
Goal-Oriented Time-Series Forecasting: Foundation Framework Design

Anonymous Author(s)

Affiliation

Address

email

Abstract

Conventional forecasting minimizes overall error but overlooks the differing importance of forecast ranges in downstream tasks. We propose a method that partitions predictions into segments and dynamically reweights them at inference, enabling flexible, on-demand targeted forecasting without retraining. Experiments on benchmarks and a new wireless dataset show improved accuracy in regions of interest and measurable downstream gains, highlighting the benefits of tighter integration between prediction and decision-making.

1 Introduction

Time-series forecasting (TSF) is a core area of machine learning (ML) with applications in economics [15], energy [26, 31], transportation [9], meteorology [39], inventory management [19], and healthcare [13, 32]. At its core, TSF builds predictive models for sequential data by leveraging historical patterns to forecast future values. Approaches range from classical statistical models, such as ARIMA [17] and ETS [6], to deep learning models, including MLPs [41], RNNs [42], LSTMs [35], TCNs [16], and Transformers [27, 20]. More recently, Large Time-Series Models (LTSMs), such as Timer [22], Moirai [37], TimesFM [11], Chronos [3], Moment [14], and Toto [10], have emerged, leveraging large-scale pretraining for zero-shot forecasting.

Most TSF methods minimize predictive error while overlooking downstream integration, where forecast errors have unequal importance. This mismatch, observed in challenges such as *Predict+Optimize for Renewable Energy Scheduling* [4] and the *M5 Competition* [25], highlights the need for models aligned with task-specific objectives. Prior work includes integrating learned weights into optimization [5], predictive loss functions tailored to optimization [12], and broader E2E learning paradigms [30].

However, existing approaches assume fixed task specifications and static regions of importance. In practice, domains such as wireless traffic prediction [40, 29] demand adaptation to shifting priorities, for example, optimizing base-station deactivation during low-traffic periods or power allocation under extremes. Since such thresholds are rarely known a priori, TSF frameworks must allow post-hoc adjustment of importance during inference (Figure 1).

Contributions. We address this gap by: (1) proposing a training methodology that enables multivariate TSF models to adapt to multiple downstream tasks at inference without retraining, (2) conducting extensive experiments on synthetic and real-world traces with comprehensive baselines and ablations, and (3) releasing a new wireless mobile network dataset to support future research.

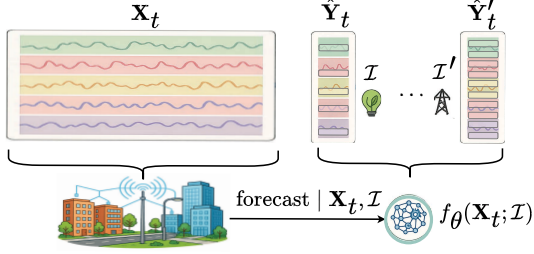


Figure 1: The figure depicts TSF in a wireless network, from data collection to training and downstream use. Applications include energy efficiency, which depends on forecasting low-traffic periods, and power allocation, requiring accuracy across traffic bands. Our method allows a single model to adapt to diverse tasks at inference, unlike traditional task-specific approaches.

2 The Forecasting Problem

Time-Series. A multivariate time-series is a temporally ordered sequence of vectors. Let $\mathbf{x}_t \in \mathbb{R}^n$ denote the n -dimensional observation at time $t \in [T]$, with $\mathbf{x}_t = (x_{t,i})_{i \in [n]}$. The series is $\{\mathbf{x}_1, \dots, \mathbf{x}_T\}$, where each component lies in a bounded set $\mathcal{X} \subset \mathbb{R}$. Temporal dependence is assumed, with \mathbf{x}_t potentially influenced by past values \mathbf{x}_{t-k} for $k > 0$.

