DOING MORE WITH LESS: DENSE TOKENIZATION FOR RECURRENT MONTHLY FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Many forecasting tasks rely on monthly observations, either because finer observations do not exist or because figures are reconciled only at the end of each period. We ask whether changing how these observations are represented can improve a model's accuracy. We propose flat monthly tokenization, which repeats each monthly average across several tokens that carry known calendar features. This representation enables multiple recurrent updates per observation while preserving the information in the monthly history and calendar. On monthly tasks built from two public daily datasets, a small GRU with 90 tokens per month reduces CRPS relative to the monthly baseline by 20 to 49% on retail and 28 to 36% on electricity. On retail, it also achieves lower CRPS than N-HiTS and the same network trained on daily observations. Ablations indicate contributions from the input representation and repeated recurrent updates, with an additional contribution from varying calendar features on electricity. Decomposing the score into median and nonmedian components shows that increasing density from 30 to 90 tokens improves both components on electricity, whereas on retail the improvement is confined to the nonmedian component. These results identify token density as a useful modeling choice for monthly forecasting. On the two evaluated datasets, dense monthly tokenization recovers or exceeds the accuracy of the same network trained on daily observations, using only monthly observations and known calendar features.

1 INTRODUCTION

Many real-world forecasting problems do not have measured, reconciled, or audited data at a daily or hourly frequency. Figures such as GDP, net income and inflation are reported only monthly or quarterly. Even where daily records exist, they are often noisy. Batch operations, such as bulk orders and shipments, create isolated spikes in the daily series. A shipment that slips by a few days within the same month, for example, moves a spike from one day to another without changing the monthly total. A model trained on the daily series must learn to ignore this timing noise, whereas the monthly aggregate does not contain it.

We ask how a forecasting model should process a series when only monthly observations are available. Deep networks improve with training data, as scaling studies of language models document (Kaplan et al., 2020). When unique data run out, language models are trained for several epochs on the same tokens, with diminishing returns (Muennighoff et al., 2023) at a rate that depends on the architecture (Wu et al., 2026). A monthly series offers only twelve observations per year, which gives a monthly model few examples to learn from. The observations cannot be increased, but the tokens that represent them can, so we distinguish *observation frequency*, the available measurements, from *token density*, the number of sequence positions per measurement. We hold the monthly history fixed and test whether several tokens per value improve forecasts. Extra tokens add no information beyond the history and calendar (Proposition 1), but they change the computation a recurrent model performs on that history (Equation 5).

Patching varies how many *observations* a token contains (Nie et al., 2023; Peršak et al., 2024). We examine the complementary choice, the number of *tokens* that represent one observation. Our flat

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

representation repeats the monthly average and attaches known calendar features to each token. We construct monthly tasks from daily datasets so that the same network trained on the original daily series can be compared with the monthly-input models, and this comparison measures how much accuracy is lost by aggregating the daily data. We measure accuracy from 1 to 150 tokens per month and compare with trained baselines, zero-shot foundation models and the daily-input reference. Our contributions are this representation and evidence, on two public datasets, that a compact recurrent model reading only monthly observations recovers or exceeds the accuracy of the same network trained on the daily series (Section 5.2). Section 6 explains where the gain arises.

2 Related work

Temporal aggregation. Aggregating a series changes its variance and serial dependence, and whether the aggregate is easier or harder to forecast depends on the generating process (Silvestrini & Veredas, 2008). For simple demand processes, aggregation can reduce forecast error (Rostami-Tabar et al., 2013). Methods that combine several aggregation levels (Kourentzes et al., 2014) and temporal hierarchies (Athanasopoulos et al., 2017) build coarser series from the observed frequency, monthly included, and gain accuracy by pooling the levels. The hierarchies are reconciled by trace minimization (Wickramasuriya et al., 2019) or learned end to end (Rangapuram et al., 2023). All of these methods start from a fixed set of observations and choose the levels at which to model them. We instead ask how each monthly observation should be represented to the model.

Disaggregation and temporal coherence. Classical temporal disaggregation estimates a finer trajectory consistent with observed aggregates, using related indicators (Chow & Lin, 1971; Fernández, 1981; Litterman, 1983) or smoothness penalties (Denton, 1971). Learned approaches have the same goal, and on quarterly GDP regularized linear models are more accurate than nonlinear ones (Jung, 2025). Aggregate-disaggregate schemes forecast at an operational target frequency (Nikolopoulos et al., 2011). Our flat, aggregate-consistent path preserves each monthly average without estimating an unobserved daily pattern. Whether a finer grain helps depends on the model. In a single-series study from annual to daily grains, Moreira (2026) reports that recursive statistical models deteriorate as the grain gets finer, a linear regression is stable, and an LSTM has its lowest error at the daily grain, where daily records supply the most training observations.

Tokenization and patching. PatchTST, TimesFM and the pretrained forecasters that followed group consecutive observations into patches, so one token stands for many observations and the model processes a shorter sequence (Nie et al., 2023; Das et al., 2024; Ansari et al., 2025; Auer et al., 2025; Khwaja et al., 2026; Podest et al., 2026). Multiple-resolution tokenization (Peršak et al., 2024) builds patches of several lengths from the same series. Our setting differs in that the observations are already monthly, with no finer series available, and we represent one observation by many tokens so the model processes each observation several times. Patching and tokenization choices change forecast accuracy in ways that are hard to predict. Yu et al. (2026) show this for pretrained foundation models, and Zucchi et al. (2026) find that adaptive patching gives no gain over a well-tuned uniform patch size in models trained from scratch. FlowState (Graf et al., 2026) also separates the signal from the rate at which it is sampled, rescaling its state-space step size with that rate so that forecasts from one signal agree across sampling rates. We instead hold the monthly signal fixed and choose the rate at which the model reads it.

Computation per observation. Several methods give a network more computation on the same input. Repeat-RNN feeds each input to a recurrent network a fixed number of times (Fojo et al., 2018); adaptive computation time and PonderNet let the network choose that number (Graves, 2016; Banino et al., 2021). In language models, pause and filler tokens add extra positions for computation (Goyal et al., 2024; Pfau et al., 2024), and repeating the prompt improves accuracy without adding information (Leviathan et al., 2025). Recurrent-depth models apply one shared block repeatedly at test time (Geiping et al., 2025). On symbolic and control tasks, Takashiro et al. (2026) apply several weight-shared recurrent updates to each incoming observation before advancing the state. Flat tokenization does the same for monthly observations but makes each update distinct through calendar features and supervises only the forecast. In forecasting, YingLong (Wang et al., 2025) appends masked output patches beyond the horizon to a bidirectional encoder and gains accuracy as their number grows, with the observed history fixed. Flat tokenization adds computation on the

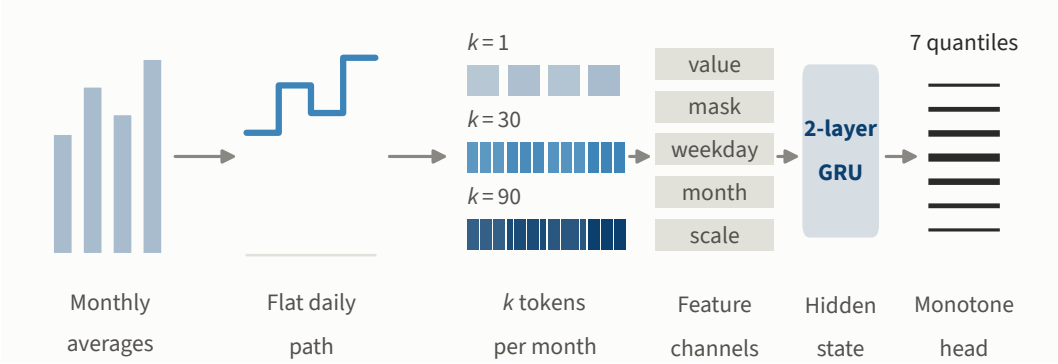

Figure 1: Overview of flat monthly tokenization. Each monthly average defines a constant path, which is sampled into k tokens per month. Each token carries the block mean and the calendar of the block's middle day, and the recurrent forecaster of Section 4.3 reads the tokens.

input side instead. Appendix D reports a variant in which all 90 tokens of a month are identical, which is Repeat-RNN without its new-token flag.

3 Monthly forecasting and the information limit

A series of monthly averages $\bar{y}_{1:M}$ is observed and the model must forecast the next H averages $\bar{y}_{M+1:M+H}$. Calendar information specific to each month, such as its number of days n_m and its weekdays, is known in advance for every month and can be used as a model input. Working with averages rather than totals is only a choice of units, as the total of month m equals the average times n_m. In the datasets studied here, each monthly average is the mean of a nonnegative daily flow $y_{m,d} \geq 0$ over the calendar days $d = 1, \ldots, n_m$ of month m,

$$\mathcal{A}_m(y) = \bar{y}_m = \frac{1}{n_m} \sum_{d=1}^{n_m} y_{m,d}. \tag{1}$$

A daily path $\tilde{y}$ is *coherent* with the observations if $\mathcal{A}_m(\tilde{y}) = \bar{y}_m$ for every $m \leq M$. Each observation constrains n_m daily values, so the daily trajectory is not identified without further assumptions.

Proposition 1 (Information limit)**.** *Let $\tilde{y} = g(\bar{y}_{1:M}, c)$ for a measurable g and a deterministic calendar c, and let $Z = \bar{y}_{M+1:M+H}$. Then $P(Z \mid \bar{y}_{1:M}, \tilde{y}) = P(Z \mid \bar{y}_{1:M})$ almost surely, so $I(Z; \tilde{y} \mid \bar{y}_{1:M}) = 0$ and the Bayes risk of any proper scoring rule for Z is the same with and without $\tilde{y}$. (Proof in Appendix A.)*

The proposition bounds what any predictor can extract from the data, but it does not describe how a trained model uses those data. A different representation changes the functions the model can learn and how they are optimized, so a trained model can still benefit from it. Observed daily values may carry information absent from the monthly averages, whereas a path built solely from those averages and the calendar cannot.

4 Method

A flat path spreads each monthly average over the days of its month (Section 4.1), a token grid samples this path into k tokens per month (Section 4.2), and a recurrent forecaster reads the tokens and emits quantiles (Section 4.3). Figure 1 summarizes the mapping.

4.1 The flat path

The flat path repeats each monthly average across all days of that month, $\tilde{y}_{m,d} = \bar{y}_m$ for all d. It is the unique coherent path that is unchanged by permuting the days, and the unique minimizer of within-month variation $\sum_d (\tilde{y}_{m,d} - \bar{y}_m)^2$. The second property holds because any coherent path can

be written $\bar{y}_m \mathbf{1} + r$ with $\sum_d r_d = 0$, so $\sum_d \tilde{y}_{m,d}^2 = n_m \bar{y}_m^2 + \sum_d r_d^2$, which is smallest at $r = 0$. If the daily values are conditionally exchangeable given the monthly history and calendar, it is also their conditional mean.

We also evaluated four smooth coherent alternatives, namely a monotone cubic interpolant, a Fourier fit, a Gaussian-process posterior mean and a trend-seasonal decomposition. All had higher CRPS than the flat path on both retail and electricity (Appendix G).

4.2 THE TOKEN GRID

For month m and a token density $k \geq 1$, the token-grid operator produces k tokens. Each contains the mean over a block $B_{m,j}$ of calendar days and the calendar features of its middle day,

$$\mathcal{T}_k(\tilde{y})_{m,j} = \Big(v_{m,j},\ c(d^*_{m,j})\Big), \qquad v_{m,j} = \frac{1}{|B_{m,j}|} \sum_{d \in B_{m,j}} \tilde{y}_{m,d}, \tag{2}$$

where $d^*_{m,j}$ is the middle day of block j and $c(d)$ collects the sine and cosine of the day of week and of the month of year of day d. Adjacent blocks may overlap, and days are repeated when $k > n_m$; Appendix I gives the boundaries. We call the forecaster of Section 4.3 trained on this grid the k-*token model*. The monthly model reads one token per month through its own input map (Appendix I) and is plotted at $k = 1$. The 30-token model uses one token per calendar day and therefore has 28 to 31 tokens per month; the 90-token model has exactly 90 tokens in every month.

For a flat path, every block mean equals the monthly average, $v_{m,j} = \bar{y}_m$ for all j and k. Thus, any value token from an observed month determines that month's average. Changing k changes the sequence length, the calendar days at which c is sampled, the mask channel (a block is observed only if all its days are) and the number of padding tokens, but leaves the monthly value unchanged. We call this *value invariance*.

Unlike the monthly input map, the dense grid encodes the day of week; Section 6.2 measures its contribution. Weekdays are known calendar information, so the dense grid stays within the setting of Proposition 1. Each dense token t carries eight channels, namely the standardized value $z_t = (\log(1 + v_t) - \mu)/\sigma$, the observation mask o_t (zero for months absent from the monthly series), the sine and cosine of the day-of-week and month-of-year angles of d^*, and the scaling statistics μ, σ across the months of the context (Appendix I).

4.3 THE FORECASTER

Let $x_{1:T}$ denote the input token sequence. A two-layer gated recurrent unit (GRU) (Cho et al., 2014) of width 128 with dropout 0.1 produces $s_t = \mathrm{GRU}(x_t, s_{t-1})$, and the final state s_T feeds a monotone quantile head on the grid $\tau_1 < \cdots < \tau_Q$ with $Q = 7$ and $\tau \in \{0.025, 0.1, 0.25, 0.5, 0.75, 0.9, 0.975\}$. A one-hidden-layer network f_ϕ maps s_T to raw outputs $r_{h,i}$, and

$$\hat{q}^z_{h,1} = r_{h,1}, \qquad \hat{q}^z_{h,i} = \hat{q}^z_{h,i-1} + \mathrm{softplus}(r_{h,i}), \quad i = 2, \ldots, Q, \qquad r = f_\phi(s_T), \tag{3}$$

for each horizon $h = 1, \ldots, H$. The superscript z denotes standardized outputs, and the original-scale forecast is $\hat{q}_{h,i} = \exp(\mu + \sigma \hat{q}^z_{h,i}) - 1$. Training minimizes the pinball loss (Koenker & Bassett, 1978) over horizons and levels in the standardized space, averaged over the episodes of a batch,

$$\mathcal{L} = \frac{1}{HQ} \sum_{h=1}^{H} \sum_{i=1}^{Q} \rho_{\tau_i}\big(z_{M+h} - \hat{q}^z_{h,i}\big), \qquad \rho_\tau(u) = u\,\big(\tau - \mathbf{1}_{u<0}\big), \tag{4}$$

with the standardized target $z_{M+h} = (\log(1 + \bar{y}_{M+h}) - \mu)/\sigma$. Two reference models share this network and head. The monthly model (L) reads one token per month, and the daily-input reference (U) reads the daily series on the calendar-day grid. The monthly model is a direct multi-horizon quantile recurrent forecaster (Wen et al., 2017), related to autoregressive recurrent forecasters such as DeepAR (Salinas et al., 2020), and is our recurrent baseline.

