LATENCY-AWARE TIME-SERIES FORECASTING WITH KNOWLEDGE DISTILLATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Time-series forecasters are scored offline as if inference took no time. On a stream sampled every Δt, the forecast of a call that takes $\mathcal{T}$ is first used $d = \lceil \mathcal{T}/\Delta t \rceil$ ticks later, when its first d steps describe instants already observed. We formalize latency-aware forecasting: a configuration issues its own stale prefix plus the $\mathcal{H}$ points the consumer needs and is scored on the remaining *usable window*. This deployment error factorizes into a device-independent error curve and a device-dependent prefix, so candidates are trained once and re-scored per device. Its optimum can be interior, since larger models lower per-step error but push the usable window further out. On a 1 kHz ECG stream and one x86 core, the usable-window error selects an MLP student a quarter the size of the one that nominal-window accuracy picks. For a selected configuration, reweighting the loss toward the usable window adds no signal to per-step read-outs and little to shared encoders, so TRIAD changes the target instead. It invokes the teacher d steps after the student, so that the teacher's forecast covers exactly the usable window, which helps when the teacher's bias grows with lead time, and adds frequency and correlation terms. With a ModernTCN teacher, TRIAD lowers the error of the selected student by about 3% on fetal ECG and 6% on ETTh2 replayed at 1 ms, in every latency setting. On a 1 kHz gait stream it gains 3 to 12% in two latency settings and loses 0.6% in the third. Removing its frequency or correlation term raises the error by at most 0.8%. Code is available at `https://anonymous.4open.science/r/TRIAD-8B7D`.

1 INTRODUCTION

Forecasts are consumed online, but inference is not free. Forecasters are evaluated on producing, at time t, the $\mathcal{H}$ points $t+1$ to $t+\mathcal{H}$, as if the inference time $\mathcal{T}$ were negligible (Zhou et al., 2021; Wu et al., 2023). When the sampling interval Δt is of the same order as $\mathcal{T}$, sampling continues during inference, and by the time the forecast returns, its first steps describe instants that have already passed (Figure 1).

This regime has reached deployment. Standard benchmarks are sampled every 10 minutes or more slowly, and even the finest series of GIFT-Eval every 10 seconds (Zhou et al., 2021; Wu et al., 2021; Aksu et al., 2024), so almost every model returns within one step. Yet a 5G slot lasts at most 1 ms (3GPP, 2024; Díaz-Ruiz et al., 2025), a robot control period 20 ms (Salzmann et al., 2023), and a phasor-measurement frame 10 to 33 ms (NASPI, 2014; She et al., 2022), and O-RAN control loops must close within 10 ms to 1 s (Lacava et al., 2025; Obiuwevwi et al., 2026). In our measurements, a call of an MLP student on a batch of 32 windows takes at most 2 ms on an A100 but up to 208 ms on one x86 core, depending on its width (Section 5.1).

Latency does not reduce what the consumer needs; it lengthens what a call must issue. With the stale prefix d, a model must issue $P = \mathcal{H} + d$ points to deliver $\mathcal{H}$ usable ones. On the gait stream, sampled every millisecond with $\mathcal{H} = 96$, the 208-step prefix of our largest MLP student on one x86 core means that it issues 304 steps and is used on its steps 209 to 304, whereas on an A100 it issues two extra steps. Because error grows with lead time, what a slow model delivers lies further out, where it is less accurate. On a given device, the error that matters, on the usable window, is therefore a function of the configuration, that is, of the model and its input length, and there are two ways to lower it: choose a better configuration, or teach the chosen one better.

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.

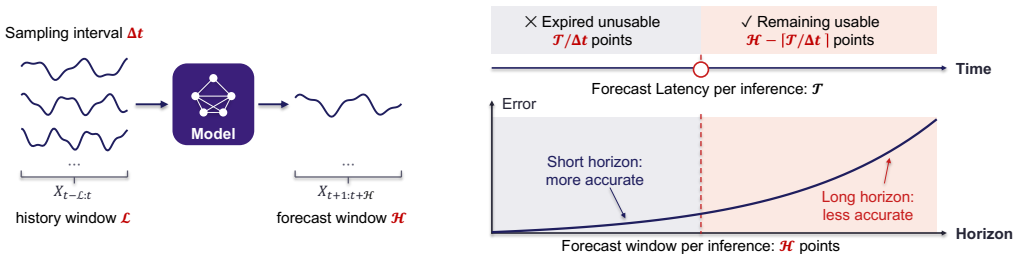


Figure 1: Inference latency turns the most accurate part of a forecast into history. **Left:** a forecaster maps a history window $X_{t-\mathcal{L}:t}$, sampled every Δt, to a forecast of the next $\mathcal{H}$ points. Offline evaluation scores all $\mathcal{H}$ of them as if inference took no time; we call these steps, 1 to $\mathcal{H}$ after the call, the *nominal window*. **Top right:** when inference takes $\mathcal{T}$, the forecast is first used $d = \lceil \mathcal{T}/\Delta t \rceil$ ticks after the call. By then its first d points, the *stale prefix*, are already observed, so an $\mathcal{H}$-point forecast leaves only $\mathcal{H} - d$ points ahead. To deliver the $\mathcal{H}$ points the consumer needs, a model must therefore issue $P = \mathcal{H} + d$; its last $\mathcal{H}$ points, steps $d+1$ to $d+\mathcal{H}$, form the *usable window*. **Bottom right:** per-step error grows with lead time, so the stale prefix removes exactly the points the model predicts best.

Current practice keeps latency out of both. Benchmarks score the nominal window. Hardware-aware architecture search treats measured latency as a constraint or penalty on unchanged offline accuracy (Tan et al., 2019; Cai et al., 2020). Distillation for forecasting (Ni et al., 2026; Li et al., 2026; Dey et al., 2026) invokes the teacher on the student's own input, so its target is anchored at the nominal window. Streaming perception and trajectory prediction account for latency, scoring outputs when they arrive or training against a later frame or a latency gap (Li et al., 2020; Yang et al., 2022; Girase et al., 2023; Lan et al., 2026), but those that train use a fixed or tuned latency rather than the model's own (Appendix A).

We treat inference latency as a rewriting of the forecasting objective rather than a constraint on it. Every deployed model, the student included, has its own stale prefix $d = \lceil \mathcal{T}/\Delta t \rceil \geq 1$, set by the 0.95-quantile latency $\mathcal{T}$ of its call on the target device (Section 3.1). With $\Delta t = 1\,\text{ms}$ and latency timed on a batch of 32 windows, our MLP student has $d \leq 2$ at every width on an A100. On one x86 core its prefix ranges from 5 to 208 steps across our three streams and is longest at the largest width on each (Section 5.1). For the first route (Section 3), the deployment error of a configuration is its error on the usable window set by its own latency; it factorizes into a device-independent per-step error curve and a device-dependent prefix, so a pool of candidates, including models deployed as they are, is measured once and re-scored for each device. For the second route (Section 4), reweighting the loss toward the usable window changes little (Lan et al., 2026), so we change the target. TRIAD invokes a teacher, chosen from a pool by its validation error, d steps after the student, so that its forecast, the *usable-window target*, covers exactly the steps the student is scored on, which pays when the teacher's bias grows with lead time. It adds a frequency term that reweights the pointwise discrepancy between the modulus and the phase of each Fourier coefficient, keeping the phase a modulus-only term cannot see, and a correlation term, since the squared loss leaves residual correlation free at a fixed per-variable error; the shifted target and the two terms are the triad. Our contributions are:

- **Latency-aware forecasting:** the stale prefix, the usable window, the deployment error, its factorization, and a selection procedure that trains each candidate once (Section 3); Appendix B analyzes when the optimum is interior and how it moves.
- **TRIAD:** distillation at the usable-window target, with an account of when the shift helps, and frequency and correlation terms on forecasts alone, so that any student can be paired with any teacher (Section 4).
- **Measurements on both routes:** a ModernTCN teacher and MLP students of 13k to 13.4M parameters, on ECG and gait streams at 1 kHz and on ETTh2 replayed at 1 ms. Latency is measured on an A100 GPU and one x86 core, and an edge setting takes ten times the

prefix of that core. We run the baseline, TimeDistill, with the teacher called d steps after the student, as in TRIAD, so the two differ only in the objective (Section 5).

In Section 5, latency changes which student is chosen, and TRIAD helps in every setting on two of three streams. On ECG and one x86 core, the nominal-window error picks a 3.3M-parameter MLP student with a 20-step prefix. The usable-window error is U-shaped in student size and picks the 837k student, whose prefix is 8 steps. From the A100 to the x86 core, the selected student shrinks from 3.3M to 837k parameters on fetal ECG and from 13M to 209k on gait. In all three settings, TRIAD lowers the error of the selected student below no distillation by 2.9 to 3.3% on ECG and by 6.1 to 6.4% on ETTh2. On gait its gain ranges from -0.6 to 12.4%, ahead of TimeDistill in two of three settings. TimeDistill is worse than no distillation in every ECG setting. In an ablation at injected prefixes, most of the gain of TRIAD over no distillation remains when both the frequency and the correlation term are removed. Removing either term raises the error by at most 0.8%. On the A100, the ModernTCN teacher deployed as it is, with a three-step prefix, beats the selected student under every training signal on ECG and gait. On ETTh2 it is worse than both distilled students.

2 RELATED WORK

Forecasters and their cost. Forecasters range from linear maps (Zeng et al., 2023) to convolutional and Transformer models (Zhou et al., 2021; Wu et al., 2023; Luo & Wang, 2024), whose inference cost grows with their size. Benchmarks score all of them on the nominal window, and cost enters only as a budget that admits or excludes a model; our deployment error scores each model on the outputs its own cost leaves usable.

