Does Synthetic Data Help? Empirical Evidence from Deep Learning Time Series Forecasters

Hugo Cazaux
Department of Engineering
Reykjavík University
Iceland, Menntavegur 1, 102
hugot@ru.is

Eyjólfur Ingi Ásgeirsson
Department of Engineering
Reykjavík University
Iceland, Menntavegur 1, 102

Hlynur Stefánsson
Department of Engineering
Reykjavík University
Iceland, Menntavegur 1, 102

Abstract

Synthetic data is now a routine ingredient of time series foundation-model pre-training, yet when it actually helps forecasters remains poorly understood. We present a large-scale empirical study: 5,265 runs across thirteen experiment groups, five architectures, five synthetic generators and nine datasets, plus a controlled pre-training ablation. In our model pool, channel-mixing architectures are markedly more receptive to synthetic augmentation than channel-independent ones: they improve in most trials (TimesNet 67%, iTransformer 64%), whereas channel-independent architectures are consistently degraded, which is why augmentation hurts in 67% of trials over a benchmark pool dominated by the latter. Model identity explains 36–54% of the variance in outcome and the binary channel-handling distinction recovers 44–84% of that, while every generator setting we sweep accounts for under 5%; the split reproduces under an external generator (KernelSynth), and part of the channel-mixing advantage depends on the extra optimisation that augmentation adds. Bundle choice is a matter of generator–target match: in a transfer matrix each bundle is the best training source for its own character, Seasonal-Trend is the right default for seasonal benchmarks, and the datasets that are extreme on another axis (Weather, Exchange) prefer the bundle matching it. Low-resource augmentation beats the sparse baseline in only 4 of 16 settings for TimesNet and 3 of 16 for iTransformer, and hard curriculum switching is harmful (+24% MSE). In pre-training, adding synthetic data to a channel-joint foundation model's corpus reduces zero-shot MSE on ETT by 30% (three seeds), but is neutral to slightly harmful across 19 held-out GIFT-Eval datasets, and a pre-registered test finds that which datasets gain is not explained by their statistical distance to the synthetic data. We give practical guidance and release all results. Code is available at `https://github.com/hugoiscracked/synthetic-ts`.

1 Introduction

Foundation models for time series forecasting have created a demand for data that the field struggles to meet: unlike text, which can be scraped, time series often sit on private servers or behind paywalls, so building a large, diverse and representative corpus is a challenge.

Synthetic data for time series long predates large language models: generating and perturbing series to enlarge small training sets is established practice Forestier et al. [2017], Fawaz et al. [2018], Lin et al. [2019]. What is new is the *scale* at which synthetic series now enter pre-training corpora for time series foundation models, and the fact that this practice was imported largely on the strength of results from language and vision rather than from evidence specific to forecasting. Our motivation is therefore not that synthetic data is an LLM technique being transferred, but that a technique

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

already common in this field is now being applied at a scale where its failure modes have not been characterised. This paper characterises them in the supervised setting.

Synthetic series offer unlimited supply, direct sampling instead of storage, and fully controllable curricula, but if the synthetic distribution does not match the target domain, augmentation can degrade rather than improve performance.

We build four difficulty-conditioned bundles representing the major archetypes of time series data, compare five models across nine datasets and sweep thirteen experiment groups to determine the factors that consistently affect downstream performance, then test whether the same data helps a foundation model in pre-training.

The research questions investigated in this article: **(Q1)** Does synthetic augmentation help when real data is abundant? **(Q2)** Does it help in low-resource settings? **(Q3)** Which bundle types transfer best to real-world forecasting? **(Q4)** How does difficulty conditioning affect downstream performance? **(Q5)** Do curriculum annealing and cross-channel correlation provide additional benefit?

Contributions. We contribute (1) a controllable difficulty-conditioned synthetic time series generator with four bundle types and Dirichlet variance allocation; (2) a large-scale empirical study that comprises 5,265 runs across thirteen experiment groups, five architectures, and nine datasets; (3) the identification of architecture as the primary determinant of outcome, and of generator–target match as the mechanism behind bundle choice, tested directly with a transfer matrix; and (4) a controlled, three-seed pre-training ablation with a pre-registered held-out evaluation, showing that the benefit of synthetic data in pre-training is likewise target-dependent. Section 3 describes the generator; Section 4 details the setup; Sections 5–6 present and interpret the findings.

2 Related work

Augmenting sparse datasets with synthetic data is an established technique in time series analysis Forestier et al. [2017], Lin et al. [2019], Fawaz et al. [2018]. In the era of large foundational forecasters, synthetic data has become a central ingredient: Amazon's Chronos2 Ansari et al. [2025] uses synthetic multivariate and covariate-informed series during pre-training and even releases a version trained exclusively on synthetic data that remains highly competitive with its real-data counterpart; Google's TimesFM Das et al. [2024] adds 20% synthetic data to its curriculum and validates the gain via ablation; Salesforce's MOIRAI Woo et al. [2024] follows a similar mixed pre-training strategy. All three achieve better or equivalent results to full-shot models in zero-shot and few-shot scenarios.

Synthetic data has also driven gains in computer vision Paulin and Ivasic-Kos [2023], Borkman et al. [2021], healthcare Gonzales et al. [2023], Giuffrè and Shung [2023] and language models Patel et al. [2024], Gan and Liu [2024], with benefits in privacy Liu et al. [2024a] and bias Paproki et al. [2024]; scaling laws for synthetic data are emerging for text Kang et al. [2025], Goyal and Chaturvedi [2023] but remain absent for time series.

Synthetic pre-training and prior-fitted networks. A distinct line of work trains forecasters on synthetic data *exclusively*, descending from prior-fitted networks Müller et al. [2022]: a model is trained on samples from a prior so that a forward pass approximates posterior inference. ForecastPFN Dooley et al. [2023] applies this to zero-shot forecasting with a purely synthetic curriculum, and TimePFN Taga et al. [2025] extends it to the multivariate case using Gaussian process kernels with linear coregionalization. Chronos Ansari et al. [2024] introduces KernelSynth, which composes Gaussian process kernels by random binary operations, and CauKer Xie et al. [2025] combines kernel composition with structural causal models, reporting clean scaling laws for synthetic corpora where real ones scale irregularly. We reproduce KernelSynth and include it as a fifth generator, so that our architectural findings can be checked against an external generator rather than only our own.

Positioning. The survey of Liu et al. Liu et al. [2025] organises this literature by *how* synthetic data is produced and *where* it enters the pipeline. Our study sits in a cell that survey identifies as under-examined: dataset-level augmentation for supervised forecasters, as opposed to corpus-level generation for foundation-model pre-training, which is where most reported successes come from. Complementary to us, Shidani et al. Shidani et al. [2026] analyse synthetic data as a regulariser and

derive an optimal synthetic-to-real ratio that depends on the Wasserstein distance between the real and synthetic distributions, predicting a U-shaped test error in that ratio. Our controlled transfer experiment is consistent with that account: error grows with generator-target mismatch, although the simple profile statistics we test do not capture that mismatch well enough to predict it (Sections 5.3 and 5.5).

Despite this momentum, there is no consensus on how to use synthetic data for time series: which generator types transfer, in what quantities, and whether curriculum scheduling matters. This paper fills that gap with a controlled, large-scale study spanning multiple architectures and datasets.

3 Methodology

The generator is organized around three components: a *bundle system* that defines the temporal character of each channel (§3.1); difficulty conditioning and cross-channel correlation controls that modulate complexity and inter-variable structure (§3.2); and data-mixing strategies that govern how synthetic and real samples are combined during training (§3.3).

3.1 Bundle System

A *bundle* is a stochastic template that defines the temporal archetype of a univariate channel: whether it is dominated by seasonality, structural breaks, long-range dependence, or volatility clustering. A D-channel series $\mathbf{X} \in \mathbb{R}^{T \times D}$ is constructed by drawing a bundle type $\mathcal{B}_j \sim \mathrm{Categorical}(\mathbf{w}_{\mathcal{B}})$ independently for each channel j, where $\mathbf{w}_{\mathcal{B}}$ are the mixture probabilities (uniform across the four types by default), and sampling a univariate series from the corresponding template.

Variance allocation. Within each bundle, components are combined so that each contributes a controlled share of total output variance. Each component $c_k(t)$ is standardised to unit variance: $\tilde{c}_k(t) = (c_k(t) - \bar{c}_k)/\sqrt{\mathrm{Var}(c_k) + \epsilon}$. Mixing weights are drawn from a Dirichlet distribution:

$$\mathbf{w} \sim \mathrm{Dir}\big(\alpha(d) \cdot \boldsymbol{\tau}\big), \qquad \alpha(d) = (1-d)\,\alpha_{\mathrm{easy}} + d\,\alpha_{\mathrm{hard}}, \tag{1}$$

with $\alpha_{\mathrm{easy}} = 80$, $\alpha_{\mathrm{hard}} = 6$, and bundle-specific target fractions $\boldsymbol{\tau}$. The output $y(t) = \sum_k \sqrt{w_k}\,\tilde{c}_k(t)$ has approximately unit variance. High concentration (low d) clusters weights near $\boldsymbol{\tau}$; low concentration (high d) allows free variation.

Four bundles cover the principal real-world temporal archetypes:

Seasonal-Trend (ST) combines Fourier seasonality, a deterministic trend and an AR(1) residual, $y_{\mathrm{ST}} = \sqrt{w_s}\,\tilde{s} + \sqrt{w_{\mathrm{tr}}}\,\widetilde{\mathrm{tr}} + \sqrt{w_r}\,\tilde{r} + \varepsilon$, with Dirichlet weights as in (1) and i.i.d. observation noise ε. The seasonal component superposes $K \in [1, 6]$ harmonics of a shared period $P \in \{24, 48, 96, 168, 336\}$ with amplitudes $a_k \propto k^{-\gamma(d)}$, where the decay exponent $\gamma(d)$ decreases with difficulty; trends are linear, quadratic or mildly exponential.