The recurrence applies one update at every token position, so the state at the end of month m is the composition of the updates over that month's r_m tokens,

$$s_m = \big(F_{x_{m,r_m}} \circ \cdots \circ F_{x_{m,1}}\big)(s_{m-1}), \qquad F_x(s) = \mathrm{GRU}(x, s), \tag{5}$$

where $r_m = k$ on a fixed grid and r_m is the number of calendar days on the calendar-day grid. Repeating a monthly value therefore applies more recurrent updates with the same parameters, and the updates differ because the calendar features change from token to token. Section 6.1 analyzes this computation.

4.4 The metric

We score each forecast on the original scale, separately for each horizon. The score is twice the mean pinball loss over the seven quantiles divided by the mean absolute month-to-month change of the observed context, a scaled discrete analog of the quantile representation (Laio & Tamea, 2007; Gneiting & Ranjan, 2011) of the continuous ranked probability score (Gneiting & Raftery, 2007),

$$\mathrm{CRPS}_h(\hat{q}_h, \bar{y}_{M+h}) = \frac{1}{\delta}\frac{2}{Q}\sum_{i=1}^{Q}\rho_{\tau_i}\big(\bar{y}_{M+h} - \hat{q}_{h,i}\big), \qquad \delta = \frac{1}{M-1}\sum_{m=2}^{M}\big|\bar{y}_m - \bar{y}_{m-1}\big|. \tag{6}$$

The score is computed identically for every model, and we refer to it as CRPS throughout. The denominator, the mean absolute error of a naive one-month-ahead forecast over the observed context, scales each series' errors before averaging across entities.

4.5 Evaluation and model selection

Each dataset has one calendar cutoff, an expanding training window, and a validation span of six (retail) or twelve (electricity) months before the first test origin (Table 2, Appendix A).

We report every comparison as a paired contrast. For models a and b on one dataset and horizon, let $C^a_{s,e}$ be the CRPS of model a at seed s averaged over the origins of entity e, and $D_{s,e} = C^a_{s,e} - C^b_{s,e}$. The paired contrast is $\Delta = \overline{D}$, reported as a percentage of $\overline{C^b}$ with interval endpoints divided by the same baseline mean. The series-cluster bootstrap averages D over seeds and resamples entities; the two-way bootstrap also resamples seeds to include training variability (Appendix L). We call a contrast *significant* when its interval excludes zero.

The selection criterion, fixed before the density experiments, retains a candidate over a baseline when it improves the average score over the dataset-horizon pairs, significantly improves at least one pair, and is not significantly worse on any pair (Equation 11, Appendix L). We select models on validation data using this rule and the two-way bootstrap, then evaluate the selected models once on the test data, using the series-cluster bootstrap.

5 Experiments

5.1 Setup

We construct monthly forecasting tasks from two public daily datasets (Table 1). The monthly-input models receive only the monthly aggregates, and only the daily-input reference and the foundation models given daily input read the daily series. Each model uses the learning-rate schedule selected for it on validation data, a one-cycle schedule with early stopping for the monthly and 30-token models and a warmup-stable-decay (WSD) schedule for the 90-token model (Appendix C). Test results average five training runs (Appendix L).

5.2 Gains from dense tokenization

On both datasets the 90-token model lowers test CRPS relative to the monthly model at every horizon, by 20 to 49% on retail and 28 to 36% on electricity (Figure 2; Tables 3 and 7 in Appendix A give every value). It matches the daily-input reference on electricity and outperforms it on retail, where the reference has 12 to 15% higher CRPS at every horizon. On electricity the daily data give no significant gain over flat tokenization, and the daily-input reference trained with the schedule of the 90-token model ($\mathrm{U_w}$) has 8.0% higher CRPS at horizon 1. There the 90-token model also lowers CRPS against the 30-token model by 8 to 21%.

Table 1: The two datasets, aggregated from public daily records to monthly averages.

	Retail	Electricity
Source	Online Retail II (Chen, 2012)	UCI load diagrams (Trindade, 2015)
Series	2,071 products, daily sales	336 clients, daily consumption
Monthly span	Dec 2009 to Nov 2011 (24 months)	Jan 2011 to Dec 2014 (48 months)

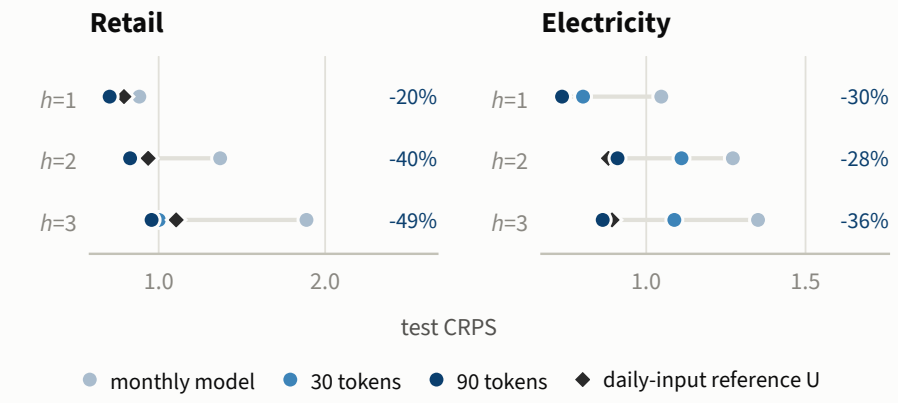

Figure 2: Test-set CRPS by forecast horizon; lower is better. Percentages are the change of the 90-token model relative to the monthly model. U is the daily-input reference (Section 4.3).

5.3 Recovery of the aggregation gap

Figure 3 expresses each monthly-input model's gain as a share of the aggregation gap, the daily-input reference's advantage over the monthly model. We use two daily-input references with the same network, U under the one-cycle schedule (Appendix C) and U_w under the WSD schedule of the 90-token model. On retail the 90-token model recovers more than the whole gap to either reference at every horizon, 116 to 193%, and on electricity 93 to 123% of it.

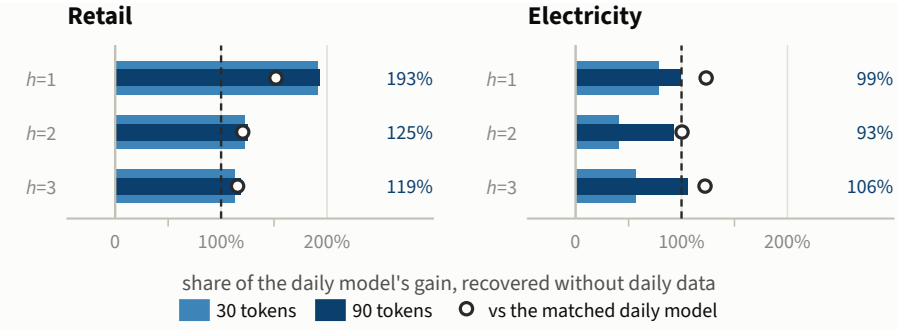

Figure 3: Share of the aggregation gap, $\mathrm{CRPS_L} - \mathrm{CRPS_U}$, recovered by the 30-token (wide bars) and 90-token (narrow bars, labeled) models. Open markers use U_w.

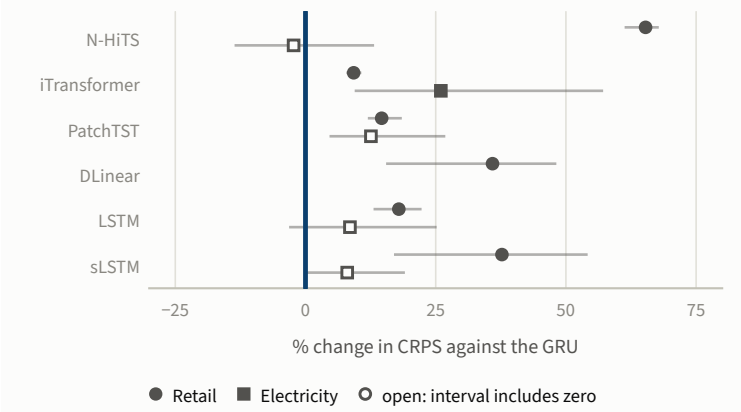

Figure 4: Trained baselines against the best GRU configuration, percentage change in CRPS; marks are the mean across horizons and bars the range; a filled mark has intervals that exclude zero at every horizon. DLinear on electricity lies beyond the axis (Table 4).

5.4 Trained baselines

Six models trained from scratch on each dataset provide the main comparison, N-HiTS (Challu et al., 2023), PatchTST (Nie et al., 2023), iTransformer (Liu et al., 2024), DLinear (Zeng et al., 2023), an LSTM (Hochreiter & Schmidhuber, 1997) and a patched sLSTM (Beck et al., 2024) (Figure 4; Table 4 in Appendix A gives every value). On retail the GRU has lower CRPS than every other trained model at every horizon, and those models have 8 to 68% higher CRPS. On electricity it has the lowest CRPS at horizon 1, N-HiTS is 13.6% lower at horizon 3, and at horizons 1 and 2 its differences from N-HiTS and the LSTM are not significant. Given the same monthly history, however, six zero-shot foundation models have similar or slightly lower median error than the GRU on retail and lower median error on electricity at longer horizons. Given the daily series, they have higher error than the GRU on retail at every horizon (Appendix B).

6 Why repetition helps

6.1 Repetition deepens the recurrent computation

Flat tokenization separates the frequency of observation from the depth of computation. With 90 tokens per month, the GRU performs 1,080 updates over the retail context and 1,620 over the electricity context, against 12 and 18 for the monthly model and about 365 and 548 for the 30-token model, with the same recurrent parameters throughout. The history remains twelve or eighteen monthly values, but each participates in many successive nonlinear transformations.

In a GRU each update is a gated interpolation. Writing one layer's retention gate as a_j and its candidate state as u_j,

$$s_j = s_{j-1} + (1 - a_j) \odot (u_j - s_{j-1}). \tag{7}$$

Each coordinate moves a fraction $1-a_j$ of the way toward its candidate. Repetition could help in two ways. The GRU can act as a leaky integrator whose gates set its timescale (Cho et al., 2014; Tallec & Ollivier, 2018), so many tokens with gates near 1 would integrate each month slowly. Alternatively, the gates and the candidate depend on the evolving state, so repetition applies a changing nonlinear update many times and deepens the computation on each value. The trained models follow the second route.

The first route requires gates near 1. With a scalar gate a and candidate u held constant over the k tokens of a month,

$$s_k = u + a^k(s_0 - u), \tag{8}$$

so a share a^k of the initial state survives the month. The gates of the trained dense models average 0.45 to 0.59 per token and their product over a month is at most 0.04 (Table 13, Appendix E), so Equation 8 predicts that each month erases the state. Yet the state moves over a month about as far

as the monthly model's state moves between two months, 0.2 to 0.6 of its norm in the second layer against 0.3 to 0.4. The candidate therefore stays close to the current state, so each token makes a small correction computed from the state and the calendar, and memory in the dense models lies in the candidate map rather than in slow gates.

The number of updates affects accuracy. Patching the flat tokens so that the recurrence takes 10 steps per month instead of 30 or 90 raises retail validation CRPS at horizon 1 by 68% and 41%, respectively (Appendix K). More updates do not help without limit, as 150 tokens do not improve on 90 (Appendix C).

6.2 Where the gain comes from

The density gains are largest on electricity, where the 90-token model has 24 to 28% lower CRPS than a model reading one dense token per month, and raising density from 30 to 90 tokens alone lowers CRPS by 6 to 14% (Table 11, Appendix D). The calendar that changes from token to token contributes to this gain, since giving every token of a month the mid-month calendar raises CRPS by 13 to 23%. On retail, the dense channels lower CRPS by 16 to 30% relative to the monthly model already at one token per month, and density lowers it by a further 4 to 22%, a gain that grows with the horizon.

6.3 Where the probabilistic improvement occurs

The score can be split into a median part and a nonmedian part. Let A_h be the absolute error of the median forecast divided by δ of Equation 6 and averaged as CRPS is, and let O_h be the mean of twice the pinball loss at the six nonmedian quantiles under the same scaling and averaging. Since twice the median pinball loss equals absolute error, the CRPS C_h satisfies exactly

$$C_h = \frac{A_h + 6O_h}{7}, \qquad O_h = \frac{7C_h - A_h}{6}, \tag{9}$$

which separates the forecast center from the other quantiles without changing the score. Figure 5 shows the loss at each level.

Against the monthly model, the 90-token model lowers both parts of the score on both datasets, median error by 22 to 51% and nonmedian loss by 20 to 49%. Against the 30-token model, it again improves both parts on electricity, lowering median error by 11.6, 21.1 and 21.7% and nonmedian loss by 7.0, 17.0 and 20.3% at horizons 1 to 3. On retail this further gain comes entirely from the nonmedian quantiles, whose loss falls by 0.9, 2.4 and 5.7%, while median error is 2.0, 1.9 and 1.1% higher.

6.4 Three sources of the gain

The 90-token model uses the network of the monthly model, reads the same monthly values and, by Proposition 1, receives no information the monthly model lacks, yet its CRPS is lower at every horizon on both datasets (Section 5.2). Three mechanisms contribute to the gap.

The first is depth of computation, which the parameter count does not constrain (Equation 5). Fewer updates per month raise CRPS (Section 6.1), each increase in density lowers test CRPS, and the 90-token model has 4 to 22% lower CRPS on retail and 24 to 28% lower on electricity than the strongest one-token model with the dense input map (Appendix D).

The second is the calendar. Electricity depends on the month-of-year channel, whose removal raises CRPS by 82 to 90%, and retail on the weekday channel, whose removal raises CRPS by 10 to 12%. Identical weekdays on every token of a retail month cost at most 2%, which is consistent with the retail model using the weekday composition of the month (Equation 10, Appendix D) rather than its within-month sequence.

The third is invariance. A reallocation of a month's daily values that preserves its average leaves the flat input unchanged (Section 4.2), so the forecaster never has to learn which daily spikes matter, whereas the daily-input reference must. On retail, where batch orders make the daily series spiky, the reference has higher CRPS than the 90-token model at every horizon (Section 5.2). Daily observations can only lower the Bayes risk, so the 90-token model is ahead where the daily-input

reference's estimation and optimization error exceed the value of that information. None of the three sources adds parameters, and the 90-token GRU trains in 2 to 4 minutes on a laptop GPU, against 6 to 79 minutes for the LSTM and 40 to 50 minutes for the patched sLSTM (Table 18).

