TiRex-2: Generalizing TiRex to Multivariate Data and Streaming

Patrick Podest[*,1,2,3] **Marco Pichler**[*,1,2,3]
Elias Bürger[1,2] **Levente Zólyomi**[1,3] **Bernhard Voggenberger**[3] **Wilhelm Berghammer**[1,2]
Daniel Klotz[4] **Sebastian Böck**[3] **Günter Klambauer**[1,2,3] **Sepp Hochreiter**[1,2,3]

[1]ELLIS Unit Linz, LIT AI Lab & Institute for Machine Learning, JKU Linz, Austria
[2]NXAI Lab, Linz, Austria [3]NXAI GmbH, Linz, Austria
[4]Interdisciplinary Transformation University Austria, Linz, Austria

[*]Equal contribution {podest,pichler}@ml.jku.at

Abstract

We introduce TiRex-2, a recurrent xLSTM-based time series foundation model that generalizes the univariate TiRex to multivariate forecasting with both past and future covariates. Real-world forecasting is inherently sequential: observations arrive continuously, variables evolve jointly, and a subset of covariates is known ahead of time. Existing Transformer-based time series foundation models capture cross-variate dependencies, but the cost of ingesting each new observation grows at least linearly with context length, even with KV caching. TiRex-2 addresses these limitations with a memory-centric recurrent design built around an asymmetric group-attention variate mixer: paired with a bidirectional time mixer, it integrates future-known covariates while keeping target variables strictly causal. To our knowledge no other time series foundation model does both, and it is what makes streaming at constant per-patch cost possible. For scalable multivariate pretraining, we propose a synthetic coupling pipeline that composes multivariate samples on the fly from large univariate corpora. On GIFT-Eval, TiRex-2 improves on the zero-shot Chronos-2-Synth on all four statistics, and on Chronos-2 under a matched corpus. On fev-bench Chronos-2 stays ahead on skill score at $2.2\times$ the mean active parameters. TiRex-2 uses 38.4M active parameters in univariate mode, plus 44.1M for multivariate forecasting. Code is available at github.com/NX-AI/tirex-2.

1 Introduction

Time series arise across diverse domains, including cloud operations (Joosen et al., 2023), macroeconomics (Sims, 1980), earth sciences (Kratzert et al., 2018), healthcare (Johnson et al., 2023), and industrial monitoring (Sathishkumar V E, 2021), where the goal is to extrapolate observed dynamics into the future. Reliable forecasts underpin high-stakes decisions such as flood mitigation (Nearing et al., 2024) and predictive maintenance (Yan et al., 2024). In practice, systems are rarely described by a single signal, rather multiple interacting variates jointly determine the system state. Effective forecasting therefore requires models that capture both temporal structure within each variate and dependencies across variates.

Classical approaches, including autoregressive models (Hyndman and Khandakar, 2008) and exponential smoothing (Gardner, 1985), are typically fit to a single time series and applied to that same series. Vector autoregressive models (Sims, 1980) extend this paradigm to multiple variates but remain instance-specific, requiring re-estimation for each multivariate system. Neural approaches shifted the paradigm toward learning across collections of related time series, with LSTMs (Hochreiter, 1991;

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

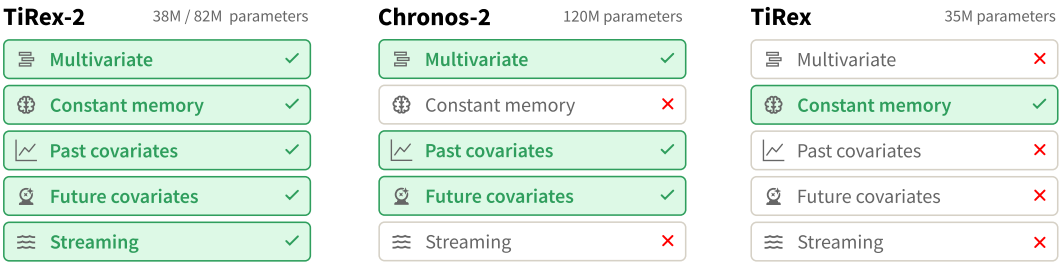


Figure 1: Comparison of TiRex-2, Chronos-2, and TiRex. Chronos-2 supports multivariate forecasting with covariates, but is neither target-causal nor constant-memory. TiRex-2 adds these properties while preserving native multivariate covariate support (for more details see Table 10).

Hochreiter and Schmidhuber, 1997) enabling explicit state tracking and the integration of multiple input variates (Nearing et al., 2024). More recently, time series foundation models (TSFMs) aim to generalize across datasets and domains, with Chronos-2 (Ansari et al., 2025), TimesFM (Das et al., 2024), MOIRAI (Woo et al., 2024), and TiRex (Auer et al., 2025b) as representative examples.

Despite this progress, a gap remains between univariate scalability and multivariate modelling. Strong univariate foundation models, such as TiRex, Reverso (Fu et al., 2026), and FlowState (Graf et al., 2025), often rely on a channel-independence assumption when applied to multivariate data, thereby neglecting cross-variate dependencies. Multivariate foundation models, including Chronos-2, MOIRAI, and GTT (Feng et al., 2024), address this limitation but are predominantly based on Transformer architectures. While effective at modeling joint dependencies, these models incur inference costs that grow with context length and require repeated processing of the full history as new observations arrive. This scaling behavior is fundamentally misaligned with streaming forecasting, where predictions must be updated continuously and efficiently.

In this work, we introduce TiRex-2, a recurrent time series foundation model that extends TiRex to multivariate forecasting with both past and future covariates while enabling true streaming inference. The model adopts a memory-centric design based on xLSTM, allowing constant-cost state updates as new data arrive. Architecturally, TiRex-2 combines a bidirectional time mixer with an asymmetric group-attention variate mixer, enabling the integration of future-known covariates without violating causality over target variables.

Our contributions are as follows (see also Figure 1):

- **Asymmetric group attention.** Asymmetric group attention allows future covariates to be used non-causally while keeping targets strictly causal, a first among TSFMs. This extends TiRex to jointly model multiple target variates with observed (past) and future covariates. Additionally, the model bypasses the variate mixer in univariate mode, activating only 38.4M parameters, and it activates the additional 44.1M only in multivariate mode.
- **Streaming inference at constant cost.** The recurrent state allows incremental forecast updates with constant computation and memory per patch, whereas attention-based models incur costs growing with context length and, if non-causal, require full-context re-computation. Even after 32M streamed time steps, forecast quality shows no sign of drift.
- **Synthetic multivariate coupling for pretraining.** We introduce a data generation pipeline that constructs diverse multivariate training instances on the fly from univariate corpora, including indirect, causal, and direct cross-variate dependencies, thereby expanding the effective training distribution. Each mechanism contributes measurably.

Section 2 reviews related work, Section 3 presents the setting, architecture, and coupling pipeline, and Section 4 reports the evaluation.

2 Related work

Univariate TSFMs. Early transformer-based TSFMs are either pretrained LLMs (Xue and Salim, 2023; Gruver et al., 2023) or trained on time-series data directly (Ansari et al., 2024). These

approaches commonly pass each time step individually to the model. However, splitting time series into non-overlapping patches (Nie et al., 2022) has become the dominant architectural choice for TSFMs (Liu et al., 2025b; Das et al., 2024; Wang et al., 2025; Liu et al., 2026a). Recent work has explored large mixture of experts models (Liu et al., 2026b; Wu et al., 2026) with billions of parameters, Liu et al. (2025a) suggesting that the different experts specialize based on the shape of patches. In contrast, smaller recurrent neural network (RNN) approaches remain competitive, both without (Fu et al., 2026; Graf et al., 2025) and with (Auer et al., 2025b) patching. In this work, we adapt the recurrent TiRex architecture proposed by Auer et al. (2025b) and extend it via the variate-mixing attention layers to learn cross-correlation patterns.

Multivariate TSFMs. Two architectural strands dominate joint multivariate forecasting. The first flattens all variates into one sequence with any-variate attention: MOIRAI (Woo et al., 2024) and MOIRAI-MOE (Liu et al., 2025a) concatenate all variates into a single sequence, while MORPHEUS (Patil et al., 2025) interleaves individual timesteps with separation tokens in between. As single-target variants, COSMIC (Auer et al., 2025a) and TimesFM-ICF (Faw et al., 2025) use the other variates to enhance the prediction (the covariate setting). Flattening, however, inflates sequence length and limits per-variate context. The second strand factorizes time and variate attention into separate layers, introduced by Crossformer (Zhang and Yan, 2022) and adopted for TSFMs by Feng et al. (2024) and Chronos-2 (Ansari et al., 2025) to scale to 8k time steps and many variates; Liu et al. (2023) push this further by embedding entire time series into single tokens.

Models differ further in whether they accept *past-only* or *future-known* covariates (whose future values are available at inference), and those handling both alongside multivariate targets remain rare (Table 10). A complementary line instead adapts univariate TSFMs through covariate-aware projections, decomposition, trainable variate-mixing layers, or in-context regression on covariates (Benechehab et al., 2025; Arango et al., 2025; Cheng et al., 2026; Ekambaram et al., 2024; Chen et al., 2023; Auer et al., 2025a; Bürger et al., 2026). We adopt factorized time/variate attention (Gao et al., 2024; Cohen et al., 2025; Zhang and Yan, 2022) on top of the recurrent TiRex backbone, yielding memory *linear* in sequence length while supporting both covariate types with multivariate targets, allowing the model to leverage, e.g., historical sensor readings alongside future-known calendar or promotion features, and adaptively disable variate-mixing for univariate inputs to preserve the efficiency of TiRex.

Synthetic data generation. In data-scarce settings, synthetic data is a viable option for training TSFMs (Ansari et al., 2025; Oreshkin et al., 2026; Moroshan et al., 2026). Existing generators mostly target univariate time series, using adapted seasonal ARIMA processes (Oreshkin et al., 2026), Gaussian processes (Ansari et al., 2024), or temporal causal models (TCMs) (Xie et al., 2025; Runge et al., 2023). For multivariate time series, Ansari et al. (2025) mention unspecified "multivariatizers" that couple independently sampled univariate series. We make this step explicit with a concrete, largely TCM-based framework of coupling mechanisms for generating diverse synthetic multivariate dependencies.

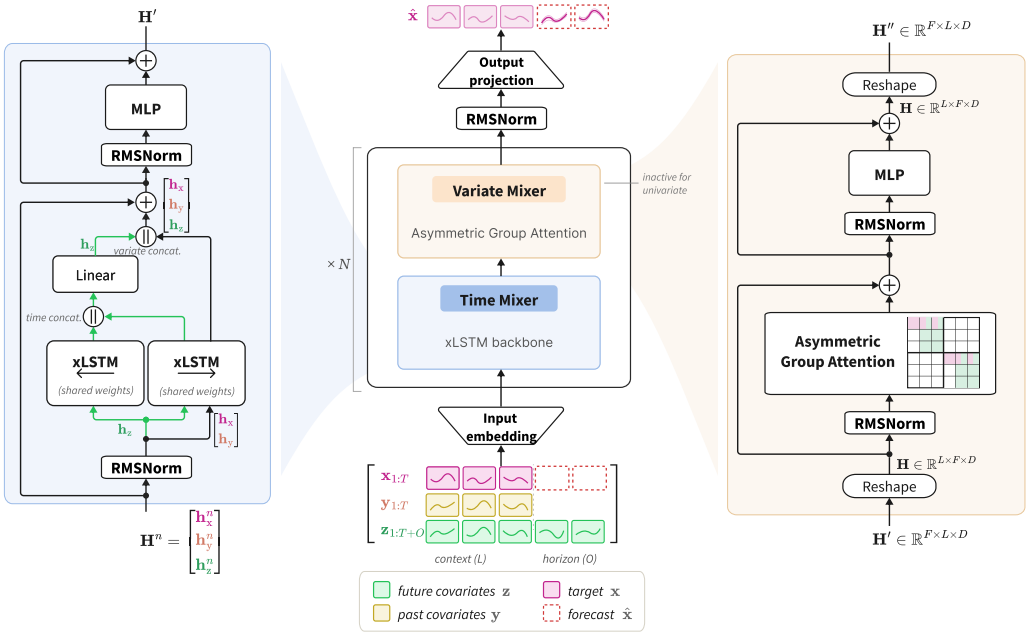
3 A TiRex architecture for multivariate forecasting with covariates

3.1 Problem setup

In multivariate time series forecasting, we want to forecast F_x target series $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_{F_x}]^\top$, each spanning T time steps of historical context and a prediction horizon of O time steps, i.e. $\mathbf{x}_{i,1:T} \in \mathbb{R}^T$ and $\mathbf{x}_{i,T+1:T+O} \in \mathbb{R}^O$. If available, the model is additionally conditioned on F_y past covariates $\mathbf{Y} \in \mathbb{R}^{F_y \times T}$ and F_z future-known covariates $\mathbf{Z} \in \mathbb{R}^{F_z \times (T+O)}$, known across the entire prediction horizon (e.g., calendar features or scheduled interventions). Then the goal is to model the conditional distribution

$$\mathcal{P}\big(\mathbf{X}_{T+1:T+O} \;\big|\; \mathbf{X}_{1:T}, \mathbf{Y}_{1:T}, \mathbf{Z}_{1:T+O}\big).$$

To this end, the model approximates the distribution above using Q evenly spaced quantiles, conditioned on the available context: $\hat{\mathbf{X}}_{T+1:T+O} = [\hat{\mathbf{X}}_{k,T+1:T+O}]_{k=1}^{Q}$, where $\hat{\mathbf{X}}_{k,T+1:T+O} \in \mathbb{R}^{F_x \times O}$ denotes the k^{th} quantile prediction.



(a) Time mixer (b) Overall architecture (c) Variate mixer

Figure 2: TiRex-2 alternates time- and variate-mixing blocks. The variate mixer's asymmetric group attention is what allows future-known covariates to be exploited bidirectionally without leaking future-known targets into earlier positions. (b) Each multivariate time series is split into non-overlapping patches before being embedded into tokens. The stack mixes information across time (time mixer), across variates (variate mixer) and inside each token (MLP). The output-projection produces Q quantile forecasts per time step in each output patch. (a) The time mixer applies a forward xLSTM to all variates, plus a weight-tied reverse pass for future-known covariates, fused linearly. (c) The variate mixer applies group attention along the variate axis with an asymmetric mask preventing target-to-covariate flow. Note: we omitted the layer super-scripts of the token representations $\mathbf{H}$ inside the blocks for clarity.

3.2 Architecture

Block signature. Figure 2b shows the high-level architecture of TiRex-2. The model first pads targets and past covariates to length $T + O$, then splits every variate independently into $L = \lceil (T + O)/P \rceil$ non-overlapping patches of size P and embeds each patch into a token, yielding a token tensor $\mathbf{H} \in \mathbb{R}^{F\times L\times D}$, with $F = F_x + F_y + F_z$ the total number of variates and D the token embedding dimension. TiRex-2 then alternates a time mixer, which mixes information across the L axis independently per variate, with a variate mixer, which mixes across the F axis independently per patch.