Non-stationary Regime (NR) is a Markov-switching AR process with 2–4 regimes whose means and transition probabilities scale with difficulty, mixed with a stochastic trend (random walk, geometric Brownian motion or Ornstein–Uhlenbeck).

Long Memory (LM) produces slowly decaying autocorrelation via ARFIMA$(0, d_{\mathrm{frac}}, 0)$, $d_{\mathrm{frac}} \in (-0.45, 0.45)$, or approximate fractional Brownian motion (Hurst 0.6–0.9), with an optional seasonal overlay.

Volatility Events (VE) combines GARCH(1,1) volatility clustering with Hawkes self-exciting event spikes whose intensity and amplitude scale with difficulty.

Example channels and full pseudo-code are in Appendices D and E.

3.2 Difficulty and Cross-Channel Correlation

Difficulty conditioning. A scalar $d \in [0, 1]$ governs the complexity of every generated series: higher d yields more harmonics, stronger trends, more frequent regime switches and larger observation noise

Table 1: Real-world datasets. D: number of channels; N: approximate total samples; Freq: sampling frequency. The four datasets marked with † are used in ablation studies (Groups 4–8, 11, 13); the first seven are used in Groups 1–3; those marked ‡ only in Group 12.

Dataset	D	N	Freq	Domain
ETTh1†	7	17 420	1 h	Electricity transformer
ETTh2	7	17 420	1 h	Electricity transformer
ETTm1†	7	69 680	15 min	Electricity transformer
ETTm2	7	69 680	15 min	Electricity transformer
Weather†	21	52 696	10 min	Meteorological
Electricity†	321	26 304	1 h	Power consumption
Traffic	862	17 544	1 h	Road occupancy
Exchange‡	8	7 588	1 day	Exchange rates
Illness‡	7	966	1 week	Influenza-like illness

(Appendix F measures these effects). By default d is drawn from a three-component truncated-normal mixture centred at 0.2, 0.5 and 0.8; the ablations use four named modes, `uniform` ($\mathcal{U}(0,1)$), `easy` ($\mathrm{Beta}(2,5)$), `medium` ($\mathrm{Beta}(2,2)$) and `hard` ($\mathrm{Beta}(5,2)$).

Cross-channel correlation. Real multivariate series exhibit inter-channel correlations absent from independently generated channels. With probability p_{latent}, the generated matrix $\mathbf{X} \in \mathbb{R}^{T\times D}$ is transformed via

$$\mathbf{Y} = \mathbf{X}\mathbf{A}^{\top}, \qquad \mathbf{A} = (1-\rho)\,\mathbf{I}_D + \rho\,\tilde{\mathbf{A}}, \tag{2}$$

where $\tilde{\mathbf{A}}_{ij} \sim \mathcal{N}(0,1)$ (row-normalised) and $\rho \sim \mathcal{U}(0.20,\ 0.50{+}0.20d)$ controls cross-channel dependence strength.

3.3 Data-Mixing Strategies

Four training modes govern the interaction between real data $\mathcal{D}_{\text{real}}$ and synthetic data $\mathcal{D}_{\text{synth}}$. In **real** mode (baseline), no synthetic data is used. In **mixed** mode, $|\mathcal{D}_{\text{synth}}| = \lfloor |\mathcal{D}^{\text{orig}}_{\text{real}}| \cdot r_{\text{synth}} \rfloor$ synthetic samples are concatenated with real training data, where $|\mathcal{D}^{\text{orig}}_{\text{real}}|$ is calculated based on the pre-sparsity real size so synthetic volume is independent of data withholding.

Data scarcity is simulated by a sparsity parameter $s \in (0,1]$: a fraction s of real training windows is retained while preserving temporal order; validation and test sets are never reduced for comparable results.

Curriculum learning is explored via two annealing modes: `anneal` moves r_{synth} from 1 to 0 (synthetic→real) and `anneal_inverse` from 0 to 1, either as a *hard* switch at epoch e_a or *gradually* by linear interpolation over the E training epochs. In all modes, validation and test sets consist exclusively of real data.

4 Experiments

4.1 Datasets

We use seven widely adopted long-term forecasting benchmarks Zhou et al. [2021], Wu et al. [2021], plus Exchange and Illness, chosen for their opposite statistical character (Section 5.3); all are summarised in Table 1. All datasets are split into train / validation / test with standard chronological partitions.

4.2 Models

We evaluate five models spanning the major architectural families: DLinear Zeng et al. [2023], SegRNN Lin et al. [2024], TimesNet Wu et al. [2023], iTransformer Liu et al. [2024b], and PatchTST Nie et al. [2023]. All models are trained for 10 epochs following the TSLib convention Wu et al. [2023], with batch size 32, Adam optimizer ($\mathrm{lr} = 10^{-4}$), and MSE loss. Input window $L = 96$; label length 48.

Table 2: Forecasting models, their key hyperparameters, and how each handles the channel dimension. *Mixing* models let information flow across variates inside the network; *independent* models apply a shared backbone to each variate in isolation. This split, rather than the linear/CNN/Transformer taxonomy, is what our results turn on.

Model	Family	Channels	Key hyperparameters
DLinear	Linear	Independent	Channel-independent (`-individual`)
SegRNN	RNN	Independent	seg_len $=48$, $d=512$, dropout$=0.5$
PatchTST	Transformer	Independent	$d=128$, $d_{\text{ff}}=256$, layers$=3$, heads$=16$, patch$=16$, stride$=8$
TimesNet	CNN	Mixing	$d=64$, $d_{\text{ff}}=64$, layers$=2$, top-$k=5$
iTransformer	Transformer	Mixing	$d=512$, $d_{\text{ff}}=512$, layers$=4$, heads$=8$

Table 3: Overview of the thirteen experiment groups. *Datasets* indicates whether the full set (7) or the ablation subset (4) is used. All groups use 3 seeds and 2 horizons $H \in \{96, 336\}$ (Illness: $\{24, 60\}$). Controlled parameters are listed as swept values; all others are held at their defaults (`data_mode`=`real`, $r_{\text{synth}}=0$, $s=1$, difficulty=`uniform`, bundle=`all`, $p_{\text{latent}}=0$).

Group	Name	Swept factor(s)	Datasets	Runs
1	Baseline	—	Full (7)	210
2	Sparsity	$s \in \{0.10, 0.25, 0.50\}$	Full (7)	630
3	Augmentation	$r_{\text{synth}} \in \{0.25, 0.50, 1.0\}$ (mixed)	Full (7)	630
4	Low-resource	$s \in \{0.10, 0.25\} \times r_{\text{synth}} \in \{0.5, 1.0, 2.0\}$ (mixed)	Ablation (4)	720
5	Difficulty	difficulty $\in$ {`uniform`, `easy`, `medium`, `hard`}	Ablation (4)	480
6	Bundle	bundle $\in$ {ST, NR, LM, VE}	Ablation (4)	480
7	Curriculum	mode $\in$ {`anneal`, `anneal_inverse`} $\times$ strategy $\in$ {`hard`, `gradual`}	Ablation (4)	480
8	Latent factor	$p_{\text{latent}} \in \{0.0, 0.3, 0.5, 0.7\}$	Ablation (4)	480
9	Cache ablation	cache_size $\in \{0, 100, 500\}$	ETTh1, ETTm1, Weather	108
10	Transfer matrix	train bundle $\times$ test bundle (5×5, incl. KernelSynth)	Synthetic	375
11	External generator	KernelSynth, mixed, $r_{\text{synth}}=1$	Ablation (4)	120
12	Flipped datasets	bundle $\in$ {ST, NR, LM, VE, KS}	Exchange, Illness	360
13	Training budget	iso-step; epochs $\in \{5, 20, 40\}$	Ablation (4)	192
			Total	**5 265**

4.3 Evaluation Protocol

We report MSE on the held-out real test set for prediction horizons $H \in \{96, 336\}$ (short-term and long-term), averaged over 3 independent seeds (2021–2023). Jobs were dispatched across GPU cluster nodes using GNU Parallel Tange [2011].

4.4 Experiment Groups

We organize 5,265 runs across thirteen groups, each isolating a specific factor. Throughout, $s \in (0, 1]$ denotes the fraction of real training windows retained and $r_{\text{synth}} \geq 0$ the number of synthetic samples added as a multiple of the original (pre-sparsity) dataset size. Table 3 provides a summary.

Group 1 provides reference MSE for every (model, dataset, H) triple and Group 2 quantifies the gap that synthetic data could close when real data is reduced. Group 3 asks whether augmentation helps when real data is plentiful, and Group 4 targets the low-resource regime where it is most likely to help. Groups 5–9 are single-factor ablations at $r_{\text{synth}}=1$, $s=1$ on the ablation subset (Group 9's cache comparison finds a fixed 500-sample cache equivalent to on-the-fly generation, Appendix I), and Groups 10–13 test the mechanisms behind these results: a bundle transfer matrix (10), an external generator (11), two datasets of opposite character (12) and training-budget controls (13).