The Learning Problem. TSF is cast as supervised regression by forming input-output pairs via sliding windows. For forecast horizon τ and window size w , the input is $\mathbf{X}_t = \mathbf{x}_{t-w:t-1} \in \mathcal{X}^{w \times n}$, and the target is $\mathbf{Y}_t = \mathbf{x}_{t:t+\tau-1} \in \mathcal{X}^{\tau \times n}$. We assume an underlying operator $f : \mathcal{X}^{w \times n} \rightarrow \mathcal{X}^{\tau \times n}$ with noisy observations $\mathbf{Y}_t = f(\mathbf{X}_t) + \epsilon_t$. A parametric model f_θ is trained on dataset $D = \{(\mathbf{X}_t, \mathbf{Y}_t)\}_{t=w}^{T-\tau}$ to minimize expected loss

$$\theta_* \in \arg \min_{\theta \in \mathbb{R}^d} \mathbb{E}_{(\mathbf{X}, \mathbf{Y}) \sim D} [l(f_\theta(\mathbf{X}), \mathbf{Y})]. \quad (1)$$

In practice, this is approximated by empirical risk with regularization $R(\theta)$ to control complexity and improve generalization [34]. Common choices of l include Mean Squared Error (MSE), estimating conditional expectations, and Mean Absolute Error (MAE), estimating conditional medians.

3 Methodology

We investigate five training policies of increasing flexibility. We begin with a baseline policy (B-Policy) corresponding to standard TSF training that ignores interval specificity, followed by a task-specific policy (E2E-Policy) that optimizes for a single target interval, representing the E2E approach when downstream tasks are known. We then introduce a continuous-interval policy (C-Policy) that incorporates intervals as covariates, enabling inference-time flexibility by exposing the model to all possible intervals during training. However, as shown in our experiments, this approach suffers from instability and yields suboptimal results, motivating more structured alternatives. Next, we consider a discretized policy (D_L -Policy) that samples from a fixed set of intervals covering the forecasting space, thereby reducing the interval search space. Since inference-time regions of interest need not align exactly with these discretized intervals, we propose a patching-augmented discretized policy (D_L^* -Policy), which combines finer discretization with a boundary-aware patching mechanism for improved adaptation.

Baseline Policy (B-Policy). Follows the standard supervised training setup in Section 2, computing loss across the full forecasting domain \mathcal{X} , without interval awareness.

Task-Specific Policy (E2E-Policy). Given a target interval $\mathcal{I} \subseteq \mathcal{X}$, this policy restricts the loss to \mathcal{I} . The model is learned by solving $\theta_* \in \arg \min_{\theta \in \mathbb{R}^d} \mathbb{E}_{(\mathbf{X}, \mathbf{Y}) \sim D} [l(f_\theta(\mathbf{X}), \mathbf{Y}) \cdot \mathbf{1}(\mathbf{Y} \in \mathcal{I}^{\tau \times n})]$.

Continuous-Interval Policy (C-Policy). To generalize across intervals, we encode \mathcal{I} as covariates (e.g., boundaries $[I_{\min}, I_{\max}] \in \mathcal{X}^2$). The model f_θ thus maps $\mathcal{X}^{w \times n+2} \rightarrow \mathcal{X}^{\tau \times n}$. Intervals are sampled uniformly from $\mathcal{U}_\delta = \{\mathcal{I} \subseteq \mathcal{X} : |\mathcal{I}| \geq \delta\}$, where δ controls minimum length of an interval. The custom loss is $L(f_\theta(\mathbf{X}_t, \mathcal{I}_t), \mathbf{Y}_t, \mathcal{I}_t) = l(f_\theta(\mathbf{X}_t, \mathcal{I}_t), \mathbf{Y}_t) \cdot \mathbf{1}(\mathbf{Y}_t \in \mathcal{I}_t^{\tau \times n})$, and training minimizes the expectation over both D and \mathcal{U}_δ . In this formulation, the model θ explicitly conditions on both the input sequence \mathbf{X}_t and the interval \mathcal{I}_t to produce the forecast $\hat{\mathbf{Y}}_t = f_\theta(\mathbf{X}_t, \mathcal{I}_t)$.

