve lower loss at every matched training-token budget. At 20B tokens, increasing model size from 9.51M to 327.39M reduces loss from 0.15123 to 0.13233, a 12.5% reduction. Increasing model size from 161.08M to 327.39M yields a smaller reduction of 0.00161, or 1.2%. These diminishing absolute gains are consistent with the fitted parameter-dependent term, which scales as $P^{-0.45}$.

Within the measured range, both larger predictors and longer training improve next-token denoising. The joint fit summarizes their effects on validation loss under a common tokenization and evaluation protocol.

H FINANCIAL FORECASTING: BENCHMARK AND EVALUATION PROTOCOLS

This appendix specifies the benchmark, baseline training and inference protocols, and FiRE/ReFLO configurations for the forecasting results in Table 1. The investment protocol is given separately in Appendix I.

H.1 BENCHMARK

The benchmark spans equities, cryptocurrency, and foreign exchange. Its equity component covers Shanghai (XSHG), NASDAQ (XNAS), Tokyo (XJPX), India (XNSE), Korea (XKRX), Hong Kong (XHKG), Indonesia (XIDX), Malaysia (XKLS), and Taiwan (XTAI), each at six frequencies. Table H.1 specifies their context and forecast lengths. Each setting receives equal weight in the aggregate results.

Table H.1: Context and forecast lengths, in bars.

Frequency	5-min	10-min	15-min	20-min	40-min	60-min	2-hour	4-hour	Daily
Context T_a	480	240	160	120	90	80	60	90	40
Horizon H_a	96	48	32	24	24	12	12	18	12
Settings	9	9	2	2	9	11	9	2	9

All models forecast OHLC from historical bars. Multivariate models receive the six OHLCVA channels; univariate models forecast the four price channels independently.

H.2 MODELS TRAINED FROM SCRATCH

We train DLinear (Zeng et al., 2023), iTransformer (Liu et al., 2024), FACT (Wang et al., 2026), SEER (Qiu et al., 2026), TimeFilter (Hu et al., 2025), DUET (Qiu et al., 2025), and SRSNet (Wu et al., 2025) separately for each market–frequency setting. For each model and setting, we select the checkpoint with the lowest validation loss across two capacity configurations.

Data and normalization. Training targets precede January 1, 2024, and validation targets span January 1–May 31, 2024. Normalization uses context statistics only. OHLC shares one mean and standard deviation across time and price fields; volume and amount are normalized separately.

Training objective. Following the baseline objective in Kronos (Shi et al., 2025, Appendix D.2), we combine mean squared error with temporal Pearson correlation. For a batch of N_b windows, let $\widetilde{\mathbf{y}}_{i,f} \in \mathbb{R}^{H_a}$ denote a normalized future price channel and $\widehat{\widetilde{\mathbf{y}}}_{i,f}$ its prediction. The shared objective is

$$\mathcal{L}_{\text{fit}} = \frac{1}{4N_b} \sum_{i=1}^{N_b} \sum_{f=1}^{4} \left[\frac{\|\widehat{\widetilde{\mathbf{y}}}_{i,f} - \widetilde{\mathbf{y}}_{i,f}\|_2^2}{H_a} - \lambda_{\text{corr}}\, \rho_\epsilon\Big(\widehat{\widetilde{\mathbf{y}}}_{i,f}, \widetilde{\mathbf{y}}_{i,f}\Big) \right]. \tag{H.1}$$

We use $\lambda_{\text{corr}} = 4$. SEER, TimeFilter, and DUET add their implementation's auxiliary losses during training. Validation uses only Eq. (H.1).

Optimization. Table H.2 summarizes the shared optimizer. Both capacity configurations receive the same training budget within each setting. We train baselines at most 12 epochs with early-stopping patience 3.

Table H.2: Shared optimization settings for models trained from scratch.

Component	Setting
Optimizer	Adam; $(\beta_1, \beta_2) = (0.9, 0.999)$; no weight decay
Learning rate	5×10^{-4}, constant
Batch size	256

Architecture configurations. Table H.3 lists the two capacity choices. Where applicable, common defaults are dropout 0.1, GELU, model width 256/512, feedforward width 512/1024, depth 2/3, and

Table H.3: Capacity configurations for models trained from scratch. Parameter ranges include prediction heads and cover both capacities across all context and horizon lengths.

Model	Smaller / larger configuration	Parameters (M)
DLinear	Shared / channel-specific linear projections	0.001–0.554
iTransformer	Width 128/256; depth 3/5; heads 4/8	0.404–2.784
FACT	Width 256/512; dilation lists of length 3/5	1.787–11.664
SEER	Width 256/512; depth 2/3; heads 4/8	1.982–11.644
TimeFilter	Width 256/512; depth 2/3; heads 4/8	0.694–5.456
DUET	Width 256/512; depth 2/3; heads 4/8	1.165–8.635
SRSNet	Hidden width 128/256	0.024–1.523

4/8 attention heads. Patch length is 8 for $T_a = 40$ and 16 otherwise. Paired values below follow the smaller/larger order in the table.

iTransformer uses feedforward width 256/512. FACT uses four kernels, core ratio 0.5, and dilation sequences $(1, 2, 1)$ / $(1, 2, 3, 2, 1)$. SEER uses four experts with top-1 routing, threshold 0.2, score width equal to model width, hidden width 256, half-patch stride, and head dropout 0.1. TimeFilter uses $\alpha = 0.1$ and top-$p = 0.5$; its patch length is reduced to $\gcd(T_a, \text{patch length})$ when required for exact tiling. DUET uses four experts with top-1 routing, a moving-average window of 25, fully connected dropout 0.2, and channel-independent expert processing. SRSNet uses patch-length stride, head dropout 0.1, affine normalization, and linear mode with $\alpha = 2$.

H.3 Pretrained Baselines

We evaluate six general-purpose models—Chronos-2 (Ansari et al., 2025), TimesFM 3.0 (Google Research, 2026), TiRex-2 (Podest et al., 2026), Toto 2.0 (Khwaja et al., 2026), Sundial (Liu et al., 2025c), and Moirai 2.0 (Liu et al., 2025a)—and two financial foundation models: FinCast (Zhu et al., 2025b) and Kronos Base (Shi et al., 2025). All use released weights without task-specific fine-tuning. Table H.4 identifies the evaluated checkpoints.

Table H.4: Pretrained baseline checkpoints and forecast readouts. Multivariate models use OHLCVA; univariate models forecast OHLC separately. Sampled readouts average 32 paths for price and return prediction and use one path for volatility prediction.

Model	Checkpoint	Input	Readout
Chronos-2	`amazon/chronos-2`	Multivariate	Median
TimesFM 3.0	`google/timesfm-3.0-pytorch`	Multivariate	Point forecast
TiRex-2	`NX-AI/TiRex-2`	Multivariate	Median
Toto 2.0	`Datadog/Toto-2.0-2.5b`	Multivariate	Median
Sundial	`thuml/sundial-base-128m`	Univariate	Sampled
Moirai 2.0	`Salesforce/moirai-2.0-R-small`	Multivariate	Median
FinCast	`Vincent05R/FinCast` (`v1.pth`)	Univariate	Mean
Kronos Base	`NeoQuasar/Kronos-base`	Multivariate	Sampled

Inference settings. Chronos-2 disables cross-series learning. TimesFM disables symmetric averaging and forced positivity. TiRex-2 and Moirai use no external covariates. Toto groups all six channels together and uses masked left padding to its patch size. FinCast uses four experts with top-2 routing, masked context padding to a multiple of 32, and no optional normalization or positional embeddings. Sundial uses 50 flow steps. Kronos uses `Kronos-Tokenizer-base`, temperature 0.6, top-$p = 0.9$, top-$k = 0$, input clipping at 5, and maximum context 512.

Forecast readout. Sundial and Kronos average 32 sampled trajectories for price and return prediction and use one trajectory for volatility. DUET averages 32 forward passes through its stochastic graph for price and return, and uses one pass for volatility. The remaining models use the point, median, or mean readouts listed in Table H.4.

H.4 FiRE AND ReFLO

Evaluated checkpoint. We evaluate the 40.46M-parameter FiRE predictor with a frozen 256-bit tokenizer. The backbone and flow head are described in Appendices C.4 and C.5. The training run uses batch size 128, learning rate 8×10^{-4}.

For the price and return columns of Table 1, FiRE uses ODE16 from $\tau = 0$ with $K = 100$ endpoint samples and $N = 1$ trajectory. Volatility prediction uses $K = N = 1$. The sampling and aggregation rules are defined in Appendix C.6.

Sampling comparison. Table H.5 compares the main-table configuration ($K = 100$, $N = 1$) with trajectory averaging ($K = 1$, $N = 32$), using the same pretrained checkpoint and ODE16 from $\tau = 0$. Trajectory averaging yields higher PCC, SCC, and RankIC, while the main-table configuration yields higher IC. Both exceed the best baseline for each metric.