5 Results

We report MSE on held-out real test sets, averaged over three independent seeds. Throughout this section, *MSE improvement* is defined as $(\text{MSE}_{\text{base}} - \text{MSE}_{\text{aug}})/\text{MSE}_{\text{base}} \times 100\%$, so *positive values mean the augmented model is better* and negative values mean it is worse. Baselines are stable across

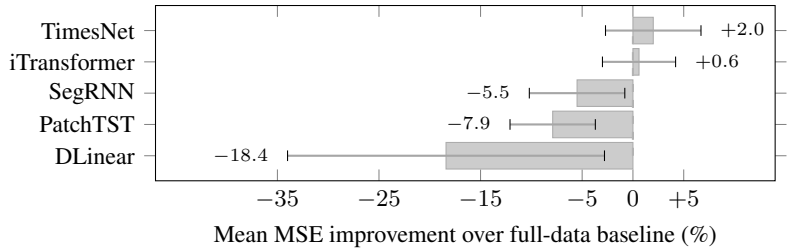


Figure 1: MSE improvement from synthetic augmentation (Group 3, $r_{\text{synth}} = 1$, $s = 1$). Positive values mean the augmented model is better than real-only training. Error bars show ± 1 std across the 14 (dataset, H) pairs. TimesNet and iTransformer gain slightly on average; the remaining three architectures are consistently harmed.

seeds: the median coefficient of variation across all (model, dataset, H) triples is 0.40% (median std $= 0.0013$), so the effects reported below are not masking noisy measurements. Per-model baselines (Group 1) are in Appendix C and agree with published results.

Overall picture. Over all five architectures in Group 3, synthetic data hurts in 67% of trials, but TimesNet and iTransformer improve in 67% and 64% of theirs while DLinear improves in only 7%: the question is not whether synthetic data helps, but when.

5.1 Full-Data Augmentation and Architecture Receptiveness (Q1)

Figure 1 shows the mean MSE improvement over the full-data baseline when synthetic data is added at $r_{\text{synth}} = 1$ (Group 3). Bars to the right of zero mean the augmented model beats the real-only model; bars to the left mean it degrades.

Two of five architectures are receptive at equal epochs. Over the 42 dataset–horizon–ratio combinations of Group 3, TimesNet improves in 28 (67%) and iTransformer in 27 (64%), against 9, 3 and 1 for SegRNN, DLinear and PatchTST. At $r_{\text{synth}} = 1$ (Figure 1) mean improvements are small for the receptive pair: $+2.0\%$ (TimesNet) and $+0.6\%$ (iTransformer). The other three degrade SegRNN -5.5%, PatchTST -7.9%, DLinear -18.4% (error bars: ± 1 std over the 14 dataset/horizon pairs). Increasing r_{synth} from 0.25 to 1.0 provides marginal additional gain for receptive models but deepens the penalty for non-receptive ones.

We offer a hypothesis, which our design does not isolate: TimesNet decomposes the input into 2D temporal-frequency maps and may profit from the spectral diversity of synthetic data, and iTransformer attends over variate tokens and may use the inter-channel diversity of synthetic multivariate batches, whereas channel-independent models (DLinear, PatchTST) see each channel in isolation, so the synthetic data adds distribution mismatch without that structure.

Is the gain only extra optimisation? In `mixed` mode synthetic samples are added to the real ones, so an augmented epoch contains more gradient steps than a real-only one. Group 13 separates the two (Table 10, Appendix A). In this control both models are harmed by augmentation, and capping optimiser steps at the real-only count adds 2.9 pp of harm for TimesNet against 0.7 pp for DLinear: extra optimisation reduces the channel-mixing model's harm more, and the gap between the two classes survives under both budgets. The control does not by itself say how much of Group 3's positive gain is due to extra steps. For these two models, training length is not a confound: across 5, 20 and 40 epochs the real-only baseline moves by at most 0.19% and the augmentation effect is flat. Group 13 was run in a different compute environment from Group 3 and its runs involving synthetic data do not reproduce Group 3 exactly (Section 6), so every comparison in it uses a baseline and an uncapped arm from the same environment.

5.2 Low-Resource Augmentation (Q2)

Figure 4 (Appendix A) shows MSE as a function of training data fraction for the two receptive models on Weather (TimesNet at $H = 336$, iTransformer at $H = 96$).

Table 4: Bundle transfer matrix. Each row is the bundle trained on, each column the bundle used for the held-out synthetic test set. Entries are MSE relative to the best training bundle for that column (1.00 = best), averaged over five architectures and three seeds. Absolute MSE differs by orders of magnitude between columns, so only within-column comparisons are meaningful.

Trained on	ST	NR	LM	VE	KS
ST	**1.00**	1.22	1.22	2.55	1.33
NR	2.15	**1.00**	1.05	2.46	1.36
LM	1.90	1.02	**1.00**	2.35	1.21
VE	2.41	1.40	1.32	**1.00**	1.73
KS	1.33	1.10	1.05	2.43	**1.00**

Synthetic augmentation can surpass the full-data baseline. TimesNet trained on only 10% of Weather with synthetic augmentation achieves MSE $= 0.2841$, beating the full-data real-only model (0.2850; dashed line). At 25% sparsity, the augmented model reaches 0.2814, 1.3% below the full-data baseline. This is the only regime in our study where synthetic data provides an unconditional win over more real data. iTransformer similarly improves MSE by 3.6% relative to the sparse baseline at $s = 0.10$ (0.1826 vs. sparse 0.1894), recovering 42.8% of the sparse-to-full gap, though it does not surpass the full-data model (0.1735).

These successes are the exception, not the rule. Across all dataset–sparsity combinations, augmentation beats the sparse baseline in only 4/16 for TimesNet and 3/16 for iTransformer, and in fewer than 2/16 for the other three models. The effect is possible but not typical: in a low-resource regime a practitioner should expect no benefit by default.

5.3 Ablation Studies (Q3–Q5)

The ablations (Groups 5–8, $s = 1$, $r_{\text{synth}} = 1$) vary one factor at a time and average over all five architectures, receptive or not, so every condition shows net degradation; the question is which setting minimises the harm (Figure 2; difficulty and latent-factor panels in Figure 3, Appendix A).

Bundle type (Q3). No bundle type produces a net improvement on average. Aggregated over all datasets and architectures, the least harmful is Seasonal-Trend (ST, -10.5%), ahead of LM (-12.5%), VE (-16.7%) and NR (-17.0%). The ordering is not fixed, however. Because architectures differ enormously in how much they are harmed, we compare bundles per dataset with exact two-sided binomial sign tests *paired* within every (model, horizon, seed) cell ($n = 30$), which cancels architecture and horizon effects (per-dataset means in Table 7, Appendix A). ST beats every other bundle on ETTh1 (21–22/30, $p \leq 0.043$), ETTm1 (21–24/30, $p \leq 0.043$) and Illness (23–25/30, $p \leq 0.005$), and is tied with LM on Electricity ($p = 0.099$), so it remains the right default for seasonal benchmarks. Two datasets overturn it significantly: on Weather **VE** beats ST (23/30, $p = 0.005$) and on Exchange **LM** beats ST (24/30, $p = 0.001$); each is best by mean rank, though not separable from the runner-up (LM on Weather, NR on Exchange; both 18/30, $p = 0.36$). In each case the winner owns the axis on which the dataset is extreme (Table 15): Exchange has an autocorrelation decay of 57.8 steps, $2.2\times$ the next-longest, and LM is the long-memory bundle; Weather has the heaviest tails of the ablation subset, and VE is the heaviest-tailed bundle. Because this axis was identified after observing the winner, we test the mechanism directly.

A direct test: the bundle transfer matrix. We train on each bundle and evaluate on a held-out synthetic test set from each bundle, giving the 5×5 matrix of Table 4 (KernelSynth included as an external generator). Because training and evaluation data are both synthetic, no real dataset mediates the comparison; validation uses the *training* bundle, as a practitioner would, and entries are normalised within each evaluation column since error scales differ between targets. *The diagonal is the best entry in every column*: each bundle is the best training source for its own target. Computed separately per architecture, the diagonal is strictly best in 23 of 25 architecture–column combinations and within 1% in all 25, the two exceptions being ties with the bundle closest in character to the target. Bundle quality is therefore not a property of the bundle but of the match between generator and target, which explains why a benchmark suite dominated by seasonal data ranks ST first; KernelSynth, built on an unrelated mechanism, obeys the same law.

Table 5: Bundle effect on datasets with the opposite statistical character (Group 12). MSE improvement (%) over the real-only baseline on that dataset: median over the ten (model, horizon) cells, each averaged over three seeds; the median is used because DLinear's degradation on Exchange (up to 327%) dominates any mean. Bold marks the best bundle by mean rank over those cells. On Illness ST beats every other bundle in paired sign tests ($p \leq 0.005$); on Exchange LM beats ST, VE and KS but is not separable from NR ($p = 0.36$). The final row is the mean over the four seasonal ablation benchmarks (Group 6), for comparison.

Dataset	ST	NR	LM	VE	KS
Exchange	−3.5	+0.4	**+3.0**	−6.1	+1.7
Illness	**+1.1**	−0.5	+1.0	−15.6	+1.4
Seasonal benchmarks	**−10.5**	−16.9	−12.5	−16.7	—

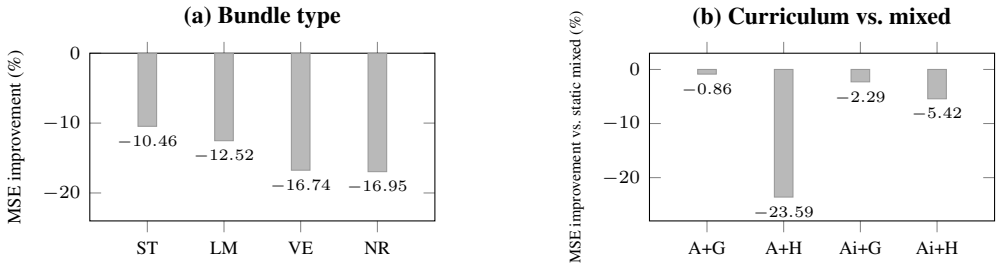


Figure 2: Bundle and curriculum ablations (Groups 5–8, $s = 1$, $r = 1$, averaged over all architectures). *All y-axes show MSE improvement vs. the full-data baseline*: lower (more negative) values indicate greater degradation. (a) Every bundle type degrades MSE on average; ST causes the least harm. (b) Curriculum strategies vs. static mixed at $r = 1$: gradual annealing performs the same as static mixing; the hard switch (**A+H**) is catastrophic. A = `anneal`, Ai = `anneal_inverse`, G = gradual, H = hard. Difficulty and latent-factor panels are in Figure 3.