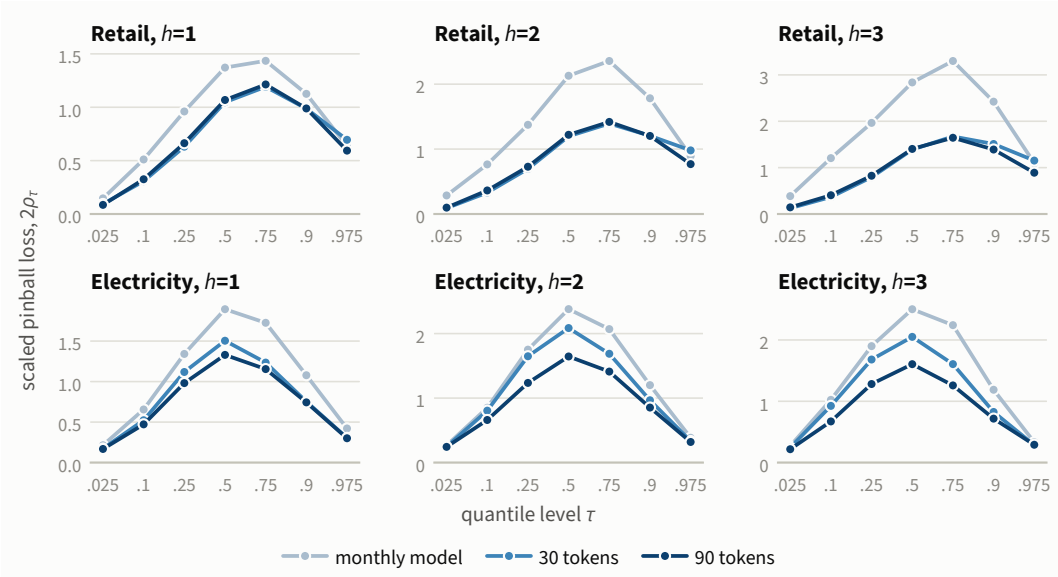

Figure 5: Scaled pinball loss at each of the seven quantile levels on the test set for the monthly, 30-token and 90-token models; the mean over the seven levels is the CRPS of Equation 6.

7 Conclusion

We asked whether a model that sees only monthly values can forecast them better by reading each value several times. On two public datasets, a 174k-parameter GRU with 90 calendar-tagged tokens per month lowers CRPS against the monthly model by 20 to 49% on retail and 28 to 36% on electricity without adding parameters, matches the same network reading the daily series on electricity, and outperforms it on retail at every horizon. Among the trained models, it has the lowest CRPS at every retail horizon and at electricity horizon 1, and N-HiTS has the lowest at electricity horizons 2 and 3. The gain has three sources, namely the depth of computation from applying the same recurrent update many times to each value, the calendar features on each token, and the invariance of the flat input to how daily values are distributed within a month. Token density is therefore a useful modeling choice for monthly forecasting. The evidence covers two datasets, two recurrent architectures and horizons of one to three months.

Reproducibility statement

Section 4 specifies the operators, model, training loss, and evaluation metric, Table 2 the splits, and Section 4.5 the contrasts, both bootstraps and the selection criterion. Appendix C gives the optimizer configurations and the selection outcome, Appendix K the definition of each adapted architecture, Appendix M the GPU hours per experiment, Appendices F to H the decomposition study, priors and robustness checks, and Appendix L the seed caveats. All datasets are public. Training of the GRU family is deterministic given the seed. The supplementary material contains the code, the scored results behind every figure and table, and the scripts that regenerate them.

Ethics statement

The work uses two public datasets of retail transactions and electricity consumption. The Online Retail II source includes a pseudonymous customer number; our cleaning aggregates transactions to daily units per product and discards that field, so no customer-level information enters the experiments. The electricity dataset contains no personal identifiers.

AI USE STATEMENT

In this paper, we used generative AI tools to implement methods, and the training, evaluation and plotting code was written or reviewed with AI assistance under the authors' supervision. We have not used generative AI tools to develop the theoretical framework, formulate or prove the mathematical claims, propose or refine hypotheses, design the research methodology or experiments, or interpret results. All code was reviewed and tested by the authors, and we take responsibility for the final content of this work, including code or artifacts produced with the aid of generative AI.

REFERENCES

Taha Aksu, Gerald Woo, Juncheng Liu, Xu Liu, Chenghao Liu, Silvio Savarese, Caiming Xiong, and Doyen Sahoo. GIFT-Eval: A benchmark for general time series forecasting model evaluation. In *NeurIPS 2024 Workshop on Time Series in the Age of Large Models*, 2024.

Abdul Fatir Ansari, Oleksandr Shchur, Jaris Küken, Andreas Auer, Boran Han, Pedro Mercado, Syama Sundar Rangapuram, Huibin Shen, Lorenzo Stella, Xiyuan Zhang, Mononito Goswami, Shubham Kapoor, Danielle C. Maddix, Pablo Guerron, Tony Hu, Junming Yin, Nick Erickson, Prateek Mutalik Desai, Hao Wang, Huzefa Rangwala, George Karypis, Yuyang Wang, and Michael Bohlke-Schneider. Chronos-2: From univariate to universal forecasting. *arXiv preprint arXiv:2510.15821*, 2025. doi: 10.48550/arXiv.2510.15821.

V. Assimakopoulos and K. Nikolopoulos. The theta model: a decomposition approach to forecasting. *International Journal of Forecasting*, 16(4):521–530, 2000. doi: 10.1016/S0169-2070(00)00066-2.

George Athanasopoulos, Rob J. Hyndman, Nikolaos Kourentzes, and Fotios Petropoulos. Forecasting with temporal hierarchies. *European Journal of Operational Research*, 262(1):60–74, 2017. doi: 10.1016/j.ejor.2017.02.046.

Andreas Auer, Patrick Podest, Daniel Klotz, Sebastian Böck, Günter Klambauer, and Sepp Hochreiter. TiRex: Zero-shot forecasting across long and short horizons with enhanced in-context learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Andrea Banino, Jan Balaguer, and Charles Blundell. PonderNet: Learning to ponder. In *8th ICML Workshop on Automated Machine Learning (AutoML 2021)*, 2021.

Maximilian Beck, Korbinian Pöppel, Markus Spanring, Andreas Auer, Oleksandra Prudnikova, Michael Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. xLSTM: Extended long short-term memory. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

William R. Bell and Steven C. Hillmer. Modeling time series with calendar variation. *Journal of the American Statistical Association*, 78(383):526–534, 1983. doi: 10.1080/01621459.1983.10478005.

Cristian Challu, Kin G. Olivares, Boris N. Oreshkin, Federico Garza, Max Mergenthaler-Canseco, and Artur Dubrawski. NHITS: Neural hierarchical interpolation for time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 6989–6997, 2023. doi: 10.1609/aaai.v37i6.25854.

Daqing Chen. Online Retail II. UCI Machine Learning Repository, 2012.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1724–1734, 2014. doi: 10.3115/v1/D14-1179. URL https://aclanthology.org/D14-1179/.

Gregory C. Chow and An-loh Lin. Best linear unbiased interpolation, distribution, and extrapolation of time series by related series. *The Review of Economics and Statistics*, 53(4):372–375, 1971. doi: 10.2307/1928739.

William S. Cleveland and Susan J. Devlin. Calendar effects in monthly time series: Detection by spectrum analysis and graphical methods. *Journal of the American Statistical Association*, 75 (371):487–496, 1980. doi: 10.1080/01621459.1980.10477500.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pp. 10148–10167, 2024.

Frank T. Denton. Adjustment of monthly or quarterly series to annual totals: An approach based on quadratic minimization. *Journal of the American Statistical Association*, 66(333):99–102, 1971. doi: 10.1080/01621459.1971.10482227.

Roque B. Fernández. A methodological note on the estimation of time series. *The Review of Economics and Statistics*, 63(3):471–476, 1981. doi: 10.2307/1924371.

Daniel Fojo, Víctor Campos, and Xavier Giró-i Nieto. Comparing fixed and adaptive computation time for recurrent neural networks. In *International Conference on Learning Representations, Workshop Track*, 2018. URL https://openreview.net/forum?id=SkZq3vyDf.

Jonas Geiping, Sean McLeish, Neel Jain, John Kirchenbauer, Siddharth Singh, Brian R. Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, and Tom Goldstein. Scaling up test-time compute with latent reasoning: A recurrent depth approach. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Tilmann Gneiting and Adrian E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007. doi: 10.1198/016214 506000001437.

Tilmann Gneiting and Roopesh Ranjan. Comparing density forecasts using threshold- and quantile-weighted scoring rules. *Journal of Business & Economic Statistics*, 29(3):411–422, 2011. doi: 10.1198/jbes.2010.08110.

Sachin Goyal, Ziwei Ji, Ankit Singh Rawat, Aditya Krishna Menon, Sanjiv Kumar, and Vaishnavh Nagarajan. Think before you speak: Training language models with pause tokens. In *International Conference on Learning Representations (ICLR)*, 2024.

Lars Graf, Thomas Ortner, Stanisław Woźniak, and Angeliki Pantazi. FlowState: Sampling-rate-equivariant time-series forecasting. In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*, volume 306 of *Proceedings of Machine Learning Research*, 2026.

Alex Graves. Adaptive computation time for recurrent neural networks. *arXiv preprint arXiv:1603.08983*, 2016.

Léo Grinsztajn, Klemens Flöge, Oscar Key, Felix Birkel, Philipp Jund, Brendan Roof, Mihir Manium, Shi Bin Hoo, Magnus Bühler, Anurag Garg, Dominik Safaric, Jake Robertson, Benjamin Jäger, Simone Alessi, Adrian Hayler, Vladyslav Moroshan, Lennart Purucker, Philipp Singer, Alan Arazi, Julien Siems, Jan Hendrik Metzen, Georg Grab, Nick Erickson, Siyuan Guo, Eliott Kalfon, Simon Bing, David Salinas, Clara Cornu, Lilly Charlotte Wehrhahn, Diana Kriuchkova, Kursat Kaya, Lydia Sidhoum, Marie Salmon, Jerry Chen, Madelon Hulsebos, Yann LeCun, Samuel Müller, Bernhard Schölkopf, Sauraj Gambhir, Noah Hollmann, and Frank Hutter. TabPFN-3: Technical report. *arXiv preprint arXiv:2605.13986*, 2026.

Alexander Hägele, Elie Bakouch, Atli Kosson, Loubna Ben Allal, Leandro Von Werra, and Martin Jaggi. Scaling laws and compute-optimal training beyond fixed training durations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8): 1735–1780, 1997. doi: 10.1162/neco.1997.9.8.1735.

Shi Bin Hoo, Samuel Müller, David Salinas, and Frank Hutter. From tables to time: Extending TabPFN-v2 to time series forecasting. *arXiv preprint arXiv:2501.02945*, 2025.

Shengding Hu, Yuge Tu, Xu Han, Ganqu Cui, Chaoqun He, Weilin Zhao, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Xinrong Zhang, Zhen Leng Thai, Chongyi Wang, Yuan Yao, Chenyang Zhao, Jie Zhou, Jie Cai, Zhongwu Zhai, Ning Ding, Chao Jia, Guoyang Zeng, Dahai Li, Zhiyuan Liu, and Maosong Sun. MiniCPM: Unveiling the potential of small language models with scalable training strategies. In *First Conference on Language Modeling (COLM)*, 2024.

Rob J. Hyndman and George Athanasopoulos. *Forecasting: Principles and Practice*. OTexts, Melbourne, Australia, 3rd edition, 2021. URL https://otexts.com/fpp3/.

Rob J. Hyndman and Yeasmin Khandakar. Automatic time series forecasting: The forecast package for R. *Journal of Statistical Software*, 27(3):1–22, 2008. doi: 10.18637/jss.v027.i03.

Ayush Jain and Rajat Sen. TimesFM-3: A zero-shot foundation model for multivariate forecasting. Google Research blog, 31 August, 2026. URL https://research.google/blog/timesfm-3-a-zero-shot-foundation-model-for-multivariate-forecasting/. Accessed 9 September 2026.

Yonggeun Jung. Temporal disaggregation of GDP: When does machine learning help? *arXiv preprint arXiv:2506.14078*, 2025. doi: 10.48550/arXiv.2506.14078. Version 3, April 2026.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Emaad Khwaja, Chris Lettieri, Gerald Woo, Eden Belouadah, Marc Cenac, Guillaume Jarry, Enguerrand Paquin, Xunyi Zhao, Viktoriya Zhukov, Othmane Abou-Amal, Chenghao Liu, Ameet Talwalkar, and David Asker. Toto 2.0: Time series forecasting enters the scaling era. *arXiv preprint arXiv:2605.20119*, 2026. doi: 10.48550/arXiv.2605.20119.

Taesung Kim, Jinhee Kim, Yunwon Tae, Cheonbok Park, Jang-Ho Choi, and Jaegul Choo. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International Conference on Learning Representations (ICLR)*, 2022. URL https://openreview.net/forum?id=cGDAkQo1C0p.

Leslie Kish. *Survey Sampling*. John Wiley & Sons, New York, 1965.

Roger Koenker and Gilbert Bassett. Regression quantiles. *Econometrica*, 46(1):33–50, 1978. doi: 10.2307/1913643.

Nikolaos Kourentzes, Fotios Petropoulos, and Juan R. Trapero. Improving forecasting by estimating time series structural components across multiple frequencies. *International Journal of Forecasting*, 30(2):291–302, 2014. doi: 10.1016/j.ijforecast.2013.09.006.

Francesco Laio and Stefania Tamea. Verification tools for probabilistic forecasts of continuous hydrological variables. *Hydrology and Earth System Sciences*, 11(4):1267–1277, 2007. doi: 10.5194/hess-11-1267-2007.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Prompt repetition improves non-reasoning LLMs. *arXiv preprint arXiv:2512.14982*, 2025.

Robert B. Litterman. A random walk, Markov model for the distribution of time series. *Journal of Business & Economic Statistics*, 1(2):169–173, 1983. doi: 10.1080/07350015.1983.10509336.

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

Marcel Meyer, Sascha Kaltenpoth, Kevin Zalipski, and Oliver Müller. Rethinking evaluation in the era of time series foundation models: (Un)known information leakage challenges. *arXiv preprint arXiv:2510.13654*, 2025.

Hugo Moreira. The granularity paradox: How temporal disaggregation inflates in-sample fit and compounds out-of-sample error. *arXiv preprint arXiv:2607.05450*, 2026. doi: 10.48550/arXiv.2607.05450.

Niklas Muennighoff, Alexander M. Rush, Boaz Barak, Teven Le Scao, Nouamane Tazi, Aleksandra Piktus, Sampo Pyysalo, Thomas Wolf, and Colin Raffel. Scaling data-constrained language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, 2023.