Latency-aware evaluation. Streaming perception scores each output against the world at the time it arrives (Li et al., 2020; Yang et al., 2022), action and trajectory forecasters are trained for a given latency (Girase et al., 2023; Lan et al., 2026; Wei et al., 2025), and 5G traffic forecasters have been reported with their measured latency next to their error (Karashevych & Sulima, 2026). Hardware-aware architecture search uses measured latency as a constraint or penalty (Tan et al., 2019; Cai et al., 2019). We are not aware of prior forecasting work that derives the scored steps from the model's own measured latency, which is what the usable window does.

Distilling forecasters. TimeDistill transfers multi-scale and multi-period structure from a supervised teacher to an MLP (Ni et al., 2026), DistilTS weights the horizon steps and aligns hidden states (Li et al., 2026), and Guard routes several teachers per sample (Dey et al., 2026). Relational distillation transfers pairwise structure across samples (Park et al., 2019; Peng et al., 2019), and teachers that see longer, complete, or later observations have been distilled into students that do not (Das et al., 2024; Yu et al., 2025; He et al., 2023). TRIAD instead calls the teacher at the student's measured prefix and matches the spectrum and the cross-variable correlation of the forecasts on the usable window; Appendix A discusses each line in detail.

3 LATENCY-AWARE FORECASTING

A forecast is useful only from the moment it arrives. Sampling continues during inference, so the forecast opens with d steps already observed, its stale prefix, where d counts the sampling intervals the call takes on its device (Figure 1). A call must therefore issue $\mathcal{H} + d$ steps, and only the last $\mathcal{H}$, the usable window, reach the consumer; the deployment error scores a configuration on them alone (Section 3.1). This error depends on the device only through d, which shifts where a per-step error curve, measured once per configuration, is read (Section 3.2). A larger model usually lowers its curve, but once its prefix grows it is read further out, where the curve is typically higher. On a given device the optimum within a family can then be interior, at intermediate capacity (Section 3.3).

3.1 SETTING AND DEPLOYMENT ERROR

A stream $y_1, y_2, \cdots \in \mathbb{R}^C$ of C variables arrives every Δt. A configuration $c = (\theta, \mathcal{L})$, a model with an input length, is drawn from a finite pool $\mathcal{F}$ of trainable families at several capacities and, optionally, models deployed as they are. Invoked at s, it reads $y_{s-\mathcal{L}+1:s}$ and issues $\hat{y}_{s,1}, \ldots, \hat{y}_{s,P}$,

where $\hat{y}_{s,h}$ forecasts y_{s+h}. In an environment v, a processor with its runtime, one call at batch size one takes $\mathcal{T}(c, P; v)$, measured at the 0.95 quantile. The forecast becomes usable at tick $s + d$ with $d = \lceil \mathcal{T}/\Delta t \rceil$, when the consumer needs $y_{s+d+1}, \ldots, y_{s+d+\mathcal{H}}$, so the issued length must cover its own stale prefix, $P \geq \mathcal{H} + \lceil \mathcal{T}(c, P; v)/\Delta t \rceil$. The least such P, written $P^\star(c; v)$, is reached by fixed-point iteration from $P = \mathcal{H}$ (Lemma 1, Appendix B), and $d(c; v) = P^\star(c; v) - \mathcal{H}$. Latency is counted in ticks, so every call has $d \geq 1$, and $d = 1$ is standard evaluation one tick later.

Definition 1 (Deployment error)**.** *The deployment error of configuration c in environment v is*

$$E(c; v) = \frac{1}{\mathcal{H}} \sum_{k=1}^{\mathcal{H}} \mathbb{E}_s \frac{1}{C} \left\| \hat{y}_{s,\, d+k}(c) - y_{s+d+k} \right\|_2^2, \qquad d = d(c; v), \tag{1}$$

where s ranges over the invocation times of the evaluation split, and latency-aware forecasting selects $c^\star(v) = \arg\min E(c; v)$ over the configurations with $d(c; v) \leq \mathcal{H}$.

Of the $P^\star = \mathcal{H} + d$ issued steps, the first d form the *stale prefix* and the last $\mathcal{H}$, steps $d+1$ to $d+\mathcal{H}$, form the *usable window* $\mathcal{W}$. Steps 1 to $\mathcal{H}$, which latency-free evaluation scores, form the *nominal window*. E contains no latency budget: it exceeds the nominal-window error by $\big(\sum_{h=\mathcal{H}+1}^{\mathcal{H}+d} e_h - \sum_{h=1}^{d} e_h\big)/\mathcal{H}$, with e_h the error at step h, so it depends on the shape of each error curve, not only on offline error and latency; under a shared lead-time profile it reduces to offline error times a factor of the prefix (Appendix B.3). E assumes d calls in flight.

3.2 Factorization and Selection

Let $d_{\max}$ exceed every prefix of interest and $P_{\max} = \mathcal{H} + d_{\max}$. We train each configuration once at issued length $P_{\max}$ and cache its per-step error on validation data, $e_h(c) = \mathbb{E}_s \frac{1}{C} \|\hat{y}_{s,h}(c) - y_{s+h}\|_2^2$ for $h \leq P_{\max}$; keeping the first $P^\star$ rows of its linear read-out deploys exactly its first $P^\star$ outputs, and a model deployed as it is has its curve recorded from forecasts of length $P_{\max}$. For every environment v with $d(c; v) < d_{\max}$, averaging over the same split then gives

$$E(c; v) = \frac{1}{\mathcal{H}} \sum_{k=1}^{\mathcal{H}} e_{\,d(c;v)+k}(c), \tag{2}$$

where the curve $e(c)$ does not depend on v, which enters only through the integer $d(c; v)$. Selection thus costs one training run per configuration and one latency measurement per configuration and environment, after which $E(\cdot; v)$ is a table lookup, and without pruning a new environment with prefixes below $d_{\max}$ needs no training (Algorithm 1, Appendix B.5).

3.3 What the Objective Implies

Because the prefix is an integer, many configurations of a family share one; among them the largest is best when accuracy grows with capacity, so the minimizer on a grid lies at the top of a level set of the prefix, and only those configurations need training (Remark 1). Under a separable model of the error curve with an unrounded prefix, Appendix B shows that the optimum can be interior, where the relative gain from capacity equals the elasticity of latency times that of the window error (Proposition 3), and how it moves with the environment (Proposition 4); selection itself uses only equation 2.

4 Triad: Distilling on the Usable Window

Each trainable configuration is a student; nothing below changes its size, latency, or prefix d. Reweighting its loss toward $\mathcal{W}$ leaves per-step read-outs unchanged (Proposition 2) and barely helps even with a shared encoder (Lan et al., 2026), so Triad changes its target, in three ways (Figure 2): it calls the teacher d steps later (Section 4.1), penalizes the modulus and the phase of the forecast spectrum separately (Section 4.2), and matches the correlation among variables (Section 4.3).

4.1 The Usable-Window Target

A call at s issues a forecast tensor $\hat{Y} \in \mathbb{R}^{P_{\max} \times C}$ whose row h forecasts y_{s+h}; $\hat{Y}^S$ is the student's, $\hat{Y}^T$ the teacher's, and Y the observed future. The teacher is chosen from a pool of forecasters by

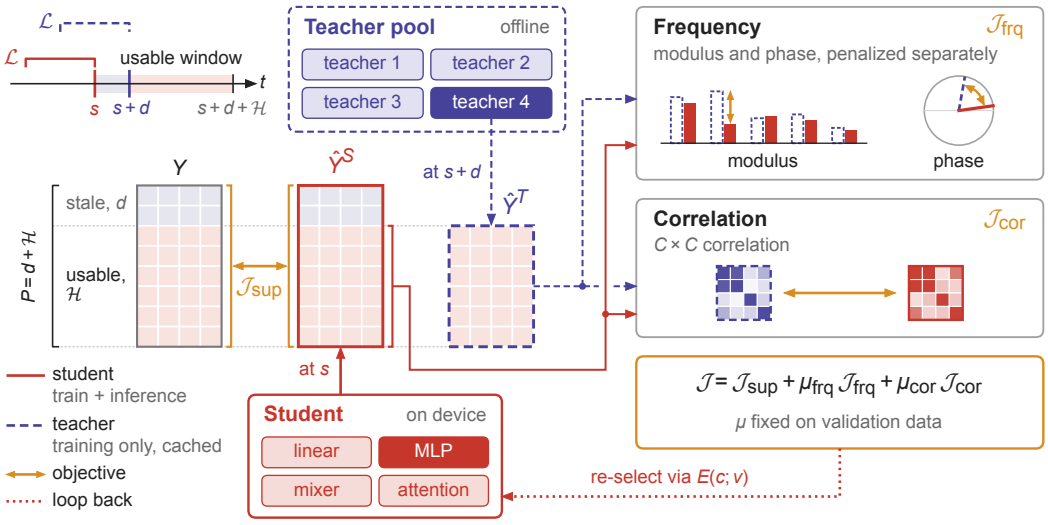


Figure 2: TRIAD changes the target of a fixed configuration. The student (solid) reads $\mathcal{L}$ steps ending at s and is the only branch at inference; of its $P = d + \mathcal{H}$ issued rows, the first d are the stale prefix and the last $\mathcal{H}$ the usable window. The teacher (dashed) is selected from a pool by its validation error on the usable window, invoked at $s + d$, and cached offline, so its first $\mathcal{H}$ steps cover the student's usable window. The frequency term penalizes modulus and phase separately, and the correlation term matches the $C \times C$ correlation of the two forecasts. The distilled error curve re-enters the deployment error, and the configuration is re-selected.

its validation error on its own first $\mathcal{H}$ steps, which is its usable-window error at every d, before any student is trained; it runs offline, so its latency never enters d. Published distillation invokes it on the student's input at s, so on the rows of $\mathcal{W}$ its forecast lies at lead times $d+1$ to $d+\mathcal{H}$, as far out as the student's; we call this the *nominal-window target*. TRIAD invokes it at $s + d$,

$$\hat{Y}^T_{d+h,:} = \hat{y}_{s+d,h}(c_{\text{tea}})^\top, \qquad h = 1, \dots, \mathcal{H}, \tag{3}$$

which reaches the same rows at lead times 1 to $\mathcal{H}$, the *usable-window target*. One cache per teacher and dataset, up to horizon $P_{\max}$, serves every prefix, student, and baseline.