Datasets with the opposite character (Q3). To test the same prediction on real targets we add two datasets chosen for a profile opposite to our seven benchmarks: **Exchange**, near random-walk daily FX rates with the longest autocorrelation decay we measure and weak seasonality, and **Illness**, weekly influenza rates with epidemic regime shifts, short memory and the heaviest tails (short-horizon grid, $L = 36$, $H \in \{24, 60\}$). ST wins *none* of the ten (model, horizon) cells on Exchange, where it is the worst bundle by mean rank and LM the best, against six of ten on Illness, whose annual cycle keeps it strongly periodic despite its heavy tails (Table 5). The ranking of bundles reverses with the character of the target, as the mechanism predicts. Augmentation is far more harmful on Exchange than on any other benchmark (DLinear degrades by up to 327%), consistent with injected structure being misinformation for a near random walk; per-model results are in Appendix H.

Curriculum scheduling (Q5). Relative to static mixing at $r_{\text{synth}} = 1$, gradual synth→real annealing is neutral (−0.9% improvement), while the hard switch is catastrophic (−23.6%, i.e. MSE rises by 23.6%) from the abrupt distribution shift at epoch 5.

Difficulty (Q4) and latent factor (Q5). All four difficulty modes degrade MSE by 7.7–10.0%, a spread under 2.4 pp: difficulty is a minor factor, with uniform marginally best. The latent-factor wrapper reduces the degradation monotonically, from 6.3% to 4.8% on the high-dimensional datasets and from 9.1% to 6.5% on ETT as p_{latent} rises from 0 to 0.7, at no cost.

5.4 How Much Does Architecture Actually Explain?

To test the claim that architecture is the primary determinant of outcome quantitatively, we decompose the variance of MSE improvement over the swept factors (Table 8, Appendix A), reporting ω^2, which corrects for degrees of freedom so that five-level model identity is not mechanically favoured over the binary channel-handling split. Model identity explains 36–54% of the variance in outcome, an order of magnitude more than any generator setting, and the channel-mixing versus channel-independent

Table 6: Pre-training ablation, three seeds per arm. Top: MSE reduction (%) from adding synthetic data, zero-shot on held-out ETT (mean ± sd over seeds). Bottom: held-out GIFT-Eval, geometric-mean ratio with/without over 67 configurations (< 1 favours synthetic data).

	H	ETTh1	ETTh2	ETTm1	ETTm2
ETT, MSE reduction	96	-15.8 ± 0.9	-4.6 ± 0.1	54.9 ± 3.7	7.2 ± 2.0
	192	2.6 ± 2.2	1.5 ± 0.3	56.5 ± 2.4	12.3 ± 2.8
GIFT-Eval held-out		MASE 1.030 ± 0.006		WQL 1.016 ± 0.013	

distinction alone recovers 44–84% of that with a quarter of the degrees of freedom. Generator settings are indistinguishable from zero against total variance; *within* a fixed (model, dataset, horizon) cell, the question a practitioner who has already chosen an architecture faces, they matter modestly (partial ω^2: latent factor 0.35, ratio 0.15, bundle and difficulty 0.06). The split is also stable in magnitude: averaged within each group, channel-mixing models sit between -1.8% and $+1.0\%$ and channel-independent models between -9.0% and -22.4%.

Does the split survive a generator we did not design? We repeat the $r_{\text{synth}} = 1$ experiment with **KernelSynth**, the Gaussian-process generator of Ansari et al. [2024], which shares no mechanism with our bundles (Table 9, Appendix A). The split reproduces more sharply: channel-mixing models average -0.7% against -23.6% for channel-independent ones, with the model ordering preserved (TimesNet $+0.7\%$, DLinear -55.2%). The effect is a property of how architectures consume synthetic multivariate data, not of our generator.

5.5 Synthetic Data in Foundation-Model Pre-training

Our supervised setting is not where synthetic data is used at scale, so we test the pre-training regime directly. We pre-train a Chronos-2-style channel-joint model Ansari et al. [2025] (39.0 M parameters) on a corpus built from GIFT-Eval pre-training windows, in two arms that differ *only* in composition: *with synthetic* (6.42 M training windows, 31.9% synthetic, from 30 shards of six generator families) and *without* (the same corpus minus those shards, 4.37 M windows). Both arms take exactly 12,500 optimiser steps of ≈137 windows with identical schedule and hyperparameters, so they see the same amount of data. Each arm is trained with three seeds.

ETT. On the four ETT datasets, excluded from pre-training in both arms, synthetic data reduces zero-shot mean MSE by $30.4 \pm 2.2\%$ over seeds (paired t-test $p = 0.003$; Table 6). The effect is highly reproducible (every cell has the same sign in all three seeds) but not uniform: six of eight cells improve, ETTh1 and ETTh2 at H=96 are consistently worse, and ETTm1 accounts for most of the gain ($+55\%$; absolute values in Table 12).

Held-out GIFT-Eval. To test whether this generalises we scored all six checkpoints on every GIFT-Eval dataset outside both arms' corpus, with the dataset list, metrics and analysis fixed and committed before any score was computed (Appendix B). The rule leaves 19 datasets and 67 configurations; one further dataset (`solar`) is excluded because it shares its source with a corpus dataset, and ETT is reported above. Here synthetic data is neutral to slightly harmful: the geometric-mean ratio of with to without is 1.030 ± 0.006 for MASE and 1.016 ± 0.013 for WQL, and 31 of 67 configurations improve (sign test $p = 0.63$). Individual datasets move in both directions, by up to -5% (electricity, bizitobs-l2c) and $+34\%$ (bitbrains).

Is the variation a matter of match? The supervised study predicts that a target benefits to the extent that the synthetic data resembles it. In a second pre-registered test we profiled each held-out dataset and the synthetic corpus on the four statistics of Table 15 and correlated distance with outcome over the 16 datasets long enough to profile. There is no relation (Spearman $\rho = +0.05$, one-sided $p = 0.42$; WQL $\rho = -0.01$). Pre-training thus reproduces the supervised picture: the effect of synthetic data depends on the target, and these profile statistics do not predict which targets gain.

6 Discussion

Supervised augmentation versus pre-training. In language models synthetic data has been most valuable in pre-training at scale, and time series foundation models (Chronos-2, TimesFM, MOIRAI) follow the same recipe. Section 5.5 qualifies that recipe: synthetic data gives a channel-joint model a large gain on ETT but none on average across a broad held-out benchmark. Our supervised setting, which trains a model from random weights on one target is the harder case, and there only channel-mixing architectures avoid harm. The two regimes agree on the main point: synthetic data helps some targets and not others, and in our pool channel-independent models are the least able to profit from it.

Bundle choice is a matching problem. The transfer matrix and the flipped datasets show that a bundle helps to the extent that the structure it adds is structure the target contains (a dependence on real-synthetic distance formalised in Shidani et al. Shidani et al. [2026]), which is why ST dominates an ETT-heavy benchmark suite. It does not yet yield a selection rule: profile-based rules fail in the supervised study (Section 5.3) and in pre-training (Section 5.5), and with few datasets we decline to fit one we cannot validate. We offer a diagnostic: profile the target, and sweep the bundle when it is extreme on an axis the default does not cover. We regard a validated selection rule as the main open problem this work leaves.

Limitations. Our architectural claim is made on the channel-handling axis alone: with five models, the linear/non-linear, CNN/Transformer and encoder-only/encoder-decoder cuts each leave one or two models per cell, and we make no claim about them. Seven of our nine datasets are trend-dominated benchmarks, four of them ETT; Exchange and Illness extend the range to heavy tails, long memory and regime shifts, but sparse or intermittent series remain untested. Supervised models are trained from random weights for 10 epochs, and in `mixed` mode augmentation also adds optimisation steps, which accounts for part of the channel-mixing benefit (Section 5.1). The three seeds vary initialisation and data order but not the synthetic draw. The pre-training ablation uses one model size, one corpus and one synthetic mixture. Groups 1–9 and 10–13 ran on different clusters; real-only baselines reproduce across them but runs with synthetic data do not, most plausibly because library versions perturb the generator's random stream, so those are compared only within one environment. Finally, the supervised study evaluates point forecasts (MSE, MAE) only; WQL enters only the pre-training evaluation.

7 Conclusion

We presented a large-scale empirical study of synthetic time series augmentation of 5,265 supervised runs across thirteen experiment groups and a controlled pre-training ablation. The effect of synthetic data is architecture- and target-conditional: in our model pool, channel-mixing models (TimesNet, iTransformer) are far more receptive than channel-independent ones (DLinear, PatchTST), which are reliably harmed. Three guidelines follow:

(1) **Prioritise channel-mixing architectures.** Architecture outweighs every generator setting combined. Even in foundation-model pre-training, synthetic data helps some targets (ETT, -30% MSE) but not a broad held-out benchmark, so evaluate on the target.
(2) **Use ST for seasonal targets; sweep the bundle otherwise.** ST is significantly best on ETTh1, ETTm1 and Illness, while Weather prefers VE and Exchange LM over it. Use a gradual synth$\rightarrow$real schedule, never a hard switch ($+24\%$ MSE).
(3) **Expect no low-resource benefit by default.** Augmentation beats the sparse baseline in only 4/16 (TimesNet) and 3/16 (iTransformer) settings.

The decision that matters is architectural, and it is made before any generator is configured.

Acknowledgements. The results presented in this article were obtained on the Jülich Supercomputing Centre JSC [2026] and the Elja IREI cluster IREI [2026]. The authors thank the JSC and IREI teams for their support.