Input layer. The input layer maps each raw variate $\mathbf{u} \in \mathbb{R}^{T+O}$, which is a row of either $\mathbf{X}_{1:T}$, $\mathbf{Y}_{1:T}$ or $\mathbf{Z}_{1:T+O}$ padded to length $T + O$, to a sequence of L tokens in three steps: scaling, patching, and embedding. Scaling builds on reversible instance normalization (Kim et al., 2021) to mitigate distribution shift across variates, and adds the inverse hyperbolic sine transform from econometrics (Burbidge et al., 1988) and energy price forecasting (Uniejewski and Weron, 2018). This prevents heavy-tailed variates from dominating the loss. Concretely, we standardize $\mathbf{u}$ with its empirical mean $\hat{\mu}$ and standard deviation $\hat{\sigma}$ computed over the observed context ($t < T$, ignoring missing entries) and apply

$$\tilde{u}_t \;=\; (1-b)\,\operatorname{arcsinh}\!\left(\frac{u_t-\hat{\mu}}{\hat{\sigma}}\right) + b\,u_t,$$

which behaves linearly near the origin and logarithmically in the tails. The binary gate b handles a separate pathology: sparse binary covariates are degenerate under naive standardization, since the gap between the two standardized levels is $1/\sqrt{p(1-p)}$, which scales as $1/\sqrt{p}$ for rare positive

classes ($p \to 0$) and diverges symmetrically as $p \to 1$. We therefore detect binary variates with the indicator $b = \mathbf{1}[\forall\, 0 \leqslant t < T : x_t \in \{0, 1\}]$ and bypass the affine transform when $b = 1$, preserving the canonical $\{0, 1\}$ encoding regardless of context-window sparsity. We defer the full specification to Section D. After scaling, each variate is split into $L = \lceil (T + O)/P \rceil$ non-overlapping patches of length P; unobserved positions (future targets, future past-covariate values, and missing entries) are padded. A two-layer residual MLP shared across variates and patches $\mathbb{R}^P \to \mathbb{R}^D$ then embeds each patch into a token of dimension D, yielding the input tensor $\mathbf{H}$ consumed by the first block.

Time mixer. The time mixer acts along the patch axis L independently for every variate, with directionality determined by the variate's type. Target and past-covariate tokens are processed by a strictly forward xLSTM (Beck et al., 2024). Future-known covariate tokens are additionally processed in reverse by the *same* weight-tied xLSTM (Schmidinger et al., 2025), and the two directions are combined by a linear fusion layer, so that each future-known covariate token encodes information from both before and after its position. Concretely, the time mixer maps the input tensor $\mathbf{H}$ to $\mathbf{H}'$. Writing $\bar{\mathbf{H}} = \mathrm{RMSNorm}(\mathbf{H})$ for the pre-normalized input, $1 \leqslant l \leqslant L$ indexing the patches, θ the (weight-tied) parameters of the xLSTM, and $[\,\cdot\,,\cdot\,]$ and $[\,\cdot\,;\,\cdot\,]$ for concatenation along the sequence and variate axes, respectively,

$$
\begin{aligned}
\tilde{\mathbf{h}}_{x,l} &= \mathrm{xLSTM}\big(\bar{\mathbf{H}}_{x,1:l}; \theta\big), & \overrightarrow{\mathbf{h}}_{z,l} &= \mathrm{xLSTM}\big(\bar{\mathbf{H}}_{z,1:l}; \theta\big), \\
\tilde{\mathbf{h}}_{y,l} &= \mathrm{xLSTM}\big(\bar{\mathbf{H}}_{y,1:l}; \theta\big), & \overleftarrow{\mathbf{h}}_{z,l} &= \mathrm{xLSTM}\big(\bar{\mathbf{H}}_{z,L:l}; \theta\big), \\
\tilde{\mathbf{h}}_{z,l} &= \mathrm{Linear}\Big(\Big[\overleftarrow{\mathbf{h}}_{z,l},\, \overrightarrow{\mathbf{h}}_{z,l}\Big]\Big), & \tilde{\mathbf{h}}_{l} &= \Big[\,\tilde{\mathbf{h}}_{x,l}\,;\,\tilde{\mathbf{h}}_{y,l}\,;\,\tilde{\mathbf{h}}_{z,l}\,\Big] \in \mathbb{R}^{F \times D}.
\end{aligned}
$$

The time-mixed output $\mathbf{H}' = \mathbf{H} + \tilde{\mathbf{H}}$, where $\tilde{\mathbf{H}} = [\tilde{\mathbf{h}}_l]_{l=1}^{L}$ further passes through a residually connected MLP creating the final block output. The time mixer processes just the future known covariates twice (forward and reverse) while targets and past covariates remain strictly forward, making streaming forecasts well-defined (Sec. 3.3).

Following Beck et al. (2024), who report benefits from mLSTM blocks on long-context tasks, we instantiate the time mixer as an xLSTM stack that alternates mLSTM and sLSTM blocks, rather than using the sLSTM-only backbone of the original TiRex (Auer et al., 2025b).

Variate mixer. The number of variates F varies from series to series and there is no canonical ordering. We therefore implement the variate mixer as multi-head self-attention, which natively handles arbitrary F and is permutation-equivariant within each variate type. We use block-diagonal *group* attention (Feng et al., 2024; Cohen et al., 2025; Ansari et al., 2025), which prevents interactions across concatenated series and avoids padding to the largest F in a batch. Within each group we further apply an *asymmetric mask* M: target queries may read covariate keys, but covariate queries cannot read target keys. With $1 \leqslant i, j \leqslant F$ indexing (query, key) within the group and $\mathcal{X}, \mathcal{Y}, \mathcal{Z}$ the target, past- and future-covariate index sets, we have:

$$
m_{ij} = \begin{cases} 0 & \neg(i \in \mathcal{Y} \cup \mathcal{Z} \wedge j \in \mathcal{X}), \\ -\infty & \text{otherwise.} \end{cases} \tag{1}
$$

Entries $m_{ij} = -\infty$ pull the attention weight to 0 in the $\mathrm{softmax}$ operation preventing information flowing from token $j \to i$ (target to covariate) while entries $m_{ij} = 0$ allow it (covariate to target).

Lemma 1 (One-block target dependence)**.** *Let $\mathbf{h}_{x,l}^{n}$ denote the l^{th} target token after the first n blocks. Then $\mathbf{h}_{x,l}^{n+1}$ is a function only of $\mathbf{H}_{x,1:l}^{n}$, $\mathbf{H}_{y,1:l}^{n}$, and $\mathbf{H}_{z,1:L}^{n}$.*

The forward-only xLSTM restricts target and past-covariate tokens at patch l to indices $\leqslant l$, while the asymmetric mask in Eq. (1) blocks every covariate query from reading a target key, so the variate mixer cannot reintroduce a future-known target dependency through the covariate channel.

Proposition 1 (Target-causality)**.** *By induction on the block depth using Lemma 1, for every $n \leqslant N$ and patch index l the target token $\mathbf{h}_{x,l}^{n}$ does not depend on $\mathbf{h}_{x,l'}^{0}$ for any $l' > l$.*

Intuitively, future-known covariates may carry information backward along the patch axis, but the mask prevents those tokens from ever reading target tokens. Combined with the output head, the next-patch prediction emitted from position l therefore cannot read the target patch it is trained to predict. A full proof is given in Section C.2.

To our knowledge, this makes TiRex-2 the first TSFM that exploits future-known covariates bidirectionally while keeping target streams strictly causal in a single forward pass. The closest prior designs are TimeXer (Wang et al., 2024), Timer XL (Liu et al., 2024) and CITRAS (Yamaguchi et al., 2025). All of them impose related but weaker asymmetries and are task-specific or fully causal along time. Section C.4 provides a detailed comparison.

Output layer and loss. A residual, two-layer MLP maps the target hidden representations $\mathbf{H}_x \in \mathbb{R}^{F_x \times L \times D}$ produced by the final time and variate mixers to patch-wise next-patch quantile forecasts (following Auer et al., 2025b), $\hat{\mathbf{X}} \in \mathbb{R}^{F_x \times Q \times (L-1) \times P}$, i.e., Q quantiles for each of the P time steps of every output patch. Predictions are returned to the original data domain by applying the inverse of the input scaler, which clips its argument to a conservative bound before applying $\sinh$ to suppress implausibly large outlier predictions and prevent overflow in the output datatype (see Section D). The model is trained with the pinball loss (Koenker and Bassett, 1978) applied at every output time step (not only on the horizon), giving up to an $(L-1)$-fold denser gradient signal than supervising on the horizon alone:

$$\mathcal{L} = \frac{1}{F_x \, Q \, |\mathcal{T}_{\text{obs}}|} \sum_{t \in \mathcal{T}_{\text{obs}}} \sum_{f=1}^{F_x} \sum_{k=1}^{Q} \left[q_k \, \max(x_{f,t} - \hat{x}_{f,k,t}, \, 0) + (1 - q_k) \, \max(\hat{x}_{f,k,t} - x_{f,t}, \, 0) \right],$$

where $\mathcal{T}_{\text{obs}}$ is the set of observed (non-missing) target time steps and q_k denotes the k^{th} quantile level. Missing values are excluded from both sum and normalisation.

3.3 Long-context efficiency and streaming forecasts

The recurrent xLSTM time mixer keeps the per-block forward time complexity at $\mathcal{O}(L)$ rather than the $\mathcal{O}(L^2)$ of attention, the variate dimension F stays constant and therefore does not affect scaling in L. The maintained hidden state of xLSTM additionally enables constant-time updates. Streaming workloads predominantly involve target and past-covariate streams observed up to the current time step, both routed through the forward-only xLSTM. In an online forecasting setting, which we refer to as streaming, each newly arrived patch can be ingested and a forecast emitted in constant time. In contrast, a transformer-based time mixer using KV-caching would pay $\mathcal{O}(L)$ per new patch, its latency growing linearly with the number of patches L. TiRex-2 can therefore be fed continuously and emit forecasts in time proportional to the length of the increment, not to the full lookback horizon.

3.4 Synthetic multivariate coupling

A model that natively consumes multivariate inputs with covariates only learns to use them if the training distribution actually contains diverse cross-variate dependencies. Existing curated

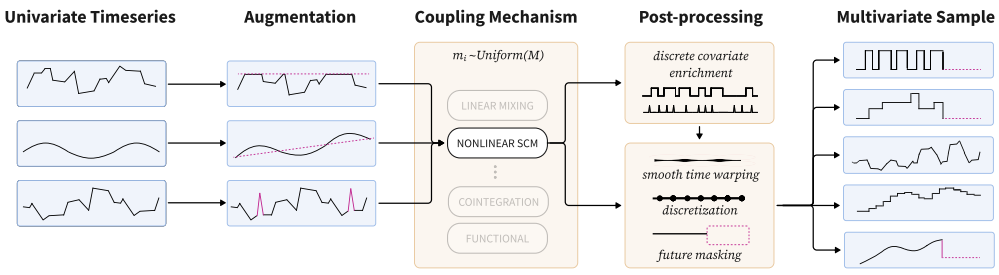


Figure 3: Synthetic multivariate coupling pipeline. A batch of univariate series is first independently augmented (amplitude trends, censoring, spike injection). A coupling mechanism $c_i \sim \text{Uniform}(C)$ is then sampled from $C = \{$ identity, univariate, linear mixing, linear structural causal model (SCM), nonlinear SCM, cointegration, functional $\}$ and, except for the identity and univariate pass-through cases, transforms the augmented series into jointly dependent variates. Post-processing adds realistic structure via covariate enrichment and applies smooth time warping, discretization, and future masking, producing the final multivariate training sample.

multivariate corpora are too narrow to enforce this learning process, while large univariate corpora are abundant. We bridge this gap with a synthetic coupling pipeline that, at training time, draws a batch of univariate series from a shared pool and *couples* them on the fly into multivariate samples with controllable cross-variate dependencies (Fig. 3):

Each series is first independently perturbed with piecewise-linear amplitude trends, quantile censoring, and synthetic spikes (Gaussian, triangular, or rectangular kernels), then cropped or NaN-padded to length T. Given F such augmented series $\mathbf{u}_1, \ldots, \mathbf{u}_F \in \mathbb{R}^T$, one of the following mechanisms is sampled to produce F output variates $\mathbf{x}_1, \ldots, \mathbf{x}_F \in \mathbb{R}^T$, written entrywise as $x_{f,t}$, with a known dependency structure:

1. **Identity / pass-through:** $x_{f,t} = u_{f,t}$ or a single univariate output, preserving univariate forecasting ability as a no-coupling control.
2. **Functional coupling:** $x_{f,t} = \phi_f(u_{0,t}) + \varepsilon_{f,t}$ with monotone, compressive, discretizing, or piecewise-linear ϕ_f, yielding direct pointwise dependence as in sensor redundancies.
3. **Linear mixing:** $x_{f,t} = \sum_{i=1}^{F} a_{fi}\, u_{i,t}$, with the singular-value spectrum of the mixing matrix $A = (a_{fi})$ sampled from dominant, uniform, or power-law regimes, mimicking shared latent drivers as in factor models.
4. **Cointegration:** $x_{f,t} = \sum_k \Lambda_{fk}\, s_{k,t} + \xi_{f,t}$ with shared random-walk trends $s_{k,t}$ and stationary AR(1) residuals $\xi_{f,t}$, reproducing long-run equilibria between nonstationary variates.
5. **Linear structural causal model:** a random directed acyclic graph with lagged edges, $x_{f,t} = \sum_{i \in \mathrm{pa}(f)} \alpha_{if}\, u_{i,\, t-\tau_{if}} + \varepsilon_{f,t}$, introducing directed lead–lag structure.
6. **Nonlinear structural causal model:** using nonlinearities g_{if} and an optional multiplicative gate h, $x_{f,t} = h(u_{k,\, t-\tau_k}) \sum_{i \in \mathrm{pa}(f)} g_{if}(u_{i,\, t-\tau_{if}})$, adding state-dependent coupling.

The resulting samples are finally enriched with realistic covariate structure: variate permutation, smooth per-variate time warping via Brownian-bridge lags, patch masking with contiguous NaN blocks (Auer et al., 2025b) per-variate, partial future observability by truncating future portions of random covariates, and discretization in value (uniform, quantile, power-law) and time (freezes, staircases, duty cycles). See Section G for details.

4 Experiments

We evaluate TiRex-2 along two axes. First, we compare against other zero-shot time series foundation models on two separate benchmarks, fev-bench (Shchur et al., 2025) and GIFT-Eval (Aksu et al., 2024). Second, we use synthetic data to isolate streaming behavior, long-horizon forecasting, and covariate-shift sensitivity. We conclude with architectural ablations that quantify the contribution of the main components of TiRex-2.

Training setup. We pretrain TiRex-2 for 700,000 steps on 2 NVIDIA H100 GPUs, followed by a short long-context post-training phase at context length of 8,192 and prediction horizon 512 time steps to adapt the model to longer sequences. The backbone consists of $N = 12$ alternating mLSTM/sLSTM xLSTM blocks. Full hyperparameters and training details for both phases are given in Section E.

Evaluation models. In the following experiments, we compare TiRex-2 to publicly available time series foundation models on fev-bench (Shchur et al., 2025) and GIFT-Eval (Aksu et al., 2024), restricting to models for which zero-shot evaluations and inference code were published on the respective leaderboard. This yields two benchmark-specific comparison sets: on fev-bench, Chronos-Bolt, MOIRAI-2, Toto-1.0 (Cohen et al., 2024), TiRex (Auer et al., 2025b), TimesFM-2.5 (Das et al., 2024), and Chronos-2 (Ansari et al., 2025); Figure 5 additionally includes TabPFN-TS (Hoo et al., 2024), Sundial-Base (Liu et al., 2025b) and the statistical baselines of the benchmark itself. On GIFT-Eval, Chronos-2-Synth, PatchTST-FM-r1 (Granite and base) (Wen et al., 2026), TiRex, TimesFM-2.5, and FlowState-r1.1 (Graf et al., 2025). On the other experiments we compare TiRex-2 against purely multivariate TSFMs on synthetic data, and therefore relax the zero-shot constraint. We include Moirai (1.0 and MoE) (Woo et al., 2024; Liu et al., 2025a), GTT (Feng et al., 2024), Toto, and Chronos-2.