Which target is better depends on the teacher: write its forecast as $\hat{y}_{s,h}(c_{\text{tea}}) = \mathbb{E}[y_{s+h} \mid \mathcal{F}_s] + b_{s,h}$, with $\mathcal{F}_s$ the history up to s and $b_{s,h}$ its conditional bias, and let the student read x_s, a function of $\mathcal{F}_s$.

Proposition 1 (Target shift). *For either target τ on row $d+k$ of $\mathcal{W}$, the minimizer $f^\star = \mathbb{E}[\tau \mid x_s]$ of $\mathbb{E}\|f(x_s) - \tau\|_2^2$ has squared error $\mathbb{E}\|y_{s+d+k} - \mathbb{E}[y_{s+d+k} \mid x_s]\|_2^2 + \mathbb{E}\|\mathbb{E}[\beta \mid x_s]\|_2^2$ on that row, with $\beta = b_{s+d,k}$ for the usable-window target and $\beta = b_{s,d+k}$ for the nominal-window one. For an ideal teacher, $b \equiv 0$, the two minimizers coincide, and the usable-window target has the larger conditional variance, by $\mathbb{E}[\|\tau^{\text{use}} - \tau^{\text{nom}}\|_2^2 \mid x_s]$.*

For pointwise matching the shift thus helps only through the teacher's bias, passing on its bias at lead k instead of $d+k$, and costs the variance of the forecast revision between s and $s+d$, which grows with d; with an ideal teacher it cannot help, even with the supervised term added (Appendix C.1), and for the nonlinear terms below that variance does not average out (Appendix C.4).

4.2 FREQUENCY TERM

For column j of a tensor on $\mathcal{W}$, let $\check{Y}_j[k]$ be its discrete Fourier coefficient at bin k, and $\Delta\phi_j[k]$ the phase gap between teacher and student. By Parseval's identity, the squared discrepancy on $\mathcal{W}$ is a sum over bins of $|\check{Y}^S_j[k] - \check{Y}^T_j[k]|^2$, and each term splits into a modulus part and a phase part (Appendix C). A modulus-only objective, such as the spectral term of Ni et al. (2026), cannot see the timing of a move: for a zero-mean target y, the forecast $-y$ matches every modulus while its squared error is four times that of the constant predictor. The frequency term keeps both parts and

weights them separately,

$$\mathcal{J}_{\mathrm{frq}} = \frac{1}{C\mathcal{H}^2} \sum_{j=1}^{C} \left[\sum_{k=0}^{\lfloor \mathcal{H}/2 \rfloor} \left(\left|\check{Y}_j^S[k]\right| - \left|\check{Y}_j^T[k]\right| \right)^2 + \frac{\xi}{2} \sum_{k \in \mathcal{B}} \left|\check{Y}_j^T[k]\right|^2 \left| U_j^S[k] - U_j^T[k] \right|^2 \right], \quad (4)$$

with $\xi > 0$, $U_j[k] = \check{Y}_j[k] / \sqrt{|\check{Y}_j[k]|^2 + \eta^2}$ for a small floor η, and $\mathcal{B} = \{1, \ldots, \lceil \mathcal{H}/2 \rceil - 1\}$, omitting the zero-frequency and Nyquist bins. Where both moduli are large against η, $\frac{1}{2}|U_j^S[k] - U_j^T[k]|^2 \approx 1 - \cos \Delta\phi_j[k]$, which has no branch cut. Weighting the phase by the teacher's squared modulus stops the student from evading it by suppressing its own modulus and gives no weight to bins without energy. Since pointwise matching charges the same discrepancy with fixed weights, $\mathcal{J}_{\mathrm{frq}}$ adds no new target but sets how an irreducible residual divides between the size and the timing of a move; $\xi = 2$ reproduces pointwise matching up to scale near convergence (Appendix C.2).

4.3 Correlation Term

With $\hat{Y}^\circ$ a forecast tensor on $\mathcal{W}$ with each column centered, the correlation of the variables' forecast paths over the rows of $\mathcal{W}$ and the correlation term are

$$R(\hat{Y})_{jl} = \frac{\langle \hat{Y}^\circ_{:,j},\, \hat{Y}^\circ_{:,l} \rangle}{\max\big(\|\hat{Y}^\circ_{:,j}\|_2, \eta\big)\, \max\big(\|\hat{Y}^\circ_{:,l}\|_2, \eta\big)}, \qquad \mathcal{J}_{\mathrm{cor}} = \frac{1}{C(C-1)} \sum_{j \neq l} \big(R(\hat{Y}^S)_{jl} - R(\hat{Y}^T)_{jl} \big)^2. \quad (5)$$

The squared loss, pointwise or supervised, gives the same value to forecasts with equal nonzero per-variable errors whose residual correlations differ, and no objective separable across variables, $\mathcal{J}_{\mathrm{frq}}$ included, has a gradient term coupling two variables; when $C \leq \mathcal{H} - 2$, $R(\hat{Y}^S)$ can even take different values at fixed per-variable errors (Proposition 5). At the error level the prefix holds the student at, a separable objective thus supplies no first-order signal separating one correlation structure from another. $\mathcal{J}_{\mathrm{cor}}$ supplies it from the most accurate multivariate teacher, as the analogue across variables of relational distillation (Park et al., 2019; Peng et al., 2019); a univariate teacher's R holds only the co-movement inherited from correlated inputs, so comparing the two isolates what joint modeling adds. E registers $\mathcal{J}_{\mathrm{cor}}$ only through per-variable error, and matching R does not bound the error of an aggregate such as the sum of the variables, so that effect is empirical. Ni et al. (2026) name a covariance-matching loss as future work; Proposition 5 is why we adopt it.

4.4 Training and Re-Selection

The student minimizes

$$\mathcal{J} = \mathcal{J}_{\mathrm{sup}} + \mu_{\mathrm{frq}}\, \mathcal{J}_{\mathrm{frq}} + \mu_{\mathrm{cor}}\, \mathcal{J}_{\mathrm{cor}}, \qquad \mathcal{J}_{\mathrm{sup}} = \frac{1}{P_{\max} C} \sum_{h=1}^{P_{\max}} \big\| \hat{Y}^S_{h,:} - Y_{h,:} \big\|_2^2, \quad (6)$$

with $\mathcal{J}_{\mathrm{frq}}$ and $\mathcal{J}_{\mathrm{cor}}$ on $\mathcal{W}$ and the supervised term on all issued steps, so every read-out row stays trained. The weights μ_{frq}, μ_{cor}, and ξ are chosen on validation data and held fixed, and every term is training-only. The curve distilled at $d(c; v)$ replaces e_h in equation 2, and the configuration is re-selected (Algorithm 1); a new environment costs distillation runs but no new teacher pass.

5 Experiments

We ask whether selecting by the usable window chooses better configurations, whether TRIAD teaches the chosen one better, and why.

5.1 Setup

Streams and devices. ECG has the four abdominal channels of the noninvasive fetal ECG challenge (Silva et al., 2013) and GAIT twelve of the 22 force-plate channels of a gait dataset (Katmah et al., 2026), both sampled at 1 kHz and from PhysioNet (Pollard et al., 2026); ETTh2 (Zhou et al., 2021) is hourly ($C = 7$), and we read each of its steps as 1 ms. Every model reads $\mathcal{L} = 720$ steps

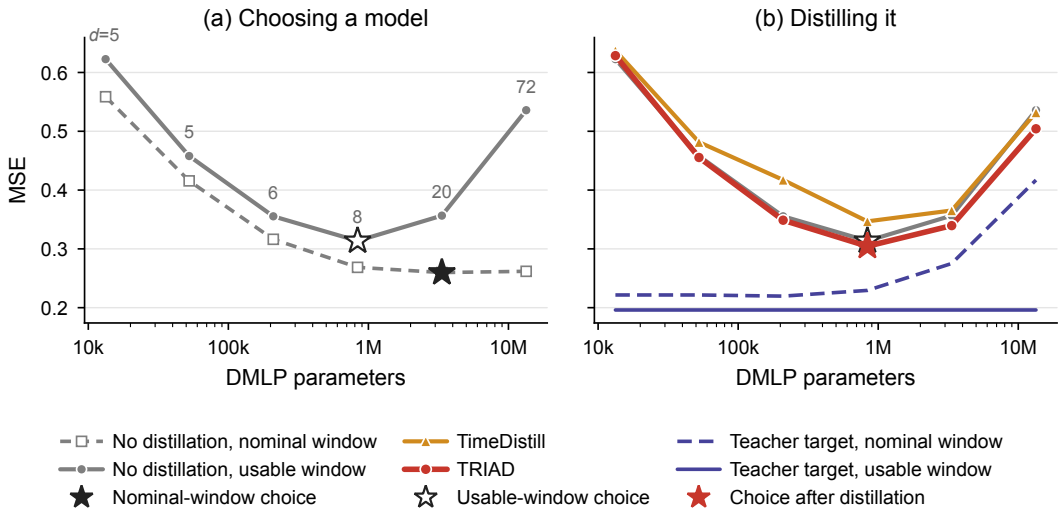


Figure 3: Choosing a model, then distilling it: ECG, one x86 core, the DMLP ladder, the ModernTCN teacher, one seed. **(a)** Selecting by the nominal-window error picks the 3.3M DMLP, whose stale prefix is 20 steps (numbers next to the markers); the usable-window error is U-shaped and selects the 837k DMLP with $d = 8$. **(b)** Everything on the usable window. The teacher's usable-window target is flat in the student's size, while its nominal-window target degrades as the student's prefix grows; TRIAD lies below no distillation at every width but the smallest, and TimeDistill with its published objective lies above it at every width but the largest.

and serves $\mathcal{H} = 96$, that is, 96 ms. $\mathcal{T}$ is the 0.95 quantile of calls on a batch of 32 windows, timed at $\mathcal{H} = 96$ before training on an A100 and on one x86 core, and $d = \lceil \mathcal{T}/1\,\mathrm{ms} \rceil$; the edge setting takes ten times the x86 prefix. On the A100 every DMLP has $d \leq 2$ and ModernTCN $d = 3$; on the x86 core d grows from 5–8 at the smallest widths to 72, 208, and 127 at the largest on ECG, GAIT, and ETTh2.