References

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

Abdul Fatir Ansari, Oleksandr Shchur, Jaris Küken, Andreas Auer, Boran Han, Pedro Mercado, Syama Sundar Rangapuram, Huibin Shen, Lorenzo Stella, Xiyuan Zhang, et al. Chronos-2: From univariate to universal forecasting. *arXiv preprint arXiv:2510.15821*, 2025.

Steve Borkman, Adam Crespi, Saurav Dhakad, Sujoy Ganguly, Jonathan Hogins, You-Cyuan Jhang, Mohsen Kamalzadeh, Bowen Li, Steven Leal, Pete Parisi, et al. Unity perception: generate synthetic data for computer vision. *arXiv preprint arXiv:2107.04259*, 2021.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. In *Forty-first International Conference on Machine Learning*, 2024.

Samuel Dooley, Gurnoor Singh Khurana, Chirag Mohapatra, Siddartha V. Naidu, and Colin White. ForecastPFN: Synthetically-trained zero-shot forecasting. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Data augmentation using synthetic data for time series classification with deep residual networks. *arXiv preprint arXiv:1808.02455*, 2018.

Germain Forestier, François Petitjean, Hoang Anh Dau, Geoffrey I Webb, and Eamonn Keogh. Generating synthetic time series to augment sparse datasets. In *2017 IEEE international conference on data mining (ICDM)*, pages 865–870. IEEE, 2017.

Zeyu Gan and Yong Liu. Towards a theoretical understanding of synthetic data in llm post-training: A reverse-bottleneck perspective. *arXiv preprint arXiv:2410.01720*, 2024.

Mauro Giuffrè and Dennis L Shung. Harnessing the power of synthetic data in healthcare: innovation, application, and privacy. *NPJ digital medicine*, 6(1):186, 2023.

Aldren Gonzales, Guruprabha Guruswamy, and Scott R Smith. Synthetic data in health care: A narrative review. *PLOS Digital Health*, 2(1):e0000082, 2023.

Mahesh Kumar Goyal and Rahul Chaturvedi. Synthetic data revolutionizes rare disease research: How large language models and generative ai are overcoming data scarcity and privacy challenges. *International Journal on Recent and Innovation Trends in Computing and Communication*, 11(11): 1368–1380, 2023.

IREI. Elja icelandic research infrastructure for iceland, 2026. URL `https://www.irei.is/#/`.

JSC. Jülich supercomputer center, 2026. URL `https://www.fz-juelich.de/en`.

Feiyang Kang, Newsha Ardalani, Michael Kuchnik, Youssef Emad, Mostafa Elhoushi, Shubhabrata Sengupta, Shang-Wen Li, Ramya Raghavendra, Ruoxi Jia, and Carole-Jean Wu. Demystifying synthetic data in llm pre-training: A systematic study of scaling laws, benefits, and pitfalls. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 10750–10769, 2025.

Shengsheng Lin, Weiwei Lin, Wentai Wu, Feiyu Zhao, Ruichao Mo, and Haotong Zhang. SegRNN: Segment recurrent neural network for long-term time series forecasting. *arXiv preprint arXiv:2308.11200*, 2024.

Zinan Lin, Alankar Jain, Chen Wang, Giulia Fanti, and Vyas Sekar. Generating high-fidelity, synthetic time series datasets with doppelganger. *arXiv preprint arXiv:1909.13403*, 2019.

Xu Liu, Taha Aksu, Juncheng Liu, Qingsong Wen, Yuxuan Liang, Caiming Xiong, Silvio Savarese, Doyen Sahoo, Junnan Li, and Chenghao Liu. Empowering time series analysis with synthetic data: A survey and outlook in the era of foundation models. *arXiv preprint arXiv:2503.11411*, 2025.

Yintong Liu, U Rajendra Acharya, and Jen Hong Tan. Preserving privacy in healthcare: A systematic review of deep learning approaches for synthetic data generation. *Computer Methods and Programs in Biomedicine*, page 108571, 2024a.

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

Samuel Müller, Noah Hollmann, Sebastian Pineda Arango, Josif Grabocka, and Frank Hutter. Transformers can do Bayesian inference. In *International Conference on Learning Representations (ICLR)*, 2022.

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

Anthony Paproki, Olivier Salvado, and Clinton Fookes. Synthetic data for deep learning in computer vision & medical imaging: A means to reduce data bias. *ACM Computing Surveys*, 56(11):1–37, 2024.

Ajay Patel, Colin Raffel, and Chris Callison-Burch. Datadreamer: A tool for synthetic data generation and reproducible llm workflows. *arXiv preprint arXiv:2402.10379*, 2024.

Goran Paulin and Marina Ivasic-Kos. Review and analysis of synthetic dataset generation methods and techniques for application in computer vision. *Artificial intelligence review*, 56(9):9221–9265, 2023.

Amitis Shidani, Tyler Farghly, Yang Sun, Habib Ganjgahi, and George Deligiannidis. Beyond real data: Synthetic data through the lens of regularization. In *Proceedings of the 29th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2026.

Ege Onur Taga, Muhammed Emrullah Ildiz, and Samet Oymak. TimePFN: Effective multivariate time series forecasting with synthetic data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 20761–20769, 2025.

Ole Tange. GNU Parallel: The command-line power tool. *;login: The USENIX Magazine*, 36(1): 42–47, 2011.

Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified training of universal time series forecasting transformers. 2024.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in neural information processing systems*, 34:22419–22430, 2021.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. In *International Conference on Learning Representations*, 2023.

Shifeng Xie, Vasilii Feofanov, Marius Alonso, Ambroise Odonnat, Jianfeng Zhang, Themis Palpanas, and Ievgen Redko. CauKer: Classification time series foundation models can be pretrained on synthetic data only. *arXiv preprint arXiv:2508.02879*, 2025.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 11121–11128, 2023.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 11106–11115, 2021.

A Supplementary Results for Section 5

This appendix collects tables and figure panels whose key numbers are quoted in the main text.

Table 7: Bundle effect per dataset (Group 6): MSE improvement (%) over the full-data baseline, averaged over five architectures, two horizons and three seeds. Bold marks the best bundle for each dataset. The best choice differs across datasets. Bold marks a bundle that is best by a paired sign test over the 30 (model, horizon, seed) cells; Electricity has no bold entry because ST and LM are not separable there ($p = 0.099$).

Dataset	ST	LM	NR	VE
ETTh1	$\mathbf{-7.7}$	-10.6	-13.9	-17.9
ETTm1	$\mathbf{-4.3}$	-11.7	-20.8	-24.5
Weather	-10.7	-10.8	-9.5	$\mathbf{-2.4}$
Electricity	-19.2	-17.0	-23.6	-22.1
Mean	$\mathbf{-10.5}$	-12.5	-17.0	-16.7

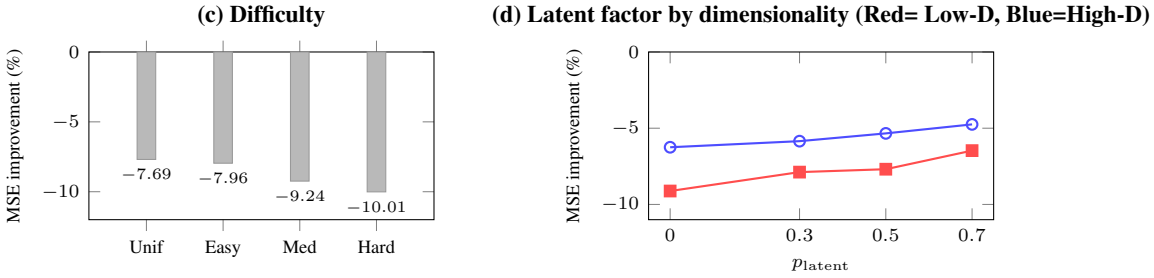


Figure 3: Difficulty and latent-factor ablations (Groups 5 and 8, $s = 1$, $r = 1$, averaged over all architectures; MSE improvement vs. the full-data baseline). (c) Difficulty mode: negligible spread (< 2.4 pp); uniform is marginally best. (d) Latent factor: higher p_{latent} reduces the degradation for both high-D datasets (Weather 21-ch, Electricity 321-ch) and low-D ETT datasets.

Table 8: Variance in MSE improvement explained by each design factor (ω^2, corrected for degrees of freedom). Computed separately within each experiment group, over all runs in that group. *Swept factor* is whichever factor the group varies. The final row reports the partial ω^2 of that factor after removing the (model, dataset, horizon) cell mean.

	Group 3 (ratio)	Group 5 (difficulty)	Group 6 (bundle)	Group 8 (p_{latent})
Model identity (5 levels)	0.382	0.389	0.543	0.356
Channel handling (2 levels)	0.250	0.290	0.241	0.300
Dataset	0.098	0.151	0.028	0.183
Horizon	0.012	0.021	0.006	0.021
Swept factor	0.002	< 0	< 0	< 0
Swept factor, partial	0.148	0.060	0.059	0.346

Table 9: External-generator check (Group 11). MSE improvement (%) over the full-data baseline at $r_{\text{synth}} = 1$, averaged over the four ablation datasets, two horizons and three seeds. *KernelSynth* is the Gaussian-process generator of Ansari et al. [2024]; *Ours* is the equivalent Group 3 setting restricted to the same datasets. The channel-handling split is the claim under test: it should not depend on whose generator is used.

Model	Channels	KernelSynth	Ours
TimesNet	mixing	+0.7	+0.3
iTransformer	mixing	-2.0	+0.1
PatchTST	independent	-3.5	-9.9
SegRNN	independent	-12.0	-8.5
DLinear	independent	-55.2	-16.1
Mean, channel-independent		-23.6	—
Mean, channel-mixing		-0.7	—

Table 10: Training-budget controls (Group 13). MSE improvement (%) from synthetic augmentation at $r_{\text{synth}} = 1$; positive means augmentation helps. *Iso-step* caps optimiser steps per epoch to the number the real-only baseline receives. All three conditions were run in the same environment and share a matched baseline. The epoch columns sweep the training budget; the real-only baseline moves by less than 0.3% across them, so the models converge well before the shortest budget.