Within-token averaging reuses the Transformer context, whereas each complete trajectory requires its own autoregressive continuation. Our timing measurements place the cost of $K = 100$ at roughly two to three times that of $K = 1$. The main table uses $K = 100$, $N = 1$ to balance forecast quality and inference time.

Table H.5: **Forecasting with within-token and trajectory averaging.** Both FiRE rows use the same pretrained checkpoint and ODE16 sampler. Bold and underlined values indicate the best and second-best results in each column.

Model / sampling	Price series		Returns	
	PCC ↑	SCC ↑	IC ↑	RankIC ↑
Best Baseline	0.0468	0.0442	0.0752	0.0703
FiRE ($K = 100$, $N = 1$; main table)	<u>0.0491</u>	<u>0.0450</u>	**0.1085**	<u>0.0853</u>
FiRE ($K = 1$, $N = 32$)	**0.0566**	**0.0533**	<u>0.0934</u>	**0.0903**

Best Baseline is the best score per metric in Table 1: Kronos Base for PCC/SCC, Moirai 2.0 for IC, and DUET for RankIC. K denotes within-token endpoint samples; N denotes complete forecast trajectories.

Task-specific post-training. Price, return, and volatility adaptation start from the same pretrained checkpoint and update the full predictor with the tokenizer frozen. The price reward averages temporal PCC and SCC; the return reward combines Pearson and rank-based candidate credits; the volatility reward uses a log-transformed QLIKE loss with an invalid-price penalty. Appendix F.3 defines these rewards.

For each parameter update, we sample four batches from each of the 62 market–frequency settings, yielding 248 batches in total. Each batch, referred to as a panel, contains 32 historical input windows from the same setting. Conditioned on each window, the model independently generates 16 candidate future trajectories. Each update therefore uses 7,936 historical windows ($62 \times 4 \times 32$) and 126,976 candidate trajectories ($7{,}936 \times 16$). Gradients from all 248 panels are computed with the model parameters held fixed and averaged before a single parameter update.

I QLIB BACKTESTING PROTOCOL AND ECONOMIC EVALUATION

Following the investment evaluation in Kronos (Shi et al., 2025), we use Qlib (Yang et al., 2020) to assess the portfolio performance of forecast-derived stock rankings after transaction costs. This appendix details the signals, ranking-reward adaptation, and trading protocol used in Section 5.2.

I.1 MARKET DATA AND SIGNAL CONSTRUCTION

Universe and chronology. We evaluate portfolios over July 1, 2024–December 31, 2025 (368 trading days), using point-in-time CSI1000 constituents as the investment universe and the CSI1000 index (SH000852) as the benchmark. Signals dated t use observations through that day's close and determine orders for the open of $t+1$. All models use the same 90-bar lookback window of daily data as input. Throughout this appendix, date offsets count trading sessions, and $O_{t,i}$ and $C_{t,i}$ denote the observed opening and closing prices of stock i, respectively. Hats denote forecasts made at time t.

Supervised baselines. For this experiment, the seven baselines are trained from scratch to predict the five-day open-to-open return,

$$y_{t,i}^{(5)} = \frac{O_{t+6,i}}{O_{t+1,i}} - 1. \tag{I.1}$$

Training covers January 2011–February 2024 and validation covers March–May 2024. For each architecture, we select the configuration in Table I.1 with the lowest validation return MSE. Training runs for at most 12 epochs, with early stopping after three epochs without improvement. DLinear and SRSNet receive closing prices; the remaining models receive all OHLCVA fields.

Table I.1: Supervised baseline configurations for the Qlib backtest. Paired values denote the smaller / larger setting; dashes indicate inapplicable fields.

Model	Layers	d_{model}	d_{ff}	Heads
iTransformer	3/5	128/256	256/512	4/8
FACT	3/5	256/512	512/1024	–
SEER	2/3	256/512	512/1024	4/8
TimeFilter	2/3	256/512	512/1024	4/8
DUET	2/3	256/512	512/1024	4/8
SRSNet	–	256/512	–	–
DLinear	`individual`: False / True			

d_{model} and d_{ff} denote model and feed-forward widths. FACT layers count convolutional blocks. SRSNet also uses hidden width 128/256. DLinear compares shared and channel-specific linear maps.

Pretrained baselines. Sundial, TimesFM, and FinCast forecast closing prices from the historical close series; their five-day ranking score is $\widehat{C}_{t+5,i}/C_{t,i} - 1$. Chronos-2, Moirai, Toto, TiRex-2, and Kronos Base receive all OHLCVA fields and forecast six bars, from which we compute $\widehat{O}_{t+6,i}/\widehat{O}_{t+1,i} - 1$. Appendix H describes the baseline models.

FiRE signals. FiRE forecasts 31 daily bars and uses the implied 30-session open-to-open return as its ranking score:

$$s_{t,i} = \frac{\widehat{O}_{t+31,i}}{\widehat{O}_{t+1,i}} - 1. \tag{I.2}$$

Both FiRE checkpoints generate one autoregressive trajectory, averaging $K = 100$ continuous endpoint samples before thresholding at each token step (Appendix C.6). Sampling uses 16 ODE steps. The signal horizons reflect model-specific empirical behavior. In exploratory comparisons, the baselines performed better with shorter-horizon signals aligned with the portfolio's holding and replacement timescale, whereas FiRE benefited from a longer readout. The latter may better capture persistent relative price movements in the generated trajectories, making stock rankings less sensitive to short-term fluctuations. The forecast horizon nevertheless does not determine the holding period: all signals are refreshed daily and evaluated under identical portfolio constraints, execution rules, and transaction costs. The comparison thus assesses model-specific signal constructions within a common trading protocol, with horizon selection forming part of the evaluated configuration.

I.2 RANKING-REWARD POST-TRAINING

For the investment experiment, we adapt FiRE with a cross-sectional ranking reward. Each sampled date provides a cross-section of 100 stocks, with $G = 16$ candidate trajectories per stock. For candidate group g and future day offset d, we compute predicted and realized one-session returns:

$$\widehat{r}^{(g)}_{t,i,d} = \frac{\widehat{O}^{(g)}_{t+d+1,i}}{\widehat{O}^{(g)}_{t+d,i}} - 1, \qquad r_{t,i,d} = \frac{O_{t+d+1,i}}{O_{t+d,i}} - 1. \tag{I.3}$$

The reward at each offset is the Spearman correlation between predicted and realized returns across stocks. The ten offsets $d = 1, \ldots, 10$ receive equal weight, with candidate-level credits computed separately at each offset. Credits are standardized within each stock's candidate group, and their negative advantages weight the repulsive updates. Flow probes are placed at the corresponding return endpoints.

I.3 PORTFOLIO CONSTRUCTION AND RETURN ACCOUNTING

Portfolio and execution. A daily top-k/drop-n strategy ranks stocks by descending signal value and targets 50 long positions with a 95% investment budget. At each rebalance, at most 20 holdings are replaced ($n_{\text{drop}} = 20$), subject to a minimum holding period of three trading days. Both buy and sell orders execute at the next session's open. Transaction costs are 5 basis points for buys and 10 basis points for sells, with a minimum charge of CNY 5 per order. The simulator applies Qlib's tradability checks and a uniform 9.5% price-limit threshold. These settings are identical across models, and portfolio returns are recorded daily regardless of the signal horizon.

Performance metrics. Let r_t be the daily portfolio return before transaction costs, c_t the cost as a fraction of portfolio value, and b_t the daily index return. The net portfolio return is $\widetilde{r}_t = r_t - c_t$, and the net excess return is $e_t = \widetilde{r}_t - b_t$. Using 252 trading days per year, we report annualized excess return (AER) and excess Sharpe ratio (ESR):

$$\text{AER} = 252\,\overline{e}, \qquad \text{ESR} = \sqrt{252}\,\frac{\overline{e}}{\text{sd}(e_t)}, \tag{I.4}$$

where $\overline{e}$ and $\text{sd}(e_t)$ are the sample mean and sample standard deviation of daily net excess returns. AER uses arithmetic annualization, while ESR is the information ratio relative to CSI1000.