Student, teacher, and baseline. The student is DMLP, the MLP of TimeDistill (Ni et al., 2026), at widths 8 to 8192 (13k to 13.4M parameters); on each device every width is trained for $d + \mathcal{H}$ steps and scored on the last $\mathcal{H}$, and a model trained for $\mathcal{H} = 96$ gives the nominal-window error. The teacher is ModernTCN (Luo & Wang, 2024) (1.2M to 1.4M parameters), also deployed as it is on the A100. TimeDistill with its published objective is the baseline; it and TRIAD call the teacher at $s + d$ and share the budget (20 epochs, batch 32) and loss weights. We report the usable-window MSE, E, from one seed.

5.2 Choosing and Distilling a Student

In Figure 3a the nominal-window error falls along the DMLP ladder up to 3.3M parameters, but the usable-window error turns up beyond 837k, once the prefix grows faster than accuracy improves, and the two minimizers differ by a factor of four in size.

The choice also moves with the device (Table 1, top rows). From the A100 to the x86 core the selected student shrinks from 3.3M to 837k parameters on ECG, from 13M to 209k on GAIT, and from 3.3M to 209k on ETTh2, and the edge setting selects the 52.5k student on ETTh2 while ECG and GAIT keep their x86 choices. Each column then fixes that student, so the three training signals are compared on one configuration.

In Table 1, TimeDistill and TRIAD call the same teacher at $s + d$ and differ only in the objective. On ECG and ETTh2, TRIAD is the best student in every column, 2.9 to 3.3% and 6.1 to 6.4% below no distillation, and its gain is nearly independent of the prefix from $d = 1$ to 80; TimeDistill trails no distillation in every ECG column. On GAIT, TRIAD gains 12.4% on the A100 and 3.4% in the edge setting, against 7.1% and 1.1% for TimeDistill; on the x86 core, with the small 209k student,

Table 1: Usable-window MSE of the student that the usable window selects without distillation, under each training signal, with the ModernTCN teacher and one seed. TimeDistill and TRIAD call the teacher at $s + d$ and differ only in the objective. The edge setting takes ten times the prefix of the x86 core. The pale row is the teacher at zero latency, which no deployment attains; the deployed teacher runs on the A100 at its own prefix (–: not measured on that device).

	ECG			GAIT			ETTh2		
	A100	x86	Edge	A100	x86	Edge	A100	x86	Edge
Selected student	3.3M	837k	837k	13M	209k	209k	3.3M	209k	52.5k
Stale prefix d	1	8	80	2	8	80	2	8	60
Teacher									
ModernTCN, zero latency	0.196	0.196	0.196	0.052	0.052	0.052	0.182	0.182	0.182
ModernTCN, deployed ($d = 3$)	**0.212**	–	–	**0.055**	–	–	0.212	–	–
No distillation	0.264	0.314	0.539	0.082	**0.099**	0.186	0.206	0.216	0.234
TimeDistill	0.277	0.347	0.582	0.076	**0.099**	0.184	0.194	0.205	0.224
TRIAD (ours)	0.255	**0.304**	**0.523**	0.072	**0.099**	**0.180**	**0.193**	**0.202**	**0.219**
Gain of TRIAD (%)	3.3	2.9	3.1	12.4	−0.6	3.4	6.1	6.4	6.2

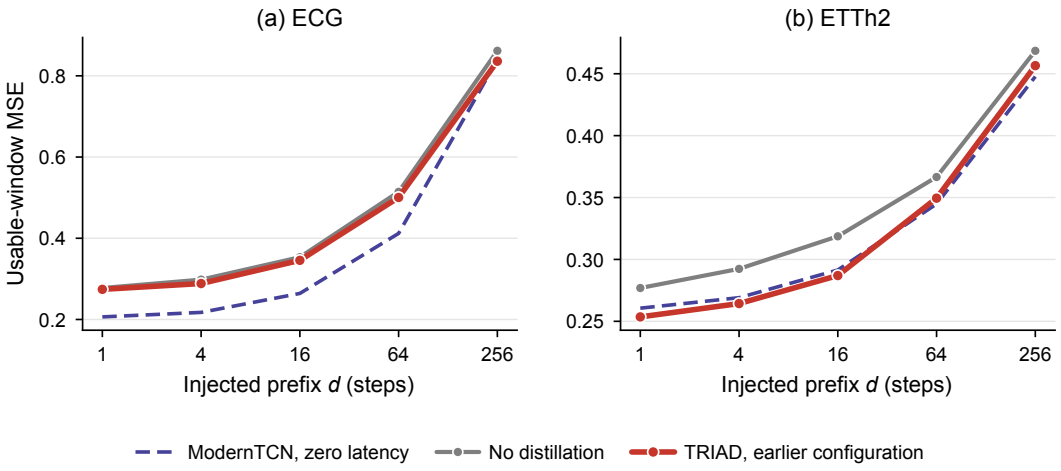


Figure 4: Usable-window MSE against an injected prefix for the 837k DMLP, with ModernTCN trained for the same horizon at zero latency, one seed. **(a)** ECG. **(b)** ETTh2. The distilled student is an earlier configuration of TRIAD, within 0.4% of Table 2 at $d = 4$ and 64.

it is 0.6% worse than no distillation and TimeDistill level with it. On the A100 the teacher deployed at $d = 3$ is best on ECG and GAIT, at 0.212 and 0.055 against 0.255 and 0.072 for the best distilled students, and worse than both distilled students on ETTh2.

5.3 How the Gain Depends on the Prefix

Figure 4 fixes the 837k DMLP and injects the prefix from 1 to 256. Every error grows with d. On ECG the teacher's lead over the undistilled student grows from 0.071 at $d = 1$ to 0.102 at $d = 64$ and collapses to 0.022 at $d = 256$, where the distilled student is also slightly below the teacher, after staying above it at every smaller prefix; on ETTh2 the lead stays between 0.016 and 0.027. The gain of this configuration over no distillation stays between 1.2 and 3.2% on ECG, and on ETTh2 rises to 9.9% at $d = 16$ before falling to 2.6% at $d = 256$; there the distilled student is below the teacher up to $d = 16$.

Table 2: Ablation of TRIAD for the 837k DMLP at injected prefixes, one seed. Each variant removes the frequency term, the correlation term, or both, which leaves the supervised and multi-scale terms; in parentheses the change against TRIAD (%), positive when worse. The correlation term is kept only where adding it lowers the validation error (last row); where it is dropped, TRIAD and its variant without it are the same run.

	ECG		ETTh2	
	$d = 4$	$d = 64$	$d = 4$	$d = 64$
TRIAD	**0.2873**	**0.4997**	**0.2646**	**0.3495**
w/o frequency term	0.2896 (+0.8)	**0.4997** (0.0)	0.2650 (+0.2)	0.3497 (+0.1)
w/o correlation term	**0.2873** (0.0)	0.5000 (+0.1)	**0.2646** (0.0)	**0.3495** (0.0)
w/o both terms	0.2896 (+0.8)	0.4999 (0.0)	0.2650 (+0.2)	0.3497 (+0.1)
No distillation	0.2981 (+3.8)	0.5143 (+2.9)	0.2924 (+10.5)	0.3666 (+4.9)
Correlation term kept	no	yes	no	no

5.4 ABLATIONS

Table 2 removes each term from TRIAD. Every variant stays 2.8 to 9.5% below no distillation, so most of the gain remains without either term. Removing the frequency term raises the error by 0.0 to 0.8%, removing the correlation term by 0.0 to 0.1%, and removing both by 0.0 to 0.8%, so the full objective is best or tied in every setting. The correlation term is kept only on ECG at $d = 64$, where adding it lowers the validation error; in the other settings TRIAD and its variant without it are the same run.

6 CONCLUSION

Inference latency changes what a forecaster is scored on: the deployment error factorizes into a device-independent error curve and a device-dependent prefix, and its optimum moves with the device. TRIAD teaches a trainable configuration with a teacher invoked d steps later and with frequency and correlation terms. In our measurements the usable-window error selects an 837k student on ECG and one x86 core, where the nominal-window error selects a 3.3M one, and TRIAD lowers the usable-window error of the selected student by 2.9 to 3.3% on ECG and 6.1 to 6.4% on ETTh2 in every latency setting, while on GAIT its gain ranges from -0.6 to 12.4%. On the A100, the teacher deployed at its own three-step prefix remains the most accurate model on ECG and GAIT.

Limitations. The measured evidence is narrow: one seed, one teacher, one student family, and three streams, one of them hourly data replayed at 1 ms. The single teacher leaves the teacher pool of Section 4 untested, and every width is trained for each device rather than once at $P_{\max}$, so the runs do not test the factorization that measures a pool once and re-scores it per device. Configurations are selected on test errors, the only ones we report, so the usable-window choice is optimistic, and the edge setting multiplies the x86 prefix by ten rather than timing a device. Both distillation methods call the teacher at $s + d$, so the runs do not compare a student distilled toward the usable-window target with one distilled toward the nominal-window target. Removing the frequency term, the correlation term, or both raises the error by at most 0.8%, so most of the gain of TRIAD over no distillation comes from the multi-scale term that every variant keeps. The deployment error assumes d contention-free workers and ignores calls slower than the latency quantile, and the attainable phase and correlation targets shrink with the prefix.

Future work. Each limitation points to a next experiment: several seeds with selection on validation errors, a timed edge device, several teachers selected from a pool, other student families, other baselines such as DistilTS and Guard, and one student trained toward the nominal-window and toward the usable-window target, which would isolate what calling the teacher d steps later contributes.