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

Konstantinos Nikolopoulos, Aris A. Syntetos, John E. Boylan, Fotios Petropoulos, and Vassilios Assimakopoulos. An aggregate–disaggregate intermittent demand approach (ADIDA) to forecasting: an empirical proposition and analysis. *Journal of the Operational Research Society*, 62 (3):544–554, 2011. doi: 10.1057/jors.2010.32.

Egon Peršak, Miguel F. Anjos, Sebastian Lautz, and Aleksandar Kolev. Multiple-resolution tokenization for time series forecasting with an application to pricing. *arXiv preprint arXiv:2407.03185*, 2024. doi: 10.48550/arXiv.2407.03185.

Jacob Pfau, William Merrill, and Samuel R. Bowman. Let's think dot by dot: Hidden computation in transformer language models. In *First Conference on Language Modeling (COLM)*, 2024.

Patrick Podest, Marco Pichler, Elias Bürger, Levente Zólyomi, Bernhard Voggenberger, Wilhelm Berghammer, Daniel Klotz, Sebastian Böck, Günter Klambauer, and Sepp Hochreiter. TiRex-2: Generalizing TiRex to multivariate data and streaming. *arXiv preprint arXiv:2607.01204*, 2026. doi: 10.48550/arXiv.2607.01204.

Zhongzheng Qiao, Sheng Pan, Anni Wang, Viktoriya Zhukova, Yong Liu, Xudong Jiang, Qingsong Wen, Mingsheng Long, Ming Jin, and Chenghao Liu. It's TIME: Towards the next generation of time series forecasting benchmarks. In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*, volume 306 of *Proceedings of Machine Learning Research*, 2026.

Syama Sundar Rangapuram, Shubham Kapoor, Rajbir S. Nirwan, Pedro Mercado, Tim Januschowski, Yuyang Wang, and Michael Bohlke-Schneider. Coherent probabilistic forecasting of temporal hierarchies. In *Proceedings of the 26th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 206 of *Proceedings of Machine Learning Research*, pp. 9362–9376, 2023.

Bahman Rostami-Tabar, M. Zied Babai, Aris Syntetos, and Yves Ducq. Demand forecasting by temporal aggregation. *Naval Research Logistics*, 60(6):479–498, 2013. doi: 10.1002/nav.21546.

David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 36(3): 1181–1191, 2020. doi: 10.1016/j.ijforecast.2019.07.001.

Oleksandr Shchur, Abdul Fatir Ansari, Caner Turkmen, Lorenzo Stella, Nick Erickson, Pablo Guerron, Michael Bohlke-Schneider, and Yuyang Wang. fev-bench: A realistic benchmark for time series forecasting. *arXiv preprint arXiv:2509.26468*, 2025.

Andrea Silvestrini and David Veredas. Temporal aggregation of univariate and multivariate time series models: A survey. *Journal of Economic Surveys*, 22(3):458–497, 2008. doi: 10.1111/j.1467-6419.2007.00538.x.

Leslie N. Smith and Nicholay Topin. 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, pp. 1100612. SPIE, 2019. doi: 10.1117/12.2520589.

Aris A. Syntetos and John E. Boylan. The accuracy of intermittent demand estimates. *International Journal of Forecasting*, 21(2):303–314, 2005. doi: 10.1016/j.ijforecast.2004.10.001.

Shota Takashiro, Masanori Koyama, Takeru Miyato, Yusuke Iwasawa, Yutaka Matsuo, and Kohei Hayashi. Exploration of fast-slow latent recurrence for train-short, test-long generalization. *arXiv preprint arXiv:2604.01577*, 2026.

Corentin Tallec and Yann Ollivier. Can recurrent neural networks warp time? In *International Conference on Learning Representations (ICLR)*, 2018.

Artur Trindade. ElectricityLoadDiagrams20112014. UCI Machine Learning Repository, 2015.

Xue Wang, Tian Zhou, Jinyang Gao, Bolin Ding, and Jingren Zhou. Output scaling: YingLong-delayed chain of thought in a large pretrained time series forecasting model. *arXiv preprint arXiv:2506.11029*, 2025.

Ruofeng Wen, Kari Torkkola, Balakrishnan Narayanaswamy, and Dhruv Madeka. A multi-horizon quantile recurrent forecaster. In *NIPS 2017 Time Series Workshop*, 2017.

Shanika L. Wickramasuriya, George Athanasopoulos, and Rob J. Hyndman. Optimal forecast reconciliation for hierarchical and grouped time series through trace minimization. *Journal of the American Statistical Association*, 114(526):804–819, 2019. doi: 10.1080/01621459.2018.1448825.

Boqian Wu, Qiao Xiao, Patrik Okanovic, Tomasz Sternal, Maurice van Keulen, Mykola Pechenizkiy, Elena Mocanu, Torsten Hoefler, and Decebal Constantin Mocanu. When data is scarce: Scaling sparse language models with repeated training. In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*, volume 306 of *Proceedings of Machine Learning Research*, 2026.

Annan Yu, Danielle C. Maddix, Boran Han, Xiyuan Zhang, Abdul Fatir Ansari, Oleksandr Shchur, Christos Faloutsos, Andrew Gordon Wilson, Michael W. Mahoney, and Yuyang Wang. Understanding the implicit biases of design choices for time series foundation models. In *International Conference on Learning Representations (ICLR)*, 2026. URL `https://openreview.net/forum?id=5jkzTzV5Ao`.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023. doi: 10.1609/aaai.v37i9.26317.

Federico Zucchi, Yi Xie, Chao Zhang, Keyuan Luo, Thomas Lampert, and Ziyue Li. Adaptive patching is harder than it looks for time-series forecasting. *arXiv preprint arXiv:2606.04074*, 2026. doi: 10.48550/arXiv.2606.04074.

A TEST-SET TABLES AND RECOVERY

Proof of Proposition 1. $\tilde{y}$ is $\sigma(\bar{y}_{1:M})$-measurable, so $\sigma(\bar{y}_{1:M}, \tilde{y}) = \sigma(\bar{y}_{1:M})$ and the conditional distribution of Z, of which a Bayes-optimal forecast is a function, is unchanged. □

For each dataset and horizon, we average the score of Equation 6 over seeds, then over forecast origins within each entity, and finally over entities. Validation origins start the month after the last training target, so the first scored validation targets fall one month into the reserved span. Retail products enter the panel if they have at least 60 days of sales over the whole record, so eligibility uses activity after the training cutoff. As a result, training and validation episodes exist for 1,745 products and 160 clients, and test episodes for 2,071 products and 336 clients, so the test set includes entities with no training history.

Tables 3, 7 and 8 give the test contrasts behind Figures 2 and 3, and Table 4 places every trained test-set model beside the best GRU configuration. The last column of that table is the classical model with the lowest unscaled horizon-1 validation CRPS, the naive forecast on retail and ETS on electricity, rescored under Equation 6. Against it, the best GRU has 5.4% lower CRPS at retail horizon 1, is level at horizon 2 and 1.8% higher at horizon 3, and on electricity it is 22.7% lower at horizon 1, 5.7% lower at horizon 2 and level at horizon 3. Under the entity mean without the divisor (Appendix F), the GRU family is lower than every classical model at every horizon.

Table 2: Contexts, horizons and split boundaries of the two datasets. Electricity models forecast six horizons; the main text reports the first three and Table 3 all six.

	Retail	Electricity
Context M, horizons H	12 months, 3	18 months, 6
Training targets	through Nov 2010	through Jun 2012
Reserved validation months	Dec 2010 to May 2011	Jul 2012 to Jun 2013
Scored validation targets	Jan to May 2011	Aug 2012 to Jun 2013
Test origins	May and Aug 2011	Jun 2013, Dec 2013, Jun 2014

Table 3: Test-set CRPS, mean of five random seeds, and percentage contrasts with 95% series-cluster bootstrap intervals; * excludes zero. Columns: the monthly, 30-token and 90-token models and the packed 90-token model. Percentages are relative to the second model named. Bold marks the lowest CRPS in a row.

Dataset	h	monthly	30-token	90-token	90-token packed	% 90-token − monthly	% 90-token − 30-token
Retail	1	0.884	0.708	**0.706**	0.706	−20.2 [−23.7, −16.7]*	−0.3 [−1.5, +1.0]
	2	1.369	0.841	**0.828**	0.866	−39.5 [−44.6, −34.6]*	−1.6 [−2.7, −0.3]*
	3	1.888	1.001	**0.957**	1.075	−49.3 [−54.8, −44.1]*	−4.4 [−5.5, −3.1]*
Electricity	1	1.047	0.802	0.736	**0.723**	−29.8 [−46.4, −15.7]*	−8.2 [−17.5, −1.4]*
	2	1.272	1.111	**0.910**	0.911	−28.5 [−44.2, −15.4]*	−18.1 [−27.8, −10.7]*
	3	1.351	1.088	**0.863**	0.869	−36.1 [−53.1, −23.2]*	−20.7 [−32.9, −11.4]*
	4	1.408	1.094	**1.032**	1.033	−26.7 [−47.0, −13.5]*	−5.6 [−19.3, +5.4]
	5	1.385	**1.019**	1.051	1.022	−24.1 [−45.6, −10.7]*	+3.1 [−7.0, +11.4]
	6	1.631	**1.206**	1.280	1.267	−21.5 [−36.5, −11.8]*	+6.1 [−0.1, +10.9]

The divisor δ of Equation 6 is computed over the observed months of the context; when every month-to-month change is zero it falls back to the mean absolute level of the context, and to 1 when that is also zero. We use the divisor because, without it, the entity mean of the electricity panel has a Kish effective sample size (Kish, 1965) of 8.6 to 9.0 of 336 clients, $(\sum_e c_e)^2 / \sum_e c_e^2$ for the entity means c_e, so a few large clients dominate the mean. On retail, by contrast, seed variance exceeds entity variance (Appendix L).

Table 4: The best of our GRU configurations against the trained baselines on the test set, mean of five seeds. In each row the first column is the lowest value among the monthly-input GRU configurations of Table 6. The upper block is the CRPS of Equation 6, the lower block the scaled absolute error of the median forecast. Bold marks the lowest value in a row and underline the second lowest. The best retail configuration is the four-layer 90-token model of Appendix J.

Dataset	h	GRU (best, ours)	N-HiTS	LSTM 90	PatchTST 90	iTransformer 90	DLinear 90	sLSTM 90	Best classical
CRPS									
Retail	1	**0.677**	1.091	0.765	0.802	0.739	0.975	0.791	0.716
	2	**0.792**	1.329	0.938	0.887	0.877	0.915	1.124	0.795
	3	0.941	1.569	1.151	1.067	1.015	1.394	1.451	**0.924**
Electricity	1	**0.664**	0.752	0.831	0.843	0.740	3.980	0.791	0.859
	2	0.866	**0.810**	0.896	0.906	0.947	3.394	0.910	0.918
	3	0.782	**0.675**	0.757	0.831	1.229	3.172	0.781	0.781
Median AE									
Retail	1	**1.022**	1.349	1.133	1.230	1.156	1.538	1.191	1.030
	2	1.188	1.635	1.358	1.353	1.366	1.272	1.712	**1.164**
	3	1.389	1.939	1.631	1.631	1.617	1.934	2.178	**1.359**
Electricity	1	1.173	**1.156**	1.455	1.442	1.229	5.276	1.291	1.477
	2	1.529	**1.342**	1.573	1.501	1.638	6.638	1.503	1.596
	3	1.407	**1.149**	1.326	1.517	2.148	5.155	1.300	1.298

Table 5: The best of our GRU configurations against the zero-shot foundation models at the monthly input on the test set, first column as in Table 4. Blocks, bold and underline as in Table 4; a dash marks a model whose released interface returns nine deciles. † marks a model whose published training corpus overlaps the electricity dataset.

Dataset	h	GRU (best, ours)	Chronos-2†	TimesFM 3	TiRex 2†	Toto 2	Toto 2 (2.5B)	TabPFN-TS-3
CRPS								
Retail	1	0.677	**0.659**	–	–	–	–	0.675
	2	0.792	**0.740**	–	–	–	–	0.769
	3	0.941	**0.858**	–	–	–	–	0.915
Electricity	1	0.664	0.700	–	–	–	–	**0.660**
	2	0.866	0.777	–	–	–	–	**0.719**
	3	0.782	0.735	–	–	–	–	**0.604**
Median AE								
Retail	1	1.022	0.988	0.979	**0.966**	1.036	1.039	0.993
	2	1.188	1.110	1.106	**1.092**	1.171	1.167	1.132
	3	1.389	1.281	1.266	**1.261**	1.350	1.359	1.307
Electricity	1	1.173	1.241	1.156	1.332	1.285	1.304	**1.129**
	2	1.529	1.392	1.290	1.535	1.496	1.552	**1.278**
	3	1.407	1.232	1.077	1.269	1.404	1.443	**1.039**

Table 6: The recurrent family on the test set, mean of five seeds: the 90-token GRU, the other GRU configurations and the LSTM at the same three densities. Blocks, bold and underline as in Table 4. The daily-input GRU U reads the daily series; every other column reads the monthly averages or the flat path built from them.

Dataset	h	GRU 90 (ours)	GRU 90, 4 layers	GRU 30	GRU monthly	GRU, daily-input reference U	GRU 90, one-cycle	GRU monthly WSD	LSTM 90	LSTM 30	LSTM monthly
CRPS											
Retail	1	0.706	**0.677**	0.708	0.884	0.792	0.749	0.785	0.765	0.750	0.804
	2	0.828	**0.792**	0.841	1.369	0.937	0.868	1.111	0.938	0.869	0.974
	3	0.957	**0.941**	1.001	1.888	1.105	1.010	1.454	1.151	1.037	1.178
Electricity	1	0.736	0.726	0.802	1.047	0.733	**0.664**	1.070	0.831	0.780	1.279
	2	0.910	0.876	1.111	1.272	0.881	**0.866**	1.335	0.896	0.998	1.535
	3	0.863	0.839	1.088	1.351	0.893	0.782	1.382	**0.757**	0.941	1.446
Median AE											
Retail	1	1.068	**1.022**	1.047	1.372	1.190	1.125	1.219	1.133	1.144	1.276
	2	1.220	**1.188**	1.197	2.130	1.381	1.264	1.657	1.358	1.327	1.464
	3	1.403	1.411	**1.389**	2.838	1.614	1.439	2.099	1.631	1.565	1.725
Electricity	1	1.330	1.316	1.504	1.894	1.339	**1.173**	1.940	1.455	1.334	2.626
	2	1.645	1.618	2.086	2.380	1.531	**1.529**	2.485	1.573	1.753	2.915
	3	1.605	1.552	2.050	2.501	1.538	1.407	2.553	**1.326**	1.644	2.814

Table 7: Test-set recovery of the aggregation gap, mean of five seeds. Columns: CRPS of the monthly model, the 30- and 90-token models and the two daily-input references, U under the one-cycle schedule and $\mathrm{U_w}$ under the WSD schedule of the 90-token model; recovery $(\mathrm{CRPS_L} - \mathrm{CRPS}_X)/(\mathrm{CRPS_L} - \mathrm{CRPS_U})$ and the same against $\mathrm{U_w}$.