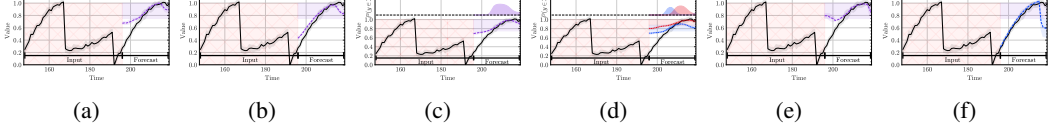


Figure 2: Subfigures (a)–(f) present the performance of the iTransformer model trained on the SynthDS trace. Each subfigure corresponds to a different policy: E2E, C, D₄, D₈, D₈¹, and B, respectively. The purple region marks the interval of interest, $\mathcal{I} = [0.75, 1.0]$. In the 1-strategy, eight intervals are patched into four. The filled region represents the classifier’s predicted probability $\mathbb{P}(y \in \mathcal{I})$. Under the D₈¹ policy, two intervals must be patched to generate a forecast for the selected interval \mathcal{I} . Black lines indicate target forecast values, while dashed lines denote the mean prediction over time, with shaded areas showing the standard deviation computed over 10 random seeds.

68 **Discretized-Interval Policy (D_L-Policy).** Instead of sampling from \mathcal{U}_δ , intervals are drawn from
69 a discrete set \mathcal{C}_L of L disjoint intervals covering \mathcal{X} , i.e., $\bigcup_{\mathcal{I} \in \text{supp}(\mathcal{C}_L)} \mathcal{I} = \mathcal{X}$.

70 **Patching-Augmented Policy (D_L^{*}-Policy).** We refine D_L-Policy by introducing two key components: (1) a *patching mechanism* that integrates forecasts from constituent intervals to enhance
71 predictions within a target interval of interest, and (2) a *soft boundary loss* (decay function) designed
72 to mitigate uncertainty near patching boundaries. A decay function modulates boundary contributions
73 $d_\nu(y, \mathcal{I}) = \exp(-\nu \max(0, |y - \Delta_{\text{avg}}| - \Delta_{\text{diff}}))$, where Δ_{avg} and Δ_{diff} are the midpoint and
74 half-length of \mathcal{I} .¹ As $\nu \rightarrow \infty$, this reduces to the indicator function. The weighted regression and
75 classification losses are
76

$$L_\nu = l(f_\theta(\mathbf{X}_t, \mathcal{I}_t), \mathbf{Y}_t) \prod_{i,t} d_\nu(y_{t,i}, \mathcal{I}_t), \quad L'_\nu = l'(f_\theta^c(\mathbf{X}_t, \mathcal{I}_t), \mathbb{1}(\mathbf{Y}_t \in \mathcal{I}_t)) \prod_{i,t} d_\nu(y_{t,i}, \mathcal{I}_t), \quad (2)$$

77 respectively. The joint training objective is then $\theta_\star = \arg \min_\theta \mathbb{E}_{(\mathbf{X}, \mathbf{Y}), \mathcal{I} \sim \mathcal{C}_L} [L_\nu + \phi \cdot L'_\nu]$, where
78 ϕ is a hyperparameter controlling the trade-off between regression and classification objectives.