I.4 BACKTEST RESULTS

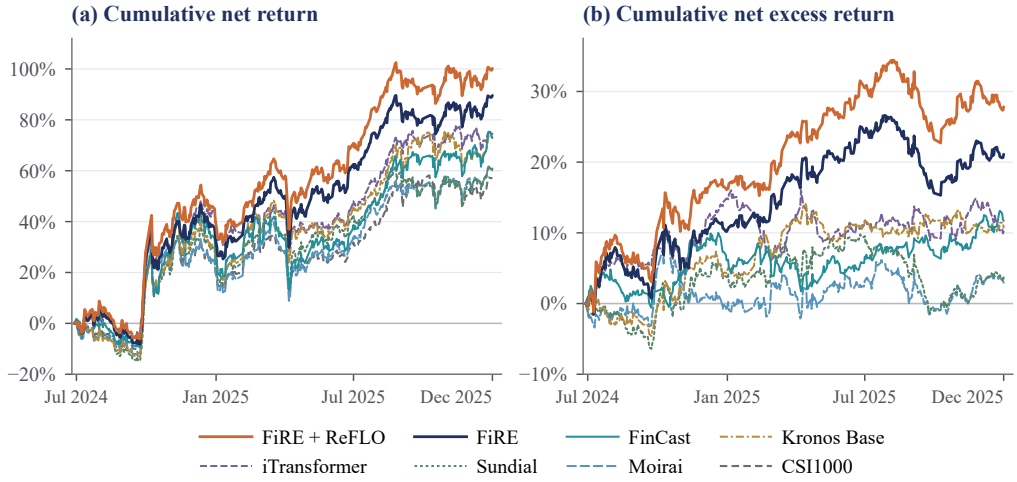


Figure I.1: **CSI1000 investment simulation.** (a) Cumulative portfolio returns after transaction costs, with CSI1000 as the benchmark. (b) Cumulative daily net excess returns relative to CSI1000. For readability, the figure shows the seven models with positive AER; Table 1 reports all 17 baselines.

Table 1 reports the full model comparison. Figure I.1 complements these annualized statistics with the cumulative return paths. FiRE and FiRE + RL end the period with net portfolio returns of 89.6% and 100.2%, respectively, compared with 57.1% for CSI1000.

J SYNTHETIC CANDLESTICK GENERATION: ADDITIONAL VISUALIZATIONS

Figures J.1–J.3 extend Figure 5 to the remaining seven market–frequency settings. These comparisons examine the distributional agreement between real and synthetic candlestick sequences, complementing the fidelity and predictive-utility results in Table 2.

J.1 GENERATION AND EVALUATION PROTOCOL

Generation. FiRE samples one continuation per historical context using Gaussian initialization, 16 ODE steps, and one flow sample per token ($K = 1$). The context and generation lengths are $(T, H) = (120, 96)$ for 15-minute data and $(40, 35)$ for daily data.

Predictive utility. The train-on-synthetic, test-on-real (TSTR) evaluation (Yoon et al., 2019; Shi et al., 2025) uses a two-layer LSTM trained on synthetic trajectories and evaluated on real trajectories. It predicts the final 16 bars from the preceding 80 for 15-minute data, and the final 5 bars from the preceding 30 for daily data. We compute temporal Pearson IC and Spearman RankIC for each OHLC channel, average over valid sample–channel pairs, and weight the eight settings equally.

J.2 DISTRIBUTIONAL COMPARISONS

Following Kronos (Shi et al., 2025), we compare real and synthetic sequences using t-SNE (van der Maaten & Hinton, 2008) and kernel density estimation (KDE). Each comparison uses 2,000 real and 2,000 synthetic trajectories, with each channel min–max normalized within its trajectory. For each model, t-SNE is fitted jointly to the flattened real and synthetic trajectories. Gaussian KDEs pool normalized values across time and channels.

Across the seven settings, FiRE's samples show substantial overlap with real data in the t-SNE projections. FiRE also reproduces the main features of the normalized density profiles, including the pronounced low-value peak in Taiwan's 15-minute data and the flatter profiles in foreign exchange. For 15-minute cryptocurrency data, the synthetic density is higher near the boundaries and lower in the interior than the real density.

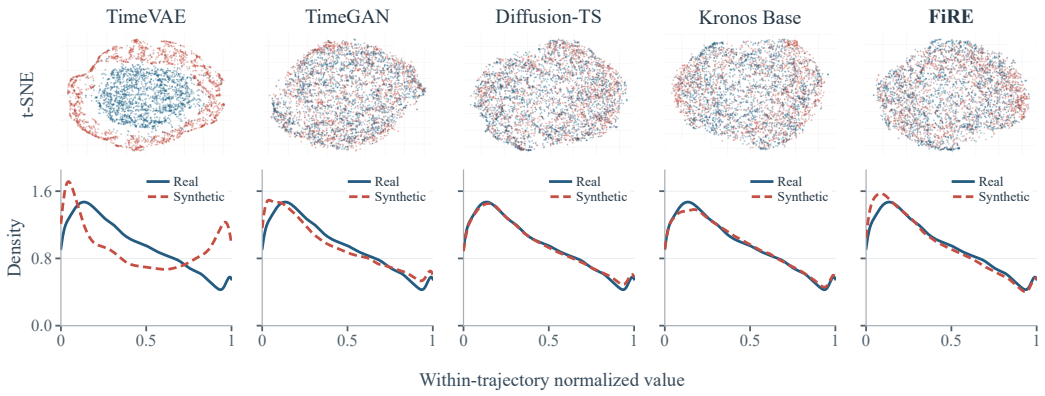
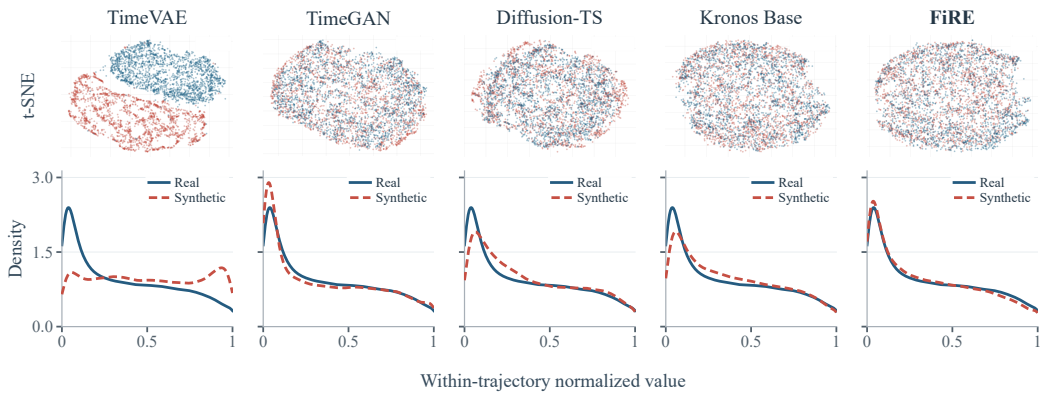
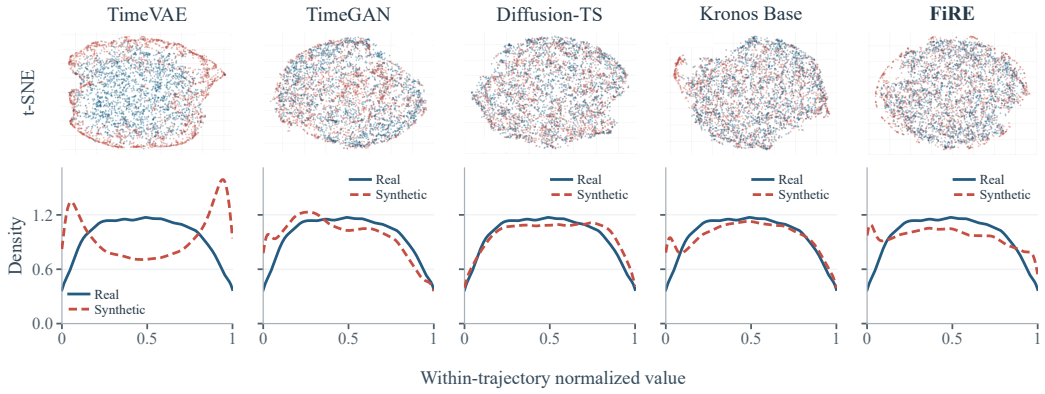


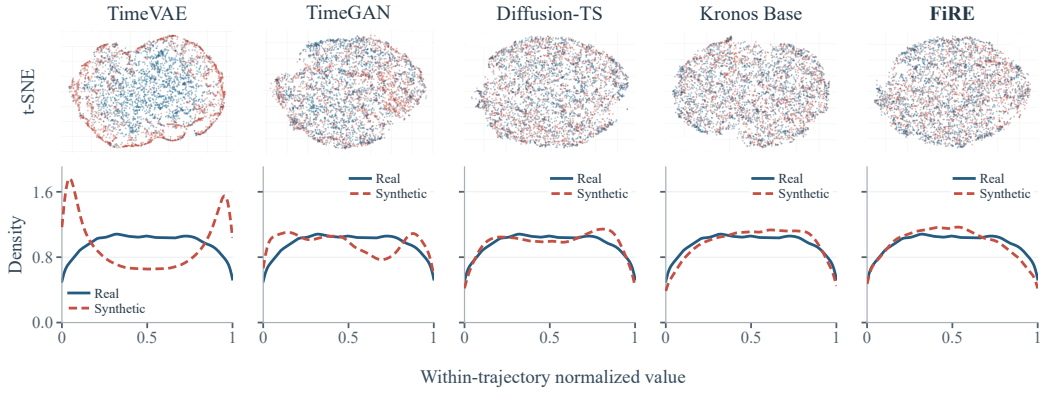
Figure J.1: **Shanghai stock exchange, daily bars.** Each column compares real (blue) and synthetic (red) data for one model. Top: t-SNE projections fitted jointly to real and synthetic trajectories. Bottom: KDEs of within-trajectory, per-channel normalized values.