		CRPS					Recovery vs U (%)		Recovery vs $\mathrm{U_w}$ (%)	
Dataset	h	monthly	30-token	90-token	U	$\mathrm{U_w}$	30-token	90-token	30-token	90-token
Retail	1	0.884	0.708	0.706	0.792	0.766	191	193	150	152
	2	1.369	0.841	0.828	0.937	0.920	122	125	118	120
	3	1.888	1.001	0.957	1.105	1.083	113	119	110	116
Electricity	1	1.047	0.802	0.736	0.733	0.794	78	99	97	123
	2	1.272	1.111	0.910	0.881	0.912	41	93	45	100
	3	1.351	1.088	0.863	0.893	0.952	57	106	66	122
	4	1.408	1.094	1.032	0.992	1.075	75	90	94	113
	5	1.385	1.019	1.051	1.033	1.131	104	95	144	132
	6	1.631	1.206	1.280	1.155	1.296	89	74	127	105

Table 8: The two daily-input references' CRPS relative to the 90-token model, with 95% series-cluster intervals, positive when the reference has the higher CRPS; * excludes zero. Companion to Table 7.

Dataset	h	% daily-input reference U − 90-token	% daily-input reference $\mathrm{U_w}$ − 90-token
Retail	1	+12.2 [+8.9, +15.7]*	+8.6 [+6.4, +10.9]*
	2	+13.1 [+9.7, +16.7]*	+11.1 [+8.3, +14.0]*
	3	+15.4 [+12.0, +19.2]*	+13.1 [+10.0, +16.4]*
Electricity	1	−0.4 [−6.7, +7.4]	+8.0 [+1.8, +15.5]*
	2	−3.1 [−8.6, +3.5]	+0.2 [−3.7, +4.9]
	3	+3.4 [−7.5, +17.4]	+10.3 [−2.4, +27.6]
	4	−3.9 [−14.2, +9.3]	+4.2 [−9.9, +23.2]
	5	−1.7 [−8.5, +7.1]	+7.6 [−5.2, +24.2]
	6	−9.7 [−14.2, −4.7]*	+1.3 [−7.7, +12.7]

Table 9: Scaled absolute error of the median on the retail test set at horizons 1 to 3, its percentage change against the 90-token GRU, four-layer on retail, and CRPS at horizon 1, for the zero-shot foundation models and N-HiTS (Appendix B). A dash marks a model that returns nine deciles and * an interval that excludes zero.

		Median AE			% vs the 90-token GRU			
Model	Input	h=1	h=2	h=3	h=1	h=2	h=3	CRPS (h=1)
Chronos-2	monthly	0.988	1.110	1.281	−3.3*	−6.6*	−9.2*	0.659
Chronos-2	flat daily path	1.099	1.274	1.438	+7.6*	+7.2*	+1.9	0.821
Chronos-2	daily	1.267	1.352	1.515	+24.0*	+13.8*	+7.4*	0.769
TimesFM 3	monthly	0.979	1.106	1.266	−4.2*	−6.9*	−10.3*	–
TimesFM 3	flat daily path	1.011	1.149	1.318	−1.1	−3.3*	−6.6*	–
TimesFM 3	daily	1.274	1.372	1.543	+24.7*	+15.5*	+9.3*	–
TiRex 2	monthly	0.966	1.092	1.261	−5.5*	−8.1*	−10.7*	–
TiRex 2	flat daily path	0.992	1.136	1.320	−2.9*	−4.4*	−6.5*	–
TiRex 2	daily	1.298	1.393	1.567	+27.0*	+17.3*	+11.0*	–
Toto 2	monthly	1.036	1.171	1.350	+1.3	−1.4	−4.4*	–
Toto 2	flat daily path	1.014	1.141	1.310	−0.8	−4.0*	−7.2*	–
Toto 2	daily	1.299	1.412	1.606	+27.1*	+18.8*	+13.8*	–
Toto 2 (2.5B)	monthly	1.039	1.167	1.359	+1.6	−1.8	−3.7*	–
Toto 2 (2.5B)	flat daily path	1.003	1.120	1.294	−1.9	−5.7*	−8.3*	–
Toto 2 (2.5B)	daily	1.307	1.416	1.606	+27.9*	+19.2*	+13.8*	–
TabPFN-TS-3	monthly	0.993	1.132	1.307	−2.9*	−4.7*	−7.4*	0.675
TabPFN-TS-3	flat daily path	1.352	1.375	1.435	+32.3*	+15.7*	+1.7	0.994
TabPFN-TS-3	daily	1.315	1.404	1.579	+28.7*	+18.2*	+11.9*	0.865
N-HiTS (library defaults)	monthly	1.239	1.377	1.489	+21.2*	+15.9*	+5.5*	1.557
N-HiTS (matched)	monthly	1.349	1.635	1.939	+32.0*	+37.6*	+37.4*	1.091
GRU, 90-token, 4 layers	flat daily path	1.022	1.188	1.411	–	–	–	0.677
GRU, 90-token	flat daily path	1.068	1.220	1.403	+4.5*	+2.7*	−0.6	0.706
GRU, 30-token	flat daily path	1.047	1.197	1.389	+2.5*	+0.8	−1.6*	0.708
GRU, monthly	monthly	1.372	2.130	2.838	+34.2*	+79.2*	+101.1*	0.884

B FOUNDATION MODELS AND N-HITS

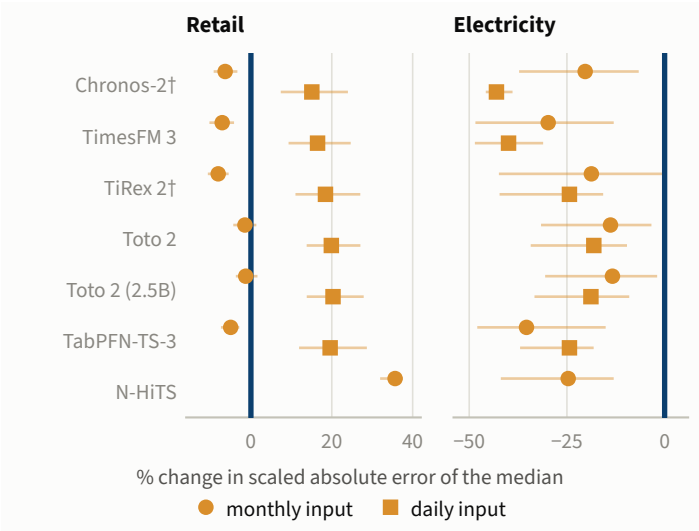

Figure 6: Percentage change in the scaled absolute error of the median against the 90-token GRU, four-layer on retail, on the test set; left of the zero line is better than the GRU. Circles read the monthly averages and squares the daily series; markers show the mean across horizons and lines the range. A dagger marks reported pretraining overlap with the evaluation dataset. Table 9 gives every retail value and Table 5 the electricity levels on the monthly input.

Six zero-shot models from five families forecast the same test episodes as the GRU: TimesFM 3 (Jain & Sen, 2026), Chronos-2 (Ansari et al., 2025), Toto 2 at 313M and 2.5B parameters (Khwaja et al., 2026), TiRex 2 (Podest et al., 2026) and TabPFN-TS-3 (Hoo et al., 2025; Grinsztajn et al., 2026), which conditions the TabPFN-3 time-series checkpoint on each episode's calendar features in context, without gradient updates. Moirai 2.0, Sundial and PatchTST-FM are not compared.

Unlike these models, N-HiTS (Challu et al., 2023) is trained from scratch per dataset on one series per entity, with gradient windows ending at the GRU's last training target, early stopping on the GRU's validation months, month-of-year sine and cosine as future exogenous inputs, and the same quantile loss, forecast cap and step budget as the GRU; entities without a GRU training episode are excluded from the fit. The tables also report N-HiTS (library defaults), with no calendar inputs and early stopping on each training episode's own label months, so that its training targets end three or six months before the GRU's. Its retail 97.5% quantile sits at the ten-times-context cap in 79 to 89% of test forecasts, against 28 to 48% for the matched run; Figure 6 and Table 9 give both runs' errors.

Each foundation model is evaluated separately with three inputs, the monthly averages that the GRU reads, the 90-token flat path, and the daily series, which the monthly-input models do not receive. In all three the target is the monthly average; flat and daily forecasts are averaged within each month at each quantile level, which preserves quantile ordering but does not generally yield quantiles of the monthly average. The flat- and daily-input comparisons therefore reflect both the input representation and the forecast-aggregation procedure.

TimesFM 3, Toto 2 and TiRex 2, whose released inference interfaces return nine deciles, cannot express the outer levels of our grid, so the comparison statistic is the absolute error of the median divided by the same δ and averaged as CRPS is. CRPS is reported for Chronos-2, whose grid brackets ours, TabPFN-TS-3, which reads any level off its predictive distribution, and N-HiTS.

The listed training corpora of Chronos-2 and of the TiRex 2 checkpoint we use contain UCI electricity, the base Toto 2 checkpoints are pretrained without public data, TabPFN-TS-3 is pretrained on synthetic data only, and the TimesFM 3 model card lists GIFT-Eval Pretrain, Wikipedia pageviews, Google Trends and synthetic data. Online Retail II appears in no listed corpus. Overlap between pretraining corpora and legacy benchmarks is a known confound (Meyer et al., 2025), and benchmarks that exclude or flag pretraining overlap exist (Aksu et al., 2024; Shchur et al., 2025; Qiao et al., 2026).

With the monthly averages as input, four foundation models have lower scaled absolute error of the median than the best GRU on retail, the four-layer 90-token model: Chronos-2, TimesFM 3 and TiRex 2 by 3.3 to 10.7% across horizons 1 to 3 (Chronos-2 by 3.3% $[-5.2, -1.5]$ at horizon 1) and TabPFN-TS-3 by 2.9 to 7.4%, whereas the two Toto models are within 1.6% at horizon 1, within 1.8% at horizon 2 and 3.7 to 4.4% lower at horizon 3; intervals exclude zero in 14 of the 18 pairs. On electricity the differences at horizon 1 have intervals that include zero, the foundation models have significantly lower error from horizon 2 to 4 depending on the model, and at horizons 4 to 6 every model has 12 to 48% lower error.

Given the daily series, the same models have 7 to 29% higher error than the GRU on retail at every horizon and 14 to 49% lower error on electricity at horizons 4 to 6. Given the flat path, TimesFM 3, TiRex 2 and Toto 2 have 1 to 8% lower error than the GRU on retail, with intervals that include zero at horizon 1 for TimesFM 3 and both Toto 2 models, whereas TabPFN-TS-3 has 32.3% $[+29.1, +35.5]$ and 15.7% higher error at horizons 1 and 2. The matched N-HiTS has 1.61 to 1.68 times the GRU's CRPS on retail, 1.02 times at electricity horizon 1 and 0.70 to 0.89 times at electricity horizons 2 to 6, with significantly lower median error there; the library-default run had 1.9 to 2.3 times on retail and 1.2 to 2.1 times on electricity.

On the validation set the ordering was reversed. Given monthly inputs, the GRU had lower error than every foundation model on electricity at every horizon, by 6 to 17% at horizon 1, and on retail at horizon 1 by 13 to 28%. The GRU's configuration, token density and early-stopping step were chosen on that span and the foundation models' were not, so the reversal on the test set is consistent with selection on validation favoring the GRU, although the change of evaluation span may also contribute. To check the comparison beyond the median, rescoring every comparison with a quantile score on the three levels all models emit, 0.1, 0.5 and 0.9, changes the sign of the monthly-input contrast in 2 of the 54 zero-shot pairs, both Toto models at electricity horizon 1, and in none on retail. Both N-HiTS runs have a higher three-level score than the GRU on retail at every horizon.

C Optimizer study

The *one-cycle schedule* (Smith & Topin, 2019) with early stopping was selected in the decomposition study (Appendix F); the *WSD schedule* (Hu et al., 2024; Hägele et al., 2024), warmup-stable-decay at twice the peak learning rate with a plateau-triggered decay, was selected in the study reported here.

Table 10: Optimizer study on the validation set, averaged over three random seeds, against the one-cycle schedule; mean percentage change in CRPS over the nine dataset-horizon pairs of retail and electricity, and the number of pairs with intervals excluding zero in each direction. Rows: cosine to zero at 8,000 steps; warmup-stable-decay at peak learning rates 2, 1 and 4×10^{-4}; WSD at 2×10^{-4} with a trapezoid weighting of the pinball levels; one-cycle without weight decay on one-dimensional parameters, which every other row also omits. Bold rows are the schedule selected for the dense models (WSD at 4×10^{-4}, chosen on the 30-token model) and tested for the monthly model (WSD at 2×10^{-4}).

Model	Schedule	Mean %	Better	Worse	Pairs
30-token	Cosine	−3.3	0	0	9
	WSD, lr 2×10^{-4}	−0.4	0	0	9
	WSD, lr 1×10^{-4}	−1.9	0	0	9
	WSD, lr 4×10^{-4}	**−4.4**	**3**	**1**	**9**
	WSD, lr 2×10^{-4}, trapezoid-weighted loss	−3.7	5	0	9
	One-cycle, no bias decay	−1.1	0	0	9
monthly	Cosine	+3.4	1	1	9
	WSD, lr 2×10^{-4}	**−1.0**	**3**	**0**	**9**
	WSD, lr 1×10^{-4}	+10.5	0	3	9
	WSD, lr 4×10^{-4}	+10.4	0	8	9
	WSD, lr 2×10^{-4}, trapezoid-weighted loss	+2.4	5	3	9
	One-cycle, no bias decay	+0.2	0	1	9

The decomposition study selected a one-cycle schedule (Smith & Topin, 2019) at peak learning rate 2×10^{-4}, weight decay 0.01 and 20,000 steps from a 12-configuration grid over learning rate, weight decay and step budget at one seed, and training stopped early at 5,500 to 8,000 steps, before learning-rate annealing. In the optimizer study of Table 10, we evaluated learning-rate schedules, peak learning rates, weight decay on one-dimensional parameters, and quantile-loss weighting, averaged over three seeds on the validation set. The GRU family and the adapted architectures of Appendix K are trained with AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.95$) on batches of 128 episodes, with gradient-norm clipping at 1.0, weight decay on matrix parameters only, validation every 500 steps, and early stopping under the one-cycle schedule after eight validation checks without improvement; the WSD schedule decays on the same plateau signal.