Reproducibility Statement

The formal results of Sections 3 and 4 and Appendices B and C state their assumptions and are proved where they are stated, Proposition 1 in Appendix C.1; the separable model of Appendix B.3 and the attenuation of Appendix C.4 are stated as models, without error bounds. Algorithm 1 specifies selection and distillation. Section 5.1 gives the streams, the student widths, the teacher, the baseline, the training budget, the latency protocol, and the measured prefixes. All reported runs use one seed; latency is the 0.95 quantile on a batch of 32 windows, and the edge setting multiplies the x86 prefix by ten instead of timing a device. Code is available at the anonymous repository linked in the abstract.

Policy on LLM Use for Research and Writing Papers

We used large language models to polish the writing of this paper and to assist with the experiments and code. The authors take full responsibility for the content of the paper.

References

3GPP. NR; physical channels and modulation. Technical Specification TS 38.211, 3rd Generation Partnership Project, 2024.

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. arXiv preprint arXiv:2410.10393, 2024.

Han Cai, Ligeng Zhu, and Song Han. ProxylessNAS: Direct neural architecture search on target task and hardware. In *International Conference on Learning Representations*, 2019.

Han Cai, Chuang Gan, Tianzhe Wang, Zhekai Zhang, and Song Han. Once-for-all: Train one network and specialize it for efficient deployment. In *International Conference on Learning Representations*, 2020.

Sourav Das, Guglielmo Camporese, Shaokang Cheng, and Lamberto Ballan. Distilling knowledge for short-to-long term trajectory prediction. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2024.

Rupasree Dey, Abdul Matin, Nathan Orwick, Yao Zhang, Shrideep Pallickara, and Sangmi Lee Pallickara. When to trust, how to distill: Multi-foundation model guidance for lightweight, robust scientific time series forecasting. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2026.

Francisco Díaz-Ruiz, Francisco J. Martín-Vega, José Antonio Cortés, Gerardo Gómez, and Mari Carmen Aguayo-Torres. Deep learning-based CSI prediction framework for channel aging mitigation in TDD 5G systems. arXiv preprint arXiv:2510.24400, 2025.

Harshayu Girase, Nakul Agarwal, Chiho Choi, and Karttikeya Mangalam. Latency matters: Real-time action forecasting transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

Jianping Gou, Baosheng Yu, Stephen John Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819, 2021. doi: 10.1007/s11263-021-01453-z.

Jun-Yan He, Zhi-Qi Cheng, Chenyang Li, Wangmeng Xiang, Binghui Chen, Bin Luo, Yifeng Geng, and Xuansong Xie. DAMO-StreamNet: Optimizing streaming perception in autonomous driving. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, 2023.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531, 2015.

Tao Huang, Shan You, Fei Wang, Chen Qian, and Chang Xu. Knowledge distillation from a stronger teacher. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Yevhen Karashevych and Svitlana Sulima. Latency-aware accuracy evaluation of network traffic prediction models for proactive QoS control in 5G/IoT networks. *Information and Telecommunication Sciences*, 17(1):20–26, 2026. doi: 10.20535/2411-2976.12026.20-26.

Rickard K. A. Karlsson, Martin Willbo, Zeshan Hussain, Rahul G. Krishnan, David Sontag, and Fredrik D. Johansson. Using time-series privileged information for provably efficient learning of prediction models. In *Proceedings of the 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pp. 5459–5484. PMLR, 2022.

Rateb Katmah, Aamna AlShehhi, Doua Kosaji, Nour Al-Rahmani, Muhammad Abdullah, Abdul Aziz Vaqar Hulleck, and Kinda Khalaf. A multimodal gait dataset of brain activity, muscle activity, kinematics and ground forces in young adults (version 1.0.0). PhysioNet, 2026.

Simran Khanuja, Melvin Johnson, and Partha Talukdar. MergeDistill: Merging language models using pre-trained distillation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 2874–2887. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.findings-acl.254.

Tobias King, Yexu Zhou, Tobias Röddiger, and Michael Beigl. MicroNAS: Memory and latency constrained hardware-aware neural architecture search for time series classification on microcontrollers. arXiv preprint arXiv:2310.18384, 2023.

Andrea Lacava, Leonardo Bonati, Niloofar Mohamadi, Rajeev Gangula, Florian Kaltenberger, Pedram Johari, Salvatore D'Oro, Francesca Cuomo, Michele Polese, and Tommaso Melodia. dApps: Enabling real-time AI-based open RAN control. *Computer Networks*, 269:111342, 2025.

Zhengxing Lan, Lingshan Liu, Haiyang Yu, and Yilong Ren. LatenAux: Toward latency-aware trajectory prediction for autonomous driving via consolidated auxiliary learning. *Communications in Transportation Research*, 6(1):9640010, 2026. doi: 10.26599/COMMTR.2026.9640010.

Mengtian Li, Yu-Xiong Wang, and Deva Ramanan. Towards streaming perception. In *European Conference on Computer Vision*, 2020.

Yuqi Li, Kuiye Ding, Chuanguang Yang, Szu-Yu Chen, and Yingli Tian. Distilling time series foundation models for efficient forecasting. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2026.

Shengsheng Lin, Weiwei Lin, Wentai Wu, Haojun Chen, and Junjie Yang. SparseTSF: Modeling long-term time series forecasting with 1k parameters. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 30211–30226. PMLR, 2024.

Chenxi Liu, Hao Miao, Qianxiong Xu, Shaowen Zhou, Cheng Long, Yan Zhao, Ziyue Li, and Rui Zhao. Efficient multivariate time series forecasting via calibrated language models with privileged knowledge distillation. In *IEEE International Conference on Data Engineering*, 2025.

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

Yuang Liu, Wei Zhang, and Jun Wang. Adaptive multi-teacher multi-level knowledge distillation. *Neurocomputing*, 415:106–113, 2020. doi: 10.1016/j.neucom.2020.07.048.

David Lopez-Paz, Léon Bottou, Bernhard Schölkopf, and Vladimir Vapnik. Unifying distillation and privileged information. In *International Conference on Learning Representations*, 2016.

Donghao Luo and Xue Wang. ModernTCN: A modern pure convolution structure for general time series analysis. In *International Conference on Learning Representations*, 2024.

Aditya K. Menon, Ankit Singh Rawat, Sashank Reddi, Seungyeon Kim, and Sanjiv Kumar. A statistical perspective on distillation. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 7632–7642. PMLR, 2021.

Seyed-Iman Mirzadeh, Mehrdad Farajtabar, Ang Li, Nir Levine, Akihiro Matsukawa, and Hassan Ghasemzadeh. Improved knowledge distillation via teacher assistant. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 2020.

NASPI. Synchrophasor technology fact sheet. Technical report, North American SynchroPhasor Initiative, October 2014.

Cuong V. Nguyen, Tal Hassner, Matthias Seeger, and Cedric Archambeau. LEEP: A new measure to evaluate transferability of learned representations. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*. PMLR, 2020.

Juntong Ni, Zewen Liu, Shiyu Wang, Ming Jin, and Wei Jin. TimeDistill: Efficient long-term time series forecasting with MLP via cross-architecture distillation. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.1*, pp. 1113–1124. Association for Computing Machinery, 2026. doi: 10.1145/3770854.3780218.

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

Lawrence Obiuwevwi, Krzysztof J. Rechowicz, Sampath Jayarathna, Safdar Hussain Bouk, Fahmida Afrin, C. Nicolas Barati, Neda Moghim, Valentina Nanou, Muhammad Enayetur Rahman, and Sachin Shetty. Enabling real-time AI in O-RAN: Deploying and measuring AI inside a Near-RT RIC xApp. In *Proceedings of the IEEE International Conference on Computer Communications and Networks (ICCCN)*, 2026.

Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3967–3976, 2019.

Baoyun Peng, Xiao Jin, Jiaheng Liu, Shunfeng Zhou, Yichao Wu, Yu Liu, Dongsheng Li, and Zhaoning Zhang. Correlation congruence for knowledge distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5006–5015, 2019.

Jiaqi Peng, Tai Wang, Jiangmiao Pang, and Yuan Shen. Towards latency-aware 3D streaming perception for autonomous driving. In *IEEE International Conference on Robotics and Automation*, pp. 3727–3733, 2025. doi: 10.1109/ICRA55743.2025.11128729.

T. Pollard, B. E. Moody, L. Lehman, B. Gow, C. Fernandes, C. Xie, A. Johnson, R. G. Mark, and T. Heldt. PhysioNet as a global platform for biomedical research. *Nature Health*, 1(8):792–795, 2026. doi: 10.1038/s44360-026-00096-z.

Tim Salzmann, Elia Kaufmann, Jon Arrizabalaga, Marco Pavone, Davide Scaramuzza, and Markus Ryll. Real-time neural MPC: Deep learning model predictive control for quadrotors and agile robotic platforms. *IEEE Robotics and Automation Letters*, 8(4), 2023.

Buxin She, Yuqing Dong, and Yilu Liu. Time delay of wide area damping control in urban power grid: Model-based analysis and data-driven compensation. *Frontiers in Energy Research*, 10: 895163, 2022.

Jingzhe Shi, Qinwei Ma, Huan Ma, and Lei Li. Scaling law for time series forecasting. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Ikaro Silva, Joachim Behar, Reza Sameni, Tingting Zhu, Julien Oster, Gari D. Clifford, and George B. Moody. Noninvasive fetal ECG: The PhysioNet/Computing in Cardiology challenge 2013. In *Computing in Cardiology*, volume 40, pp. 149–152, 2013.

Samuel Stanton, Pavel Izmailov, Polina Kirichenko, Alexander A. Alemi, and Andrew Gordon Wilson. Does knowledge distillation really work? In *Advances in Neural Information Processing Systems*, volume 34, 2021.

Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V. Le. MnasNet: Platform-aware neural architecture search for mobile. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

Hao Wang, Licheng Pan, Zhichao Chen, Degui Yang, Sen Zhang, Yifei Yang, Xinggao Liu, Haoxuan Li, and Dacheng Tao. FreDF: Learning to forecast in the frequency domain. In *International Conference on Learning Representations*, 2025.

Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Y. Zhang, and Jun Zhou. TimeMixer: Decomposable multiscale mixing for time series forecasting. In *International Conference on Learning Representations*, 2024.

Dong Wei, Xiaoning Sun, Xizhan Gao, Shengxiang Hu, and Huaijiang Sun. ALIEN: Implicit neural representations for human motion prediction under arbitrary latency. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In *Advances in Neural Information Processing Systems*, 2021.

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

Zhijian Xu, Ailing Zeng, and Qiang Xu. FITS: Modeling time series with $10k$ parameters. In *International Conference on Learning Representations*, 2024.

Jinrong Yang, Songtao Liu, Zeming Li, Xiaoping Li, and Jian Sun. Real-time object detection for streaming perception. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

Kaichao You, Yong Liu, Jianmin Wang, and Mingsheng Long. LogME: Practical assessment of pre-trained models for transfer learning. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*. PMLR, 2021.

Shan You, Chang Xu, Chao Xu, and Dacheng Tao. Learning from multiple teacher networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1285–1294. Association for Computing Machinery, 2017. doi: 10.1145/3097983.3098135.

Chengqing Yu, Fei Wang, Chuanguang Yang, Zezhi Shao, Tao Sun, Tangwen Qian, Wei Wei, Zhulin An, and Yongjun Xu. Merlin: Multi-view representation learning for robust multivariate time series forecasting with unfixed missing rates. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. Association for Computing Machinery, 2025.

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

Hailin Zhang, Defang Chen, and Can Wang. Confidence-aware multi-teacher knowledge distillation. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2022.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021.

Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 27268–27286. PMLR, 2022.

A Extended Related Work

Forecasters and their cost. Forecasters span a wide range of cost. DLinear showed a linear map from lookback to horizon to be competitive on the standard benchmarks (Zeng et al., 2023), PatchTST, TimesNet, iTransformer, and TimeMixer restored the advantage of deep models (Nie et al., 2023; Wu et al., 2023; Liu et al., 2024; Wang et al., 2024), and FITS and SparseTSF shrink the parameter count to the order of 10^4 and 10^3 (Xu et al., 2024; Lin et al., 2024). ModernTCN (Luo & Wang, 2024) is a supervised convolutional forecaster with a long receptive field. Even offline, each training-set size has its own optimal lookback (Shi et al., 2024). Across this range, cost decides whether a model is admitted to a budget and accuracy is then scored on the nominal window; our objective scores a model on the outputs its own cost leaves usable.

Latency-aware evaluation and prediction. Streaming perception and the detectors built for it charge a detector's latency to its accuracy by scoring each output against the world at the time it arrives (Li et al., 2020; Yang et al., 2022; He et al., 2023), and Peng et al. (2025) treat that latency as a runtime distribution rather than a constant. RAFTformer requires action forecasts to finish before the target instant and finds that accuracy then peaks at an intermediate latency (Girase et al., 2023), LatenAux removes a fixed latency gap from trajectory targets and aligns its predictor to an auxiliary branch that also sees the gap (Lan et al., 2026), and ALIEN predicts motion under an arbitrary latency (Wei et al., 2025). Closest to our setting, Karashevych & Sulima (2026) score 5G traffic forecasters on prediction error alongside measured inference latency, throughput, and memory, but the scores stay separate and latency does not change which forecast steps enter the error. Hardware-aware architecture search treats latency measured on the target device as a constraint or a penalty, including for time-series classification on microcontrollers (Tan et al., 2019; Cai et al., 2019; King et al., 2023), and once-for-all training specializes one network to each device without retraining (Cai et al., 2020). Streaming evaluation thus scores each configuration at its own measured latency and can select among configurations by it (Li et al., 2020), the streaming methods that train do so against a fixed or tuned latency, and once-for-all specialization changes what is deployed but not what is scored. Our objective uses one quantity, the prefix induced by the model's own measured cost, both to rank candidates, rather than filter them, and to set the target each is trained against. We are not aware of prior work in time-series forecasting that derives the usable window from the forecaster's own measured latency.

Distilling forecasters. Distillation chooses what to transfer and from which input. Beyond matching the teacher's outputs (Hinton et al., 2015), methods match intermediate features or relations among samples, which Gou et al. (2021) formalize as response-, feature-, and relation-based knowledge; relational objectives transfer the pairwise structure of a batch (Park et al., 2019; Peng et al., 2019), and our correlation term is their analogue across the variables of a single forecast (Section 4.3). For time series, TimeDistill transfers multi-scale and multi-period structure from one supervised teacher to an MLP that sees the same window, the latter as a softmax of the amplitude spectrum matched by a KL divergence (Ni et al., 2026); with a ModernTCN teacher, removing that term leaves the Electricity, ETT, and Solar errors unchanged at 0.157, 0.342, and 0.184 and raises the Traffic and Weather errors by 0.004 and 0.001, but with an iTransformer teacher it raises the ETT error from 0.345 to 0.383 (their Tables 4 and 25). DistilTS trains DLinear and iTransformer students with horizon weights that increase toward far steps and with hidden states aligned to the teacher's, and its authors note that with unlimited data and computation uniform averaging would suffice (Li et al., 2026), which Proposition 2 states for per-step read-outs. TimeKD distills a language-model teacher that sees the ground-truth future into a student that sees only the history, and aligns a $C \times C$ encoder attention matrix, which exists only if the student attends over variables (Liu et al., 2025). The Fourier transform also appears outside distillation: FEDformer places it in the representation and supervises with a time-domain squared error (Zhou et al., 2022), and FreDF places it in the objective, penalizing the complex spectral difference with an ℓ_1 norm, which keeps the argument but does not weight it separately (Wang et al., 2025). Teachers that see more than the student are studied as privileged information (Lopez-Paz et al., 2016; Karlsson et al., 2022), and teachers with longer, complete, or later observations have been distilled into students without them, the last because the student's output is consumed with delay (Das et al., 2024; Yu et al., 2025; He et al., 2023). Ni et al. (2026) name a covariance-matching loss as future work in their Appendix L, on the strength of a post-hoc observation that a channel-independent student's correlation matrix drifts toward its

teacher's; what Section 4.3 adds is an account of what such a term supplies: an objective separable across variables has no gradient term coupling two variables, and at the per-variable error the prefix holds the student at the squared loss stays constant while the correlation moves (Proposition 5), so no choice of weight in their objective, every term of which is computed per variable, produces a signal separating one correlation structure from another; whether matching the correlation lowers an error is an empirical question (Appendix C.3). These methods align activations, which ties the student to one teacher's representation, or align forecasts pointwise, which leaves implicit what structure is transferred, and they weight every horizon step or the far ones. The objectives of TRIAD (Section 4) are defined on forecasts alone. The frequency term splits the discrepancy between teacher and student forecasts on the usable window into modulus and phase and sets how a residual the student cannot remove is divided between them, and the correlation term matches the correlation among variables, which the squared loss leaves free at fixed per-variable error. Both read the teacher at an input shifted by the student's measured prefix, so what the teacher supplies is its more accurate short-lead forecast of the steps the student is scored on (Proposition 1) rather than a longer window, a privileged modality, or a tuned anchor time, and both carry weight only on the steps the device leaves usable.

Adaptive transfer weights. With several teachers, prior work decides how much to trust each one per sample: multi-teacher distillation averages teacher outputs or learns instance-level weights for them (You et al., 2017; Liu et al., 2020), confidence-aware variants weight teachers by a sample-wise reliability score (Zhang et al., 2022), and several language models are distilled into one student (Khanuja et al., 2021). Guard applies this to forecasting: a router trained against cached teacher errors weights two teachers per sample from context statistics and teacher uncertainties, and a gating network sets a per-sample distillation temperature (Dey et al., 2026). A more accurate teacher is not always a better one, and students often fail to match even the teacher they have (Mirzadeh et al., 2020; Huang et al., 2022; Menon et al., 2021; Stanton et al., 2021); transferability scores such as LEEP and LogME rank pretrained models for a target task from one forward pass over its data, without training (Nguyen et al., 2020; You et al., 2021). In all of these the unit of trust is a sample, a task, or a whole model, and in distillation the weights are learned jointly with the student. TRIAD selects one teacher per dataset from a pool by its validation error on the usable window, and uses the best multivariate teacher for the correlation term. The transfer weights are chosen on validation data before the student is trained, so the target depends on the teacher and the prefix alone and does not move as the student moves; an unreliable teacher is replaced before training rather than down-weighted during it. Routing several teachers per sample operates over teachers rather than over objectives and is orthogonal to the shift; we do not use it.

B THEORY OF LATENCY-AWARE SELECTION

B.1 LATENCY QUANTILE AND ISSUED LENGTH

We take $\mathcal{T}$ at the quantile $q = 0.95$: underestimating the latency makes forecasts arrive late, whereas overestimating it by under one interval costs at most one step of lead time. Quantiles are made nondecreasing in the issued length P by a running maximum.

Lemma 1 (Issued length)**.** *Let $\varphi(P) = \mathcal{H} + \lceil \mathcal{T}(c, P; v)/\Delta t \rceil$ with $\mathcal{T}$ nondecreasing in P, and call P admissible if $P \geq \varphi(P)$. If some P is admissible, as when each additional issued step costs at most $\beta < \Delta t$, the iteration $P_0 = \mathcal{H}$, $P_{j+1} = \varphi(P_j)$ is nondecreasing and reaches, in finitely many steps, the least admissible length $P^\star(c; v)$, which satisfies $P^\star = \varphi(P^\star)$.*

Proof. Under the increment bound, $\varphi(P) \leq \mathcal{H} + 1 + \big(\mathcal{T}(c, \mathcal{H}; v) + \beta(P - \mathcal{H})\big)/\Delta t \leq P$ for large P. For the least admissible $P^\star$, monotonicity gives $\varphi(\varphi(P^\star)) \leq \varphi(P^\star)$, so $\varphi(P^\star)$ is admissible and equals $P^\star$. By induction $P_j \leq P_{j+1} \leq P^\star$, since $\mathcal{H} \leq \varphi(\mathcal{H})$, $\mathcal{H} \leq \varphi(P^\star) = P^\star$, and φ is nondecreasing; the integer sequence therefore stops at an admissible $\bar{P} \leq P^\star$, which is $P^\star$. □

For a model deployed as it is, equation 2 is exact when its first $P^\star$ outputs do not depend on the requested length.