(a) Taiwan equities, 15-minute bars

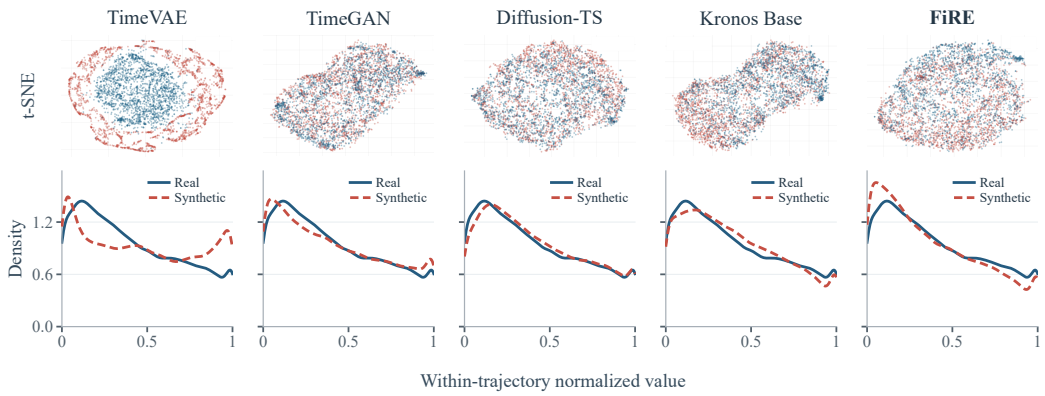


(b) Cryptocurrency, 15-minute bars

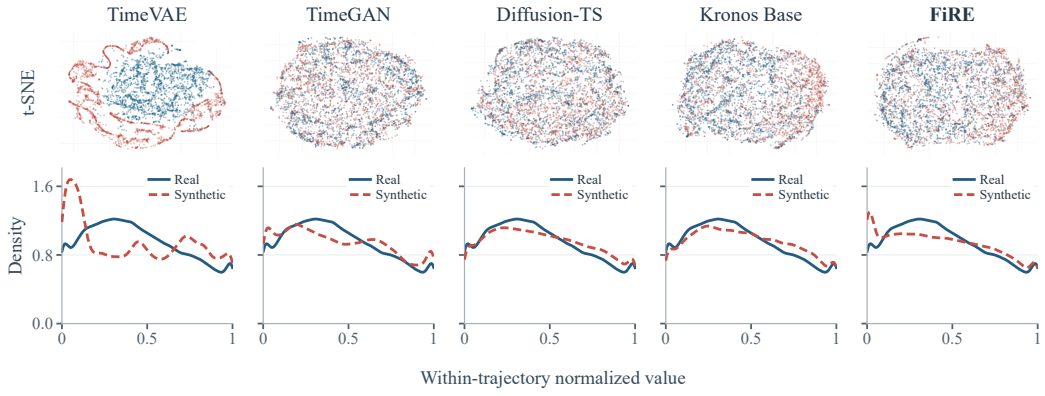


(c) Foreign exchange, 15-minute bars

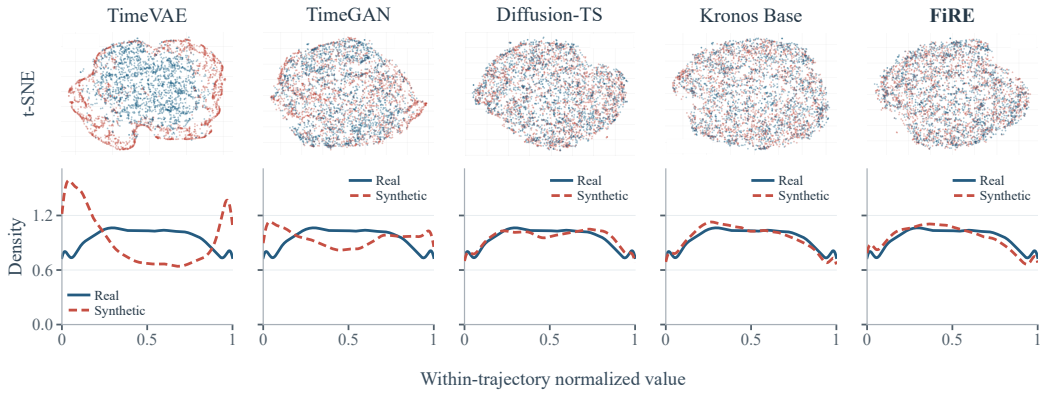
Figure J.2: **Additional 15-minute settings.** Distributional comparisons for (a) Taiwan stock exchange, (b) cryptocurrency, and (c) foreign exchange. Each group shows t-SNE projections (top) and KDEs of within-trajectory, per-channel normalized values (bottom). Blue denotes real data and red denotes synthetic data.



(a) Taiwan equities, daily bars



(b) Cryptocurrency, daily bars



(c) Foreign exchange, daily bars

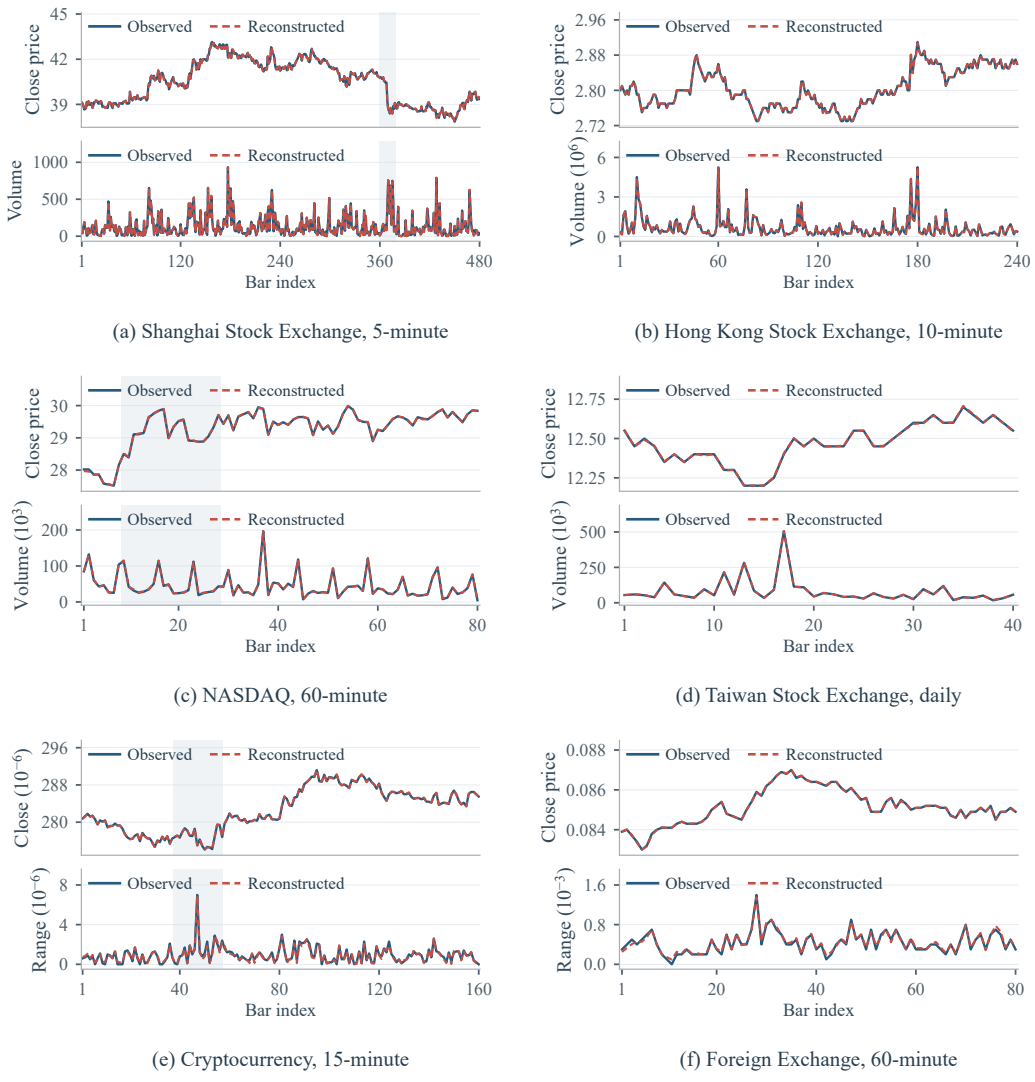
Figure J.3: **Additional daily settings.** Distributional comparisons for (a) Taiwan stock exchange, (b) cryptocurrency, and (c) foreign exchange. Each group shows t-SNE projections (top) and KDEs of within-trajectory, per-channel normalized values (bottom). Blue denotes real data and red denotes synthetic data.