For the 30-token model the WSD schedule at 4×10^{-4} lowers retail CRPS by 8.3% $[-16.5, -2.9]$, 15.1% and 20.0% at horizons 1 to 3 and has an interval that includes zero on electricity except at horizon 4, where it is 1.8% $[+0.1, +4.4]$ worse, and we kept it for the dense models. The trapezoid-weighted loss is significantly better on five pairs and worse on none (-3.7% mean), so it satisfies Equation 11, which the WSD schedule fails because of the electricity horizon-4 pair. The WSD schedule was kept for its larger mean improvement. For the monthly model the best schedule is WSD at 2×10^{-4} (retail -2.4 to -3.2%, all three significant; electricity interval includes zero), and 4×10^{-4} has 11 to 21% higher CRPS on electricity at every horizon. Each model kept its own best schedule in the density study. Over eight seeds neither retuned schedule met the selection criterion against the one-cycle schedule, so the monthly and 30-token models kept the one-cycle schedule for the test evaluation.

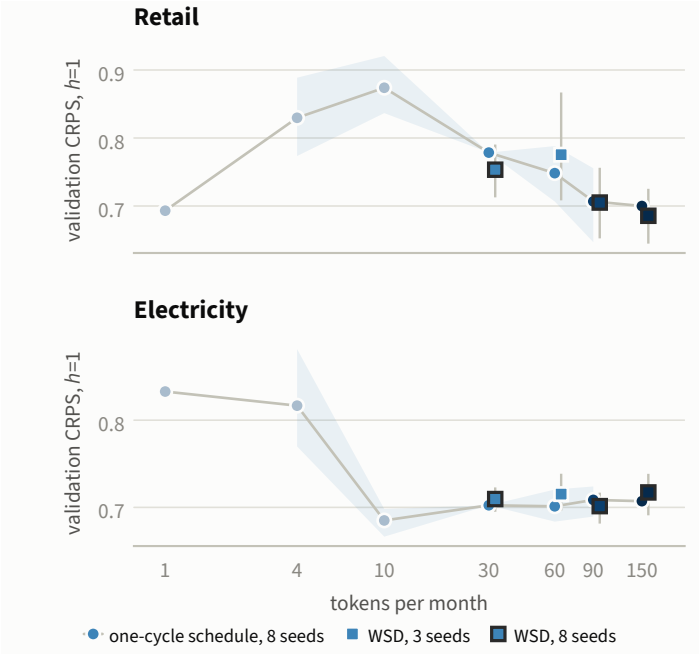

Figure 7: Validation CRPS at $h = 1$ against tokens per month on a logarithmic axis. Circles show the one-cycle schedule over eight seeds, squares the WSD schedule over three seeds, and outlined squares densities rerun over eight seeds; bars and the shaded band are 95% intervals of the contrast with the 30-token model.

On the validation span, increasing token density does not improve accuracy monotonically on either dataset (Figure 7). On retail it degrades from 1 to 10 tokens per month and improves above 10, and the monthly model has the lowest retail validation CRPS at horizon 1; this ranking reverses on the test set (Table 3). On electricity the monthly model has the highest CRPS and every $k \geq 10$ lies within 3% of the 30-token model. Neither $k = 60$ nor $k = 150$ improves on $k = 90$, the latter over eight seeds. Averaged over eight seeds against the 30-token model under the one-cycle schedule, the 90-token model under the WSD schedule changes CRPS by -9.4% $[-16.2, -2.9]$, -14.4% and -18.4% on retail and between -1.5 and $+1.7\%$ on electricity with no significant pair; against the

30-token model under the WSD schedule it remains lower on retail (-6.4% $[-11.8, -1.1]$, -9.0%, -10.6%) and has an interval that includes zero on electricity.

Under WSD, averaged over three seeds and against the 90-token model with the same configuration, $k = 150$ has lower CRPS at retail horizon 1 (-7.0% $[-9.1, -4.7]$) and electricity horizon 2 (-3.8% $[-8.4, -1.4]$) with no pair significantly worse, so it meets Equation 11 on three seeds. Over eight seeds the contrasts are -2.8% $[-5.9, +0.1]$, -4.3% and -3.2% on retail and between -2.5 and $+2.2\%$ on electricity, no interval excludes zero, and the criterion is not met. $k = 60$ improves nowhere.

The packed recurrence skips the leading padding and unobserved tokens, so the state starts at the first observed month rather than running over the padded prefix. We trained each of $k \in \{30, 60, 90, 150\}$ with both the default and the packed recurrence under each model's best schedule. Packed recurrence on the monthly model raises CRPS by $+16$, $+28$ and $+39\%$ on retail and by $+35$, $+16$ and $+8\%$ on electricity at horizons 1 to 3, every interval excluding zero. The packed forward pass matches the default one on full-length inputs, so the two differ only on padded inputs. Training histories contain at least six observed months and are left-padded to the input length, and because the training window precedes the validation span they never reach the full context. The packed recurrence sees only real tokens and must therefore extrapolate in length at evaluation, which may contribute to the degradation of the packed monthly model and of the packed 90-token model on the test set. That model improved on validation (retail -4.4% $[-7.3, -1.4]$, -7.5% and -6.6% against the default 90-token model, with an interval that includes zero on electricity), whereas on the test set it has higher CRPS at retail horizons 2 and 3 ($+4.5\%$ $[+3.4, +5.7]$ and $+12.3\%$ $[+10.6, +14.0]$), an interval that includes zero at retail horizon 1, and differences within $\pm 3\%$ on electricity.

D Input map, density and calendar ablations on the test set

Tables 11 and 12 report five ablation models trained after the test evaluation of Section 5.2 and scored once on the test set over five seeds with the series-cluster bootstrap of Section 4.5. The one-token-per-month model uses the dense input map with the calendar of the month's middle day and the one-cycle schedule, so it differs from the monthly model only in the input map and from the 30-token model only in density. The 30-token model under the WSD schedule holds the schedule fixed for the 90-versus-30 contrast; on retail the schedule-matched density gain is 0.7 to 3.4%, on electricity 6 to 14% at horizons 1 to 3 with the sign reversed at horizons 5 and 6. The calendar-mid model gives every token of a month the calendar of the month's middle day, so its 90 tokens are identical copies and the recurrence sees no within-month variation. On electricity it has higher CRPS at horizons 1 to 4 and lower at horizons 5 and 6 (-7.0% $[-9.3, -5.2]$ and -3.1% $[-6.9, -0.4]$).

The dropout-0 models remove the dropout of 0.1 between the GRU layers and in the head from both the 30-token and the 90-token model. Without dropout the 90-token model's advantage over the monthly model is unchanged (21, 39 and 48% on retail against 20, 40 and 49% in Table 3), and the retail 90-versus-30 contrast under Equation 6 is at most 1.1% at every horizon; the horizon-3 gain of Table 3 over the 30-token model therefore depends on dropout. On electricity the density gain without dropout is 14 to 31% at every horizon.

We trained further ablations after the test evaluation, five seeds each, and Tables 6 and 11 give the levels of the schedule ablations. The 90-token model under the one-cycle schedule has 15.3%, 36.6% and 46.5% lower CRPS than the monthly model on retail and 36.6%, 31.9% and 42.1% lower on electricity; against the WSD schedule it is 4.8 to 6.1% higher on retail and 4.9 to 9.7% lower on electricity, every interval excluding zero. The monthly model under its own WSD schedule lowers retail CRPS by 11.2%, 18.8% and 23.0% and raises electricity CRPS by 2.2 to 5.0%, and the 90-token model remains 10.1 to 34.1% lower on retail and 31.3 to 37.5% lower on electricity. The two calendar channels matter on different datasets, as zeroing the weekday channels of the 90-token model raises CRPS by 10.5%, 9.6% and 12.2% on retail and by 0.6% $[-5.7, +5.4]$, 6.6% and 5.3% on electricity, whereas zeroing the month-of-year channels changes retail CRPS by $+1.5$, -1.6 and $+3.4\%$ and raises electricity CRPS by 90.1%, 82.9% and 81.8%. The dense input map at one token per month under the WSD schedule at 2×10^{-4} or 4×10^{-4}, whichever is lower per pair, reaches 0.738, 0.973 and 1.222 on retail and 1.022, 1.196 and 1.183 on electricity, and the 90-token model is 4.4%, 14.8% and 21.6% lower on retail and 28.0%, 23.9% and 27.0% lower on electricity, every interval excluding zero.

An LSTM with the same width, depth and head in place of the GRU (Table 6) also improves over its monthly model with dense tokens. Its 90-token model has 4.9%, 3.7% and 2.3% lower CRPS than its monthly model on retail and 35.0%, 41.6% and 47.6% lower on electricity, with intervals excluding zero except at retail horizon 3. Against the GRU at the same density the LSTM is 8.4 to 20.2% worse on retail at 90 tokens and 3.3 to 5.9% worse at 30 tokens, whereas on electricity it is 13.0% worse at horizon 1 and 12.3% better at horizon 3 at 90 tokens; its monthly model is 9 to 38% better than the GRU's on retail and 7 to 22% worse on electricity.

The calendar-mid penalty on electricity comes from the constant weekday value rather than from the identical tokens, since giving every token the calendar of the month's first day costs 11.8 to 12.4% at 90 tokens whereas zeroing the weekday channels costs at most 7%. On retail the constant weekday is as informative as the varying one.

One route for the electricity calendar effect is weekday composition. Suppose the daily flow is $y_{m,d} = \ell_m + b_{w(d)} + \varepsilon_{m,d}$ with weekday effects b_w; then

$$\bar{y}_m = \ell_m + \sum_{w=1}^{7} \frac{n_{m,w}}{n_m} b_w + \bar{\varepsilon}_m, \tag{10}$$

where $n_{m,w}$ counts occurrences of weekday w. Weekday composition is a standard calendar predictor of monthly averages (Hyndman & Athanasopoulos, 2021, Section 7.4), with a long history in seasonal adjustment (Cleveland & Devlin, 1980; Bell & Hillmer, 1983), and it is available even when daily values are not. The calendar sequence also lets the GRU adjust its state differently at different calendar phases.

Table 11: Scaled CRPS of the ablations on the test set, including the one-token dense input map under the WSD schedule at two learning rates, and the input-map and density contrasts, with 95% series-cluster bootstrap intervals; a star marks an interval that excludes zero.

Dataset	h	1-token	1-token WSD 2×10^{-4}	1-token WSD 4×10^{-4}	30-token WSD	90-token	90-token cal.-mid	90-token no dropout	% 1-token − monthly	% 30-token − 1-token
Retail	1	0.740	0.738	0.746	0.710	0.706	0.707	0.700	−16.3 [−19.2, −13.6]*	−4.4 [−6.1, −2.7]*
	2	1.002	1.009	0.973	0.844	0.828	0.833	0.836	−26.8 [−30.5, −23.3]*	−16.0 [−18.6, −13.4]*
	3	1.325	1.276	1.222	0.991	0.957	0.976	0.991	−29.8 [−33.4, −26.3]*	−24.4 [−27.7, −21.2]*
Electricity	1	1.114	1.074	1.022	0.783	0.736	0.832	0.743	+6.4 [−2.5, +12.8]	−28.1 [−39.2, −13.2]*
	2	1.321	1.225	1.196	1.038	0.910	1.083	0.883	+3.9 [−6.8, +11.2]	−15.9 [−28.1, −0.4]*
	3	1.339	1.337	1.183	1.005	0.863	1.062	0.822	−0.9 [−9.9, +4.6]	−18.8 [−32.1, −2.1]*
	4	1.365	1.479	1.285	1.009	1.032	1.102	0.967	−3.0 [−13.9, +3.3]	−19.9 [−34.0, −2.6]*
	5	1.408	1.491	1.302	0.965	1.051	0.978	1.028	+1.7 [−8.7, +7.6]	−27.6 [−37.3, −16.7]*
	6	1.705	1.793	1.627	1.150	1.280	1.239	1.205	+4.5 [−4.7, +9.7]	−29.3 [−34.7, −22.9]*

Table 12: Percentage contrasts of the ablations of Table 11: density under the matched WSD schedule, calendar-mid and dropout-0 against the 90-token model, density without dropout, and the 90-token model against the lower of the two tuned one-token models; 95% series-cluster intervals, a star marks an interval that excludes zero.

Dataset	h	% 90-token − 30-token WSD	% cal.-mid − 90-token	% no dropout − 90-token	% 90-token − 30-token, no dropout	% 90-token − best 1-token
Retail	1	−0.7 [−1.4, +0.1]	+0.2 [−0.8, +1.1]	−0.8 [−1.6, −0.1]*	−1.1 [−1.8, −0.2]*	−4.4 [−6.6, −2.1]*
	2	−1.8 [−2.5, −1.1]*	+0.5 [−0.4, +1.3]	+0.9 [+0.2, +1.7]*	−0.7 [−1.7, +0.3]	−14.8 [−17.3, −12.4]*
	3	−3.4 [−4.1, −2.7]*	+1.9 [+0.9, +2.9]*	+3.5 [+2.7, +4.4]*	+0.3 [−0.9, +1.7]	−21.6 [−24.5, −18.8]*
Electricity	1	−6.0 [−9.0, −3.5]*	+13.2 [+10.6, +15.7]*	+1.0 [−4.3, +5.9]	−14.0 [−30.3, −1.7]*	−28.0 [−46.3, −12.2]*
	2	−12.3 [−14.8, −10.2]*	+19.0 [+16.4, +21.7]*	−3.0 [−7.8, +1.2]	−21.7 [−34.0, −11.7]*	−23.9 [−36.6, −11.9]*
	3	−14.1 [−18.3, −10.4]*	+23.0 [+20.4, +25.7]*	−4.8 [−9.9, +1.4]	−31.4 [−47.8, −18.3]*	−27.0 [−40.2, −14.7]*
	4	+2.3 [−3.5, +7.1]	+6.7 [+4.5, +8.8]*	−6.3 [−9.9, −1.8]*	−24.5 [−45.1, −8.6]*	−19.7 [−34.1, −8.1]*
	5	+8.9 [+4.4, +12.9]*	−7.0 [−9.3, −5.2]*	−2.2 [−7.1, +4.3]	−17.4 [−36.6, −1.9]*	−19.3 [−31.0, −10.2]*
	6	+11.3 [+7.8, +14.4]*	−3.1 [−6.9, −0.4]*	−5.9 [−8.7, −2.5]*	−14.9 [−30.8, −2.9]*	−21.4 [−27.9, −15.6]*

E GATE MEASUREMENT

We unroll the trained GRU step by step over the test episodes and record the retention gate at every token for the five seeds of each model. Table 13 gives, per layer, the mean gate, the product of gates over a month, the share of units whose mean gate exceeds 0.9, and the relative displacement of the state over a month.