4.1 Zero Shot

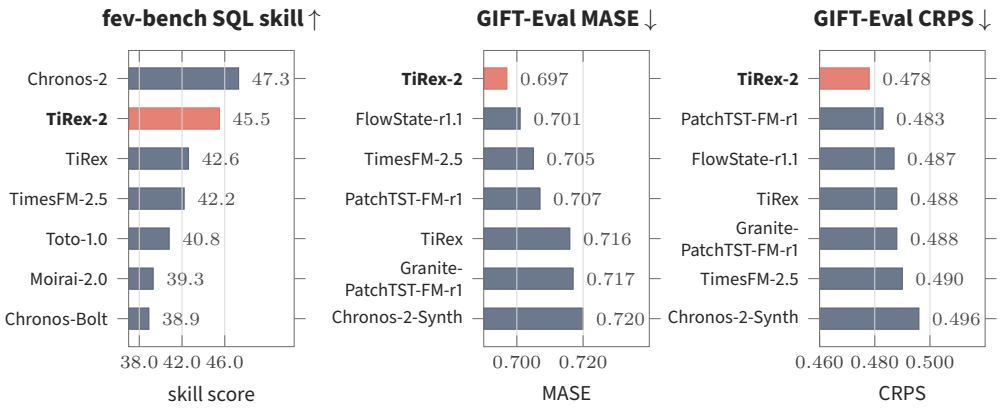


Figure 4: Zero-shot performance of TiRex-2 against the best time series foundation models on each metric: the fev-bench SQL skill score against Seasonal Naive, where higher is better, and GIFT-Eval MASE and CRPS, where lower is better. Panels are sorted with the best model at the top, so the leaderboard position is read the same way in all three.

We evaluate zero-shot forecasting on two complementary benchmarks. fev-bench (Shchur et al., 2025) probes the ability to exploit past and future-known covariates, whereas GIFT-Eval (Aksu et al., 2024) probes generalization across diverse domains, sampling frequencies, and horizons. To rule out training-test leakage, we pretrain a separate checkpoint per benchmark, with overlapping datasets removed from the respective training corpus (see Section F). Chronos-2 includes train-splits of GIFT-Eval during pretraining, so the zero-shot comparison there is against Chronos-2-Synth, which TiRex-2 improves on across all four statistics. Under a corpus matched to Chronos-2, TiRex-2 improves on Chronos-2 on all four as well (Section B.2). On fev-bench Chronos-2 remains ahead (Figure 4, Section B.1). Figure 5 gives the pairwise picture against every other model on the leaderboard: TiRex-2 wins 65–100% of tasks against all of them but Chronos-2, with which it splits tasks evenly (51.0, CI 41.0–60.0) while trailing by 1.8 SQL skill points.

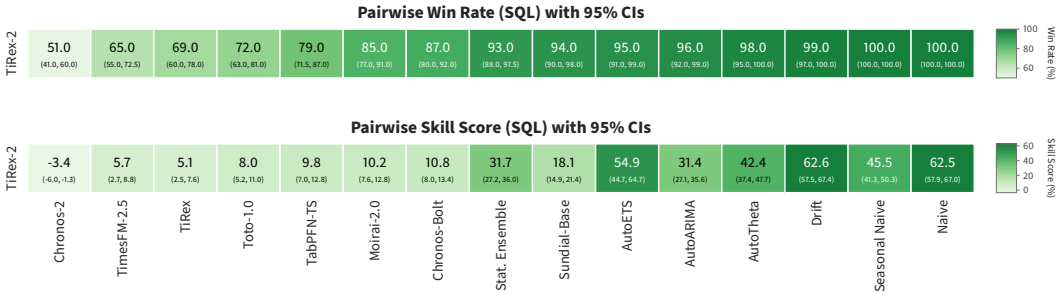


Figure 5: Zero-shot performance of TiRex-2 against time series models on fev-bench (SQL) using both Pairwise Win Rate and Pairwise Skill Score with 95% confidence intervals.

4.2 Sensitivity to streaming, covariates and forecast horizon

Streaming. The causal, recurrent design of TiRex-2 (Sec. 3.3) lets the model ingest arbitrarily long contexts patch by patch at constant per-patch cost for target and past-covariate streams and emit a forecast after every update. We stream up to 32M time steps of a first-order autoregressive target driven by a noisy, lagged covariate registered as future-known, and report MAE per emitted patch (Fig. 6, left). Forecast quality remains stable far past the 8k post-training context: the recurrent state extrapolates cleanly to context lengths 4000× beyond training, without any sign of saturation or drift. On real data, Pichler et al. (2026) confirm that TiRex-2 retains its accuracy on the full history of the ten fev-bench tasks exceeding 50k steps, while all other evaluated TSFMs degrade beyond their

trained context. Section B.3 presents additional results on retained forecast quality as well as latency and memory measurements.

Long-horizon forecasting. Long-horizon forecasting on chaotic systems is effectively a covariate-utilisation stress test: models must extract signal from the remaining state variables to maintain accuracy at long horizons. The `dysts` benchmark (Gilpin, 2023) provides 135 chaotic trajectories at three temporal granularities. We forecast one channel as target with the rest as future-known covariates, evaluating TiRex-2 and Chronos-2 at $h \in \{32, 64, \ldots, 1056\}$.

Covariate shift sensitivity. Since real-world covariates are rarely perfectly aligned with the target (e.g. Podobnik et al., 2010; Zhao et al., 2023), we probe shift tolerance by pairing synthetic random-walk targets with a shifted, noisy covariate, $z_t^\tau = x_{t-\tau} + \varepsilon_t$, $\varepsilon_t \sim \mathcal{N}(0,\, 0.1^2)$, where x is the target and τ the shift in time steps, and tracking the MAE against τ (Fig. 6). TiRex-2 remains informative well beyond the range where Chronos-2 has fallen back to its no-covariate baseline. For Toto-1.0, GTT and Moirai-MOE the covariate was provided as a past covariate since this is the setting which the models support, one can therefore see that the models only make gains for negative shifts, where the future is shifted into the past.

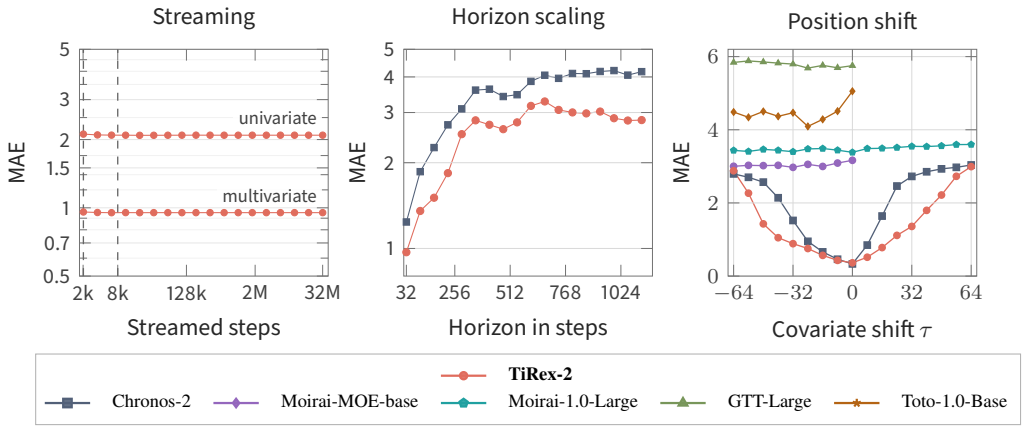


Figure 6: Left: streaming. MAE is stable far beyond the 2k pretraining and 8k post-training context boundaries (dashed). Middle: long-horizon forecasting on `dysts` (log y-axis), averaged over temporal granularities. TiRex-2 attains lower MAE than Chronos-2 at all horizons, with the gap widening as the horizon grows (per-granularity results in Figure 9). Right: covariate shift sensitivity. TiRex-2 retains useful covariate signal over a wider lag range than Chronos-2. Setup in Sections 4.2 and B.7.

4.3 Ablations

We ablate the core design choices of TiRex-2 on fev-bench, which contains enough genuinely multivariate tasks to produce meaningful signal. Moreover, we investigate the impact of each mechanism of the synthetic multivariate coupling on fev-bench and compare those impacts on GIFT-Eval. Each configuration is trained with up to six seeds. We report mean and standard deviation of the SQL and MASE skill score metrics for fev-bench, and the MASE and CRPS metrics for GIFT-Eval.

For the architecture ablations, relative to the full model, we remove or replace group attention, asymmetric attention masking, binary-aware scaling, bidirectional context mixing, and the mixed sLSTM/mLSTM backbone. We further test the pre-processor and future-known covariates. Providing future-known covariates yields a consistent gain, indicating that forecast-window signal outweighs extended autoregressive history (Table 1).

In the coupling mechanism ablation, we disable each coupler separately and show the results in Table 2. In the multivariate evaluation (fev-bench), removing any single coupler lowers both skill scores by roughly 2–3 points, so each mechanism contributes; the nonlinear SCM causes the largest drop in SQL, while identity and functional coupling matter most for MASE. As expected, on the largely univariate GIFT-Eval the effects are an order of magnitude smaller and mostly within seed variance, with functional coupling the only coupler whose removal measurably hurts CRPS.

Table 1: Ablation study on fev-bench and the subset with future-known covariates. The baseline row reports mean SQL skill score ± std.; other rows report mean deltas with original std. over six seeds. Rows below the baseline remove or replace a single component of TiRex-2 (higher is better).

Configuration	All fev-bench SQL skill / Δ	Future-known cov. subset SQL skill / Δ
TiRex-2 (full)	**45.5 ± 0.15**	**45.19 ± 0.38**
No group attention	−9.67 ± 2.22	−18.73 ± 3.44
No asymm. mask	−4.35 ± 0.24	−8.05 ± 0.37
No future-known covariates	−1.87 ± 0.15	−6.59 ± 0.43
Forward only	−1.37 ± 0.12	−4.62 ± 0.37
sLSTM only	−0.26 ± 0.17	−0.16 ± 0.28
No binary-aware scaler	−0.13 ± 0.12	+0.07 ± 0.27
8k→2k context	−0.12 ± 0.13	−0.09 ± 0.15

Table 2: Leave-one-out ablation of the coupling mechanisms. fev-bench: skill score against Seasonal Naive over 100 tasks, higher is better, so a negative delta means the ablation hurt. GIFT-Eval: raw MASE and CRPS, lower is better, so a positive delta means the ablation hurt. Three seeds.

	fev-bench skill ↑		GIFT-Eval ↓	
Coupler removed	SQL	MASE	MASE	CRPS
none (TiRex-2)	45.50 ± 0.15	33.74 ± 0.23	0.6970 ± 0.0006	0.4780 ± 0.0009
Identity/passthrough	−2.21 ± 0.23	−2.62 ± 0.28	−0.0006 ± 0.0015	−0.0009 ± 0.0000
Functional	−2.09 ± 0.07	−2.60 ± 0.04	+0.0066 ± 0.0040	+0.0016 ± 0.0006
Linear mixing	−2.05 ± 0.08	−2.41 ± 0.10	+0.0018 ± 0.0025	+0.0003 ± 0.0000
Cointegration	−2.12 ± 0.05	−2.45 ± 0.10	+0.0009 ± 0.0019	+0.0007 ± 0.0006
Linear SCM	−2.12 ± 0.02	−2.48 ± 0.07	+0.0014 ± 0.0023	−0.0001 ± 0.0006
Nonlinear SCM	−2.97 ± 0.11	−2.36 ± 0.20	+0.0000 ± 0.0007	−0.0002 ± 0.0006

5 Conclusion

We presented TiRex-2, a recurrent xLSTM-based foundation model for multivariate time series forecasting with past and future-known covariates. By pairing a bidirectional time mixer with an asymmetric group-attention variate mixer, TiRex-2 exploits future-known covariates while keeping target and past-covariate states strictly causal, which in turn enables constant-cost streaming inference at arbitrary context lengths. Together with a synthetic coupling pipeline that supplies the cross-variate diversity missing from curated multivariate corpora, this yields competitive zero-shot accuracy at a modest parameter budget: on GIFT-Eval TiRex-2 improves on the zero-shot Chronos-2-Synth on all four statistics and on Chronos-2 under a matched corpus, and it wins 65–100% of fev-bench tasks against every baseline but Chronos-2, with which it splits tasks evenly on SQL (51.0, CI 41.0–60.0) while trailing on aggregate skill score. A natural direction for future work is dynamic covariate selection. Just as TiRex-2 bypasses the variate mixer entirely in univariate mode, an in-context estimate of covariate-target correlation could prune individual covariates from the group attention whenever their contribution is negligible, saving FLOPs at inference.

Limitations. Streaming at constant per-patch cost covers targets, past covariates, and future-known covariates whose values are available in advance: since covariate queries never attend to target keys (Eq. (1)), the bidirectional covariate representations depend on covariate inputs alone and are computed once before the stream starts. Only revisions of already-provided future-known values, e.g. an updated weather forecast, require recomputing their representations and the target states that read them. This is rarely binding: such covariates (calendars, scheduled events) are typically fixed, and revisable ones can be shifted into the causal region as past covariates, which restores full streaming (Figure 6). Streaming also operates at patch granularity (32 steps). Our long-context comparisons run baselines past their trained context; a sliding window at their native length would narrow the gap, though the per-patch cost advantage holds.

Acknowledgments and Disclosure of Funding

We thank Sebastian Lehner for extensive discussions throughout the development of the model, and Michael List and Andreas Mayr for their valuable feedback. We further thank Christian Ganhör, Daniil Yefimov, and Martin Loretz for their contributions to the open-source code repository, and Suad Demiri for porting TiRex-2 to Apple devices. The LIT AI Lab, and the Institute for Machine Learning are supported by the Federal State Upper Austria. We acknowledge the EuroHPC Joint Undertaking for awarding us access to Leonardo at CINECA (Italy) and MareNostrum 5 at BSC (Spain).

References

Aksu, T., Woo, G., Liu, J., Liu, X., Liu, C., Savarese, S., Xiong, C., and Sahoo, D. (2024). GIFT-Eval: A Benchmark for General Time Series Forecasting Model Evaluation.

Ansari, A. F., Shchur, O., Küken, J., Auer, A., Han, B., Mercado, P., Rangapuram, S. S., Shen, H., Stella, L., Zhang, X., Goswami, M., Kapoor, S., Maddix, D. C., Guerron, P., Hu, T., Yin, J., Erickson, N., Desai, P. M., Wang, H., Rangwala, H., Karypis, G., Wang, Y., and Bohlke-Schneider, M. (2025). Chronos-2: From Univariate to Universal Forecasting. arXiv:2510.15821.

Ansari, A. F., Stella, L., Turkmen, A. C., Zhang, X., Mercado, P., Shen, H., Shchur, O., Rangapuram, S. S., Arango, S. P., Kapoor, S., Zschiegner, J., Maddix, D. C., Wang, H., Mahoney, M. W., Torkkola, K., Wilson, A. G., Bohlke-Schneider, M., and Wang, B. (2024). Chronos: Learning the Language of Time Series. *Transactions on Machine Learning Research*.

Arango, S. P., Mercado, P., Kapoor, S., Ansari, A. F., Stella, L., Shen, H., Senetaire, H. H. J., Turkmen, A. C., Shchur, O., Maddix, D. C., Bohlke-Schneider, M., Wang, B., and Rangapuram, S. S. (2025). ChronosX: Adapting Pretrained Time Series Models with Exogenous Variables. In *Proceedings of The 28th International Conference on Artificial Intelligence and Statistics*, pages 2242–2250. PMLR.

Auer, A., Parthipan, R., Mercado, P., Ansari, A. F., Stella, L., Wang, B., Bohlke-Schneider, M., and Rangapuram, S. S. (2025a). Zero-Shot Time Series Forecasting with Covariates via In-Context Learning. arXiv:2506.03128 [cs].

Auer, A., Podest, P., Klotz, D., Böck, S., Klambauer, G., and Hochreiter, S. (2025b). TiRex: Zero-Shot Forecasting Across Long and Short Horizons with Enhanced In-Context Learning.

Beck, M., Pöppel, K., Spanring, M., Auer, A., Prudnikova, O., Kopp, M. K., Klambauer, G., Brandstetter, J., and Hochreiter, S. (2024). xLSTM: Extended Long Short-Term Memory.

Benechehab, A., Feofanov, V., Paolo, G., Thomas, A., Filippone, M., and Kégl, B. (2025). AdaPTS: Adapting Univariate Foundation Models to Probabilistic Multivariate Time Series Forecasting.