B.2 REWEIGHTING

Proposition 2 (Reweighting). *Let the parameters split as $\psi = (\psi_1, \dots, \psi_P)$, where ψ_h enters only the output at step h and the representation it reads is fixed, and let the loss be $\sum_h w_h \ell_h(\psi_h)$ with $w_h \geq 0$ and any per-step losses ℓ_h. For every step with $w_h > 0$, the minimizers of ψ_h do not depend on w; in particular, restricting the loss to steps $d+1$ to $d+\mathcal{H}$ leaves every used step at the minimizers of the unweighted loss.*

Proof. ψ_h appears only in the h-th term, and for $w_h > 0$ minimizing $w_h \ell_h$ is minimizing ℓ_h. □

The proof never uses the target, so it covers a teacher's forecast too; with a jointly trained representation the weights do change the fit, although a loss restricted to the usable window barely changes accuracy in trajectory prediction (Lan et al., 2026).

B.3 WHEN THE OPTIMUM IS INTERIOR

Fix the input length and v, index a family by a capacity n, and model its error curve as $e_h(n) \approx \varepsilon(n)\, g(h)$, where capacity scales the whole curve and a positive, nondecreasing g describes how error grows with lead time. With the prefix relaxed to $d(n) = \mathcal{T}(n)/\Delta t$, $E(n) = \varepsilon(n)\, \bar{g}(d(n))$ with $\bar{g}(x) = \frac{1}{\mathcal{H}} \sum_{k=1}^{\mathcal{H}} g(x+k)$, and with the offline error $O(n) = \varepsilon(n)\, \bar{g}(0)$, $E(n) = O(n)\, \bar{g}(d(n))/\bar{g}(0)$: under a shared lead-time profile, the deployment error is the offline error times a factor of the prefix. Its logarithmic derivative splits into a gain and a cost,

$$\frac{\mathrm{d}\log E}{\mathrm{d}\log n} = -\alpha + \omega(d)\,\lambda, \qquad \alpha = -\frac{\mathrm{d}\log \varepsilon}{\mathrm{d}\log n}, \quad \lambda = \frac{\mathrm{d}\log \mathcal{T}}{\mathrm{d}\log n}, \quad \omega = \frac{\mathrm{d}\log \bar{g}}{\mathrm{d}\log x}. \tag{7}$$

Proposition 3 (Interior optimum). *Let ε, $\mathcal{T}$, and g be positive and differentiable, with $\mathcal{T}$ and g nondecreasing, and assume* (A1) *α is nonincreasing,* (A2) *λ is nondecreasing, and* (A3) *ω is nondecreasing. Then $\log E$ is convex in $\log n$, and on $[n_{\min}, n_{\max}]$ a minimizer is $n_{\min}$ if $\alpha \leq \omega(d)\lambda$ at $n_{\min}$, $n_{\max}$ if $\alpha \geq \omega(d)\lambda$ at $n_{\max}$, and otherwise an interior $n^\star$ with $\alpha(n^\star) = \omega(d(n^\star))\,\lambda(n^\star)$.*

Proof. With $u = \log n$ and $F(u) = \log E(e^u) = \log \varepsilon + \log \bar{g}(d)$, the chain rule and $\log d = \log \mathcal{T} - \log \Delta t$ give equation 7. Since $\mathcal{T}$ and $\bar{g}$ are nondecreasing, $d(n)$ is nondecreasing and $\omega, \lambda \geq 0$, so by (A1)–(A3) both $-\alpha$ and $\omega(d(n))\,\lambda(n)$ are nondecreasing; hence F' is nondecreasing and F is convex. If $F' \geq 0$ at $u_{\min}$ or $F' \leq 0$ at $u_{\max}$, that endpoint is a minimizer; otherwise F' changes sign and, by the intermediate value property of derivatives, vanishes at an interior minimizer. □

(A1) holds for power-law decay toward an irreducible level (Shi et al., 2024), (A2) for a launch cost plus compute growing as a power of capacity, and (A3) for linear, power, or exponential growth of error in lead time; (A2) fails where latency jumps and (A3) once error saturates.

Remark 1 (Integer prefix). If $e_h(n)$ is nonincreasing in n for every h, then with the rounded prefix $d = \lceil \mathcal{T}(n)/\Delta t \rceil$, E is nonincreasing in n while d is constant, and when d grows by one it changes by $(e_{d+\mathcal{H}+1}(n) - e_{d+1}(n))/\mathcal{H}$, an increase whenever error grows with lead time. The largest configuration of a family that shares a prefix therefore has the lowest E, and the minimizer on a grid lies at the top of a level set of the prefix; Propositions 3 and 4 describe the relaxed objective, and since the level sets move between environments, a slower environment can still select a larger configuration.

B.4 HOW THE OPTIMUM MOVES

Proposition 4 (Environments). *Keep the relaxed prefix and the model of Appendix B.3 for one family, let* (A1) *and* (A2) *hold in environment v, and let* (A3) *hold on the prefixes of v and of another environment v'. If $\mathcal{T}(n; v') \geq \mathcal{T}(n; v)$ and $\lambda(n; v') \geq \lambda(n; v)$ for all n, as when v' is uniformly slower, $\mathcal{T}(n; v') = \sigma\, \mathcal{T}(n; v)$ with $\sigma > 1$, then $E(\cdot; v')$ has a minimizer $n^\star(v') \leq n^\star(v)$.*

Proof. With F_v and $F_{v'}$ as in the proof of Proposition 3, the gain α does not depend on the environment, and $d(n; v') \geq d(n; v)$ with (A3) gives $\omega(d(n; v')) \geq \omega(d(n; v)) \geq 0$, so with $\lambda(n; v') \geq \lambda(n; v) \geq 0$, $F'_{v'} \geq F'_v$. If $u^\star$ minimizes F_v, then $F'_v \geq 0$ on $(u^\star, u_{\max}]$, since F'_v is nondecreasing and F_v cannot fall below its minimum; so $F_{v'}$ is nondecreasing there and has a minimizer in $[u_{\min}, u^\star]$. For $\mathcal{T}(n; v') = \sigma\, \mathcal{T}(n; v)$, λ is unchanged. □

B.5 Selection and Distillation Procedure

Algorithm 1 Latency-aware selection and distillation

Require: candidates $\mathcal{F}$, environments $\mathcal{V}$, quantile q, teacher pool $\mathcal{P}$ (empty for selection alone)

1: measure the latency of each $c \in \mathcal{F}$ in each v; compute $d(c; v)$ (Lemma 1); drop it if $d(c; v) > \mathcal{H}$
2: $P_{\max} \leftarrow \mathcal{H} + \max d(c; v) + 1$; $\mathcal{K} \leftarrow$ the largest configuration per family, input length, and prefix (Remark 1)
3: train each $c \in \mathcal{K}$ once at issued length $P_{\max}$; cache its validation curve $e(c)$
4: if $\mathcal{P} \neq \emptyset$: cache each teacher's forecasts; keep the most accurate on its first $\mathcal{H}$ steps
5: **for** $v \in \mathcal{V}$ **do**
6: if $\mathcal{P} \neq \emptyset$: distill each $c \in \mathcal{K}$ on equation 6, teacher read at $s + d(c; v)$; its curve replaces $e(c)$
7: $E(c; v) \leftarrow \frac{1}{\mathcal{H}} \sum_k e_{d(c;v)+k}(c)$ for $c \in \mathcal{K}$ and models deployed as they are; $c^\star(v) \leftarrow \arg\min E(\cdot; v)$
8: **end for**
9: **return** $c^\star(v)$ with the first $P^\star(c^\star(v); v)$ rows of its read-out

The family is trained in full when larger members may not dominate, for instance when they overfit. Without distillation one run per configuration serves every environment, and distillation costs one run per configuration and environment; selection uses validation data only.