Table 13: Learned retention gates of the monthly, 30-token and 90-token models on the test set, mean of five seeds.

Dataset	Model	Layer	Mean gate	Product over a month	Units with gate > 0.9	State displacement
Retail	monthly	1	0.43	0.43	0%	0.44
		2	0.35	0.35	1%	0.40
	30 tokens	1	0.52	0.00	0%	0.55
		2	0.59	0.04	11%	0.23
	90 tokens	1	0.50	0.00	0%	0.57
		2	0.58	0.02	10%	0.21
Electricity	monthly	1	0.39	0.39	0%	0.34
		2	0.28	0.28	0%	0.31
	30 tokens	1	0.45	0.00	0%	0.75
		2	0.50	0.02	3%	0.41
	90 tokens	1	0.50	0.00	0%	0.93
		2	0.53	0.01	1%	0.56

F THE DECOMPOSITION STUDY

The decomposition study compared the monthly model, a daily-input reference of the same architecture that reads the daily series, the 30-token flat model, a parameter-matched two-model monthly ensemble, and priors (Appendix G), averaged over five seeds under identical normalization and schedule. The tables report the entity-mean score without the divisor of Equation 6, the metric of that study. Table 14 gives test-set CRPS. The aggregation gap, daily reference minus monthly model, is significant on retail and electricity at every horizon. The representation channel, flat model minus monthly model, recovers 75% of the gap on retail and 83% on electricity at horizon 1 in that metric (Figure 3 reports the corresponding recovery measure using the scaled test score), and the flat model has lower CRPS than the matched ensemble on both datasets.

Table 14: Test-set CRPS of the decomposition study at $h = 1$ to H, entity mean without the divisor, averaged over five seeds, one-cycle schedule. Electricity is in units of 10^3. Columns: best classical baseline on validation among naive, seasonal naive, ETS, theta, ARIMA and Croston-SBA (Hyndman & Khandakar, 2008; Assimakopoulos & Nikolopoulos, 2000; Syntetos & Boylan, 2005); monthly model; daily-input reference; 30-token flat model; 30-token monotone-cubic model; 30-token model with a donor weekday profile; 30-token model with a learned cross-domain inverse; two-model monthly ensemble at matched parameters. Bold marks the lowest CRPS in a row.

Dataset	h	Best classical	Monthly	Daily-input reference U	30-token flat	30-token cubic	Donor weekday	Learned inverse	Two-model ensemble
Retail	1	1.974	1.940	**1.780**	1.820	1.834	1.818	1.883	1.878
	2	2.259	2.859	**2.092**	2.213	2.221	2.210	2.370	2.469
	3	2.688	3.950	**2.502**	2.723	2.807	2.718	2.919	3.213
Electricity	1	1.059	1.174	0.605	0.703	0.733	0.683	**0.582**	1.326
($\times 10^3$)	2	1.207	1.713	**0.829**	1.058	1.116	1.221	1.054	1.818
	3	0.897	2.012	**0.793**	1.069	1.265	1.190	1.209	2.111
	4	0.914	1.825	**0.856**	1.011	1.170	1.006	1.183	1.882
	5	1.281	1.487	0.872	0.883	0.996	**0.841**	0.918	1.500
	6	1.645	1.529	0.972	0.952	1.066	**0.934**	1.000	1.503

G RECONSTRUCTION SHAPES AND PRIORS

Four smooth coherent shapes fitted to the monthly series itself were tested in the decomposition study under the one-cycle schedule: a monotone cubic interpolant, a low-order Fourier fit, a Gaussian-process posterior mean with weekly and annual kernels, and a trend-seasonal-irregular decomposition, each renormalized per month. Averaged over horizons 1 to 3, every alternative has higher CRPS than the flat path on both datasets (monotone cubic by 1.4 to 9.3%, Fourier by 1.5 to

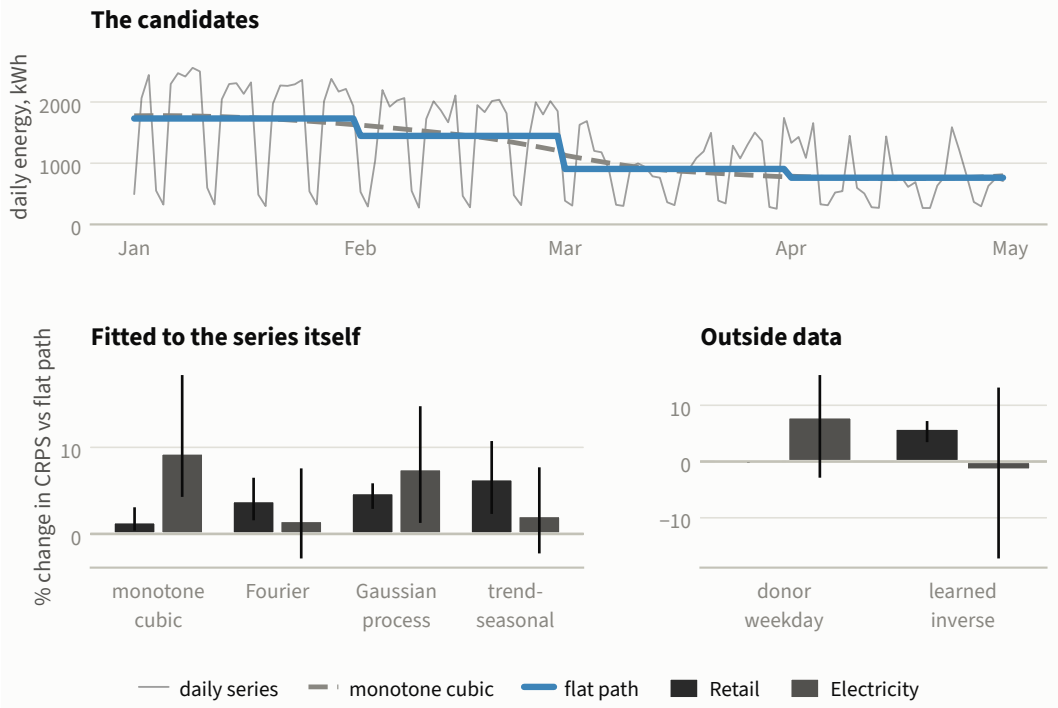

Figure 8: Top, competing coherent reconstructions on a real window spanning a level change. Bottom, every alternative against the flat path on the test set of the decomposition study; bars are the mean over horizons 1 to 3 and lines the range.

3.8%, Gaussian process by 4.8 to 7.5%, trend-seasonal by 2.1 to 6.4%; Figure 8). The ranking holds at every retail horizon; on electricity, Fourier at horizons 2 and 3 and trend-seasonal at horizon 3 are up to 2.8% below flat, with intervals that include zero. On retail, the same ranking holds under trimmed mean, median ratio, geometric mean, win rate, and log-space CRPS, with per-entity win rates of 0.29 to 0.44 against flat. On electricity, median per-entity ratios favor the smooth reconstructions (-1.0 to -4.0%), whereas the entity-mean score favors flat.

Three further variants over eight validation seeds, adaptive shrinkage toward a Fourier fit, a Denton path and an edge-blended path, fail the selection criterion against the flat path, as does Dirichlet resampling of within-month shares as a training augmentation. On retail at horizon 1, shrinkage is 4.2%, Denton 1.6% and Dirichlet 2.8% worse.

The benefit of external priors varies across datasets. In the decomposition study a donor weekday profile is -2.9% against flat on electricity at horizon 1; a learned inverse trained on other domains has the lowest CRPS on electricity at horizon 1 but higher CRPS than flat on retail, and does not outperform uniform shares on its within-month allocation objective.

H Normalization and sum-target robustness

We standardize with statistics from the monthly context rather than the input window for all models, so that normalization is identical across them. Relative to arcsinh and mean scaling, $\log(1 + x)$ standardization had the lowest score, or a difference whose interval included zero, in five of six dataset-horizon pairs. Working with averages rather than sums (Section 3) does not change the decomposition study, as rescoring every contrast on monthly sums preserves the sign and significance of all 36 contrast-horizon pairs.

I TOKEN GRID AND INPUT CHANNELS

For month m with n_m days and density k, block $j \in \{1, \dots, k\}$ of the grid in Section 4.2 covers days $\lfloor (j-1)\, n_m/k \rfloor + 1$ through $\max\left(\lfloor (j-1)\, n_m/k \rfloor + 1,\ \lceil j\, n_m/k \rceil\right)$, so adjacent blocks may share a day and days repeat when $k > n_m$; a block is observed only if all its days are. The 30-token model uses the calendar-day grid, one block per day, in place of this formula.

Every model reads the same per-episode scaling statistics, μ and σ of $\log(1+\bar{y}_m)$ over the observed monthly context, entered as the channels $\mu/10$ and $\log(1+\sigma)$. The dense tokens of Section 4.2, including the calendar-day grid of the 30-token model and the daily-input reference, carry the eight channels listed there. The monthly model reads one token per month with seven channels: the standardized value, the mask, $\log n_m - \log 30$ for the length of the month, the month-of-year pair, and the two scaling channels. The monthly model therefore sees month length where the dense models see the weekday, and the two input maps are not the $k = 1$ and $k > 1$ cases of one operator. Apart from their inputs, the models within each matched comparison share the network, forecast head, loss, inversion guard, and stated optimizer configuration. The inversion guard clips the standardized quantiles to $[-8, 8]$ before inversion and the inverted forecast to $[0,\ 10 \max(c_{\max}, 1)]$, where $c_{\max}$ is the largest observed monthly average of the context.

J ENCODER BASELINES AND GRU DEPTH

PatchTST used per-channel reversible instance normalization (Kim et al., 2022), patching scaled with k, a three-layer encoder and a flatten head, with a grid over patch length, learning rate and width, DLinear used a moving-average decomposition with a grid over kernel, learning rate and per-channel weights, and iTransformer was trained under the one-cycle schedule. On the validation set the two PatchTST configurations, patch 16 with width 64 and patch 32 with width 128, have higher CRPS than the GRU at horizon 1 by $+7.1\%$ $[-0.1, +12.8]$ to $+46.4\%$ $[+24.1, +76.3]$ on retail and by $+24.8\%$ $[+12.7, +40.0]$ to $+33.9\%$ $[+21.8, +47.7]$ on electricity. The disadvantage has an interval that includes zero on electricity by horizon 3 or 4, the wider model has lower CRPS than the GRU at electricity horizon 6 (-4.5% $[-8.9, -1.6]$ at 90 tokens and -4.1% $[-7.8, -1.2]$ at 30 tokens), and on retail the narrower model at 30 tokens is significantly lower at horizons 2 and 3 (-5.9%). Within the GRU family, a width-64 GRU at 90 tokens has intervals that include zero against the width-128 GRU (retail $+3.6\%$ $[-9.4, +26.9]$ at horizon 1, electricity within $\pm 1.2\%$). In a preliminary comparison at horizon 1 under the entity mean without the divisor, DLinear had 148 to 189% higher validation CRPS than the GRU on retail, but it has 7k parameters against the GRU's 174k, so this comparison does not establish a general disadvantage of linear models.

The main experiments use one GRU family with widths 64 and 128 and two stacked layers. A depth and width study run after the test evaluation (Table 15) compares three and four layers, width 256, and a third layer at 30 tokens and on the monthly model, each against the two-layer model at the same density under the WSD schedule, over three validation seeds. Only the four-layer 90-token model satisfies Equation 11, and it does so again over eight seeds, with retail CRPS 5.7% $[-9.5, -1.2]$ and 8.4% lower at horizons 1 and 2, and on electricity intervals that include zero at horizons 1 to 5 and 5.8% lower CRPS at horizon 6. Three layers give no gain at horizon 1 on either dataset, width 256 has higher CRPS on both datasets with the largest validation loss of any variant, a third layer at 30 tokens changes nothing on electricity, and three or four layers on the monthly model raise its electricity CRPS by 10 to 36% at horizons 1 and 2, so depth helps only with dense tokens. On the test set (Table 16) the four-layer model has 4.1%, 4.4% and 1.7% lower CRPS than the two-layer model on retail, with lower error on 66% of products at horizon 1, and 15.8 to 15.9% lower on electricity at horizons 4 to 6, where it is lower on about 80% of clients and in 14 of the 15 seed-horizon pairs. At electricity horizons 1 to 3 its mean contrast has an interval that includes zero, but the median client is 7 to 16% worse and only 26 to 36% of clients improve, so the mean there rests on a few large clients. It is the best retail configuration in Tables 4 and 5, at 369k parameters and 4.2 minutes of training.

Table 15: Depth and width study on the validation set: percentage change in scaled CRPS against the two-layer model at the same density, three seeds unless stated, two-way bootstrap; * excludes zero.

	Retail			Electricity					
Model	h=1	h=2	h=3	h=1	h=2	h=3	h=4	h=5	h=6
30-token, 3 layers	+3.5	−2.6	−3.2*	+0.5	+1.2	+1.1	+1.5	+2.0	−1.7
90-token, 3 layers	+0.4	−9.9*	−10.1	+5.3	+3.2	+3.7*	+2.1	+1.9	−2.6
90-token, 3 layers, lr 2×10^{-4}	−2.5	−2.5	+0.5	+0.9	+0.1	−0.8	−2.3	−2.4	−5.4*
90-token, 4 layers	−9.2*	−15.0*	−15.0*	+2.2	+1.8	+0.3	−1.3	−1.6	−5.1*
90-token, 4 layers, eight seeds	−5.7*	−8.4*	−7.1	+2.3	+1.7	+0.8	−0.4	−1.3	−5.8*
90-token, width 256	+7.0*	−8.4	−9.1	+2.6	+5.2*	+3.7	+2.5	+4.0*	+4.6
monthly, 3 layers	+11.4*	+16.4*	+22.1*	+32.0*	+9.7*	+3.0*	+0.7	+0.1	+0.1
monthly, 4 layers, eight seeds	−3.8	−4.3	−2.8	+36.3*	+10.2*	+2.2	−0.5	−1.1	−0.2

Table 16: The four-layer 90-token model on the test set: scaled CRPS, mean of five seeds, and percentage change against the two-layer 90-token, 30-token, monthly and daily-input models with series-cluster intervals; * excludes zero.