Burbidge, J. B., Magee, L., and Robb, A. L. (1988). Alternative Transformations to Handle Extreme Values of the Dependent Variable. *Journal of the American Statistical Association*, 83(401):123–127.

Bürger, E., Berghammer, W. F., Pichler, M., Podest, P., Böck, S., and Hochreiter, S. (2026). Covariates without retraining: Efficient conditioning of time series foundation models. In *Workshop TS-LIMITS @NeurIPS 2026 – Generalization for Time Series in Tight Settings: Latency, Inference, Memory, prIvacy and susTainability*.

Chen, S.-A., Li, C.-L., Arik, S. O., Yoder, N. C., and Pfister, T. (2023). TSMixer: An All-MLP Architecture for Time Series Forecasting. *Transactions on Machine Learning Research*.

Cheng, H., Wu, X., Shu, Y., Rao, Z., Pan, L., Yang, B., and Guo, C. (2026). CORA: Boosting Time Series Foundation Models for Multivariate Forecasting Through Correlation-Aware Adapter.

Cohen, B., Khwaja, E., Doubli, Y., Lemaachi, S., Lettieri, C., Masson, C., Miccinilli, H., Ramé, E., Ren, Q., Rostamizadeh, A., Terrail, J. O. d., Toon, A.-M., Wang, K., Xie, S., Xu, Z., Zhukova, V., Asker, D., Talwalkar, A., and Abou-Amal, O. (2025). This Time is Different: An Observability Perspective on Time Series Foundation Models. arXiv:2505.14766 [cs].

Cohen, B., Khwaja, E., Wang, K., Masson, C., Ramé, E., Doubli, Y., and Abou-Amal, O. (2024). Toto: Time Series Optimized Transformer for Observability. arXiv:2407.07874 [cs].

Das, A., Kong, W., Sen, R., and Zhou, Y. (2024). A decoder-only foundation model for time-series forecasting.

Ekambaram, V., Jati, A., Dayama, P., Mukherjee, S., Nguyen, N. H., Gifford, W. M., Reddy, C., and Kalagnanam, J. (2024). Tiny Time Mixers (TTMs): Fast Pre-trained Models for Enhanced Zero/Few-Shot Forecasting of Multivariate Time Series.

Faw, M., Sen, R., Zhou, Y., and Das, A. (2025). In-Context Fine-Tuning for Time-Series Foundation Models.

Feng, C., Huang, L., and Krompass, D. (2024). General Time Transformer: an Encoder-only Foundation Model for Zero-Shot Multivariate Time Series Forecasting. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, CIKM '24, pages 3757–3761, New York, NY, USA. Association for Computing Machinery.

Fu, X., Li, Y., Papaioannou, G., and Kim, Y. (2026). Reverso: Efficient Time Series Foundation Models for Zero-shot Forecasting. arXiv:2602.17634.

Gao, S., Koker, T., Queen, O., Hartvigsen, T., Tsiligkaridis, T., and Zitnik, M. (2024). UniTS: A unified multi-task time series model. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Gardner, E. S. (1985). Exponential smoothing: The state of the art. *Journal of Forecasting*, 4(1):1–28.

Gilpin, W. (2023). Chaos as an interpretable benchmark for forecasting and data-driven modelling.

Graf, L., Ortner, T., Woźniak, S., and Pantazi, A. (2025). FlowState: Sampling-Rate Invariant Time Series Foundation Model with Dynamic Forecasting Horizons.

Gruver, N., Finzi, M. A., Qiu, S., and Wilson, A. G. (2023). Large Language Models Are Zero-Shot Time Series Forecasters.

Hochreiter, S. (1991). Untersuchungen zu dynamischen neuronalen Netzen. Diploma thesis, Institut für Informatik, Lehrstuhl Prof. Brauer, Technische Universität München.

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

Hoo, S. B., Müller, S., Salinas, D., and Hutter, F. (2024). The tabular foundation model tabPFN outperforms specialized time series forecasting models based on simple features. In *NeurIPS Workshop on Time Series in the Age of Large Models*.

Hyndman, R. J. and Khandakar, Y. (2008). Automatic Time Series Forecasting: The **forecast** Package for *R*. *Journal of Statistical Software*, 27(3).

Johnson, A. E. W., Bulgarelli, L., Shen, L., Gayles, A., Shammout, A., Horng, S., Pollard, T. J., Hao, S., Moody, B., Gow, B., Lehman, L.-w. H., Celi, L. A., and Mark, R. G. (2023). MIMIC-IV, a freely accessible electronic health record dataset. *Scientific Data*, 10(1):1.

Joosen, A., Hassan, A., Asenov, M., Singh, R., Darlow, L., Wang, J., and Barker, A. (2023). How Does It Function? Characterizing Long-term Trends in Production Serverless Workloads. In *Proceedings of the 2023 ACM Symposium on Cloud Computing*, SoCC '23, pages 443–458, New York, NY, USA. Association for Computing Machinery.

Kim, T., Kim, J., Tae, Y., Park, C., Choi, J.-H., and Choo, J. (2021). Reversible Instance Normalization for Accurate Time-Series Forecasting against Distribution Shift.

Koenker, R. and Bassett, G. (1978). Regression Quantiles. *Econometrica*, 46(1):33–50.

Kratzert, F., Klotz, D., Brenner, C., Schulz, K., and Herrnegger, M. (2018). Rainfall–runoff modelling using Long Short-Term Memory (LSTM) networks. *Hydrology and Earth System Sciences*, 22(11):6005–6022.

Liu, C., Aksu, T., Liu, J., Liu, X., Yan, H., Pham, Q., Savarese, S., Sahoo, D., Xiong, C., and Li, J. (2026a). Moirai 2.0: When Less Is More for Time Series Forecasting. arXiv:2511.11698 [cs].

Liu, X., Liu, J., Woo, G., Aksu, T., Liang, Y., Zimmermann, R., Liu, C., Li, J., Savarese, S., Xiong, C., and Sahoo, D. (2025a). Moirai-MoE: Empowering Time Series Foundation Models with Sparse Mixture of Experts.

Liu, Y., Hu, T., Zhang, H., Wu, H., Wang, S., Ma, L., and Long, M. (2023). iTransformer: Inverted Transformers Are Effective for Time Series Forecasting.

Liu, Y., Qin, G., Huang, X., Wang, J., and Long, M. (2024). Timer-XL: Long-Context Transformers for Unified Time Series Forecasting.

Liu, Y., Qin, G., Shi, Z., Chen, Z., Yang, C., Huang, X., Wang, J., and Long, M. (2025b). Sundial: A Family of Highly Capable Time Series Foundation Models. arXiv:2502.00816 [cs].

Liu, Y., Su, X., Wang, S., Zhang, H., Liu, H., Wang, Y., Ye, Z., Xiang, Y., Wang, J., and Long, M. (2026b). Timer-S1: A Billion-Scale Time Series Foundation Model with Serial Scaling. arXiv:2603.04791.

Loshchilov, I. and Hutter, F. (2019). Decoupled Weight Decay Regularization. In *International Conference on Learning Representations*.

Moroshan, V., Siems, J., Zela, A., Carstensen, T., and Hutter, F. (2026). TempoPFN: Synthetic pre-training of linear RNNs for zero-shot time series forecasting.

Nearing, G., Cohen, D., Dube, V., Gauch, M., Gilon, O., Harrigan, S., Hassidim, A., Klotz, D., Kratzert, F., Metzger, A., Nevo, S., Pappenberger, F., Prudhomme, C., Shalev, G., Shenzis, S., Tekalign, T. Y., Weitzner, D., and Matias, Y. (2024). Global prediction of extreme floods in ungauged watersheds. *Nature*, 627(8004):559–563.

Nie, Y., Nguyen, N. H., Sinthong, P., and Kalagnanam, J. (2022). A Time Series is Worth 64 Words: Long-term Forecasting with Transformers.

Oreshkin, B. N., Jauhari, M., Selvam, R. K., Wolff, M., Pan, W., Ramasubramanian, S., Olivares, K. G., Konstantinova, T., Potapczynski, A., Cao, M., Efimov, D., Mahoney, M. W., and Wilson, A. G. (2026). Zero-shot Forecasting by Simulation Alone. arXiv:2601.00970 [cs].

Patil, P., Varshney, A., Cherukumalli, M., Deshpande, H., Eun, L., Sahoo, D., and Chittar, N. (2025). MORPHEUS : A Foundation Model for Multivariate Time Series Forecasting.

Pichler, M., Podest, P., Klambauer, G., and Hochreiter, S. (2026). Time series foundation models on the edge. In *Workshop TS-LIMITS @NeurIPS 2026 – Generalization for Time Series in Tight Settings: Latency, Inference, Memory, prIvacy and susTainability*.

Podobnik, B., Wang, D., Horvatic, D., Grosse, I., and Stanley, H. E. (2010). Time-lag cross-correlations in collective phenomena. *Europhysics Letters*, 90(6):68001.

Runge, J., Gerhardus, A., Varando, G., Eyring, V., and Camps-Valls, G. (2023). Causal inference for time series. *Nature Reviews Earth & Environment*, 4(7):487–505.

Sathishkumar V E, C. S. (2021). Steel Industry Energy Consumption.

Schmidinger, N., Schneckenreiter, L., Seidl, P., Schimunek, J., Hoedt, P.-J., Brandstetter, J., Mayr, A., Luukkonen, S., Hochreiter, S., and Klambauer, G. (2025). Bio-xlstm: Generative modeling, representation and in-context learning of biological and chemical sequences. *International Conference On Learning Representations*.

Shchur, O., Ansari, A. F., Turkmen, C., Stella, L., Erickson, N., Guerron, P., Bohlke-Schneider, M., and Wang, Y. (2025). fev-bench: A realistic benchmark for time series forecasting. *arXiv preprint arXiv:2509.26468*.

Sims, C. A. (1980). Macroeconomics and Reality. *Econometrica*, 48(1):1–48.

Smith, L. N. and Topin, N. (2019). Super-convergence: Very fast training of neural networks using large learning rates. In *Artificial intelligence and machine learning for multi-domain operations applications*, volume 11006, pages 369–386. SPIE.

Uniejewski, B. and Weron, R. (2018). Efficient Forecasting of Electricity Spot Prices with Expert and LASSO Models. *Energies*, 11(8).

Wang, X., Zhou, T., Gao, J., Ding, B., and Zhou, J. (2025). Output Scaling: YingLong-Delayed Chain of Thought in a Large Pretrained Time Series Forecasting Model. arXiv:2506.11029 [cs].

Wang, Y., Wu, H., Dong, J., Qin, G., Zhang, H., Liu, Y., Qiu, Y., Wang, J., and Long, M. (2024). TimeXer: Empowering Transformers for Time Series Forecasting with Exogenous Variables.

Wen, Y., Gifford, W. M., Reddy, C., Nguyen, L. M., Kalagnanam, J., and Julius, A. A. (2026). Revisiting the Generic Transformer: Deconstructing a Strong Baseline for Time Series Foundation Models. arXiv:2602.06909 [cs].

Woo, G., Liu, C., Kumar, A., Xiong, C., Savarese, S., and Sahoo, D. (2024). Unified Training of Universal Time Series Forecasting Transformers. In *Proceedings of the 41st International Conference on Machine Learning*, pages 53140–53164. PMLR.

Wu, S., Huang, J., Feng, W., Li, B., Zhang, X., Meng, E., Li, D., Lou, J., and Ng, S.-K. (2026). Wave-MoE: A Wavelet-Enhanced Mixture-of-Experts Foundation Model for Time Series Forecasting.

Xie, S., Feofanov, V., Alonso, M., Odonnat, A., Zhang, J., and Redko, I. (2025). CauKer: Classification Time Series Foundation Models Can Be Pretrained on Synthetic Data only. In *1st ICML Workshop on Foundation Models for Structured Data*.

Xue, H. and Salim, F. D. (2023). PromptCast: A New Prompt-based Learning Paradigm for Time Series Forecasting. arXiv:2210.08964 [stat].

Yamaguchi, Y., Suemitsu, I., and Wei, W. (2025). CITRAS: Covariate-Informed Transformer for Time Series Forecasting. arXiv:2503.24007 [cs].

Yan, P., Abdulkadir, A., Luley, P.-P., Rosenthal, M., Schatte, G. A., Grewe, B. F., and Stadelmann, T. (2024). A Comprehensive Survey of Deep Transfer Learning for Anomaly Detection in Industrial Time Series: Methods, Applications, and Directions. *IEEE Access*, 12:3768–3789.

Zhang, Y. and Yan, J. (2022). Crossformer: Transformer Utilizing Cross-Dimension Dependency for Multivariate Time Series Forecasting.

Zhao, H., Li, H., Xuan, Y., Bao, S., Cidan, Y., Liu, Y., Li, C., and Yao, M. (2023). Investigating the critical influencing factors of snowmelt runoff and development of a mid-long term snowmelt runoff forecasting. *Journal of Geographical Sciences*, 33(6):1313–1333.

Appendix

Table of Contents

A Extended results **16**

B Additional experiments and analyses **18**
B.1 Reporting under the fev-bench protocol 18
B.2 GIFT-Eval under a matched training corpus 18
B.3 Streaming: per-patch latency and memory 18
B.4 Streaming: covariate contribution 19
B.5 Architectural ablations on GIFT-Eval 19
B.6 Scaling the number of covariates 19
B.7 Experimental setup for the streaming, ablation and covariate-scaling results 20

C Asymmetric group attention: leakage derivation and details **22**
C.1 Notation and group structure 22
C.2 Proof of proposition 1 (sufficiency) 22
C.3 Necessity of the asymmetric mask: two-block leakage counterexample . . . 23
C.4 Comparison to prior cross-variate asymmetric attention designs 24
C.5 Comparison to multivariate and covariate-aware TSFMs 24

D Binary-aware tail-compressing scaler **25**

E Training setup **26**
E.1 Long-context posttraining 26

F Pre-training corpus **28**

G Synthetic multivariate coupling: background and design **31**

H Code and model checkpoints **32**

I Societal impact **33**

A Extended results

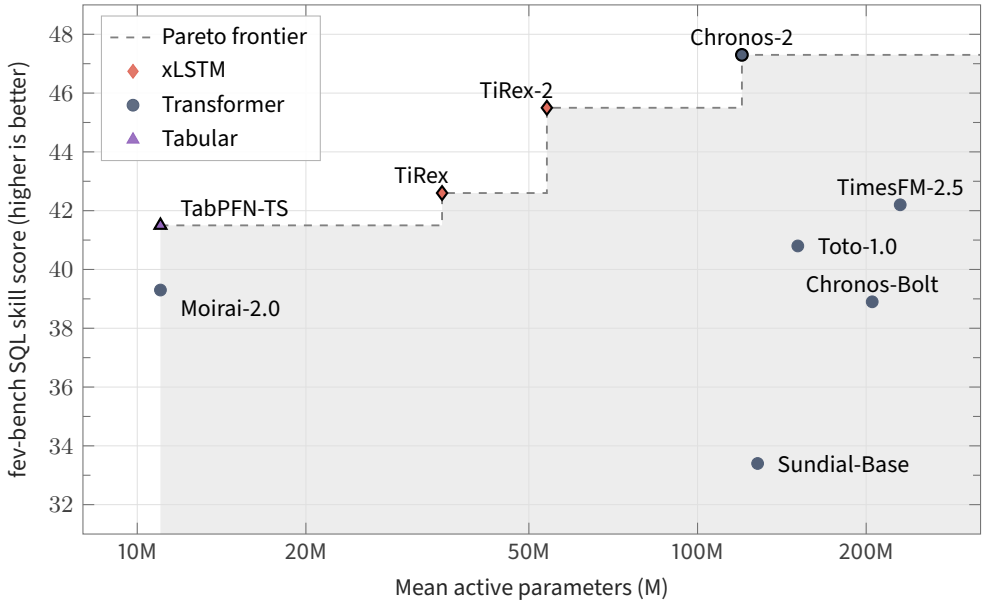


Figure 7: fev-bench SQL skill score against Seasonal Naive versus mean active parameters across nine zero-shot time series foundation models (TSFMs). Higher is better, so Pareto-optimal models (TabPFN-TS, TiRex, TiRex-2, Chronos-2) form the upper-left frontier (dashed staircase, black outlines); the shaded area marks the dominated subset of the parameter–skill plane. Chronos-2 attains the highest skill score, at $\sim 2.2\times$ the mean active parameters of TiRex-2 and $\sim 3\times$ its median runtime (0.838 against 0.273 s per 100 tasks); TiRex-2 is the strongest model below 100M active parameters. For TiRex-2 the variate mixer is skipped on purely univariate inputs (Sec. 3), so we report the task-weighted expectation $\mathbb{E}_t[P_{\text{active}}] = P_{\text{base}} + p_{\text{cov}}\, P_{\text{var-mix}} = 38.4 + 0.35 \cdot 44.1 \approx 53.84$M over the $65/100$ univariate and $35/100$ multivariate tasks of fev-bench; in a fully multivariate deployment the count is 82.5M. All other models are dense, so total and active parameter counts coincide.