79 At inference, given interval \mathcal{I} , we collect overlapping training intervals $\Xi_L(\mathcal{I}) \triangleq$
80 $\{\mathcal{I}' \in \text{supp}(\mathcal{C}_L) : \mathcal{I}' \cap \mathcal{I} \neq \emptyset\}$. Predictions are then combined by: (1) Averaging strategy (1-
81 strategy): $\hat{\mathbf{Y}} = \sum_{\mathcal{I}' \in \Xi_L(\mathcal{I})} f_\theta^c(\mathbf{X}, \mathcal{I}') f_\theta(\mathbf{X}, \mathcal{I}') (\sum_{\mathcal{I}' \in \Xi_L(\mathcal{I})} f_\theta^c(\mathbf{X}, \mathcal{I}'))^{-1}$, (2) Maximum confi-
82 dence strategy (∞ -strategy): $\hat{\mathbf{Y}} = f_\theta(\mathbf{X}, \arg \max_{\mathcal{I}' \in \Xi_L(\mathcal{I})} f_\theta^c(\mathbf{X}, \mathcal{I}'))$.

83 4 Experiments

84 **Experimental Setup.** This section reports the numerical results obtained from the proposed training
85 policies and evaluates their forecasting performance in comparison with multiple baseline methods.
86 We begin by describing the forecasting models, training policies, benchmark datasets, and training
87 configurations employed. This is followed by both qualitative and quantitative analyses of the
88 various training strategies applied to these models.² We evaluate four state-of-the-art TSF models:
89 iTransformer [21], DLinear [41], PatchTST [28], and TimeMixer [36]; further details on how
90 these architectures are augmented to incorporate interval covariates and a classification head are
91 provided in the Experimental Appendix. We consider five training policies introduced in Section 3:
92 B-Policy, E2E-Policy, C-Policy, D_L-Policy, and D_L^{*}-Policy, where the symbol \star specifies
93 the patching method ($\star = 1$ for average patching and $\star = \infty$ for maximum-confidence patching). For
94 benchmarking, we use both synthetic and real-world datasets: SynthDS (synthetic trace for controlled
95 evaluation), BLW-TrafficDS [1] (beam-level wireless traffic, released with this work), Traffic [7]
96 (road occupancy), Electricity [33] (power consumption), and Weather [38] (meteorological data).
97 We provide additional results in the Experimental Appendix on the sensitivity of different policies to
98 hyperparameters (e.g., number and length of sampled intervals, patching strategy, decay rate), as well
99 as their impact on a realistic downstream energy-saving optimization task.

¹This function is depicted in Figure 3.

²The code for all experiments is available at (Omitted for anonymity).