K Case Studies: Reconstruction and Forecasting Visualization

K.1 Tokenizer Reconstruction

Following the reconstruction visualizations in Kronos (Shi et al., 2025), we examine the information preserved by FiRE's tokenizer at the sequence and candlestick levels. Each observed window is encoded into 256-bit tokens and decoded back into OHLCVA values using the frozen tokenizer. For each of six market–frequency settings, we display the window nearest the median normalized reconstruction MSE over available fields. Normalization statistics are computed from the complete observed window (Appendix C.1).

Figure K.1 shows that reconstructed close prices follow both gradual movements and abrupt changes, while the equity volume series retain the timing and relative magnitude of the observed peaks.



(a) Shanghai Stock Exchange, 5-minute

(b) Hong Kong Stock Exchange, 10-minute

(c) NASDAQ, 60-minute

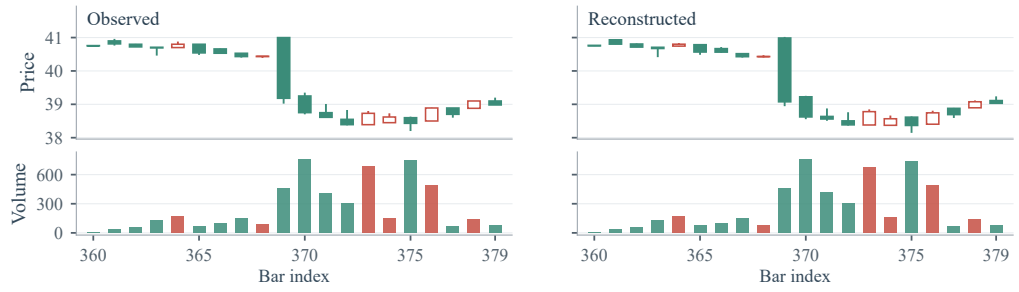
(d) Taiwan Stock Exchange, daily

(e) Cryptocurrency, 15-minute

(f) Foreign Exchange, 60-minute

Figure K.1: **Tokenizer reconstruction across markets and temporal resolutions.** Blue solid lines denote observations; red dashed lines denote reconstructions. Each example shows close price alongside volume (a–d) or the high–low range, $H - L$ (e–f). Shaded intervals identify the close-ups in Figure K.2.

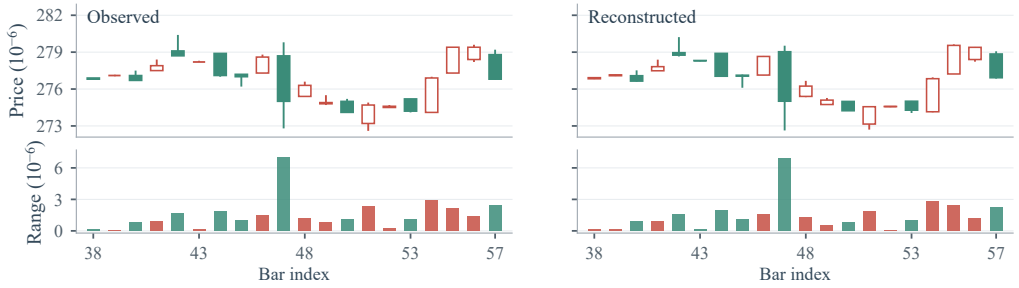
The candlestick views in Figure K.2 resolve the sharp decline and the cryptocurrency range spike in more detail. They also reveal small discrepancies in wick lengths and the direction of small-bodied candles, complementing the numerical and structural reconstruction metrics in Appendix D.5.



(a) Shanghai Stock Exchange, 5-minute



(b) NASDAQ, 60-minute



(c) Cryptocurrency, 15-minute

Figure K.2: **Candlestick-level reconstruction.** Observed and reconstructed candlesticks in the shaded intervals of Figure K.1, shown on matched axes. Red hollow candles indicate $C \geq O$; green filled candles indicate $C < O$. The lower panels show volume for Shanghai and NASDAQ, and $H - L$ for cryptocurrency, with colors matching the corresponding candle bodies.

K.2 Forecasting Examples

Figures K.3–K.10 compare pretrained FiRE, FiRE + ReFLO with separate price and volatility rewards, Kronos Base (Shi et al., 2025), FinCast (Zhu et al., 2025b), and Chronos-2 (Ansari et al., 2025). Blue lines show observations; thin red lines show all 32 independent FiRE or Kronos trajectories, and thick red lines their decoded pointwise means. FinCast and Chronos-2 use native mean and median forecasts. Dashed lines mark the forecast origin; axes are shared.

FiRE uses Gaussian noise at $t = 0$, 16 Euler steps, and inner/outer sampling $K = N = 1$ per call, repeated 32 times. Kronos uses temperature 1.0 and top-$p = 0.9$. The six original favorable windows are retained and rerun; two additional windows show near-zero close-price correlation. This retrospective gallery illustrates varied forecast quality, rather than representative performance.

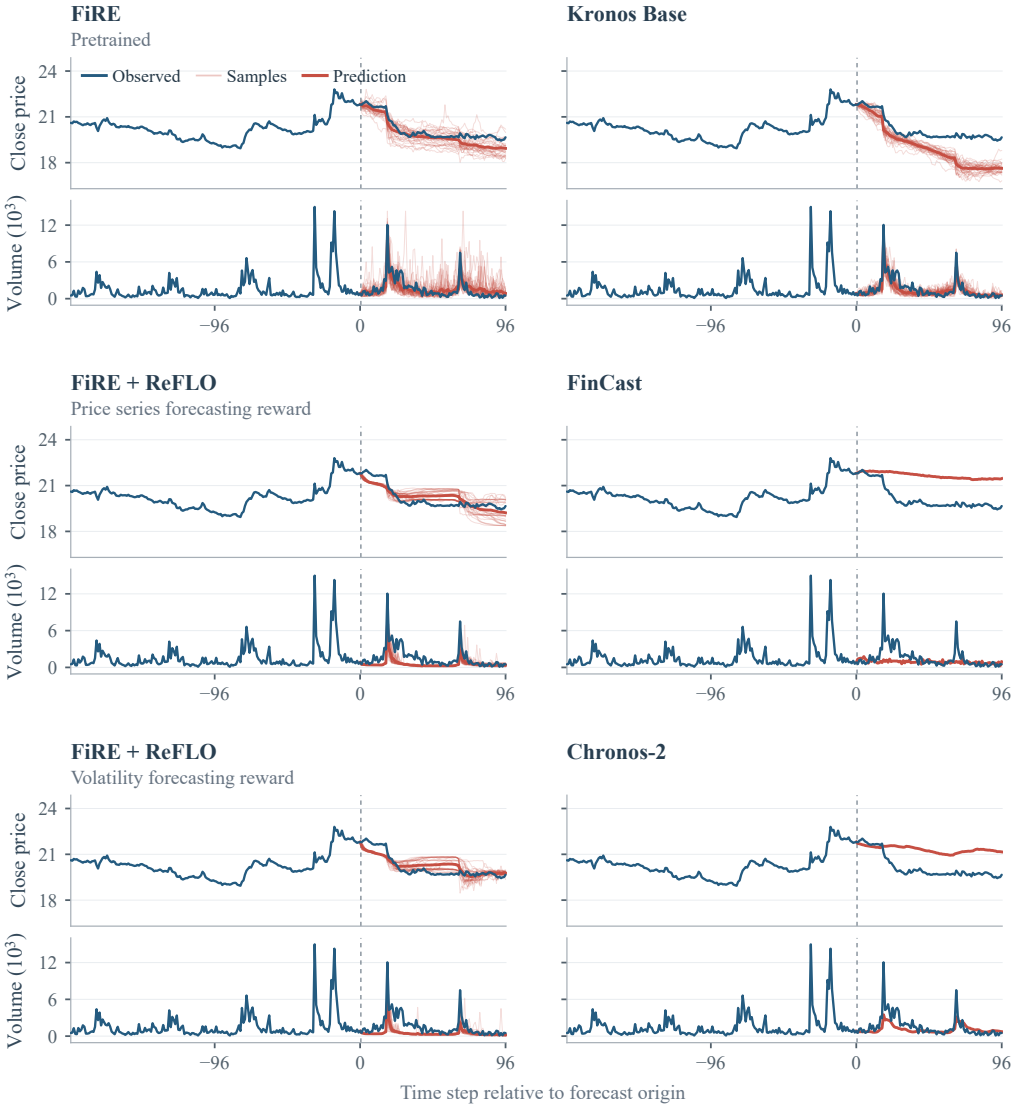


Figure K.3: **Shanghai Stock Exchange, 5-minute bars.** Models use 480 context bars to forecast 96 bars; the last 192 context bars are displayed.