	Retail			Electricity					
Model	h=1	h=2	h=3	h=1	h=2	h=3	h=4	h=5	h=6
CRPS, 90-token, 4 layers	0.677	0.792	0.941	0.726	0.876	0.839	0.868	0.885	1.078
% vs two-layer 90-token	−4.1*	−4.4*	−1.7*	−1.3	−3.7	−2.8	−15.9*	−15.8*	−15.8*
% vs 30-token	−4.4*	−5.9*	−6.0*	−9.4	−21.1*	−22.9*	−20.7*	−13.2*	−10.6*
% vs monthly	−23.5*	−42.2*	−50.2*	−30.6*	−31.1*	−37.9*	−38.3*	−36.1*	−33.9*
% vs daily-input reference U	−14.5*	−15.5*	−14.9*	−0.9	−0.6	−6.0	−12.5	−14.4*	−6.7

K ADAPTED ARCHITECTURES

Each adapted architecture uses the eight input channels, the seven-level monotone head, the horizons and the WSD schedule, and has at most six times the GRU's 174k parameters. Patch length is 3 at $k = 30$ and 9 at $k = 90$, so at $k = 90$ each month is 10 patches and every month boundary is a patch boundary. At $k = 30$ a 3-token patch can straddle a month boundary, so the causal patch decoder supervises the last patch that ends at or before each month-end, which never includes a token of the following month, whose flat value is the auxiliary horizon-one target. *Patched GRU* embeds nonoverlapping patches with a two-layer residual block before the recurrence, so the recurrence takes 10 steps per month at every k, with 302k parameters at $k = 30$ and 332k at $k = 90$. *Patched sLSTM* replaces the GRU with two sLSTM blocks (Beck et al., 2024) with four heads, no convolution and a SwiGLU of factor 2.667. *Causal patch decoder* is a three-layer pre-norm transformer over patches with rotary positions, causal attention and a bias-free SwiGLU of width 344, supervised at every month-end patch. *Causal supervision* applies the head at every month-end state and averages the pinball loss over valid month and horizon pairs, as TimesFM (Das et al., 2024) averages over patch positions and TiRex (Auer et al., 2025) over output tokens, and inference reads the last state only. *Future covariates* follow TiRex 2 (Podest et al., 2026), with a forward target stream and a bidirectional calendar stream from the same weight-tied GRU, mixed by one attention layer in which target queries read covariate keys but not the reverse, so the target at position l depends on target inputs at positions $\leq l$ only. *Contiguous patch masking* zeroes the value and mask channels over blocks of one tenth of a month during training, drawn as in TiRex (Auer et al., 2025, Appendix D.4) ($c_{\max} = 5$, $p_{\max} = 0.25$), and the *residual block layout* wraps the GRU in pre-RMSNorm residual blocks with a bias-free SwiGLU feed-forward of width 384. The scaler uses whole-context statistics, so the auxiliary targets of the causal models are mildly optimistic, whereas the scored position uses the same scaler as the baseline.

Table 17 and Figure 9 compare each variant with a GRU trained at the same density and configuration over three seeds. Under Equation 6 no variant satisfies Equation 11 against the 90-token GRU, and none improves on it at the first horizon. The patched GRU, whose recurrence takes 10 steps per month, has 67.9% and 40.7% higher retail CRPS at horizon 1 than the GRU at 30 and 90 tokens,

Table 17: Architecture study on the validation set, averaged over three random seeds; percentage change in CRPS per horizon against a GRU trained under identical conditions at the same density k (left) and against the 90-token GRU (right), * where the two-way interval excludes zero. A dash marks a pair that was not run.

			vs the GRU at the same k			vs the 90-token GRU		
Dataset	Variant	k	h=1	h=2	h=3	h=1	h=2	h=3
Retail	Future-covariate GRU	30	+3.5	−10.6	−25.9*	+10.5*	−2.5	−16.4*
		90	+16.6*	+1.6	−11.9	+16.6*	+1.6	−11.9
	Causal-supervision GRU	30	−2.7	−18.9*	−28.9*	+3.9*	−11.6*	−19.9*
		90	+4.4*	−11.7*	−20.0*	+4.4*	−11.7*	−20.0*
	Patched sLSTM	30	+5.7	−9.6	−19.4*	–	–	–
		90	+6.6*	−5.0	−8.9*	+6.6*	−5.0	−8.9*
	Residual-block GRU	30	+2.5	−5.1	−8.7	–	–	–
		90	+9.2*	+14.7*	+13.1*	–	–	–
	Contiguous patch masking	30	+7.1*	+10.2*	+13.2*	–	–	–
		90	+7.6*	+14.8*	+20.4*	–	–	–
	Causal patch decoder, lr 1×10^{-4}	30	+51.6*	+2.6	−13.6	+61.8*	+11.9	−2.6
		90	+52.0*	+14.3*	−11.1	–	–	–
	Patched GRU	30	+67.9*	+31.8	+6.2	–	–	–
		90	+40.7*	+25.6*	+18.8*	–	–	–
	Causal patch decoder, lr 4×10^{-4}	30	+43.1*	−6.8	−17.3*	+52.7*	+1.6	−6.7
		90	+41.3*	+2.6	−13.8	–	–	–
Electricity	Future-covariate GRU	30	+0.9	−2.5*	−1.6	+0.9	−0.9	−1.6
		90	+0.5	−1.7	−2.2	+0.5	−1.7	−2.2
	Causal-supervision GRU	30	+6.0	+5.3*	+4.2*	+6.0	+6.9*	+4.2*
		90	+4.1	+4.9	+5.0*	+4.1	+4.9	+5.0*
	Patched sLSTM	30	+9.7*	+2.3	+3.3	–	–	–
		90	+9.3	+1.8	+1.0	+9.3	+1.8	+1.0
	Residual-block GRU	30	+7.4*	+4.7	+6.2*	–	–	–
		90	+0.8	−0.9	−0.8	–	–	–
	Contiguous patch masking	30	+4.0*	+0.9	+0.1	–	–	–
		90	+8.2*	+5.6*	+0.4	–	–	–
	Causal patch decoder, lr 1×10^{-4}	30	+31.1*	+19.4*	+8.2*	+31.1*	+21.3	+8.2*
		90	+17.2*	+7.3	+5.0	–	–	–
	Patched GRU	30	+8.5*	+2.8	+3.4*	–	–	–
		90	+11.6*	+5.9	+4.2*	–	–	–
	Causal patch decoder, lr 4×10^{-4}	30	+56.6*	+29.2*	+12.4*	+56.6*	+31.2*	+12.5*
		90	+20.3*	+13.1*	+9.9*	–	–	–

and the causal patch decoder has 17 to 57% higher CRPS at horizon 1 on both datasets. Contiguous patch masking and the residual block layout have higher CRPS at every retail horizon at 90 tokens. Four variants gain at later horizons. Causal supervision at every month end is $+4.4\%$ $[+2.5, +6.2]$ at retail horizon 1 and -11.7% and -20.0% at horizons 2 and 3 at 90 tokens, the future-covariate design is $+16.6\%$ $[+11.6, +22.6]$ at retail horizon 1 and -6.5% $[-12.3, -0.4]$ at electricity horizon 6, the causal patch decoder at its lower learning rate and 90 tokens is -5.8% $[-11.7, -1.1]$ at electricity horizon 6, and the patched sLSTM at 90 tokens is -8.9% $[-15.1, -2.1]$ at retail horizon 3 and $+6.6\%$ $[+4.1, +9.1]$ at horizon 1. Averaged over eight seeds on the validation set, the patched sLSTM at 90 tokens per month changes CRPS by +1.9% on average over the nine dataset-horizon pairs relative to the 90-token GRU. No pair improves significantly, and two worsen significantly. The model therefore does not meet the selection criterion in Equation 11. On the test set the patched sLSTM has 12.2%, 35.7% and 51.5% higher CRPS than the 90-token GRU on retail and intervals that include zero on electricity (Table 4), at 40 to 50 minutes of training per model against three for the GRU.

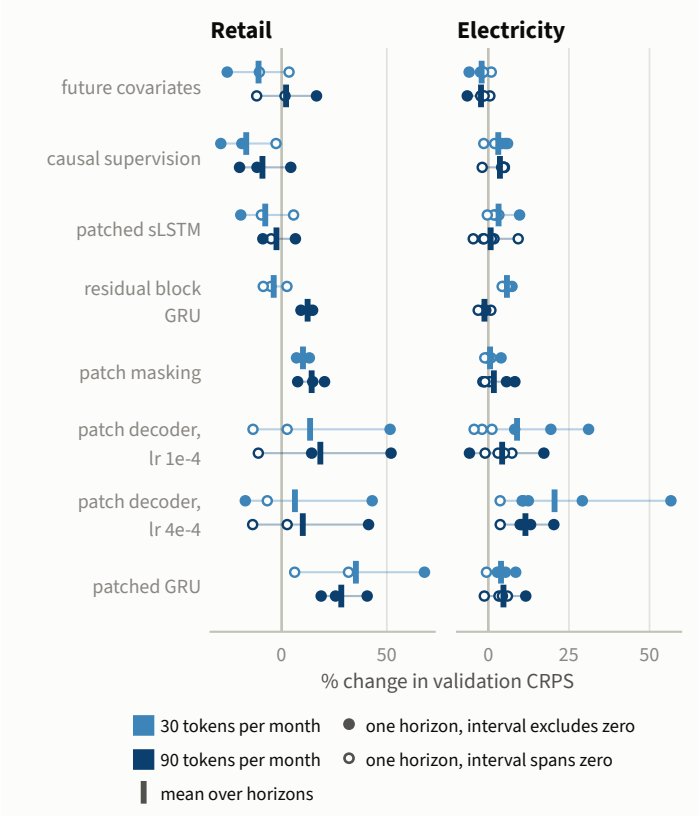

Figure 9: Architecture study on the validation set, averaged over three seeds, percentage change in CRPS against the GRU at the same density, for retail (left) and electricity (right). Color indicates token density. Thick bars are the mean over horizons; circles are single horizons, filled where the interval excludes zero. Thin lines show the range across horizons, not confidence intervals.

L BOOTSTRAPS, SELECTION CRITERION AND SEED-INFERENCE CAVEATS

Let $\mathcal{P}$ be the set of dataset-horizon pairs and $[l_p, u_p]$ the interval of Δ_p. The selection criterion, fixed before the density study, retains a candidate a over a baseline b when

$$\max_{p\in\mathcal{P}} l_p \le 0 \quad \wedge \quad \frac{1}{|\mathcal{P}|}\sum_{p\in\mathcal{P}} \Delta_p < 0 \quad \wedge \quad \min_{p\in\mathcal{P}} u_p < 0, \tag{11}$$

that is, a candidate is retained if it improves the average score, significantly improves at least one dataset-horizon pair, and is not significantly worse on any pair. Both bootstraps of Section 4.5 keep all origins of an entity together, draw 9,999 replicates and report a percentile interval; the series-cluster bootstrap averages the seed by entity matrix of differences over seeds and resamples entities with replacement, and the two-way bootstrap resamples seed rows with replacement, averages, then resamples entity columns.

The intervals of the decomposition study resample entities with seeds averaged within each dataset-horizon pair, so training-seed variance, which on retail exceeds the entity variance, does not enter them. Of 144 combinations of contrast, horizon and metric in that study, 103 are significant under the original bootstrap and 70 when both seeds and entities are resampled, and one shape contrast, the Gaussian-process path against flat, rests on a single seed of five. On retail, only the aggregation-gap contrast remains significant at every horizon. The main result holds under this stricter bootstrap, as resampling seeds as well as entities leaves the contrast between the 90-token and the monthly model significant at every horizon on both datasets, with retail intervals of $[-35, -6]$, $[-54, -21]$ and $[-63, -32]\%$ and electricity intervals of $[-44, -14]$, $[-40, -15]$ and $[-48, -23]\%$. Subse-

quent comparisons therefore use three seeds for screening, eight for confirmation, and the two-way bootstrap in every selection.

M COMPUTE

All runs used one NVIDIA GeForce RTX 5080 laptop GPU. Training compute counts the forward FLOPs of each model over its training steps and batches, with the backward pass taken as twice the forward pass; the LSTM and sLSTM minutes reflect the memory limit of the card. Table 19 reports total recorded wall-clock time for the trained models; the later PatchTST and width-matched GRU runs validated every 100 steps (Appendix J), the N-HiTS runs (both configurations) and the zero-shot inference of Section 5.4 are not included.

Table 18: Training cost of the trained models, mean of five seeds: parameters, forward GFLOP per episode, training compute in PFLOP, minutes on one RTX 5080 laptop GPU, and test CRPS at horizon 1.

Dataset	Model	Parameters	Forward GFLOP per episode	Training PFLOP	Minutes	Test CRPS, h=1
Retail	GRU monthly	171k	0.004	0.01	0.4	0.884
	GRU 30	171k	0.112	0.24	1.1	0.708
	GRU 90	171k	0.325	0.66	2.4	0.706
	GRU 90, one-cycle	171k	0.325	0.80	3.1	0.749
	GRU 90, 4 layers	369k	0.750	1.50	4.2	0.677
	GRU, daily-input reference U	171k	0.112	0.25	1.4	0.792
	LSTM 90	222k	0.434	1.43	78.6	0.765
	sLSTM 90	494k	0.113	0.26	39.7	0.791
	PatchTST 90	375k	0.005	0.02	3.9	0.802
	iTransformer 90	884k	0.003	0.00	0.9	0.739
	DLinear 90	7k	0.000	0.00	0.4	0.975
Electricity	GRU monthly	174k	0.005	0.01	0.3	1.047
	GRU 30	174k	0.168	0.39	1.7	0.802
	GRU 90	174k	0.488	1.05	3.8	0.736
	GRU 90, one-cycle	174k	0.488	1.22	4.3	0.664
	GRU 90, 4 layers	372k	1.125	2.38	6.5	0.726
	GRU, daily-input reference U	174k	0.168	0.41	1.8	0.733
	LSTM 90	225k	0.650	1.47	5.8	0.831
	sLSTM 90	497k	0.170	0.31	49.9	0.791
	PatchTST 90	468k	0.008	0.02	7.4	0.843
	iTransformer 90	955k	0.004	0.01	1.0	0.740
	DLinear 90	22k	0.000	0.00	0.4	3.980

Table 19: GPU hours per experiment.

Experiment	Runs	GPU h
Decomposition study, priors and comparisons (Appendices F, G)	285	10.8
Normalization ablations (Appendix H)	36	0.5
Optimizer, density, packing and confirmation studies (Appendix C)	568	23.0
Test evaluation (Section 5.2)	75	5.2
Daily-input references U and U_w (Section 5.3)	30	1.3
Input-map, schedule, calendar and dropout ablations (Appendix D)	79	5.8
Encoders, PatchTST and DLinear (Appendix J)	156	10.2
Architecture study and the eight-seed sLSTM confirmation (Appendix K)	140	23.6
Other exploratory runs	311	10.3
Schedule, weekday and calendar ablations (Appendix D)	80	2.8
Encoders and sLSTM on the test set (Appendix A)	60	9.4
LSTM ladder (Appendix D)	30	7.5
Depth and width study (Appendix J)	62	4.5
Total	1,912	114.9