Models	DLinear						TimeMixer						PatchTST						iTransformer					
Policies	D _L	D _L ¹	D _L ²	B	C _{0,2}	Improve	D _L	D _L ¹	D _L ²	B	C _{0,2}	Improve	D _L	D _L ¹	D _L ²	B	C _{0,2}	Improve	D _L	D _L ¹	D _L ²	B	C _{0,2}	Improve
BLW-TrafficDS ($\times 10^3$, $L = 8$)																								
\mathcal{I}_1	167.1	172.3	164.9	163.8	218.0	0.0%	127.5	148.1	138.0	125.9	143.7	0.0%	102.8	158.2	117.8	124.1	118.1	17.2%	119.3	121.1	122.8	130.8	133.1	8.8%
\mathcal{I}_2	76.0	74.9	75.7	82.4	84.8	9.1%	40.1	37.8	41.7	83.0	50.7	54.5%	31.1	30.4	35.3	77.9	34.6	60.9%	74.7	75.9	75.7	77.3	75.6	3.4%
\mathcal{I}_3	56.7	56.0	56.7	62.0	58.8	9.7%	24.3	22.9	23.8	64.9	32.0	64.7%	18.3	17.2	20.8	60.3	22.5	71.5%	57.1	58.4	57.6	59.5	56.0	5.9%
\mathcal{I}_4	44.2	43.8	44.4	48.9	43.4	11.2%	16.1	15.4	15.8	48.1	20.8	68.0%	11.4	10.4	12.6	47.2	15.2	78.0%	44.0	44.6	44.1	46.8	42.1	10.0%
\mathcal{I}_5	33.5	33.4	33.8	37.6	31.7	15.7%	10.8	10.2	10.4	35.8	13.1	71.5%	7.1	6.5	7.9	34.9	9.8	81.4%	31.4	32.1	31.9	35.8	28.7	19.2%
\mathcal{I}_6	28.1	27.9	28.2	31.8	26.4	16.9%	8.0	7.8	8.0	30.2	9.3	74.2%	4.5	4.0	5.0	29.4	6.8	86.3%	25.2	25.7	25.7	29.8	23.3	21.8%
\mathcal{I}_7	19.3	19.2	19.4	21.7	18.8	13.4%	5.5	5.5	5.6	21.8	6.1	74.8%	2.7	2.4	2.8	20.4	3.9	88.2%	19.2	19.5	19.5	20.6	18.1	12.1%
\mathcal{I}_8	11.1	11.1	11.2	12.5	10.9	12.8%	3.3	3.5	3.4	12.1	3.6	72.7%	1.7	1.5	1.8	11.7	2.6	87.2%	11.2	11.2	11.2	11.8	11.4	5.1%
$\overline{\mathcal{I}_1-\mathcal{I}_8}$	54.0	54.8	54.3	57.6	61.6	6.3%	29.4	31.4	30.9	52.7	34.9	44.2%	22.4	28.8	25.5	50.7	26.7	55.8%	47.8	48.6	48.6	51.5	48.5	7.2%
Traffic ($\times 500$, $L = 4$)																								
\mathcal{I}_1	1.70	1.53	1.67	1.79	1.82	14.5%	1.90	1.71	1.73	2.44	2.17	29.9%	0.91	1.02	1.06	1.41	1.11	27.7%	1.29	1.30	1.32	1.41	1.36	8.5%
\mathcal{I}_2	1.65	1.35	1.44	1.99	1.71	32.2%	1.25	1.23	1.27	2.02	1.52	39.2%	0.91	0.90	0.93	1.42	0.89	37.3%	1.19	1.22	1.24	1.30	1.13	20.7%
\mathcal{I}_3	0.61	0.44	0.47	0.81	0.62	45.7%	0.39	0.36	0.41	0.74	0.51	51.5%	0.32	0.30	0.35	0.65	0.30	53.9%	0.53	0.54	0.55	0.66	0.53	19.7%
\mathcal{I}_4	0.45	0.31	0.33	0.59	0.45	47.5%	0.22	0.20	0.24	0.59	0.35	66.2%	0.16	0.16	0.19	0.53	0.16	69.8%	0.35	0.36	0.37	0.51	0.35	31.4%
$\overline{\mathcal{I}_1-\mathcal{I}_4}$	1.10	0.90	0.98	1.29	1.15	30.2%	0.94	0.88	0.91	1.45	1.14	39.3%	0.62	0.59	0.63	1.01	0.62	41.6%	0.84	0.85	0.87	1.02	0.84	17.6%