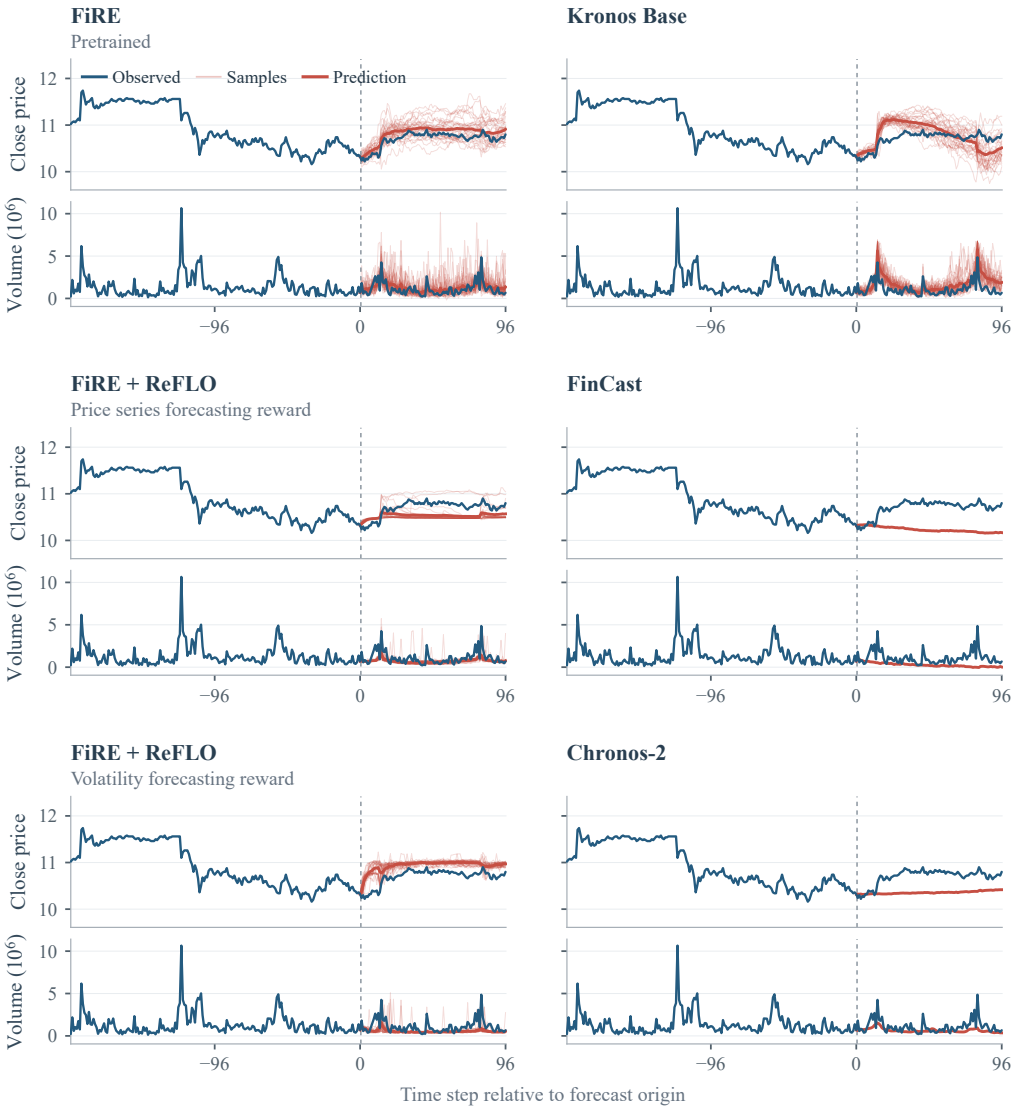


Figure K.4: **Hong Kong Stock Exchange, 5-minute bars.** Models use 480 context bars to forecast 96 bars; the last 192 context bars are displayed.

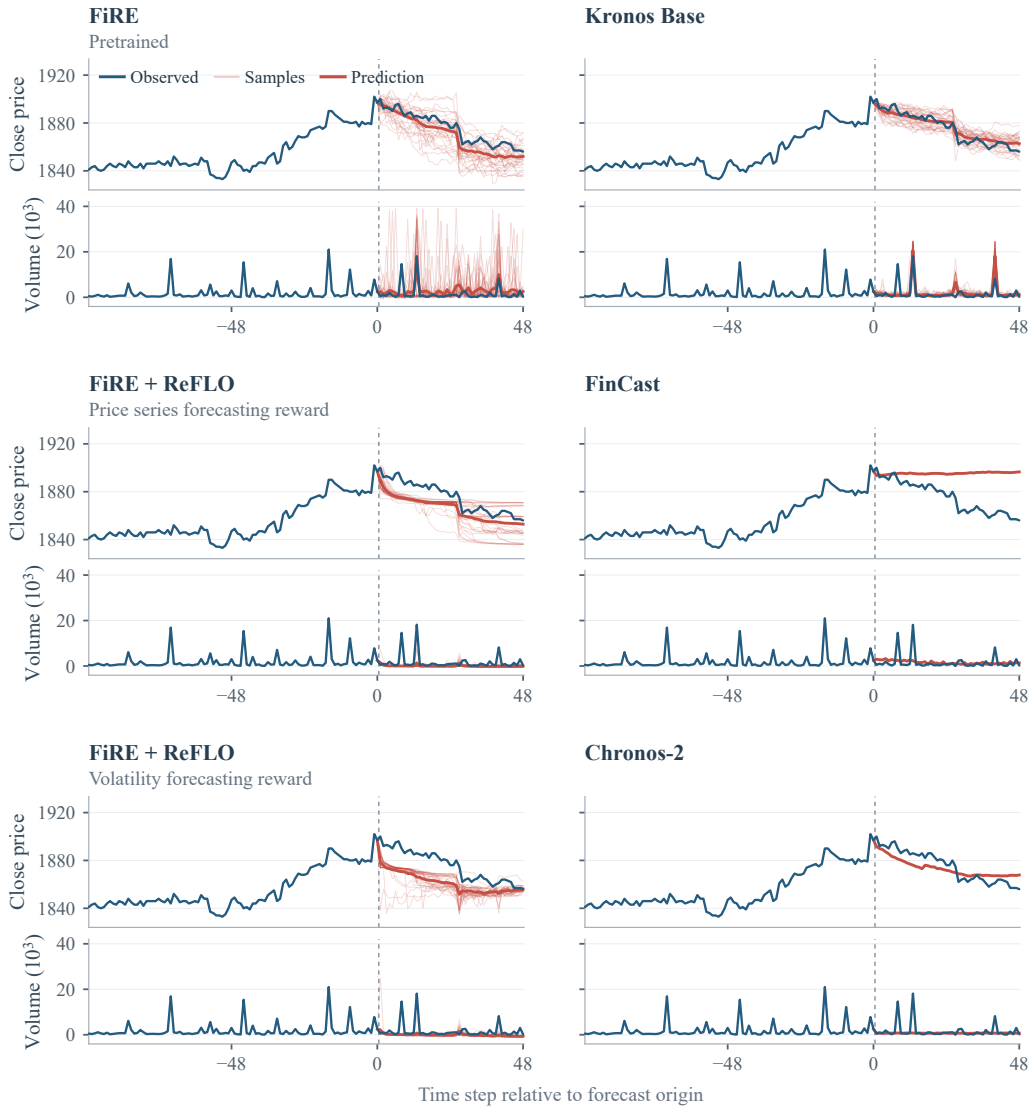


Figure K.5: **Japan Stock Exchange, 10-minute bars.** Models use 240 context bars to forecast 48 bars; the last 96 context bars are displayed.