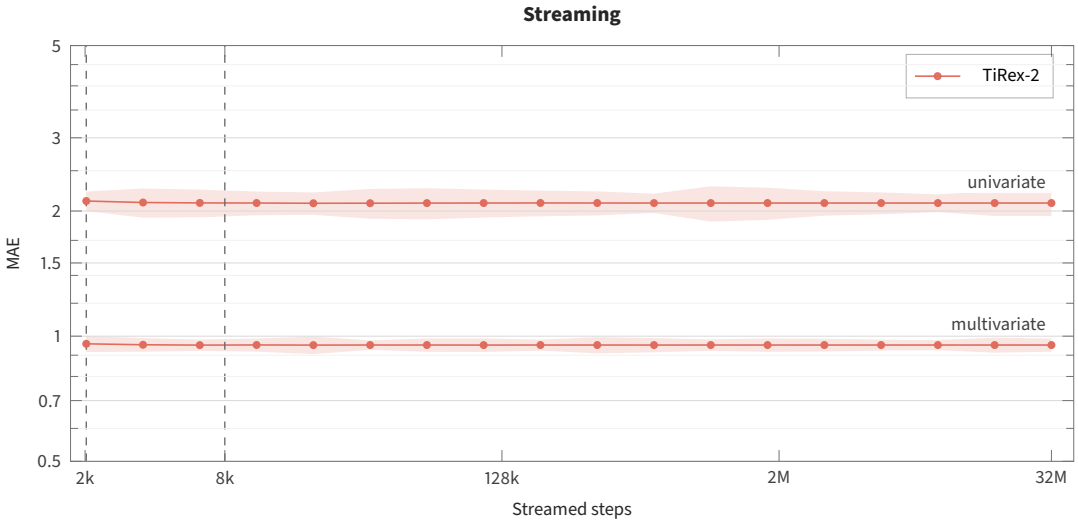


Figure 8: Streaming evaluation: MAE of TiRex-2 as a function of cumulative streamed steps. Univariate and multivariate variants are shown separately. Shaded regions indicate ± 1 standard deviation of the MAE across 8 runs. Vertical dashed lines mark the pre-training context length (2k) and post-training context length (8k).

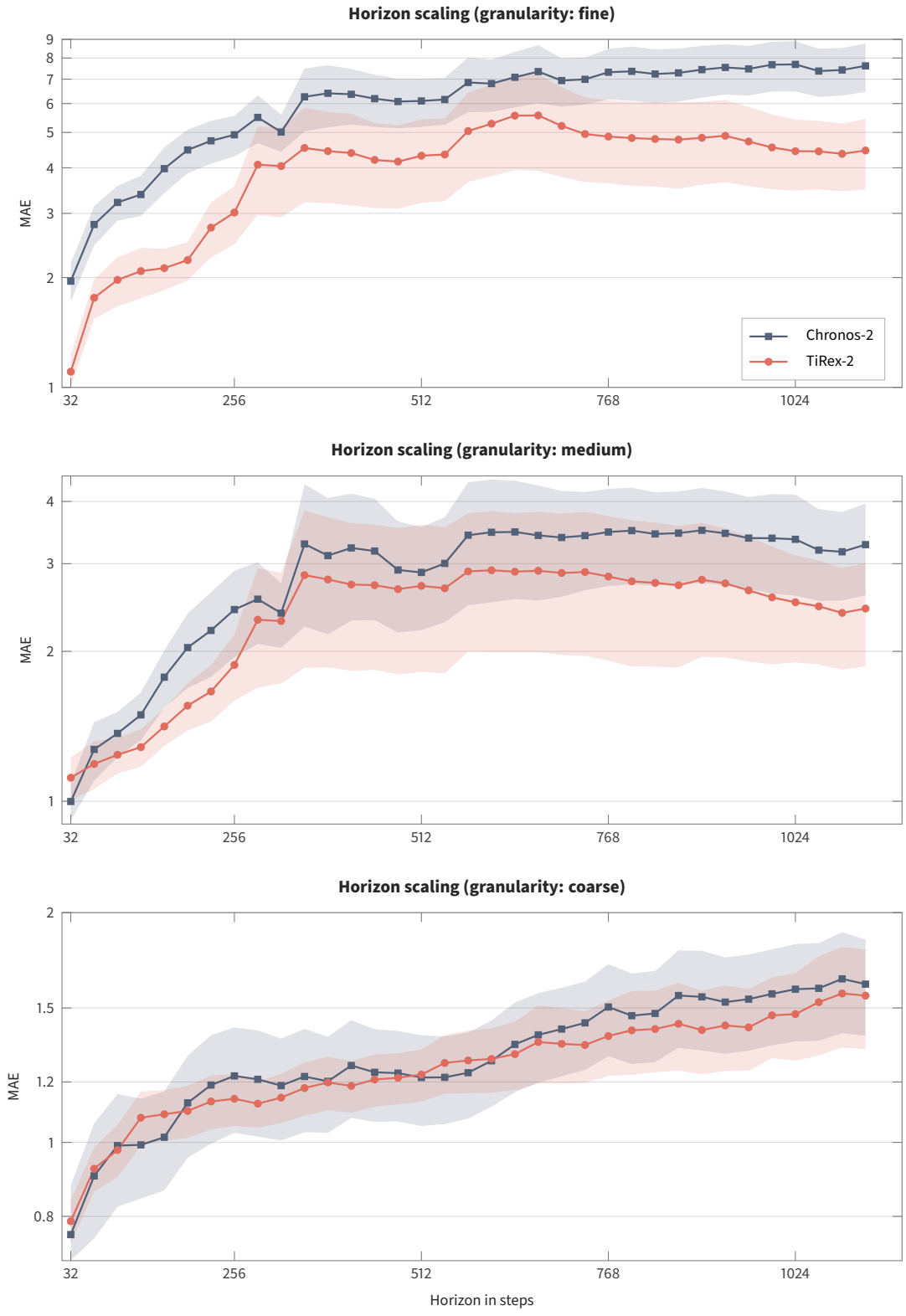


Figure 9: Long-horizon forecasting on `dysts`, MAE vs. forecast horizon on a log-y axis. Subplots show fine, medium, and coarse temporal granularity. Shaded bands indicate ± 1 SEM ($\sigma/\sqrt{N}$, N=135 tasks). TiRex-2 attains lower MAE than Chronos-2 at nearly all horizons. The gap is largest at fine resolution, where it widens with the horizon; at coarse resolution the two models are close, with overlapping SEM bands.

B Additional experiments and analyses

B.1 Reporting under the fev-bench protocol

Table 3 reports fev-bench under the benchmark's own protocol (Shchur et al., 2025). SQL denotes its scaled quantile loss, skill scores are relative to Seasonal Naive.

Table 3: fev-bench (100 tasks) under the benchmark protocol. Win rates are pairwise against the other column, with 95% confidence intervals; skill scores are relative to Seasonal Naive. Higher is better throughout.

Statistic	TiRex-2	Chronos-2
Pairwise win rate, SQL	51.0 (CI 41.0–60.0)	49.0
Pairwise win rate, MASE	44.0 (CI 34.0–54.0)	**56.0**
Skill score, SQL	45.5	**47.3**
Skill score, MASE	33.7	**35.5**

Against the remaining baselines TiRex-2 wins 65–100% of tasks on SQL and 63–99% on MASE, at roughly 2.2× fewer active parameters and roughly 3× lower median runtime (0.273 against 0.838 s per 100 tasks).

Broken down by covariate type (Table 4), TiRex-2 leads Chronos-2 on tasks with past-only covariates on both metrics, while Chronos-2 is ahead when future-known covariates are present. On tasks carrying both types, the three models supporting future-known covariates stay above 37 SQL and 24 MASE skill, while all others fall below 31 and 17.

Table 4: fev-bench skill scores against Seasonal Naive by covariate type, under the benchmark protocol; higher is better, best per column in bold, second best underlined. Subsets: all tasks ($n = 100$), known covariates only (18), past covariates only (12), both types (12), and any dynamic covariate (42); the middle three partition the last.

	Skill score, SQL					Skill score, MASE				
Model	all	known	past	both	any	all	known	past	both	any
Chronos-2	**47.3**	**52.6**	<u>42.4</u>	**42.5**	**47.0**	**35.5**	**43.2**	36.6	**30.3**	**37.9**
TiRex-2	<u>45.5</u>	<u>49.7</u>	**43.9**	37.7	<u>44.8</u>	<u>33.7</u>	<u>39.8</u>	**39.7**	24.6	<u>35.8</u>
TabPFN-TS	41.5	47.1	36.8	<u>40.6</u>	42.5	29.8	37.7	32.7	<u>28.1</u>	33.6
TiRex	42.6	42.7	40.6	29.9	38.7	30.0	31.2	<u>36.8</u>	14.1	28.5
TimesFM-2.5	42.2	41.4	37.4	30.7	37.4	30.2	31.0	33.4	16.1	27.8
Toto-1.0	40.8	37.7	39.0	26.9	35.2	28.2	26.2	34.0	12.8	25.1
FlowState	39.9	40.3	36.8	30.5	36.7	27.8	30.5	32.4	15.9	27.2
Chronos-Bolt	38.9	40.9	34.9	28.6	35.9	26.5	30.3	29.5	13.5	25.6

B.2 GIFT-Eval under a matched training corpus

Two TiRex-2 checkpoints are evaluated: one strictly zero-shot, with all overlapping datasets removed (Section F), and one trained on a corpus matched to Chronos-2. Neither configuration touches the test split.

B.3 Streaming: per-patch latency and memory

At batch size 1, Chronos-2 costs 17.8 ms up to 2k, 107 ms at 2.6×10^4 steps, 1.67 s at 2.1×10^5 and 28.4 s at 1.05M; TiRex-2 recompute costs 206 ms at 9.7×10^4, 956 ms at 4.7×10^5 and 2.07 s at 1.05M. Streaming costs 15.48–15.60 ms throughout, i.e. at 1.05M steps 133× less than recomputation and 1830× less than Chronos-2, which has no incremental path. Streamed memory is 0.318 GB at batch size 1 and 0.42 GB at batch size 64 at all contexts. At batch size 64 recompute reaches 28.8 GB (Chronos-2, 2.0×10^5 steps) and 36.9 GB (TiRex-2, 4.7×10^5 steps).

Table 5: GIFT-Eval. Lower is better throughout; ranks are averaged over tasks. Chronos-2 trains on the GIFT-Eval pre-train splits, so its zero-shot counterpart is Chronos-2-Synth; comparisons are meaningful only within a block. No configuration touches the test split.

Model	MASE	MASE rank	CRPS	CRPS rank
Zero-shot: GIFT-Eval pre-train splits unseen				
TiRex-2 (zero-shot)	**0.697**	**43.18**	**0.478**	**38.28**
Chronos-2-Synth	0.720	60.47	0.496	59.28
Trained on the GIFT-Eval pre-train splits				
TiRex-2 (matched corpus)	**0.678**	**36.20**	**0.467**	**34.68**
Chronos-2	0.698	38.96	0.485	39.50

B.4 Streaming: covariate contribution

Chronos-2 is evaluated under the same streaming regime with its 8k truncation removed, so that it sees the full stream.

Table 6: Streaming to extrapolated context lengths on the AR(1) stream of Section B.7, mean MAE $\pm$ 1 standard deviation across eight runs. UV is univariate, MV adds one informative covariate; lower is better. A value is bold only where the two models' ± 1 SD intervals do not overlap.

	Chronos-2		TiRex-2	
Context	UV	MV	UV	MV
2k	2.110 ± 0.981	0.766 ± 0.192	2.114 ± 0.322	0.959 ± 0.121
8k	2.857 ± 1.336	1.214 ± 0.490	2.092 ± 0.452	0.952 ± 0.084
32k	2.614 ± 1.570	2.588 ± 1.446	2.088 ± 0.485	$\mathbf{0.952 \pm 0.071}$
128k	2.453 ± 0.792	2.456 ± 0.806	2.090 ± 0.456	$\mathbf{0.952 \pm 0.101}$

The UV – MV gap of TiRex-2 runs from 1.155 at 2k to 1.139 at 128k; that of Chronos-2 from 1.344 at 2k to 0.026 at 32k and -0.003 at 128k.

B.5 Architectural ablations on GIFT-Eval

Table 7: Architectural ablations on GIFT-Eval. First row absolute, others deltas against it; lower is better, so a positive delta means the ablation hurt. Three seeds (Table 1: six). GIFT-Eval contains far fewer tasks with informative cross-variate structure than fev-bench, so the group-attention delta is diluted and not comparable to Table 1.

Variant	MASE	CRPS
TiRex-2 (full)	$\mathbf{0.697} \pm 0.0006$	$\mathbf{0.478} \pm 0.0009$
w/o group attention	$+0.0018 \pm 0.0027$	$+0.0032 \pm 0.0028$
w/o binary-aware scaler	-0.0013 ± 0.0011	$+0.0000 \pm 0.0012$
sLSTM only	$+0.0019 \pm 0.0020$	$+0.0022 \pm 0.0007$
Context length 2k	$+0.0156 \pm 0.0011$	$+0.0111 \pm 0.0012$

Table 7 shows architecture ablations evaluated on GIFT-Eval. Relative to the pure sLSTM backbone, the mixed backbone gives a median inference speed-up of 5 to 7%, up to roughly 20% on the smallest datasets, and trains 25% faster. Removing the binary-aware bypass costs 0.4 ± 0.1 SQL skill points on the 43 short-context fev-bench tasks, against 0.1 ± 0.1 overall.

B.6 Scaling the number of covariates

In Tables 8 and 9 we contrast TiRex-2's and Chronos-2's ability of extracting progressively more information from future known covariates. We use M independent AR(1) series whose scaled sum, perturbed by Gaussian noise, forms the target, with k of them revealed as known past and future covariates (Section B.7).

Table 8: Covariate scaling at $M = 128$, construction in Section B.7. MSE of the median forecast over 128 repetitions (SEM ≈ 0.02) and per-call latency in milliseconds; lower is better for both.

k	TiRex-2 MSE	Chronos-2 MSE	TiRex-2 ms	Chronos-2 ms
0	1.080	1.080	**0.78**	1.15
16	1.056	**1.031**	**9.71**	17.36
64	**0.913**	0.998	**44.50**	63.35
128	**0.819**	0.996	**66.20**	135.97

Table 9: Covariate scaling at $M = 256$, MSE of the median forecast (mean $\pm$ 1 SEM). Larger M lowers each covariate's individual contribution, so the two sweeps are not directly comparable at equal k.

k	TiRex-2 MSE	Chronos-2 MSE
0	1.093 ± 0.021	1.095 ± 0.021
16	1.104 ± 0.023	1.114 ± 0.020
32	1.079 ± 0.022	1.073 ± 0.020
96	$\mathbf{0.991 \pm 0.020}$	1.059 ± 0.019
256	$\mathbf{0.932 \pm 0.018}$	1.057 ± 0.019

At $M = 256$ the $96 \to 256$ step still yields a gain for TiRex-2; Chronos-2 is flat from $k \approx 32$ at $M = 128$ and $k \approx 96$ at $M = 256$. In units of covariates' worth of information ($\Delta \cdot M$), TiRex-2 extracts 33 to 41 against Chronos-2's roughly 10. At $M = 1024$ the per-covariate correlation with the target is 0.030 ($0.085 \to 0.060 \to 0.030$ across the three sweeps) and neither model improves over $k = 0$; a $k = 1024$ forward pass costs TiRex-2 0.52 s and 4.0 GB against Chronos-2's 2.28 s and 21.9 GB.