Weather ($\times 2$, $L = 4$)																								
\mathcal{I}_1	1.05	0.76	0.75	0.78	1.32	3.9%	2.25	0.73	0.71	1.00	1.94	27.0%	0.40	0.67	0.53	0.65	1.17	38.5%	0.49	0.48	0.49	0.68	1.72	29.4%
\mathcal{I}_2	0.36	0.23	0.29	0.42	0.36	45.2%	0.37	0.22	0.26	0.54	0.43	59.3%	0.22	0.23	0.28	0.37	0.22	40.5%	0.32	0.32	0.32	0.42	0.49	23.8%
\mathcal{I}_3	0.13	0.09	0.10	0.20	0.26	55.0%	0.19	0.08	0.09	0.26	0.25	69.2%	0.08	0.09	0.11	0.22	0.08	63.6%	0.18	0.17	0.18	0.23	0.39	26.1%
\mathcal{I}_4	0.16	0.14	0.21	0.21	0.27	33.3%	0.17	0.12	0.13	0.27	0.26	55.6%	0.10	0.13	0.14	0.23	0.11	56.5%	0.19	0.19	0.19	0.21	0.38	9.5%
$\overline{\mathcal{I}_1-\mathcal{I}_4}$	0.43	0.31	0.34	0.40	0.55	22.5%	0.75	0.29	0.30	0.52	0.72	44.2%	0.20	0.28	0.26	0.37	0.40	46.0%	0.30	0.29	0.29	0.38	0.74	23.7%
Electricity ($\times 10^{-1}$, $L = 4$)																								
\mathcal{I}_1	3.81	3.82	3.82	3.77	3.95	0.0%	3.83	3.95	3.95	4.50	4.86	14.9%	3.33	4.66	4.69	3.32	3.78	0.0%	12.8	10.3	10.5	3.26	3.67	0.0%
\mathcal{I}_2	1.15	1.16	1.16	1.17	1.14	2.23%	1.11	1.07	1.09	1.41	1.37	24.3%	0.919	1.24	1.34	1.01	0.990	9.37%	4.67	3.69	3.84	1.05	1.02	2.48%
\mathcal{I}_3	0.916	0.929	0.932	0.955	0.890	6.81%	0.807	0.782	0.836	1.08	0.984	27.6%	0.671	0.852	0.931	0.835	0.689	19.6%	2.08	1.84	1.98	0.861	0.822	4.53%
\mathcal{I}_4	0.369	0.379	0.380	0.399	0.353	7.52%	0.297	0.277	0.288	0.449	0.388	38.3%	0.238	0.248	0.272	0.370	0.243	35.7%	0.567	0.509	0.522	0.372	0.322	13.4%
$\overline{\mathcal{I}_1-\mathcal{I}_4}$	1.56	1.57	1.57	1.57	1.58	0.7%	1.51	1.52	1.54	1.86	1.90	18.8%	1.29	1.75	1.81	1.39	1.42	6.93%	5.02	4.09	4.21	1.39	1.46	0.0%
SynthDS ($\times 10^3$, $L = 4$)																								
\mathcal{I}_1	15.0	14.5	18.5	50.9	24.8	71.5%	16.2	15.7	17.9	45.5	17.5	65.5%	13.2	13.2	15.1	37.9	15.3	65.2%	14.1	14.1	17.5	40.2	20.0	65.0%
\mathcal{I}_2	11.7	11.9	12.8	20.3	12.8	42.4%	11.4	12.2	15.0	18.8	13.5	39.4%	9.08	10.2	11.2	26.8	11.0	66.2%	7.96	9.17	9.65	26.2	10.6	69.6%
\mathcal{I}_3	11.7	11.2	10.7	17.1	10.6	38.1%	11.5	11.7	13.3	13.9	10.8	22.4%	7.74	8.61	8.92	14.9	9.19	48.0%	7.97	8.38	8.63	16.2	8.71	50.9%
\mathcal{I}_4	16.8	16.8	18.2	41.0	20.3	59.1%	15.5	17.3	19.7	34.9	16.6	55.5%	12.7	12.9	13.3	36.8	13.7	65.6%	13.1	12.4	13.6	33.7	13.4	63.2%
$\overline{\mathcal{I}_1-\mathcal{I}_4}$	13.8	13.6	15.1	32.3	17.1	57.9%	13.7	14.2	16.5	28.3	14.6	51.7%	10.7	11.2	12.1	29.1	12.3	63.3%	10.8	11.0	12.3	29.1	13.2	62.9%