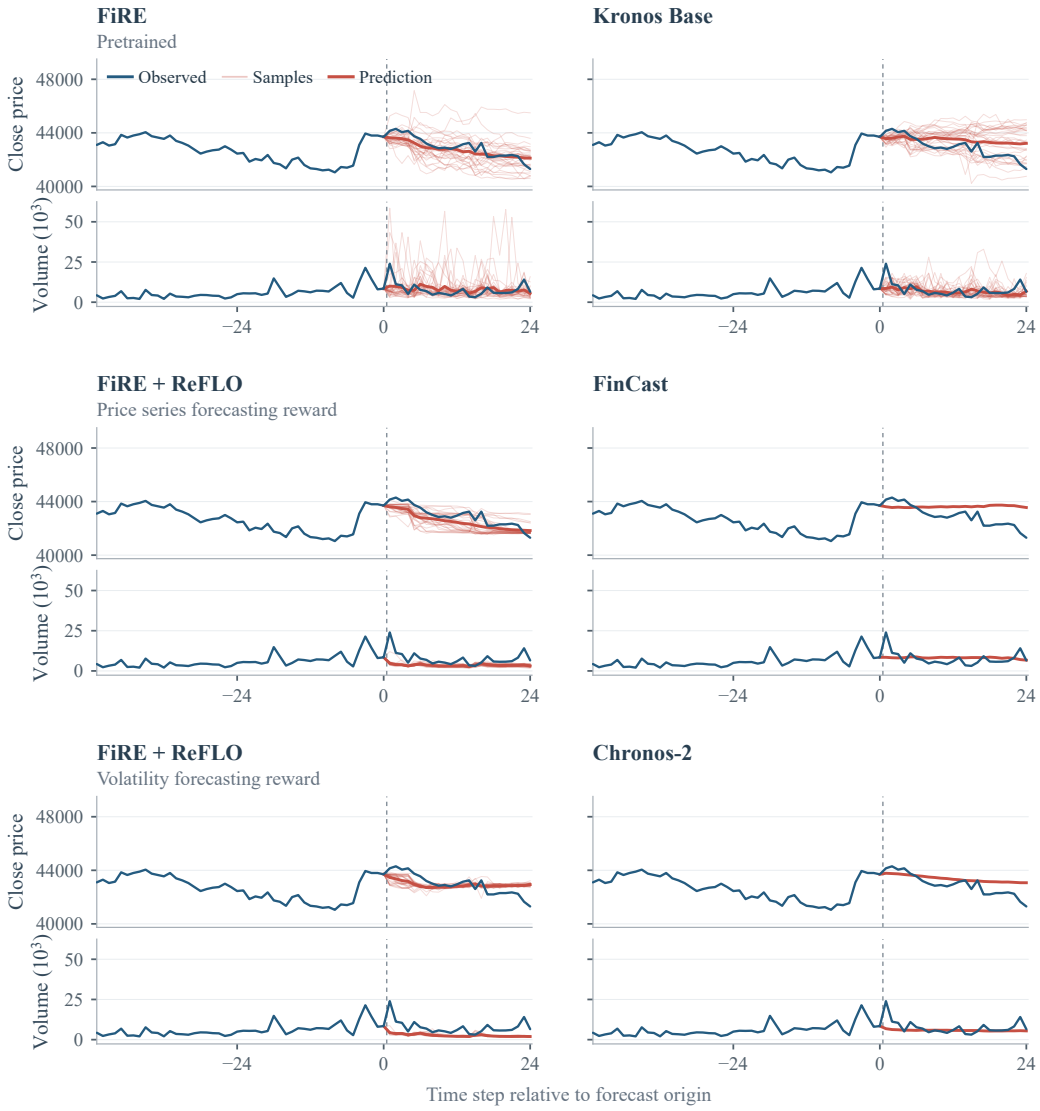


Figure K.6: **Korea Stock Exchange, 40-minute bars.** Models use 90 context bars to forecast 24 bars; the last 48 context bars are displayed.

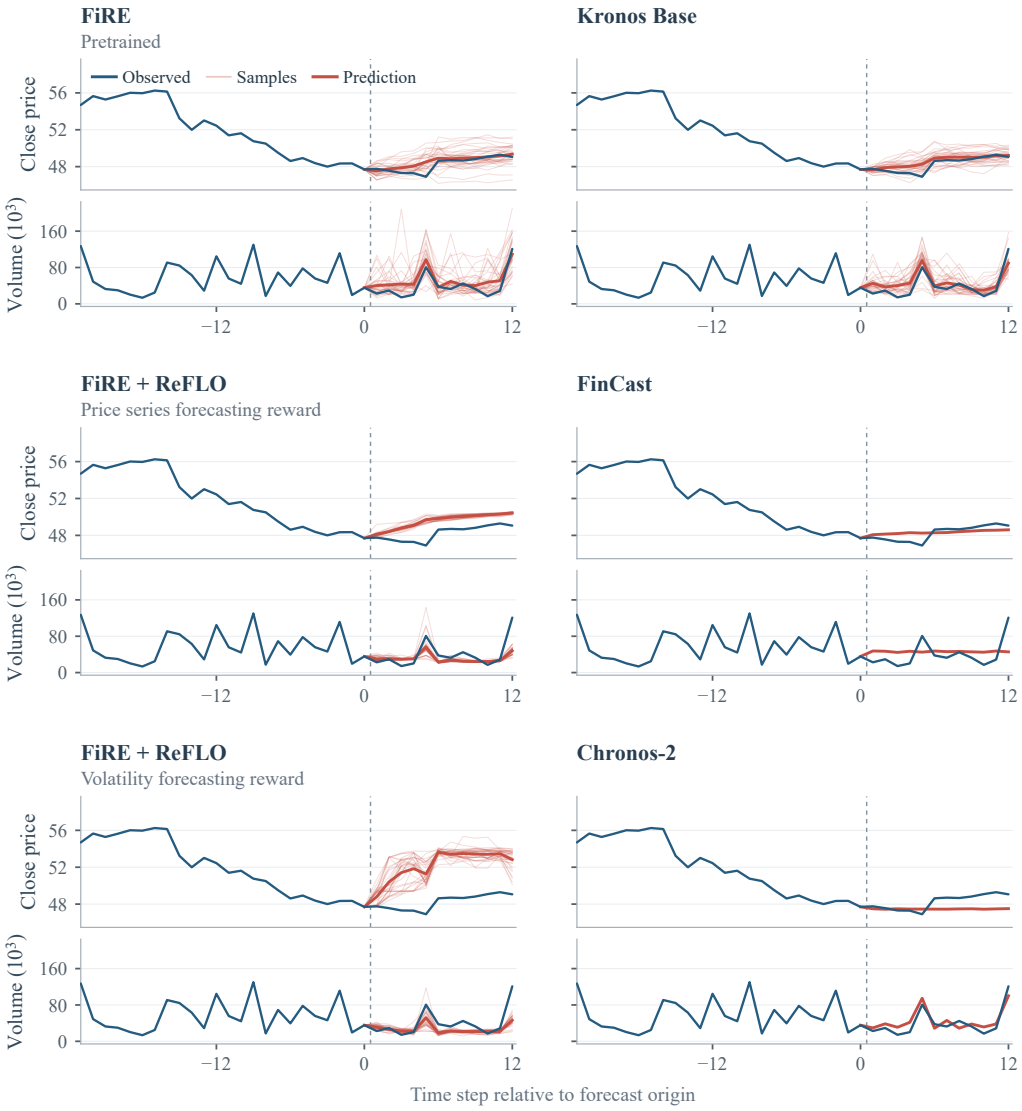


Figure K.7: **NASDAQ, 60-minute bars.** Models use 80 context bars to forecast 12 bars; the last 24 context bars are displayed.

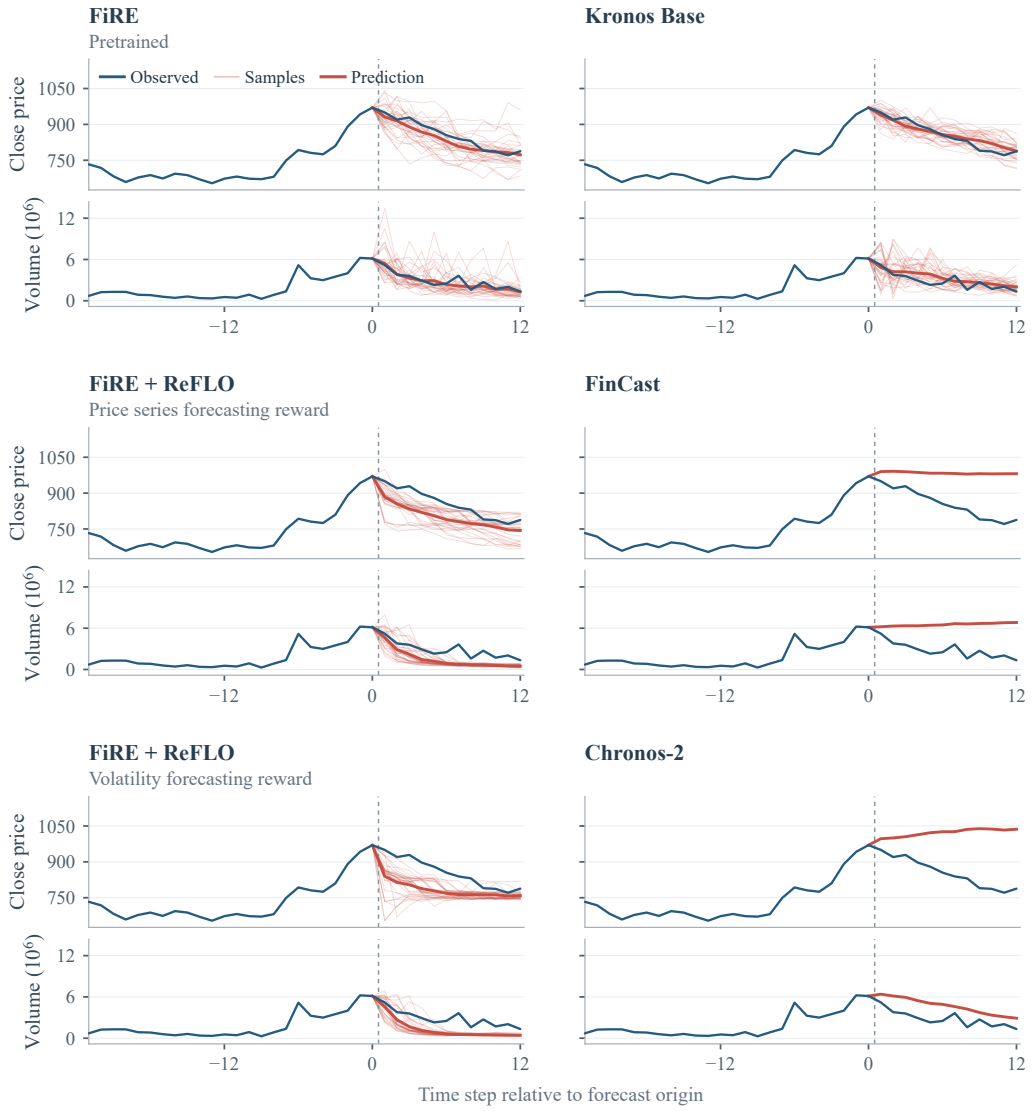


Figure K.8: **Taiwan Stock Exchange, daily bars.** Models use 40 context bars to forecast 12 bars; the last 24 context bars are displayed.