C Details of the Distillation Objectives

C.1 Proof of Proposition 1

Assume y and b square-integrable, write $y = y_{s+d+k}$ and $\mu_y(x_s) = \mathbb{E}[y \mid x_s]$, and let $\tau^{\text{use}} = \mathbb{E}[y \mid \mathcal{F}_{s+d}] + b_{s+d,k}$ and $\tau^{\text{nom}} = \mathbb{E}[y \mid \mathcal{F}_s] + b_{s,d+k}$ be the two targets on row $d + k$. Because x_s is a function of $\mathcal{F}_s \subseteq \mathcal{F}_{s+d}$, the tower property gives $\mathbb{E}\big[\mathbb{E}[y \mid \mathcal{F}_{s+d}] \mid x_s\big] = \mathbb{E}\big[\mathbb{E}[y \mid \mathcal{F}_s] \mid x_s\big] = \mu_y(x_s)$, so for either target $f^\star = \mu_y + \mathbb{E}[\beta \mid x_s]$. Then $f^\star - y = (\mu_y - y) + \mathbb{E}[\beta \mid x_s]$, and the cross term vanishes because $\mu_y - y$ has zero conditional mean given x_s, which gives the stated error. If $b \equiv 0$, both minimizers equal μ_y and $\mathbb{E}[\tau^{\text{use}} \mid \mathcal{F}_s] = \tau^{\text{nom}}$; in $\tau^{\text{use}} - \mu_y = (\tau^{\text{use}} - \tau^{\text{nom}}) + (\tau^{\text{nom}} - \mu_y)$ the cross term vanishes on conditioning on $\mathcal{F}_s$, so $\mathbb{E}[\|\tau^{\text{use}} - \mu_y\|_2^2 \mid x_s] = \mathbb{E}[\|\tau^{\text{use}} - \tau^{\text{nom}}\|_2^2 \mid x_s] + \mathbb{E}[\|\tau^{\text{nom}} - \mu_y\|_2^2 \mid x_s]$, and the same decomposition with y and $\mathcal{F}_{s+d}$ shows that an ideal teacher's target varies less about μ_y than y does. With the supervised term added, minimizing $\mathbb{E}\|f(x_s) - y\|_2^2 + \mu_{\text{kd}}\, \mathbb{E}\|f(x_s) - \tau\|_2^2$ for $\mu_{\text{kd}} > 0$ gives $f = (\mu_y + \mu_{\text{kd}} f^\star)/(1 + \mu_{\text{kd}})$, whose error on the row exceeds that of μ_y by $\big(\mu_{\text{kd}}/(1 + \mu_{\text{kd}})\big)^2\, \mathbb{E}\|\mathbb{E}[\beta \mid x_s]\|_2^2$, so the comparison between the two targets is unchanged. Neither target thus beats μ_y at the population level; a teacher target helps in finite samples, where a target that varies less than y lowers the variance of the fitted student, and for an ideal teacher the shift adds $\mathbb{E}\|\tau^{\text{use}} - \tau^{\text{nom}}\|_2^2$ to that variation. □

C.2 The Frequency Term

With $\check{Y}_j[k]$ the discrete Fourier coefficient at bin k of column j over the $\mathcal{H}$ rows of $\mathcal{W}$, Parseval's identity writes the discrepancy between the two forecasts as a sum over bins, and each term splits into a modulus part and a phase part:

$$\mathcal{D} = \frac{1}{C\mathcal{H}} \sum_{h \in \mathcal{W}} \big\|\hat{Y}^S_{h,:} - \hat{Y}^T_{h,:}\big\|_2^2 = \frac{1}{C\mathcal{H}^2} \sum_{j=1}^{C} \sum_{k=0}^{\mathcal{H}-1} \big|\check{Y}^S_j[k] - \check{Y}^T_j[k]\big|^2, \tag{8}$$

$$\big|\check{Y}^S_j[k] - \check{Y}^T_j[k]\big|^2 = \Big(\big|\check{Y}^S_j[k]\big| - \big|\check{Y}^T_j[k]\big|\Big)^2 + 2\,\big|\check{Y}^S_j[k]\big|\,\big|\check{Y}^T_j[k]\big|\big(1 - \cos\Delta\phi_j[k]\big). \tag{9}$$

The frequency term equation 4 reweights this charge, so it carries no target that $\mathcal{D}$ lacks, and four choices fix the weights. Where both moduli are large against the floor η, the phase comparison is close to $1 - \cos\Delta\phi_j[k] = 2\sin^2(\Delta\phi_j[k]/2)$, which is invariant to the wrapping of the argument, and the floor keeps it smooth at $\check{Y}^S_j[k] = 0$. The set $\mathcal{B}$ omits the zero-frequency and Nyquist

bins, where a real forecast has argument in $\{0, \pi\}$, and leaves the sign of the window mean to the supervised term. The cross term of equation 9 carries the weight $2|\check{Y}_j^S[k]||\check{Y}_j^T[k]|$, which the student could drive to zero by suppressing its own modulus, so equation 4 uses the square of the teacher's modulus, which also gives no weight to bins without teacher energy. Finally, the modulus sum counts each interior bin once where $\mathcal{D}$ counts it with its conjugate, and near convergence the phase sum carries about $\xi/4$ of the phase charge of $\mathcal{D}$, so the ratio of phase to modulus weight is $\xi/2$ times that of $\mathcal{D}$: $\xi = 2$ reproduces the division of pointwise matching at half its scale.

C.3 Cross-Variable Blindness

Fix a reference Y^{ref} on $\mathcal{W}$, the observed future for the supervised term or the teacher's forecast for pointwise matching, and write $Z = \hat{Y}^S - Y^{\mathrm{ref}}$ for the student's residual, with columns $z_1, \dots, z_C$ and per-variable errors $\|z_j\|_2^2/\mathcal{H}$.

Proposition 5 (Cross-variable blindness)**.** (i) *An objective separable across variables, $\mathcal{G} = \sum_j \Phi_j(\hat{Y}^S_{:,j})$, such as the squared loss or equation 4, has $\partial\mathcal{G}/\partial\hat{Y}^S_{h,j}$ depending on column j alone, and the squared loss depends on Z only through the per-variable errors.* (ii) *Let Σ be positive semidefinite with unit diagonal, let every per-variable error exceed $\eta^2/\mathcal{H}$, and let $\operatorname{rank}\Sigma + \operatorname{rank} Y^{\mathrm{ref}\circ} \le \mathcal{H} - 1$, with $Y^{\mathrm{ref}\circ}$ the column-centered reference. Some $\hat{Y}^S$ then attains those errors exactly, with zero-mean residual columns and $R(Z) = \Sigma$, and has $R(\hat{Y}^S)_{jl}$ strictly increasing in Σ_{jl}; so $R(\hat{Y}^S)$ is not determined by the per-variable errors whenever $\operatorname{rank} Y^{\mathrm{ref}\circ} \le \mathcal{H} - 2$, for instance when $C \le \mathcal{H} - 2$.*

Proof. (i) $\partial\mathcal{G}/\partial\hat{Y}^S_{h,j} = \partial\Phi_j/\partial\hat{Y}^S_{h,j}$ is a function of column j alone, and the squared loss is $\frac{1}{\mathcal{H}C}\sum_j \|z_j\|_2^2$. (ii) Factor $\Sigma = LL^\top$ with $L \in \mathbb{R}^{C\times r}$, $r = \operatorname{rank}\Sigma$, whose rows L_j are unit vectors. The subspace $\mathbf{1}^\perp \cap (\operatorname{span} Y^{\mathrm{ref}\circ})^\perp$ has dimension at least r, so it holds the columns of some $U \in \mathbb{R}^{\mathcal{H}\times r}$ with orthonormal columns. With $\sigma_j = (\mathcal{H} \cdot \text{per-variable error } j)^{1/2} > \eta$, set $Z_{:,j} = \sigma_j U L_j^\top$ and $\hat{Y}^S = Y^{\mathrm{ref}} + Z$. Each column of Z has zero mean and norm σ_j, which gives the prescribed errors, and with the floor inactive $R(Z)_{jl} = \langle L_j, L_l\rangle = \Sigma_{jl}$. As Z is orthogonal to the centered reference, whose columns we write y_j°, $R(\hat{Y}^S)_{jl} = (\langle y_j^\circ, y_l^\circ\rangle + \sigma_j\sigma_l\Sigma_{jl})/\big(\|y_j^\circ\|_2^2 + \sigma_j^2\big)^{1/2}\big(\|y_l^\circ\|_2^2 + \sigma_l^2\big)^{1/2}$, which is strictly increasing in Σ_{jl}; $\Sigma = ss^\top$ for $s = (1, 1, \dots, 1)$ and for $s = (1, -1, 1, \dots, 1)$ has rank one and gives two values. □

At $\mathcal{H} = C = 2$ the residuals $\left(\begin{smallmatrix}+1 & +1\\ -1 & -1\end{smallmatrix}\right)$ and $\left(\begin{smallmatrix}+1 & -1\\ -1 & +1\end{smallmatrix}\right)$ are rearrangements of one another, hence equal under every per-entry loss, yet have residual correlations $+1$ and -1. The statement concerns the loss surface at a fixed nonzero error, the level the prefix holds the student at: at zero loss the correlation is correct, and in parameter space the variables stay coupled through the shared Jacobian, although no gradient term compares two variables. When C exceeds $\mathcal{H}$, both correlation matrices are rank deficient, and equation 5 compares two noisy estimates. Matching R also does not bound the error of an aggregate such as the sum of the variables, $\frac{1}{\mathcal{H}}\|\sum_j z_j\|_2^2 = \frac{1}{\mathcal{H}}\big(\sum_j \|z_j\|_2^2 + \sum_{j\neq l}\langle z_j, z_l\rangle\big)$: when $2C < \mathcal{H}$ and the teacher's correlation equals that of the observed future, the construction of (ii) with Σ equal to that correlation and per-variable errors proportional to the variances of the variables matches it exactly, yet has a larger aggregate error than uncorrelated residuals with the same per-variable errors whenever the variables are positively correlated.

C.4 Attenuation

The usable-window target is privileged: the teacher conditions on $y_{s+1}, \dots, y_{s+d}$, which the student never reads. For a squared-error term that part averages out, since the minimizer is $\mathbb{E}[\tau \mid x_s]$ (Proposition 1), but equation 4 and equation 5 are nonlinear in $\hat{Y}^T$, so what they can attain are conditional means of the teacher's statistics, which are attenuated: off-diagonal correlations satisfy $|\mathbb{E}[R_{jl} \mid x_s]| \le \mathbb{E}[|R_{jl}| \mid x_s]$, and the phase term is minimized at the argument of $\mathbb{E}[|\check{Y}_j^T[k]|^2\, U_j^T[k] \mid x_s]$, whose modulus is below $\mathbb{E}[|\check{Y}_j^T[k]|^2\, |U_j^T[k]| \mid x_s]$ whenever the teacher's argument varies given x_s. The shift moves these targets even for an ideal teacher: with $b \equiv 0$ the nominal-window coefficient is the conditional mean of the usable-window one given $\mathcal{F}_s$, so by

Jensen's inequality the attainable modulus $\mathbb{E}[|\check{Y}_j^T[k]| \mid x_s]$ is larger for the usable-window target. Both attenuations typically grow with d, the regime in which the shift pays most, and we do not bound the gap.