Table 1: MAE (\downarrow) and relative improvement (\uparrow) over the baseline for DLinear, TimeMixer, iTransformer, and PatchTST under five policies. MAE values are rescaled per dataset ($\times 10^3$ for SynthDS/BLW-TrafficDS, $\times 500$ for Traffic, $\times 2$ for Weather, $\times 10^{-1}$ for Electricity). For BLW-TrafficDS, patching uses $L = 8, 16$ intervals. Bold indicates best per model; underline marks overall best.

Qualitative Comparison. We begin with the SynthDS trace to illustrate how training policies adapt to intervals of interest. *Baseline:* Training with B-Policy yields averaged forecasts centered around the global mean (Fig. 4b), showing no adaptation to interval-specific distributions. *End-to-End:* Using E2E-Policy with $\mathcal{I} = [0.75, 1]$ (Fig. 2a) demonstrates best-case performance: forecasts align with the interval-specific signal but require retraining per interval. *Continuous Exploration:* Under C-Policy, the model must map all possible $\mathcal{I} \subset \mathcal{X}$ to their hypotheses. As shown in Fig. 2b, this approach underfits; restricting interval length ($|\mathcal{I}| \geq \delta$) improves results but introduces bias if δ is too large (Fig. 9). *Discrete Sampling:* With D_L -Policy, the model trains on a finite set of intervals (Fig. 2c). Performance matches E2E-Policy while retaining inference-time adaptability. However, it assumes prior knowledge of relevant intervals. *Patching:* To relax this assumption, we overparameterize the interval set and apply patching at inference (Figs. 2d, 2e). Despite slight degradation relative to D_L -Policy, patched models still recover the underlying hypotheses and adapt flexibly to unseen intervals, motivating their use in foundation TSF settings.

Quantitative Comparison. Table 1 reports MAE across datasets and models. As expected, B-Policy performs worst, while C-Policy yields modest gains but struggles to separate hypotheses. D_L -Policy improves markedly when its intervals match ground truth, though this assumption is unrealistic in practice. D_L^* -Policy maintains comparable accuracy while enabling broader adaptability, and in some cases even surpasses the discrete baseline due to ensemble-like effects. These trends extend to real-world datasets, with stronger models (iTransformer, PatchTST) benefiting most, while weaker ones (DLinear) show less consistent improvements. Overall, patching provides a practical compromise between performance and flexibility.

5 Conclusion

This work introduces a training methodology that extends existing TSF models into foundation-style architectures capable of inference-time adaptation to diverse downstream tasks. Extensive empirical evaluation confirms the effectiveness of the proposed approach. Future directions include scaling to broader domains and adapting large pre-trained TSF models [22, 37, 11, 3, 14, 10] for arbitrary-interval forecasting.