Model	Channels	Uncapped	Iso-step	Δ	5 ep	20 ep	40 ep
DLinear	independent	−9.2	−10.0	−0.7	−9.2	−9.2	−9.2
TimesNet	mixing	−5.8	−8.6	−2.9	−5.7	−5.8	−5.8

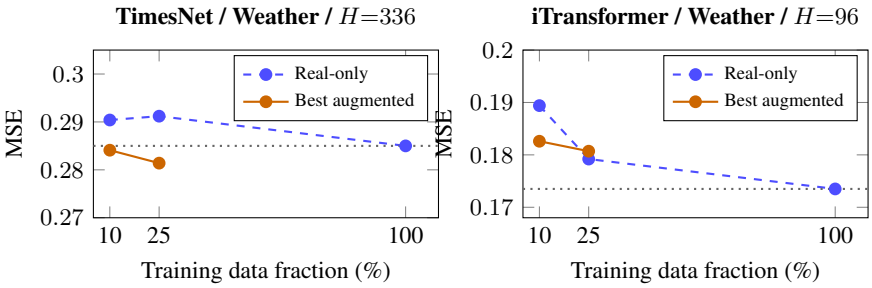


Figure 4: Low-resource augmentation (Group 4). Dashed line: real-only training at each sparsity level (Group 2). Solid line: best synthetic augmentation ratio observed in the sweep, chosen on test MSE (an oracle summary over the tested ratios, not a deployable selection) ($r_{\text{synth}} \in \{0.5, 1, 2\}$, $s \in \{0.10, 0.25\}$). Dotted horizontal: full-data baseline (Group 1). Augmented points below the dotted line surpass the full-data ceiling.

B Pre-training Ablation: Protocol and Detailed Results

Training. Both arms use the same 39.0 M-parameter channel-joint model, 12,500 optimiser steps of one micro-batch per GPU on 8 A100s (sequence budget 64 channel-sequences per GPU, ≈137 windows and ≈490 channel-sequences per step), Adam with learning rate 5×10^{-5}, 250 warm-up steps and cosine decay, seeds 2021–2023. The corpus is fixed to the same 145 dataset identifiers, index and window caps in both arms; the *without* arm drops the 30 synthetic shards. ETT is excluded from both.

Held-out GIFT-Eval protocol. Before any GIFT-Eval score was computed we committed the dataset list, metrics and analysis script to the project repository. A GIFT-Eval dataset is held out if it is absent from the *with* arm's corpus (the *without* corpus is a subset). A name screen leaves 22 of 28 datasets; a source check against our corpus descriptions removes `solar`, which derives from the same NREL data as a corpus dataset, and we omit `ett1`/`ett2`, reported under the ETT protocol. This leaves 19 datasets and 67 of the 97 canonical configurations. Metrics are the standard MASE and WQL; the primary statistic is the geometric mean over configurations of the with/without ratio, per seed. All 67 configurations were scored for all six checkpoints. Both arms are close to Seasonal Naive on the leaderboard statistic (geometric-mean MASE ratio to Seasonal Naive 1.094 with, 1.063 without; WQL 0.743 and 0.731).

Profile-distance test. A second pre-registered test profiles each held-out dataset (training portion only, 400 windows of length 256) and the synthetic corpus (2,000 windows, shards weighted by their share of training windows) on the four statistics of Table 15, scales them robustly and correlates Euclidean distance with the per-dataset log ratio. Three datasets have no series long enough to profile. Over the remaining 16: MASE $\rho = +0.05$ (one-sided permutation $p = 0.42$), WQL $\rho = -0.01$ ($p = 0.52$).

Table 11: Held-out GIFT-Eval per dataset: geometric-mean ratio with/without over the dataset's configurations, averaged over three seeds (< 1 favours synthetic data), and profile distance to the synthetic corpus (— = too short to profile).

Dataset	Configs	MASE ratio	WQL ratio	Distance
LOOP_SEATTLE	7	1.027	1.012	1.55
M_DENSE	4	0.951	0.960	1.30
SZ_TAXI	4	0.991	1.005	1.95
bitbrains_fast_storage	4	1.335	0.982	3.20
bitbrains_rnd	4	1.217	0.988	3.41
bizitobs_application	3	1.059	1.167	3.96
bizitobs_l2c	6	0.948	0.946	4.03
bizitobs_service	3	1.047	1.101	3.83
car_parts_with_missing	1	1.006	0.996	—
covid_deaths	1	1.101	1.205	—
electricity	8	0.947	0.989	3.70
hierarchical_sales	2	0.993	0.996	3.16
hospital	1	1.001	1.017	—
jena_weather	7	1.024	1.061	2.33
kdd_cup_2018_with_missing	4	0.988	0.972	2.48
restaurant	1	0.990	0.984	2.43
saugeenday	3	1.024	1.050	3.08
temperature_rain_with_missing	1	0.985	0.981	2.49
us_births	3	1.130	1.138	1.31
All (geometric mean)	67	1.030	1.016	

Table 12: Pre-training ablation on ETT, absolute zero-shot MSE (mean $\pm$ sd over three seeds; 800 windows per cell).

Dataset / H	With synthetic	Without
ETTh1 / 96	0.405 ± 0.003	0.349 ± 0.004
ETTh1 / 192	0.407 ± 0.005	0.418 ± 0.005
ETTh2 / 96	0.092 ± 0.001	0.088 ± 0.001
ETTh2 / 192	0.110 ± 0.000	0.112 ± 0.001
ETTm1 / 96	0.316 ± 0.012	0.703 ± 0.031
ETTm1 / 192	0.338 ± 0.015	0.776 ± 0.012
ETTm2 / 96	0.075 ± 0.003	0.081 ± 0.002
ETTm2 / 192	0.080 ± 0.002	0.091 ± 0.001

C Baseline MSE (Group 1)

Table 13 reports full-data baseline MSE (mean over 3 seeds) for all five models, seven datasets, and both prediction horizons. Bold marks the best (lowest MSE) per (dataset, H) pair.

Table 13: Baseline MSE (Group 1, real-only, full data). Mean over 3 seeds. Bold: best (lowest MSE) per (dataset, H) pair.

Dataset	H	DLinear	SegRNN	TimesNet	iTransformer	PatchTST
ETTh1	96	0.396	**0.374**	0.402	0.396	0.390
	336	0.487	**0.455**	0.531	0.489	0.468
ETTh2	96	0.345	0.293	0.332	0.312	**0.287**
	336	0.592	0.424	0.480	0.441	**0.418**
ETTm1	96	0.346	**0.331**	0.339	0.350	0.337
	336	0.414	**0.398**	0.432	0.461	0.409
ETTm2	96	0.193	**0.173**	0.187	0.185	0.177
	336	0.382	**0.295**	0.319	0.320	0.304
Weather	96	0.196	**0.165**	0.171	0.174	0.182
	336	0.282	**0.271**	0.285	0.280	0.282
Electricity	96	0.210	0.193	0.204	**0.161**	0.212
	336	0.223	0.224	0.263	**0.191**	0.233
Traffic	96	0.696	0.781	0.640	**0.440**	0.555
	336	0.653	0.818	0.705	**0.479**	0.562

D Bundle Examples

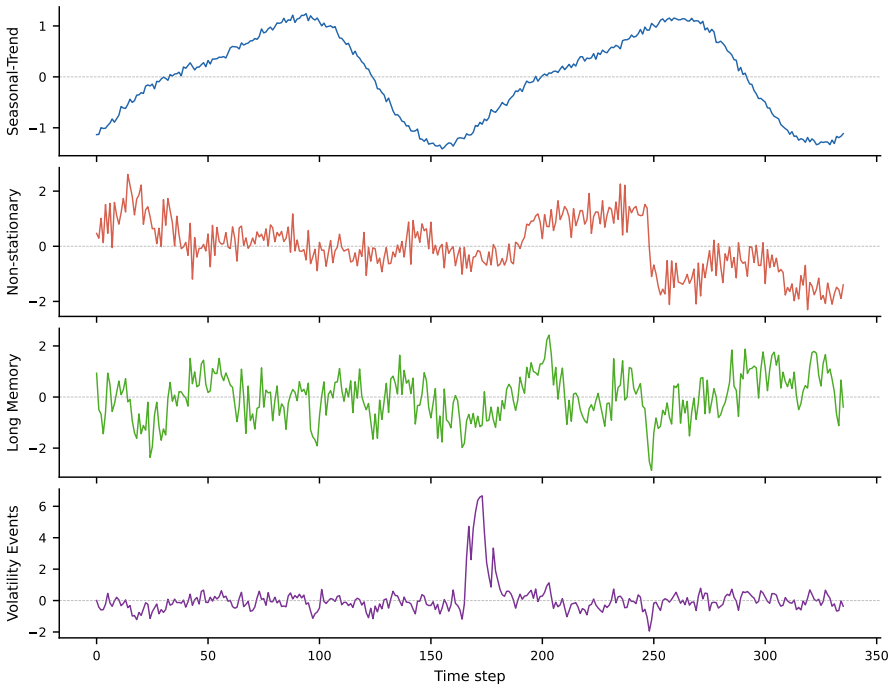


Figure 5: One representative channel drawn from each bundle type at medium difficulty ($d = 0.5$, seed fixed for reproducibility, $T = 336$ time steps). **Seasonal-Trend (ST):** dominant periodic structure with mild AR residual. **Non-stationary (NR):** Markov-switching regime changes with a stochastic trend component. **Long Memory (LM):** slowly decaying autocorrelations producing persistent, non-periodic undulations. **Volatility Events (VE):** low-amplitude baseline punctuated by a sharp GARCH/Hawkes spike.