B.7 Experimental setup for the streaming, ablation and covariate-scaling results

This section collects the experimental setups behind the results reported in Sections B.3 to B.6, one paragraph per subsection and in the same order. Every trained arm in this appendix uses three seeds and we report mean $\pm$ std across them; Table 1 in the main text uses up to six for some arms, and each caption states its own count.

Latency and memory (Section B.3). Single NVIDIA A40 with 48 GB, one process on one GPU. Both models run in float32 with `torch.set_float32_matmul_precision("highest")`. One series, context from 192 steps to 1.05×10^6, at batch size 64 and 1; a random-walk series with a fixed seed, so that context length is exactly controllable. We perform 200 timed calls per configuration after 32 untimed warmups, each call bracketed by `torch.cuda.synchronize()` and timed with `time.perf_counter()`, and report the mean end-to-end predict latency; run-to-run standard deviation is at most 0.45 ms throughout. A configuration is dropped once it exhausts the device. Chronos-2 uses patch size 16; TiRex-2 uses patch size 32, so one streamed step ingests one new patch. Latency is reported per call, not per forecast step.

Streaming covariate contribution (Section B.4; Table 6 and Figure 6). A first-order autoregressive target driven by a noisy, lagged covariate: the target responds to the covariate with a delay, so the covariate's relevant information always lies in the target's relative past. The covariate is registered as future-known. Both are ingested patch by patch with a forecast emitted after every update; eight runs per configuration.

Architectural ablations (Section B.5; Tables 2 and 7). Each arm is trained for 700k steps on three seeds, with architecture and evaluation configuration held fixed across arms. fev-bench results follow the benchmark protocol (Shchur et al., 2025) with skill scores against Seasonal Naive over 100 tasks; GIFT-Eval results are raw MASE and CRPS under that benchmark's protocol (Aksu et al., 2024).

Covariate scaling (Section B.6; Tables 8 and 9). We generate M independent AR(1) series $x_{i,t}$ with $\rho = 0.9$, each standardized to unit marginal variance via innovation variance $1 - \rho^2$. Their

sum scaled by $1/\sqrt{M}$ is a unit-variance signal, which we perturb with Gaussian noise of $\sigma = 0.3$ to form the target y; $\sigma^2 = 0.09$ is an irreducible error floor. Revealing k of the M series leaves a Bayes-optimal error of $\sigma^2 + (M - k)/M$. We supply k series as known past and future covariates, nested across k, with context 2048 and horizon 320, and score the median forecast by MSE over 128 repetitions, reporting mean $\pm$ 1 SEM. Both models see identical realizations, so the comparison is paired.

C Asymmetric group attention: leakage derivation and details

This appendix expands on the variate-mixer paragraph of Section 3 and gives (i) the full specification of the asymmetric group attention, (ii) a proof that the asymmetric mask together with the forward-only xLSTM on targets and past covariates is *sufficient* for target-causality (Proposition 1), and (iii) a counterexample showing that the asymmetric mask is *necessary*: dropping the $\mathcal{Y} \cup \mathcal{Z} \to \mathcal{X}$ block opens a concrete two-block leakage path from future targets into earlier target representations.

C.1 Notation and group structure

As in the main paper, a mixer is a map $\mathbf{H} \mapsto \mathbf{H}'$ between token tensors in $\mathbb{R}^{F \times L \times D}$, and one TiRex-2 block applies a time mixer followed by a variate mixer. We write $\mathbf{h}_{f,l}$ for the token of variate f at patch $1 \leqslant l \leqslant L$, and $\mathcal{X}, \mathcal{Y}, \mathcal{Z} \subseteq \{1, \dots, F\}$ for the index sets of the target, past-covariate and future-covariate variates. Depth enters only through block outputs: as in Lemma 1, $\mathbf{H}^n$ is the output of the n-th block and $\mathbf{H}^0$ the output of the input layer; representations inside a block carry no index. The variates are partitioned into groups. Inside the variate mixer we transpose to shape $L \times F \times D$ and apply attention along the F axis independently per patch position l. The additive mask $M \in \{0, -\infty\}^{F \times F}$ is shared across all patch positions and block-diagonal in this group structure: every entry linking variates of different groups is set to $-\infty$, while within a group

$$m_{ij} \;=\; \begin{cases} 0 & \text{if } \neg\big(i \in \mathcal{Y} \cup \mathcal{Z} \;\wedge\; j \in \mathcal{X}\big), \\ -\infty & \text{otherwise,} \end{cases} \tag{2}$$

where $i, j \in \{1, \dots, F\}$ index (query, key). Equivalently, within a group the mask sends each query to its set of admissible keys via

$$i \;\longmapsto\; \begin{cases} \mathcal{X} \cup \mathcal{Y} \cup \mathcal{Z} & \text{if } i \in \mathcal{X}, \\ \mathcal{Y} \cup \mathcal{Z} & \text{if } i \in \mathcal{Y} \cup \mathcal{Z}, \end{cases}$$

so target queries attend to target and covariate keys, whereas covariate queries attend to covariate keys only.

C.2 Proof of proposition 1 (sufficiency)

This subsection establishes that the asymmetric mask (1) together with the forward-only xLSTM on targets and past covariates is sufficient to keep target tokens free of future-target dependencies at every depth; necessity of the mask is treated separately in Section C.3.

Dependency sets. For a token $\mathbf{h}_{f,l}$ of any representation in the stack, let $\mathrm{Dep}(\mathbf{h}_{f,l}) \subseteq \{1, \dots, F\} \times \{1, \dots, L\}$ be the set of pairs (f', l') such that $\mathbf{h}_{f,l}$ is a non-trivial function of the input token $\mathbf{h}^0_{f',l'}$. Define the temporal receptive field $I_f(l) = \{1, \dots, L\}$ if $f \in \mathcal{Z}$ and $I_f(l) = \{1, \dots, l\}$ otherwise, and the variate receptive set $S_f = \mathcal{X} \cup \mathcal{Y} \cup \mathcal{Z}$ for $f \in \mathcal{X}$ and $S_f = \mathcal{Y} \cup \mathcal{Z}$ for $f \in \mathcal{Y} \cup \mathcal{Z}$. I_f encodes the directionality of the time mixer (forward-only on targets and past covariates, bidirectional on future covariates), and S_f encodes the asymmetric mask (1) (covariate queries cannot read target keys). For a mixer $\mathbf{H} \mapsto \mathbf{H}'$ this gives

$$\mathrm{Dep}\big(\mathbf{h}'_{f,l}\big) \subseteq \bigcup_{l' \in I_f(l)} \mathrm{Dep}\big(\mathbf{h}_{f,l'}\big) \qquad \text{(time mixer)},$$

$$\mathrm{Dep}\big(\mathbf{h}'_{f,l}\big) \subseteq \bigcup_{f' \in S_f} \mathrm{Dep}\big(\mathbf{h}_{f',l}\big) \qquad \text{(variate mixer)},$$

where the variate-mixer union is at fixed patch l since the mixer acts independently per patch. Normalization, the token-wise MLPs and the residual connections act on each token separately and add nothing: the residual term contributes $\mathrm{Dep}(\mathbf{h}_{f,l})$, which both unions already contain since $l \in I_f(l)$ and $f \in S_f$.

Invariant. Call a representation *admissible* if its tokens satisfy

$$(\mathbf{C}) \;\; f \in \mathcal{X},\; (f', l') \in \mathrm{Dep}\big(\mathbf{h}_{f,l}\big) \text{ and } f' \in \mathcal{X} \implies l' \leqslant l,$$

$$(\mathbf{I}) \;\; f \in \mathcal{Y} \cup \mathcal{Z} \implies \mathrm{Dep}\big(\mathbf{h}_{f,l}\big) \cap (\mathcal{X} \times \{1, \dots, L\}) = \emptyset,$$

i.e. target tokens depend on no later target input, and covariate tokens depend on no target input at all. We show that the input is admissible and that every mixer maps an admissible representation $\mathbf{H}$ to an admissible $\mathbf{H}'$.

Input layer: $\mathrm{Dep}(\mathbf{h}^0_{f,l}) = \{(f,l)\}$ satisfies both. *Time mixer*: for $f \in \mathcal{X}$, $I_f(l) \subseteq \{1,\dots,l\}$, so every target-index pair inherited from $\mathbf{H}$ has $l' \leqslant l$, preserving $(\mathbf{C})$. For $f \in \mathcal{Y} \cup \mathcal{Z}$, the union runs over tokens of the same covariate, and $(\mathbf{I})$ carries over unchanged. *Variate mixer*: for $f \in \mathcal{X}$, the union runs over $f' \in S_f = \mathcal{X} \cup \mathcal{Y} \cup \mathcal{Z}$ at fixed patch l; contributions from keys $f' \in \mathcal{X}$ satisfy $(\mathbf{C})$ in $\mathbf{H}$ (so any target-index pair has $l' \leqslant l$), while contributions from $f' \in \mathcal{Y} \cup \mathcal{Z}$ contain no target-index pair by $(\mathbf{I})$, preserving $(\mathbf{C})$. For $f \in \mathcal{Y} \cup \mathcal{Z}$, $S_f = \mathcal{Y} \cup \mathcal{Z}$ excludes $\mathcal{X}$, and $(\mathbf{I})$ holds in $\mathbf{H}$ for every $f' \in S_f$, so $(\mathbf{I})$ is preserved.

Applying this to the $2N$ mixers of the stack in order, every representation in the stack is admissible, in particular every block output $\mathbf{H}^n$, $n \leqslant N$. Property $(\mathbf{C})$ for $\mathbf{H}^n$ is Proposition 1. □

Intuitively, $(\mathbf{I})$ is what makes the bidirectional time mixer on future covariates safe: covariate tokens never accumulate target information, so the reverse pass cannot transport target content from a later patch back to an earlier one.

C.3 Necessity of the asymmetric mask: two-block leakage counterexample

The sufficiency proof in Section C.2 leaves open whether the $\mathcal{Y} \cup \mathcal{Z} \to \mathcal{X}$ block in Eq. (2) is actually needed, or whether a symmetric variate mixer would already preserve target-causality. We resolve this by exhibiting an explicit leakage path: without the $\mathcal{Y} \cup \mathcal{Z} \to \mathcal{X}$ block, a target representation at patch l depends on target inputs at positions $l' > l$, even within a single forward pass and without any autoregressive sampling. The leakage path uses one symmetric variate mixer, the bidirectional time mixer of the next block, and a second variate mixer; three components that are individually well-defined but, in combination, route information backwards along the time axis.

Assume the series contains at least one future-covariate variate (index $z \in \mathcal{Z}$) and one target variate (index $x \in \mathcal{X}$) in the same group, and follow three consecutive mixers, the primes counting the mixers applied so far:

$$\mathbf{H} \xrightarrow{\text{variate mixer}} \mathbf{H}' \xrightarrow{\text{time mixer}} \mathbf{H}'' \xrightarrow{\text{variate mixer}} \mathbf{H}'''.$$

Step 1 (symmetric variate mixer). With a symmetric mask, the future-covariate token at position l aggregates over all variates in the group, including the target variate x:

$$\mathbf{h}'_{z,l} \;=\; \mathbf{h}_{z,l} + \sum_{v \in \mathcal{X} \cup \mathcal{Y} \cup \mathcal{Z}} \alpha_{z,v,l}\, \mathbf{h}_{v,l} \quad \text{with } \alpha_{z,x,l} \neq 0 \text{ in general.}$$

Hence $\mathbf{h}'_{z,l}$ carries information about $\mathbf{h}_{x,l}$.

Step 2 (bidirectional time mixer, next block). The future-covariate stream is processed by both a forward and a reverse xLSTM. The reverse pass writes into future-covariate tokens at positions $l' < l$ a representation that is a function of all $\mathbf{h}'_{z,l''}$ with $l'' \geqslant l'$, including $l'' = l$:

$$\mathbf{h}''_{z,l'} \;=\; \phi\big(\mathbf{h}'_{z,l'},\, \mathbf{h}'_{z,l'+1},\, \dots,\, \mathbf{h}'_{z,L}\big),$$

where ϕ collects the forward output, the reverse output, and their linear fusion. By Step 1, ϕ is a function of $\mathbf{h}_{x,l}$, the target token at the *later* position l.

Step 3 (variate mixer, next block). With a symmetric variate mixer, the target token at position $l' < l$ attends to all variates of its group at the same patch index, including the future covariate z:

$$\mathbf{h}'''_{x,l'} \;=\; \mathbf{h}''_{x,l'} + \alpha'_{x,z,l'}\, \mathbf{h}''_{z,l'} + \sum_{v \neq z} \alpha'_{x,v,l'}\, \mathbf{h}''_{v,l'},$$

and by Step 2 the middle term is a function of $\mathbf{h}_{x,l}$ with $l > l'$. Composing the three steps, the target representation at the earlier patch l' is a non-trivial function of the target input at the later patch l. During training this leak corrupts the supervision signal: the loss at position l' can be reduced by copying from position $l > l'$, and at inference (where future targets are absent) the resulting representation distribution is shifted away from training. Single-pass streaming forecasts are then no longer well-defined either.

How the asymmetric mask breaks the chain. The mask in Eq. (2) forbids exactly the $\mathcal{Y} \cup \mathcal{Z} \to \mathcal{X}$ direction by setting $\alpha_{z,x,l} = 0$ for all l, thus $\mathbf{h}'_{z,l}$ is independent of any target token, and steps 2–3 cannot inject target information at later positions back into earlier target tokens.

C.4 Comparison to prior cross-variate asymmetric attention designs

Several recent models also impose an asymmetric target↔covariate information flow, but in different regimes. TimeXer (Wang et al., 2024) compresses each covariate series into a single learnable token and lets target queries cross-attend to those tokens. TiRex-2, instead, attends at every patch position, preserving the temporal alignment between target and covariate features. TimeXer is moreover a task-specific model and is not evaluated as a zero-shot foundation model. Timer-XL (Liu et al., 2024) flattens variates and time into a single token sequence and combines causal intra-variate attention with covariate-asymmetric attention inside one block. Because it is fully causal along time, it cannot exploit future covariates. Our design avoids this limitation by treating future covariates bidirectionally in the time mixer. CITRAS (Yamaguchi et al., 2025) likewise factorises time and variate attention and forms variate attention with the target as query and covariates as keys/values. For future covariates it pairs key at patch l with value at patch $l+1$ to look one step ahead. CITRAS is again task-specific and its one-step shift is a heuristic substitute for the proper bidirectional treatment of future covariates that we adopt.

C.5 Comparison to multivariate and covariate-aware TSFMs

Table 10 positions TiRex-2 against existing multivariate time series foundation models and covariate-aware single-target models along three axes: (i) whether the model can ingest past- and (ii) future covariates, and (iii) whether its architecture preserves *target causality*, i.e., whether target representations at patch l are guaranteed to be independent of target inputs at patches $l' > l$. To our knowledge TiRex-2 is the first TSFM to provide support for future covariates and stay causal on the target variate(s). This is a prerequisite for the streaming inference of Sec. 3.3 since otherwise target states at earlier patches would change as new time steps arrive.

Table 10: Past and future-covariate support and target-causality preservation across multivariate and covariate-aware time series foundation models. ✓ = supported/preserved, × = not supported/not preserved.