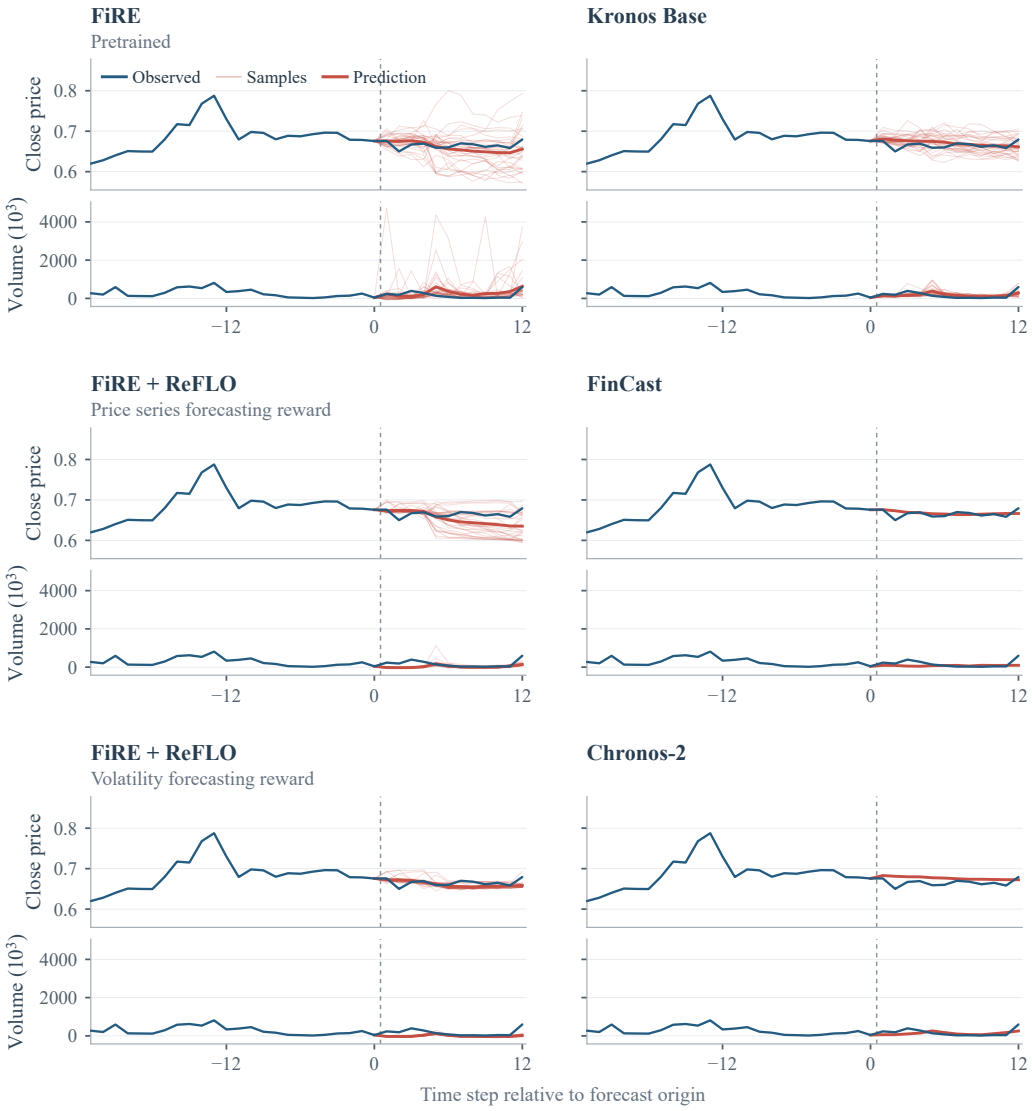


Figure K.9: **NASDAQ, 60-minute bars.** Models use 80 context bars to forecast 12 bars; the last 24 context bars are shown.

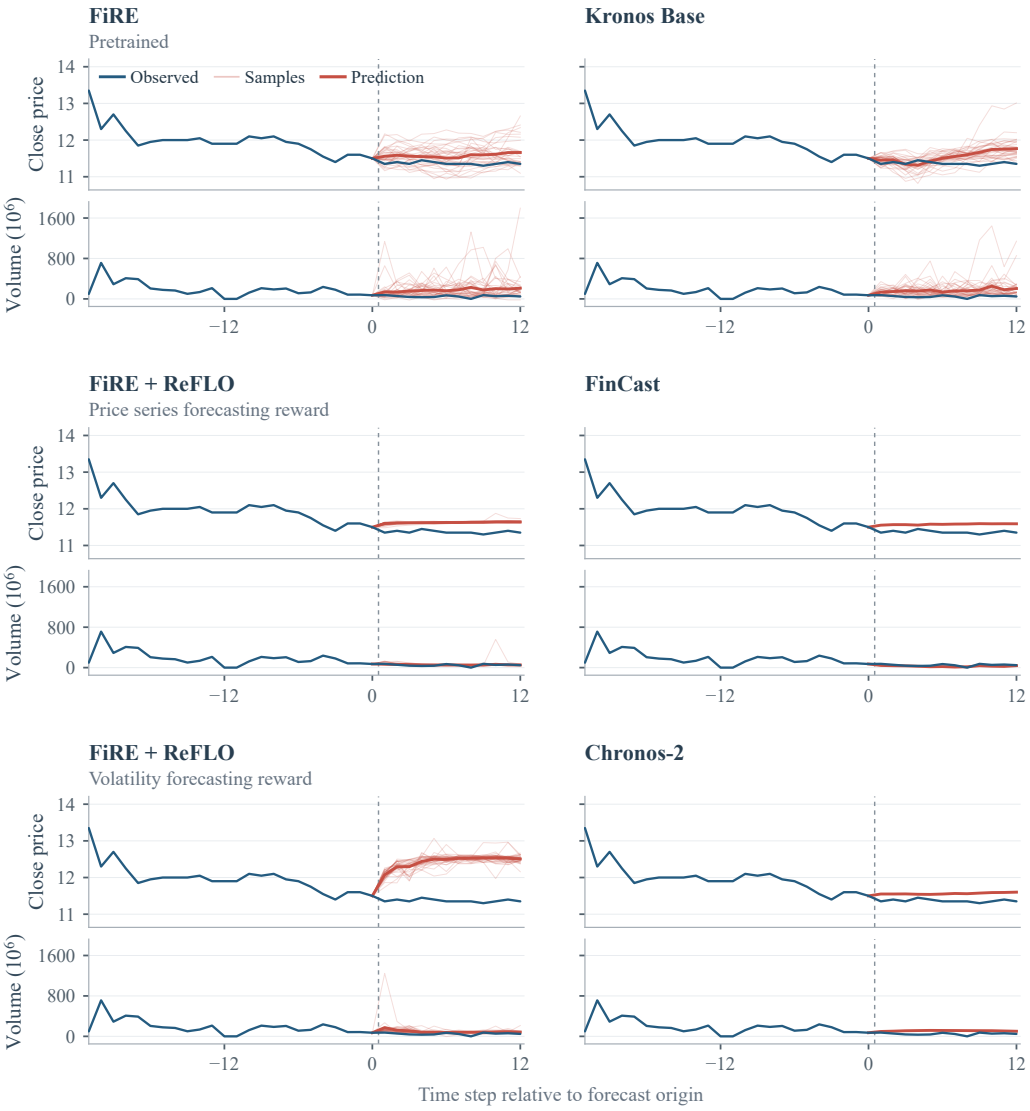


Figure K.10: **Taiwan Stock Exchange, daily bars.** Models use 40 context bars to forecast 12 bars; the last 24 context bars are shown.