E Bundle Pseudo-code

$\tilde{x}$ denotes x normalised to unit variance. Dirichlet weights $(w_i) \sim \mathrm{Dir}(\alpha(d))$ use difficulty-scaled targets and concentration; harder difficulties flatten the mixture and raise the noise floor $\sigma_{\mathrm{obs}}(d)$.

Algorithm 1 Seasonal-Trend (ST) bundle

Require: length T, difficulty $d \in [0, 1]$

$K \sim \mathrm{Discrete}\{1, \ldots, 6\}$; $P \sim \{24, 48, 96, 168, 336\}$; $\gamma \leftarrow \gamma(d)$ ▷ $\gamma\downarrow$ as $d\uparrow$

for $k = 1, \ldots, K$ **do**

  $\phi_k \sim \mathcal{U}(0, 2\pi)$; $a_k \leftarrow k^{-\gamma}$ ▷ phase and power-law amplitude

end for

$s(t) \leftarrow \sum_{k=1}^{K} a_k \sin(2\pi k t/P + \phi_k)$ ▷ seasonal component

$\phi \sim \mathcal{U}(-0.9,\ 0.9)$; generate $r(t)$ as AR(1) with coefficient ϕ ▷ residual

type $\sim \mathrm{Uniform}\{\mathrm{linear},\ \mathrm{quadratic},\ \mathrm{exponential}\}$

if type $=$ exponential **then** sample $b \sim \mathcal{U}(-0.01, 0.01)$

end if

compute $\mathrm{tr}(t)$ according to sampled type ▷ trend

$(w_s, w_{tr}, w_r) \sim \mathrm{Dir}(\alpha(d)\boldsymbol{\tau}_{\mathrm{ST}}(d))$ ▷ targets $[0.92, 0.06, 0.02] \to [0.50, 0.20, 0.30]$ easy$\to$hard

$\tilde{y}(t) \leftarrow \sqrt{w_s}\,\tilde{s}(t) + \sqrt{w_{tr}}\,\widetilde{\mathrm{tr}}(t) + \sqrt{w_r}\,\tilde{r}(t)$ ▷ unit-variance mix

$\sigma_{\mathrm{obs}} \leftarrow \sigma_{\mathrm{obs}}(d)$; $y(t) \leftarrow \tilde{y}(t) + \varepsilon(t),\ \varepsilon(t) \sim \mathcal{N}(0, \sigma^2_{\mathrm{obs}})$

return y

Algorithm 2 Non-stationary Regime (NR) bundle

Require: T, d

$M \sim \mathrm{Uniform}\{2, 3, 4\}$ ▷ number of regimes

$\mu_m \sim \mathcal{N}(0, \sigma_\mu(d))$, $\beta_m \sim \mathcal{N}(0, \sigma_\beta(d))$ for $m = 1, \ldots, M$ ▷ regime params

$p_{\mathrm{stay}} \leftarrow p_{\mathrm{stay}}(d)$ ▷ $p_{\mathrm{stay}}\downarrow$ as $d\uparrow$, giving more frequent transitions

simulate Markov chain: $s_1, \ldots, s_T$ with $\Pr(s_{t+1} = m \mid s_t = m) = p_{\mathrm{stay}}$

$g(t) \leftarrow \mathrm{AR}(1)$ in regime s_t with mean μ_{s_t} and slope β_{s_t} ▷ regime process

type $\sim \mathrm{Uniform}\{\mathrm{random\ walk},\ \mathrm{GBM},\ \mathrm{Ornstein{-}Uhlenbeck}\}$; compute $\mathrm{st}(t)$

generate AR(1) residual $r(t)$

$(w_g, w_{st}, w_r) \sim \mathrm{Dir}(\alpha(d)\boldsymbol{\tau}_{\mathrm{NR}}(d))$

$\varepsilon(t) \sim \mathcal{N}(0, \sigma^2_{\mathrm{obs}}(d))$

$y(t) \leftarrow \sqrt{w_g}\,\tilde{g}(t) + \sqrt{w_{st}}\,\widetilde{\mathrm{st}}(t) + \sqrt{w_r}\,\tilde{r}(t) + \varepsilon(t)$

return y

Algorithm 3 Long Memory (LM) bundle

Require: T, d

$d_{\mathrm{frac}} \sim \mathcal{U}(-0.45,\ 0.45)$ **or** $H \sim \mathcal{U}(0.6,\ 0.9)$ for the fBm variant

$\mathrm{lm}(t) \leftarrow \mathrm{ARFIMA}(0,\ d_{\mathrm{frac}},\ 0)$ ▷ slowly decaying autocorrelations

$p_{\mathrm{seas}}(d) \leftarrow 5\%$ (easy), 25% (medium), 45% (hard)

if $\mathcal{U}(0, 1) < p_{\mathrm{seas}}(d)$ **then**

  generate Fourier seasonal overlay $s(t)$ at reduced difficulty $0.3d$

else

  $s(t) \leftarrow 0$

end if

$n(t) \sim \mathcal{N}(0, \sigma^2_n(d))$ ▷ noise component

$(w_{\mathrm{lm}}, w_s, w_n) \sim \mathrm{Dir}(\alpha(d)\boldsymbol{\tau}_{\mathrm{LM}}(d))$

$y(t) \leftarrow \sqrt{w_{\mathrm{lm}}}\,\widetilde{\mathrm{lm}}(t) + \sqrt{w_s}\,\tilde{s}(t) + \sqrt{w_n}\,\tilde{n}(t)$

return y

Algorithm 4 Volatility Events (VE) bundle

Require: T, d

sample GARCH$(1,1)$ params $(\omega, \alpha_G, \beta_G)$ scaled by d ▷ $\omega, \alpha_G \uparrow$ with d

initialise $g(0)$ and $\sigma^2(0)$ from the stationary variance

for $t = 1, \dots, T$ **do**

  $\sigma^2(t) \leftarrow \omega + \alpha_G\, g^2(t{-}1) + \beta_G\, \sigma^2(t{-}1)$

  $g(t) \sim \mathcal{N}(0,\, \sigma^2(t))$ ▷ GARCH innovation

end for

$\lambda_0 \leftarrow \lambda_0(d),\ A \leftarrow A(d),\ \delta \leftarrow \delta(d)$ ▷ Hawkes params, all $\uparrow$ with d

$h(t) \leftarrow \lambda_0 + A \sum_{\tau<t} \exp(-\delta(t-\tau))$ ▷ self-exciting intensity

sample event times $\{t_{\mathrm{ev}}\}$ from inhomogeneous $\mathrm{Poisson}(h(t))$

$\mathrm{spikes}(t) \leftarrow A_{\mathrm{spike}}(d) \sum_{\mathrm{events}} \kappa(t - t_{\mathrm{ev}})$ ▷ κ: exponential decay kernel

$n(t) \sim \mathcal{N}(0, \sigma^2_{\mathrm{obs}}(d)); \quad \widetilde{(w_g, w_{\mathrm{sp}}}, w_n) \sim \mathrm{Dir}(\alpha(d)\boldsymbol{\tau}_{\mathrm{VE}}(d))$

$y(t) \leftarrow \sqrt{w_g}\, \tilde{g}(t) + \sqrt{w_{\mathrm{sp}}}\, \mathrm{spikes}(t) + \sqrt{w_n}\, \tilde{n}(t)$

return y

F What the Difficulty Control Does

The scalar d is a *generator control*: it sets the number of harmonics, the amplitude decay exponent, the regime-switch frequency, the Dirichlet concentration and the observation noise floor. It is not itself a measurement of the produced signal, so its meaning has to be established empirically. Table 14 sweeps d on a five-point grid and reports four measurable statistics of the resulting series, each the median over 200 independent draws of length 1024 per cell.

The control behaves as intended where each bundle has the corresponding mechanism, and is inert where it does not. Spectral entropy rises with d in ST ($0.18 \to 0.62$) and NR ($0.40 \to 0.82$) as extra harmonics and more frequent switching flatten the spectrum. The change-point rate rises in ST, NR and VE but is flat in LM ($\rho = -0.11$), which contains no regime-switching component at all. Kurtosis rises sharply in VE once the volatility mechanism engages beyond $d = 0.5$ ($0.45 \to 4.17$), and is non-monotone in NR, where low d yields rare but very large mean shifts and therefore heavy difference tails. We report this unevenness rather than smoothing it: d orders the complexity of each bundle internally, but it is not a common difficulty axis that can be compared across bundles.

G Statistical Profiles of Datasets and Bundles

Table 15 profiles every real dataset and every synthetic bundle on the four axes used throughout the discussion. All statistics are computed on random windows of 432 steps, the longest input/output span the models are trained on, because both the change-point rate and the autocorrelation decay depend on series length; comparing a 20,000-step dataset against a 1,024-step synthetic draw would compare artefacts of length rather than temporal character. Real datasets are standardised per channel using training-split statistics, exactly as the data loaders do, and only the training split is used so that the profile never sees test data.

H Flipped-Character Datasets: Per-Model Detail (Group 12)

Table 5 reports medians and mean ranks because the per-model spread on Exchange is extreme. The underlying per-model numbers are given here. MSE improvement (%) over the real-only baseline on the same dataset, averaged over both horizons and three seeds; positive means augmentation helps.

Table 14: What the difficulty control d does to the generated signal. Each entry is the median over the sampled series at that d; ρ is the Spearman rank correlation between d and the statistic across the grid. d is a generator control, not a measurement of the signal. Within each bundle it moves the statistics that bundle is designed to control, and does so monotonically ($|\rho| \geq 0.8$) in most cells, but it is not a single common axis: the change-point rate is flat in LM ($\rho = -0.11$), which has no regime-switching mechanism, and the kurtosis of NR is non-monotone because at low d that bundle produces rare but very large mean shifts.