Model	Past covariates	Future covariates	Target causality
TiRex-2 (ours)	✓	✓	✓
Multivariate TSFMs			
MOIRAI (Woo et al., 2024)	✓	✓	×
MOIRAI-MOE (Liu et al., 2025a)	✓	×	×
Toto (Cohen et al., 2025)	✓	×	✓
Chronos-2 (Ansari et al., 2025)	✓	✓	×
TabPFN-TS (Hoo et al., 2024)	×	✓	×
GTT (Feng et al., 2024)	✓	×	×
MORPHEUS (Patil et al., 2025)	✓	✓	×
Covariate-aware single-target TSFMs			
COSMIC (Auer et al., 2025a)	✓	✓	×
TimesFM-ICF (Faw et al., 2025)	✓	×	✓

D Binary-aware tail-compressing scaler

This appendix expands on the input-layer paragraph in Sec. 3 and gives (i) the motivation for the binary-aware bypass, (ii) the full forward and inverse transforms with numerical clipping, and (iii) the per-variate statistics used by the scaler.

Given a single variate $\mathbf{x} \in \mathbb{R}^T$ with index set of observed entries $\mathcal{V} \subseteq \{1, \dots, T\}$, we compute per-variate statistics over the full context window:

$$\hat{\mu} = \frac{1}{|\mathcal{V}|} \sum_{t \in \mathcal{V}} x_t, \qquad \hat{\sigma} = \max\left(\sqrt{\frac{1}{|\mathcal{V}|} \sum_{t \in \mathcal{V}} (x_t - \hat{\mu})^2},\ \epsilon \right), \tag{3}$$

where $\epsilon > 0$ is a small constant that prevents division by zero for near-constant or sparse variates.

Why binary signals need a bypass. Naive standardization is problematic for sparse binary signals. Consider the rare-positive case $\bar{p} \ll 1$ (the rare-negative case $\bar{p} \to 1$ is symmetric by exchanging the roles of the two levels): for $X \in \{0, 1\}$ with $\mathbb{P}(X = 1) = \bar{p}$, the empirical statistics become $\hat{\mu} \approx \bar{p}$ and $\hat{\sigma} \approx \sqrt{\bar{p}(1 - \bar{p})} \approx \sqrt{\bar{p}}$. The two standardized levels are then

$$z_0 \;=\; \frac{0 - \hat{\mu}}{\hat{\sigma}} \;\approx\; \frac{-\bar{p}}{\sqrt{\bar{p}}} \;=\; -\sqrt{\bar{p}}, \qquad z_1 \;=\; \frac{1 - \hat{\mu}}{\hat{\sigma}} \;\approx\; \frac{1 - \bar{p}}{\sqrt{\bar{p}}} \;=\; \frac{1}{\sqrt{\bar{p}}} - \sqrt{\bar{p}}, \tag{4}$$

yielding a gap

$$z_1 - z_0 \;\approx\; \frac{1}{\sqrt{\bar{p}}} - \sqrt{\bar{p}} - \left(-\sqrt{\bar{p}}\right) \;=\; \frac{1}{\sqrt{\bar{p}}} \tag{5}$$

that grows without bound as $\bar{p} \to 0$. The representation of both levels therefore depends on the context-window sparsity rather than the binary semantics of the signal. To avoid this, the affine transformation is bypassed for detected binary variates:

$$\mu = (1 - b)\, \hat{\mu}, \qquad \sigma = (1 - b)\, \hat{\sigma} + b, \qquad b = \mathbf{1}\Big[\forall\, t \in \mathcal{V} :\ x_t \in \{0, 1\} \Big]. \tag{6}$$

This gated parameterization replaces $(\hat{\mu}, \hat{\sigma})$ with $(0, 1)$ when $b = 1$, so the affine transform reduces to the identity without branching. The construction preserves the canonical $\{0, 1\}$ encoding and yields a stable, sparsity-invariant input regardless of the class balance observed in the context.

Forward and inverse transforms. The forward transformation standardizes and applies an $\operatorname{arcsinh}$ tail compression to non-binary variates while leaving binary ones unchanged; the inverse undoes the compression and rescales back to the original domain:

$$\tilde{x}_t = (1 - b)\ \operatorname{arcsinh}\!\left(\tfrac{x_t - \mu}{\sigma}\right) + b\, x_t, \qquad \hat{x}_t = (1 - b)\, \Big(\sigma\, \sinh\big(\operatorname{clip}_c(\tilde{x}_t)\big) + \mu \Big) + b\, \tilde{x}_t, \tag{7}$$

where $\operatorname{clip}_c(z) = \max(-c, \min(c, z))$ bounds the input to $\sinh$ to prevent its exponential growth from exceeding the representable range of the output datatype and to suppress implausibly large outlier predictions. We use a fixed bound $c = 20$ in training and at inference. Since the outputs of the model in the normalized domain typically stay well below this bound, the clip is only very rarely active in practice and acts mainly as a safeguard against overflow, as $\sinh(20) \approx 2.4 \times 10^8$ is far below the range of the output datatype. A drop-in alternative is to compute c per sample from the per-variate statistics (μ, σ) as

$$c \;=\; \alpha \cdot \operatorname{arcsinh}\left(\frac{x_{\max}^{(\mathrm{dtype})} - \mu}{\sigma} \right), \tag{8}$$

with safety factor $\alpha \in (0, 1]$, which guarantees $\sigma \sinh(c) + \mu \leqslant x_{\max}^{(\mathrm{dtype})}$ and, for $\alpha < 1$, additionally suppresses implausibly large outlier predictions. In our experiments this alternative yields virtually identical results, since model outputs typically stay well below either bound and the clip is only very rarely active.

E Training setup

We pretrain TiRex-2 for 700,000 optimizer steps on 2 NVIDIA H100 GPUs in bf16-mixed precision. Training takes approximately 50 hours wall-clock time at an effective batch size of 64 (per-GPU batch size of 32, no gradient accumulation). Optimization uses AdamW (Loshchilov and Hutter, 2019) with a weight decay of 0.01 and gradients are value-clipped at 1.0 to stabilize the update magnitude in the presence of the heavy-tailed sample-loss distribution typical of large-scale time-series corpora.

Learning-rate schedule. The learning rate follows the OneCycle schedule (Smith and Topin, 2019) as implemented by `torch.optim.lr_scheduler.OneCycleLR` with cosine annealing in both phases (`anneal_strategy='cos'`). The schedule is parameterized by a peak learning rate $\eta_{\max} = 1.2 \cdot 10^{-3}$, an initial divisor $d_0 = 50$ and a final divisor $d_f = 10^4$, which fix the boundary values

$$\eta_0 = \frac{\eta_{\max}}{d_0} = 2.4 \cdot 10^{-5}, \qquad \eta_f = \frac{\eta_0}{d_f} = \frac{\eta_{\max}}{d_0\, d_f} = 2.4 \cdot 10^{-9}. \tag{9}$$

With total training steps $T = 700{,}000$ and warmup fraction $\rho = 0.05$ (`pct_start`), the warmup phase ends at step $T_w = \rho T = 35{,}000$. Letting t denote the current optimizer step and defining the normalized phase progress

$$s_w(t) = \frac{t}{T_w}, \qquad s_a(t) = \frac{t - T_w}{T - T_w}, \tag{10}$$

the schedule is given by

$$\eta(t) = \begin{cases} \eta_{\max} + \frac{1}{2}\big(\eta_0 - \eta_{\max}\big)\big(1 + \cos(\pi\, s_w(t))\big), & 0 \leqslant t \leqslant T_w \quad \text{(cosine warmup)}, \\ \eta_f + \frac{1}{2}\big(\eta_{\max} - \eta_f\big)\big(1 + \cos(\pi\, s_a(t))\big), & T_w < t \leqslant T \quad \text{(cosine anneal)}. \end{cases} \tag{11}$$

Hence η ramps cosine-smoothly from η_0 to $\eta_{\max}$ over the first 5% of training and is then cosine-annealed back to $\eta_f \approx 0$ over the remaining 95%. The cosine-shaped warmup is particularly well suited when continuing from a pretrained checkpoint: the gradually increasing step size lifts the parameters out of the local optimum they occupy without disrupting the geometry of the learned representations, enabling stable continued training before the main annealing phase takes over.

Model configuration. The backbone consists of $N = 12$ residual blocks alternating mLSTM and sLSTM, with embedding dimension $d_{\text{model}} = 512$, $h = 4$ heads per layer, a feed-forward expansion to dimension $d_{\text{ff}} = 2{,}048$, QK-normalization, and dropout $p = 0.1$ applied within each block. The model operates on a context of $L_{\text{ctx}} = 2{,}048$ time steps and produces forecasts over a horizon of $L_{\text{pred}} = 320$ steps, with both input and output patch sizes set to $P_{\text{in}} = P_{\text{out}} = 32$. This yields $L_{\text{ctx}}/P_{\text{in}} = 64$ input tokens and $L_{\text{pred}}/P_{\text{out}} = 10$ output tokens per sample.

Training objective. We minimize the quantile loss over $K = 99$ equidistant quantile levels $\{\tau_k = k/100\}_{k=1}^{99}$, augmented with soft sample-impact capping to limit the influence of individual high-loss samples on the gradient. This prevents pathological tail samples (e.g. rare regime shifts or outliers in the synthetic mixtures) from dominating the update direction without discarding their information content entirely.

E.1 Long-context posttraining

To adapt TiRex-2 to longer sequences, we follow pretraining with a short posttraining phase that extends the context length to $L^{\text{pt}}_{\text{ctx}} = 8{,}192$ and the prediction horizon to $L^{\text{pt}}_{\text{pred}} = 512$, while keeping the input/output patch size fixed at $P_{\text{in}} = P_{\text{out}} = 32$. All architectural hyperparameters (N, d_{model}, h, d_{ff}, dropout, normalization) are inherited unchanged from pretraining.

Posttraining is initialized from the pretraining checkpoint and runs for $T^{\text{pt}} = 100{,}000$ optimizer steps, retaining the optimizer configuration of the pretraining phase (AdamW with weight decay 0.01, gradient value-clipping at 1.0, bf16-mixed precision) and the same training objective (quantile loss over $K = 99$ levels with soft sample-impact capping). The learning-rate schedule is again OneCycle but with a peak value reduced by an order of magnitude to $\eta^{\text{pt}}_{\max} = 1.2 \cdot 10^{-4}$, reflecting the fine-tuning character of this phase and preventing the model from drifting away from the pretrained solution.

While trained with a fixed prediction horizon of $L_{\text{pred}}^{\text{pt}} = 512$, the model is not limited to it at deployment: on dysts we forecast horizons of up to $1{,}056$ steps (Figure 6). Likewise, the recurrent state of the xLSTM backbone is not limited to the trained context length: via streaming, we evaluate contexts of up to $32{,}000{,}000$ steps, i.e. $4{,}000\times$ longer than the $8{,}192$ seen in posttraining.

Table 11: Training hyperparameters of TiRex-2 for the pretraining and long-context posttraining phases.

	Pretraining	Posttraining
Optimization		
Optimizer	AdamW (Loshchilov and Hutter, 2019)	
Weight decay	0.01	
Gradient clipping (value)	1.0	
Precision	bf16-mixed	
Hardware	$2\times$ NVIDIA H100	
Total steps	700,000	100,000
Effective batch size	64 (per-GPU 32, no grad. accumulation)	
Wall-clock time	≈ 50 h	
Initialization	random	
Learning-rate schedule (OneCycle)		
Peak LR $\eta_{\max}$	$1.2 \cdot 10^{-3}$	$1.2 \cdot 10^{-4}$
Warmup fraction	5%	
Initial divisor	50	
Final divisor	10^4	
Architecture		
Blocks N	12 (alternating mLSTM / sLSTM)	
Embedding dim d_{model}	512	
Heads h	4	
FFN dim d_{ff}	2,048	
Normalization	QK-norm	
Dropout	0.1	
Output clamp	Clamp of $\sinh$ input to ± 20 during re-scaling	
Quantile levels K	99	
Sequence layout		
Context length L_{ctx}	2,048	8,192
Prediction horizon L_{pred}	320	512
Input / output patch size	32 / 32	
Input / output tokens	64 / 10	256 / 16
Streamed context length (tested)	32M steps	
Objective		
Loss	Quantile loss with soft sample-impact capping	
Quantile levels K	99 (equidistant in $(0, 1)$)	

F Pre-training corpus

The univariate data sources of our pre-training corpora are inherited from TiRex and TiRex-1.1 (Auer et al., 2025b). We reproduce the description here for self-containedness and to make the boundary between the inherited univariate sources and the multivariate extension introduced in Section 3.4 explicit. In contrast to TiRex, we do *not* apply the TSMixup augmentation of Ansari et al. (2024): its role as a univariate mixing prior is subsumed by our coupling mechanism (Section 3.4), which both generates and mixes series under a richer set of cross-variate structures.

The univariate corpus comprises three components:

1. **Chronos training data (~ 30 M series).** We use the public training collection assembled by Ansari et al. (2024) as a source of real-world univariate time series drawn from heterogeneous domains. Series are z-score normalized per sample and used directly, without TSMixup-style convex combination at this stage.
2. **Synthetic Gaussian-process series (~ 15 M).** We adopt the GP-based synthetic data pipeline of TiRex (Auer et al., 2025b) verbatim, which itself is an extension of KernelSynth (Ansari et al., 2024): each series is drawn from a zero-mean Gaussian process whose kernel is randomly composed from a fixed bank under $\{+, \times\}$. We refer to Auer et al. (2025b) for the full specification.
3. **GIFT-Eval pre-training subset (~ 2.5 M).** A subset of the pre-training corpus released alongside GIFT-Eval (Aksu et al., 2024); concrete dataset filtering is handled per evaluation benchmark as described below.

Sampling protocol. At each training step a sample is drawn from this pool such that the Chronos component and the synthetic GP component are sampled with equal per-series probability, following the original TiRex pipeline. We additionally mix in, at a fixed rate of 1%, synthetic trajectories generated from `dysts` (Gilpin, 2023) to expose the model to deterministic chaotic dynamics. The corresponding `dysts` evaluation split used in Section 4 is strictly held out and not used for training. To form a multivariate training instance, we draw $V \sim \mathcal{U}\{1, \ldots, 12\}$ univariate series from this pool and pass them through the coupling mechanism of Section 3.4, which imposes cross-variate dependencies on the otherwise independently sampled series; `dysts` trajectories are exempt from this stage and enter the batch as standalone univariate samples. The coupling mechanism thereby both lifts the univariate marginal to a multivariate distribution and replaces the role of TSMixup as a univariate mixing prior.

Per-benchmark zero-leakage corpora. To ensure a strictly zero-shot evaluation, we pre-train two separate model checkpoints, one for each benchmark used in Section 4: for the GIFT-Eval (Aksu et al., 2024) and fev-bench (Shchur et al., 2025) evaluations, we remove from the training corpus any dataset overlapping with the corresponding evaluation benchmark, following each benchmark's own leakage rules. Two datasets requiring particular attention are `chronos_datasets/solar` and `chronos_datasets/solar_1h`. Following the approach of TiRex 1.0, these solar datasets do not constitute leakage for fev-bench, as we train at 5-minute and 1-hour resolutions while the benchmark evaluates at weekly and daily frequencies. For the GIFT-Eval checkpoint, we remove the leaking Alabama subsets of the solar datasets from the training data. The two training corpora therefore differ in their concrete dataset composition, while the sampling protocol described above is applied identically in both cases. Reported scores for each benchmark are produced by the checkpoint trained on the corpus from which that benchmark's data has been excluded. For the matched-corpus comparison in Section B.2, we additionally train a third checkpoint whose corpus extends the GIFT-Eval zero-shot corpus with the GIFT-Eval train splits while the test splits remain excluded.

Table 12: Training datasets — `Salesforce/lotsa_data`.

Dataset	TiRex-2-fev	TiRex-2-GIFT-Eval
lotsa_data/BEIJING_SUBWAY_30MIN	✓	✓
lotsa_data/HZMETRO	✓	✓
lotsa_data/LOS_LOOP	✓	✓
lotsa_data/PEMS03	✓	✓
lotsa_data/PEMS04	✓	✓
lotsa_data/PEMS07	✓	✓
lotsa_data/PEMS08	✓	✓
lotsa_data/PEMS_BAY	✓	✓
lotsa_data/Q-TRAFFIC	✓	✓
lotsa_data/SHMETRO	✓	✓
lotsa_data/azure_vm_traces_2017	✓	—
lotsa_data/bdg-2_bear	✓	—
lotsa_data/bdg-2_fox	✓	—
lotsa_data/bdg-2_panther	✓	—
lotsa_data/bdg-2_rat	✓	—
lotsa_data/beijing_air_quality	✓	✓
lotsa_data/borealis	✓	✓
lotsa_data/borg_cluster_data_2011	✓	—
lotsa_data/buildings_900k	✓	✓
lotsa_data/bull	—	✓
lotsa_data/cdc_fluview_ilinet	✓	✓
lotsa_data/cdc_fluview_who_nrevss	✓	✓
lotsa_data/china_air_quality	✓	✓
lotsa_data/cockatoo	—	✓
lotsa_data/covid19_energy	—	✓
lotsa_data/covid_mobility	—	✓
lotsa_data/elf	—	✓
lotsa_data/extended_web_traffic_with_missing	—	✓
lotsa_data/favorita_sales	✓	✓
lotsa_data/favorita_transactions	—	✓
lotsa_data/gfc12_load	—	✓
lotsa_data/gfc14_load	—	✓
lotsa_data/gfc17_load	—	✓
lotsa_data/godaddy	✓	✓
lotsa_data/hog	—	✓
lotsa_data/ideal	✓	✓
lotsa_data/kaggle_web_traffic_weekly	✓	✓
lotsa_data/kdd2022	—	✓
lotsa_data/largest_2017	✓	✓
lotsa_data/largest_2018	✓	✓
lotsa_data/largest_2019	✓	✓
lotsa_data/largest_2020	✓	✓
lotsa_data/largest_2021	✓	✓
lotsa_data/lcl	✓	—
lotsa_data/london_smart_meters_with_missing	—	✓
lotsa_data/monash_m3_other	—	✓
lotsa_data/oikolab_weather	—	✓
lotsa_data/pdb	—	✓
lotsa_data/pedestrian_counts	—	✓
lotsa_data/project_tycho	—	✓
lotsa_data/residential_load_power	✓	✓
lotsa_data/residential_pv_power	✓	✓
lotsa_data/sceaux	✓	✓
lotsa_data/smart	✓	✓
lotsa_data/solar_power	✓	✓
lotsa_data/spain	—	✓
lotsa_data/subseasonal	✓	✓
lotsa_data/subseasonal_precip	✓	✓
lotsa_data/sunspot_with_missing	—	✓
lotsa_data/tourism_monthly	—	✓
lotsa_data/vehicle_trips_with_missing	—	✓
lotsa_data/wiki-rolling_nips	—	✓
lotsa_data/wind_power	✓	✓

Table 13: Training datasets — autogluon/chronos_datasets.

Dataset	TiRex-2-fev	TiRex-2-GIFT-Eval
chronos_datasets_extra/brazilian_cities_temperature	✓	✓
chronos_datasets/dominick	—	✓
chronos_datasets/electricity_15min	✓	—
chronos_datasets/exchange_rate	—	✓
chronos_datasets/m4_daily	✓	—
chronos_datasets/m4_hourly	✓	—
chronos_datasets/m4_monthly	✓	—
chronos_datasets/m4_weekly	✓	—
chronos_datasets/mexico_city_bikes	✓	✓
chronos_datasets/monash_australian_electricity	—	✓
chronos_datasets/monash_cif_2016	—	✓
chronos_datasets/monash_electricity_hourly	✓	—
chronos_datasets/monash_electricity_weekly	✓	—
chronos_datasets/monash_fred_md	—	✓
chronos_datasets/monash_kdd_cup_2018	✓	—
chronos_datasets/monash_london_smart_meters	✓	✓
chronos_datasets/monash_m1_monthly	—	✓
chronos_datasets/monash_m1_quarterly	—	✓
chronos_datasets/monash_m1_yearly	—	✓
chronos_datasets/monash_m3_monthly	—	✓
chronos_datasets/monash_m3_quarterly	—	✓
chronos_datasets/monash_m3_yearly	—	✓
chronos_datasets/monash_nn5_weekly	—	✓
chronos_datasets/monash_pedestrian_counts	✓	✓
chronos_datasets/monash_rideshare	✓	✓
chronos_datasets/monash_temperature_rain	✓	—
chronos_datasets/monash_tourism_quarterly	—	✓
chronos_datasets/monash_tourism_yearly	—	✓
chronos_datasets/monash_traffic	—	✓
chronos_datasets/monash_weather	—	✓
chronos_datasets/nn5	—	✓
chronos_datasets/solar	✓	✓
chronos_datasets/solar_1h	✓	✓
chronos_datasets_extra/spanish_energy_and_weather	✓	✓
chronos_datasets/taxi_1h	✓	✓
chronos_datasets/taxi_30min	✓	✓
chronos_datasets/uber_tlc_daily	✓	✓
chronos_datasets/uber_tlc_hourly	✓	✓
chronos_datasets/ushcn_daily	✓	✓
chronos_datasets/weatherbench_daily	✓	✓
chronos_datasets/weatherbench_hourly_* (11 vars)	✓	✓
chronos_datasets/weatherbench_weekly	✓	✓
chronos_datasets/wiki_daily_100k	✓	✓
chronos_datasets/wind_farms_daily	✓	✓
chronos_datasets/wind_farms_hourly	✓	✓

G Synthetic multivariate coupling: background and design

This appendix provides the conceptual background for the synthetic coupling pipeline introduced in Section 3.4. We specify the phenomena the pipeline is designed to cover during training of TiRex-2, the rationale for their inclusion, and the design principles underlying the construction.

Motivation. Real multivariate time series rarely arise from a single generative process. They typically combine several qualitatively distinct sources of cross-variate structure: shared unobserved drivers, lagged dependencies, deterministic functional relationships between variates, and joint stochastic trends. Observational pipelines superimpose further structure (irregular sampling, partial observability, sensor dropouts, quantisation, and asynchronous updates) that is generally inseparable from the underlying signal. A foundation model intended for zero-shot generalisation across domains must therefore handle all of these regimes simultaneously, since the dominant regime is typically unknown a priori and may vary within a single dataset.

Curated multivariate corpora are insufficient to enforce this breadth, whereas large univariate corpora are abundant. We therefore construct multivariate training examples on the fly from the univariate pool by sampling from a broad menu of cross-variate dependency types, combined with a rich set of observational artefacts. The pipeline is not intended to replicate any specific real-world dataset; rather, it ensures that no single inductive bias dominates, so that shared latent factors, lagged causal influence, deterministic covariate relationships, and common stochastic trends are each represented with non-negligible probability in the training distribution.

Design principles. The pipeline is governed by three principles. *Coverage*: the training distribution spans qualitatively distinct dependency types, including indirect (latent-factor) structure, directed lagged causation, and direct pointwise functional relationships. *Compositionality*: each stage is independently randomised per example, yielding a combinatorial enlargement of the effective training distribution relative to any enumeration of fixed scenarios. *Observational realism*: the data presented to TiRex-2 reflects the structural artefacts encountered in applied forecasting, which frequently account for the gap between benchmark and deployment performance.

Pipeline overview. The pipeline comprises three stages, each randomised per example. The first stage diversifies the marginal behaviour of individual univariate series (amplitude profile, dynamic range, and the presence and shape of localised events) so that the joint structure imposed in subsequent stages is not confounded with marginal variability. The second stage introduces cross-variate dependencies by sampling from a collection of coupling mechanisms, each targeting a distinct region of dependency space. The third stage applies a sequence of randomised observational transforms. The mechanism within each stage is itself sampled, so that the joint distribution over training examples is induced by a procedural prior rather than by a single parametric model.

Coverage of dependency phenomena. The coupling mechanisms in the second stage are selected to span complementary forms of multivariate structure. Each constitutes a stylised representative of a class of phenomena; the equations from Section 3.4 are restated below to fix notation.

Indirect coupling through linear mixing,

$$\mathbf{x}_t \;=\; \mathbf{A}\,\mathbf{u}_t,$$

captures the case in which observed series arise as combinations of a smaller number of underlying drivers. The induced correlation structure varies continuously with the spectrum of $\mathbf{A}$, ranging from near-independence to near-collinearity. We sample from several qualitatively distinct spectral regimes to avoid implicit specialisation to any particular effective rank or condition number.

Directed, lagged influence is represented by structural causal models over a randomly sampled directed acyclic graph. In the linear case,

$$x_{f,t} \;=\; \sum_{i \in \mathrm{pa}(f)} \alpha_{if}\, u_{i,\,t-\tau_{if}} \;+\; \varepsilon_{f,t},$$

this class introduces dependencies that instantaneous mixing cannot produce, in particular a temporal asymmetry between cause and effect. The nonlinear extension,

$$x_{f,t} \;=\; h\big(u_{k,\,t-\tau_k}\big) \sum_{i \in \mathrm{pa}(f)} g_{if}\big(u_{i,\,t-\tau_{if}}\big),$$

adds a multiplicative modulation component, which serves as a proxy for threshold-driven and regime-switching dynamics. Such dynamics are characteristic of gated systems and of many physical and economic processes in which one variable controls the activity of another. The pipeline does not commit to a parametric form for the modulating function h or the edge-level nonlinearities g_{if}.

Common stochastic trends are represented by a cointegration mechanism,

$$\mathbf{x}_t \;=\; \mathbf{\Lambda}\,\mathbf{s}_t \;+\; \boldsymbol{\xi}_t,$$

with non-stationary shared drivers $\mathbf{s}$ and stationary deviations $\boldsymbol{\xi}$. The relevant phenomenon is that individual series may drift without bound while specific linear combinations remain stationary, a regime that is poorly approximated by either linear mixing or SCM-style coupling.

Direct functional coupling,

$$x_{f,t} \;=\; \phi_f\big(u_{0,t}\big) + \varepsilon_{f,t},$$

represents the opposite extreme: a covariate is a deterministic transformation of the target up to additive noise. This regime corresponds to calendar features, derived quantities, and sensors that observe a common underlying signal through distinct nonlinearities, and is the most informative for the forecasting target. It complements the indirect dependency classes by exposing the model to covariates carrying near-deterministic information, which is the regime the asymmetric variate mixer is designed to exploit.

The identity and univariate pass-through case,

$$x_{f,t} \;=\; u_{f,t},$$

is included explicitly to preserve univariate forecasting performance. In its absence, exposure to coupled data biases the model toward assuming cross-variate structure even when none is present, which is particularly detrimental in univariate mode, where the variate mixer is bypassed.

Observational layer. The third stage addresses the empirical observation that the gap between a well-specified generative process and the observed data is frequently dominated by observational artefacts. Variates may be reordered arbitrarily across datasets, sampled asynchronously, missing in contiguous blocks (due to joint blackouts or independent sensor faults), observed only up to the forecast origin rather than over the full horizon O, or discretised in value or time. Each of these artefacts must be handled by a deployed forecasting model and is absent from clean synthetic data. The observational layer is therefore treated as a first-class component of the pipeline. The artefact families it covers (variate permutation, smooth time warping, patch masking generalising the contiguous-patch scheme of TiRex (Auer et al., 2025b), partial future observability for a random subset of future covariates, and value- and time-discretisation) were selected on the basis of artefact patterns prevalent in applied forecasting. Partial future observability prevents TiRex-2 from becoming dependent on the future-covariate channel being fully populated, training it to exploit such information when available without conditioning on its presence.

H Code and model checkpoints

The code repository for the model is hosted on GitHub: `https://github.com/NX-AI/tirex-2`. The pretrained checkpoints are available on Hugging Face (`https://huggingface.co/NX-AI/TiRex-2`), one per training corpus of Section F:

- **GIFT-Eval, zero-shot** (`NX-AI/TiRex-2-gifteval-zs`): excludes any overlap with the GIFT-Eval pretraining and evaluation datasets from pretraining. Used for the GIFT-Eval results of Section 4.
- **GIFT-Eval, matched corpus** (`NX-AI/TiRex-2-gifteval-pretrain`): adds the GIFT-Eval pretraining collection to the zero-shot corpus. Used for the comparison with Chronos-2 in Section B.2.
- **fev-bench, zero-shot** (`NX-AI/TiRex-2-fevbench`): excludes all fev-bench evaluation datasets from pretraining, following the same approach as for GIFT-Eval. Used for all fev-bench results.

I Societal impact

As a foundation model, TiRex-2 is trained once and can then be applied to many forecasting tasks without incurring additional task-specific training costs. This can broaden access to accurate time series forecasting in domains such as energy planning, infrastructure monitoring, logistics, and early warning systems. Support for covariates can further amplify these benefits, since many real deployments contain additional signals that improve forecasts without requiring model retraining. However, forecasts can still be wrong, and errors in high-stakes environments such as healthcare, safety-critical infrastructure, or disaster response may be costly. We therefore recommend domain-specific validation, uncertainty-aware decision rules, monitoring under distribution shift, and human oversight for consequential uses.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The key contributions stated in the abstract and introduction are well in the paper's scope. Specifically, Section 3 discusses the architectural components enabling our proposed model, TiRex-2, of multivariate predictions conditioned on future covariates with its favorable scaling behavior as a function of the sequence length L and compares this to transformer-based architectures. In Table 1 we show that the architecture modifications improve upon TiRex (Auer et al., 2025b). Section 3.3 explains how TiRex-2 can be used for streaming, with Figure 6 showcasing the continued performance when streaming forecasts to 32M timesteps. In Section 3.4 we explain the modifications we made to the data pipeline creating synthetic multivariate time series from univariate samples. We show empirical evaluation of our model on two separate benchmarks in Figure 4, where TiRex-2 ranks first among zero-shot models on GIFT-Eval and second on fev-bench, behind Chronos-2.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations of our work in the dedicated paragraph in Section 5.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.

- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: For Lemma 1 and Proposition 1 we provide a combined proof in Section C.2.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We describe the architecture of the model (Section 3), the synthetic multivariate coupling pipeline (Section 3.4) in detail and provide the hyperparameters with which we trained the model and the datasets providing the univariate time series samples in Sections E and F, respectively.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.

(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The code is publicly available at `https://github.com/NX-AI/tirex-2` and the pretrained checkpoints on Hugging Face (Section H). All training datasets (Section F) and both benchmarks are publicly available.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide a detailed description of the training details in Section E. The benchmarks we evaluated on are public and standardized described in Section 4.1 and in more detail in the respective publications. We further specify the additional tasks we evaluate on in Section 4.2.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: For the architectural ablations, we trained each model with up to 6 different random seeds, and report the standard deviation observed (see Table 1). For the long-horizon forecasting, we show in Figure 9 the standard error of the mean and for the streaming evaluation, we show the standard deviation of the MAE across 8 runs, as can be seen in Figure 8.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We describe the hardware used to run each experiment in Section E alongside the wall-clock time it takes to run each one.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics and confirm that this work complies with its principles and requirements.

   Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss potential positive and negative societal impacts in Section I, including beneficial forecasting applications, risks from incorrect high-stakes use, privacy and bias concerns from covariates, and mitigation strategies.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: Our work does not pose such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The paper cites all relevant resources appropriately, including models, datasets and benchmarks: Sections 2 and 4

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We release the documented code under the Apache-2.0 license at github.com/NX-AI/tirex-2, with documentation at nx-ai.github.io/tirex-2, and three open-weights checkpoints on Hugging Face, each with its own model card documenting training corpus, intended use and limitations (Section H).

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: Our work involves neither crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.

- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: Our work involves neither crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs were not used as important, original, or non-standard components of the core methods. Their use was limited to editing assistance and routine implementation support.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.