Bundle	Statistic	d=0	d=0.25	d=0.5	d=0.75	d=1	ρ
ST	Spectral entropy	0.182	0.182	0.333	0.621	0.620	+0.90
	Change points / 1000	7.812	7.812	9.766	10.742	11.719	+0.97
	Hurst H	0.813	0.812	0.842	1.007	1.042	+0.90
	Excess kurtosis	-1.495	-1.459	-0.133	-0.024	-0.042	+0.90
NR	Spectral entropy	0.401	0.458	0.651	0.794	0.821	+1.00
	Change points / 1000	5.859	8.789	9.766	9.766	9.766	+0.89
	Hurst H	1.314	1.255	1.099	0.965	0.936	-1.00
	Excess kurtosis	18.427	32.382	0.729	0.123	0.089	-0.90
LM	Spectral entropy	0.755	0.721	0.822	0.854	0.833	+0.80
	Change points / 1000	6.836	6.348	6.836	5.859	6.836	-0.11
	Hurst H	1.217	1.243	1.092	0.882	0.974	-0.80
	Excess kurtosis	-0.031	-0.021	-0.027	-0.011	-0.022	+0.50
VE	Spectral entropy	0.907	0.909	0.788	0.709	0.751	-0.80
	Change points / 1000	2.930	2.930	5.859	10.742	8.789	+0.87
	Hurst H	0.696	0.678	0.819	0.850	0.829	+0.80
	Excess kurtosis	0.446	0.513	4.169	3.988	4.060	+0.70

Table 16: Group 12 per-model MSE improvement (%). Bold marks the best bundle for each model. On Exchange, DLinear degrades by up to 327% while the other four architectures move by single digits, which is why an unweighted mean over models is not a meaningful summary. The channel-mixing models gain most, but not exclusively: on Exchange SegRNN gains with LM and PatchTST with NR, and SegRNN gains on Illness with every bundle except NR.

Dataset	Model	ST	NR	LM	VE	KS
Exchange	DLinear	-160.2	-327.5	-242.5	$\mathbf{-38.3}$	-300.5
	SegRNN	-6.1	$+0.4$	$\mathbf{+4.4}$	-6.9	-8.7
	PatchTST	-19.3	$\mathbf{+1.6}$	-2.5	-23.8	-4.5
	TimesNet	$+2.4$	$\mathbf{+15.5}$	$+15.0$	$+0.2$	$+11.4$
	iTransformer	-1.6	$+0.8$	$+4.2$	$+3.4$	$\mathbf{+5.6}$
Illness	DLinear	-3.8	-4.8	-4.1	$\mathbf{-3.0}$	-4.1
	SegRNN	$+11.7$	-0.5	$+5.7$	$\mathbf{+12.5}$	$+10.6$
	PatchTST	$\mathbf{-6.6}$	-20.8	-13.3	-39.0	-14.8
	TimesNet	$\mathbf{+3.3}$	-0.1	$+2.0$	-38.6	$+1.3$
	iTransformer	$+5.8$	$+9.0$	$\mathbf{+9.8}$	-15.6	$+9.4$

I Cache Ablation: Detailed Results (Group 9)

Table 17 reports MSE (mean over 3 seeds) for each fully observed (model, dataset, H) combination in Group 9, along with the maximum pairwise difference across the three cache strategies. All twelve combinations show a maximum difference below 0.03, confirming that a fixed 500-sample cache is a practical drop-in replacement for on-the-fly generation.

Table 15: Statistical profile of the benchmark datasets (training split only) and of each synthetic bundle, measured over random windows of 432 steps – the longest span the models see – so that all rows are length comparable. Higher seasonality means stronger periodic structure; higher change-point rate means more frequent mean shifts; higher ACF decay means longer memory; *lower* tail index means heavier tails. The seven benchmarks of the main study occupy a narrow region (seasonality 0.69–0.98), which is the sense in which the suite is homogeneous. Note that nearest-profile matching does not by itself predict the best-performing bundle (Table 7): these marginal statistics are too coarse to capture the relevant notion of match.

	Seasonality	Chg.pts/1000	ACF decay	Tail index
Real datasets				
ETTh1	0.77	6.94	4.10	3.90
ETTh2	0.76	9.26	5.64	4.89
ETTm1	0.74	16.20	14.67	3.31
ETTm2	0.69	16.20	13.26	4.65
Weather	0.72	11.57	26.67	3.22
Electricity	0.98	1.16	3.89	5.46
Traffic	0.96	0.00	3.21	3.29
Exchange	0.62	9.26	57.80	3.43
Illness	0.65	13.89	7.92	2.45
Synthetic bundles				
ST	0.96	16.20	9.78	7.26
NR	0.63	11.57	17.22	4.81
LM	0.64	9.26	20.48	6.10
VE	0.62	6.94	3.15	4.12
KS	0.73	10.42	15.08	13.18

Table 17: Group 9 cache ablation. MSE averaged over 3 seeds. Max Δ: largest pairwise MSE difference across the three cache sizes.

Model	Dataset	H	Cache-0	Cache-100	Cache-500	Max Δ
PatchTST	ETTh1	96	0.4674	0.4670	0.4507	0.0167
PatchTST	ETTh1	336	0.5559	0.5339	0.5343	0.0220
PatchTST	ETTm1	96	0.3894	0.3913	0.3975	0.0081
PatchTST	ETTm1	336	0.4904	0.4795	0.4822	0.0109
PatchTST	Weather	96	0.1920	0.1961	0.1925	0.0041
PatchTST	Weather	336	0.2962	0.2974	0.2948	0.0026
iTransformer	ETTh1	96	0.3988	0.4111	0.4165	0.0176
iTransformer	ETTh1	336	0.5084	0.5180	0.5163	0.0095
iTransformer	ETTm1	96	0.3576	0.3515	0.3455	0.0120
iTransformer	ETTm1	336	0.4361	0.4401	0.4349	0.0052
iTransformer	Weather	96	0.1817	0.1787	0.1787	0.0030
iTransformer	Weather	336	0.2838	0.2822	0.2812	0.0026

J Full Augmentation Results by Dataset (Group 3)

Table 18 reports MSE improvement (%) for Group 3 (full-data augmentation, $r_{\text{synth}} \in \{0.25, 0.5, 1.0\}$, best over ratios) relative to the Group 1 baseline. Positive values indicate the augmented model beats the real-only baseline. Architecture-conditional behaviour is consistent across datasets: TimesNet and iTransformer show net gains on most benchmarks; DLinear and PatchTST are consistently harmed, with DLinear incurring up to -26% on ETTm1 and ETTm2.

Table 18: Group 3 MSE improvement (%) over Group 1 baseline per (dataset, H) pair, best ratio. Positive = augmentation improves MSE. The best-ratio column is an oracle summary over the tested ratios, not a deployment-time selection rule.

Dataset	H	DLinear	SegRNN	TimesNet	iTransformer	PatchTST
ETTh1	96	−4.1	−4.4	−1.2	+1.6	−7.3
	336	−2.3	−3.1	+7.7	+0.2	−6.7
ETTh2	96	−8.1	+1.5	+9.8	+4.9	−3.5
	336	+5.0	+1.9	+11.8	+3.8	−0.4
ETTm1	96	−22.7	−6.3	+2.5	+5.7	−9.0
	336	−25.7	−4.7	+1.9	+8.5	−7.6
ETTm2	96	−25.9	−0.6	+7.3	+4.4	−4.8
	336	−6.9	−1.8	+2.8	+4.0	−2.3
Weather	96	+3.5	−0.4	+4.2	+1.6	+0.5
	336	+1.2	+1.0	+3.4	+0.6	−1.6
Electricity	96	−8.5	−7.7	+1.7	−2.0	−8.2
	336	−7.0	−4.0	+1.5	−1.5	−6.9
Traffic	96	−3.2	+1.1	−0.6	+0.6	−2.2
	336	−2.4	+1.4	+6.4	−0.6	−3.4
Mean		−7.6	−1.9	+4.3	+2.3	−4.6

K Public Experiment Dataset

To support reproducibility and future benchmarking, all experiment results are released in the project repository: a consolidated CSV of all 5,265 supervised runs across Groups 1–13, and, separately, the per-configuration scores, pre-registrations and analysis scripts of the pre-training ablation (Appendix B). The supervised CSV has fields for experiment group, model, dataset, prediction horizon, seed, data mode, synthetic ratio, sparsity, difficulty, bundle, latent factor probability, annealing strategy, annealing epoch, MSE, MAE, training time, and timestamp. This allows independent reanalysis of all findings in this paper without requiring GPU resources, and provides a foundation for meta-learning approaches to hyperparameter selection for synthetic augmentation.

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 contributions of the paper are listed in the abstract in the introduction. Both the positive and negative results are mentioned.
   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: Section 6 discusses the limitations of our work provided as well as the scope of the study. We acknowledge that even a sweep with thousands of runs will have assumptions on models, datasets, hyperparameters, and we did our best to use a fair, representative grid for our experiments.

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: [N/A]

   Justification: This paper does not provide theoretical results and as such does not contain any proof.

   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: The paper provides full details for reimplementation, and the public project repository provides our full integration of the process in a standard time-series forecasting library, as well as the complete job arrays necessary to re-run the same experiments.

   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 results of all 5,265 supervised runs and of the pre-training ablation are available in the public project repository, together with the exact reproduction scripts.
   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: All information about the runs can be found in the job scripts and the dataset file.
   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: Statistical significance is reported for the relevant metrics.
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 report the aggregate compute requirements in the paper, and timestamp and training time for each experiment are available in the dataset.
   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: Removed identifiable elements from the submission.
   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: [N/A]
    Justification: No specific societal impacts for this study.
    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., deploy-

ment 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: The paper poses no 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.