References

- [1] Omitted for anonymity. NA.
- [2] 3GPP. Evolved universal terrestrial radio access (e-utra); potential solutions for energy saving for e-utran (study on energy saving enhancement for e-utran). Technical Report TR 36.927 (v12.0.0, Release 12), 3GPP / ETSI, September 2014.
- [3] Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, et al. Chronos: Learning the language of time series. 2024.
- [4] Christoph Bergmeir, Frits de Nijs, Abishek Sriramulu, Mahdi Abolghasemi, Richard Bean, John Betts, Quang Bui, Nam Trong Dinh, Nils Einecke, Rasul Esmaeilbeigi, et al. Comparison and evaluation of methods for a predict+ optimize problem in renewable energy. *arXiv preprint arXiv:2212.10723*, 2022.
- [5] Dimitris Bertsimas and Nathan Kallus. From predictive to prescriptive analytics. *Management Science*, 66(3):1025–1044, 2020.
- [6] Robert G Brown. *Exponential smoothing for predicting demand*. Little, 1956.
- [7] Caltrans. Pems traffic data. <http://pems.dot.ca.gov>, 2016.
- [8] Haluk Celebi, Yavuz Yapıcı, Ismail Güvenç, and Henning Schulzrinne. Load-based on/off scheduling for energy-efficient delay-tolerant 5g networks. *IEEE Transactions on Green Communications and Networking*, 3(4):955–970, 2019.
- [9] Chao Chen, Karl Petty, Alexander Skabardonis, Pravin Varaiya, and Zhanfeng Jia. Freeway performance measurement system: mining loop detector data. *Transportation research record*, 1748(1):96–102, 2001.
- [10] Ben Cohen, Emaad Khwaja, Kan Wang, Charles Masson, Elise Ramé, Youssef Doubli, and Othmane Abou-Amal. Toto: Time series optimized transformer for observability. *arXiv preprint arXiv:2407.07874*, 2024.
- [11] Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. In *Forty-first International Conference on Machine Learning*, 2024.
- [12] Adam N Elmachetoub and Paul Grigas. Smart “predict, then optimize”. *Management Science*, 68(1):9–26, 2022.
- [13] Marzyeh Ghassemi, Marco Pimentel, Tristan Naumann, Thomas Brennan, David Clifton, Peter Szolovits, and Mengling Feng. A multivariate timeseries modeling approach to severity of illness assessment and forecasting in icu with sparse, heterogeneous clinical data. In *Proceedings of the AAAI conference on artificial intelligence*, volume 29, 2015.
- [14] Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. Moment: A family of open time-series foundation models. *arXiv preprint arXiv:2402.03885*, 2024.
- [15] Clive William John Granger and Paul Newbold. *Forecasting economic time series*. Academic press, 2014.
- [16] Yangdong He and Jiabao Zhao. Temporal convolutional networks for anomaly detection in time series. In *Journal of Physics: Conference Series*, volume 1213, page 042050. IOP Publishing, 2019.
- [17] Rob J Hyndman and George Athanasopoulos. 8.9 seasonal arima models. *Forecasting: principles and practice. oTexts*. Retrieved, 19, 2015.
- [18] Hyungyu Ju, Seungnyun Kim, Youngjoon Kim, and Byonghyo Shim. Energy-efficient ultra-dense network with deep reinforcement learning. *IEEE Transactions on Wireless Communications*, 21(8):6539–6552, 2022.

- [19] Na Li, Donald M Arnold, Douglas G Down, Rebecca Barty, John Blake, Fei Chiang, Tom Courtney, Marianne Waito, Rick Trifunov, and Nancy M Heddle. From demand forecasting to inventory ordering decisions for red blood cells through integrating machine learning, statistical modeling, and inventory optimization. *Transfusion*, 62(1):87–99, 2022.
- [20] Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. *arXiv preprint arXiv:2310.06625*, 2023.
- [21] Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting, 2024.
- [22] Yong Liu, Haoran Zhang, Chenyu Li, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Timer: Transformers for time series analysis at scale. *arXiv e-prints*, pages arXiv–2402, 2024.
- [23] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.
- [24] J Xavier Salvat Lozano, Jose A Ayala-Romero, Andres Garcia-Saavedra, and Xavier Costa-Perez. Kairos: Energy-efficient radio unit control for o-ran via advanced sleep modes. In *IEEE INFOCOM 2025-IEEE Conference on Computer Communications*, pages 1–10. IEEE, 2025.
- [25] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. M5 accuracy competition: Results, findings, and conclusions. *International Journal of Forecasting*, 38(4):1346–1364, 2022.
- [26] Luis Martín, Luis F Zarzalejo, Jesus Polo, Ana Navarro, Ruth Marchante, and Marco Cony. Prediction of global solar irradiance based on time series analysis: Application to solar thermal power plants energy production planning. *Solar Energy*, 84(10):1772–1781, 2010.
- [27] Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. *arXiv preprint arXiv:2211.14730*, 2022.
- [28] Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers, 2023.
- [29] Shuyu Ping, Adnan Aijaz, Oliver Holland, and A. Hamid Aghvami. Green cellular access network operation through dynamic spectrum and traffic load management. In *2013 IEEE 24th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pages 2791–2796, 2013.
- [30] Meng Qi, Yuanyuan Shi, Yongzhi Qi, Chenxin Ma, Rong Yuan, Di Wu, and Zuo-Jun Shen. A practical end-to-end inventory management model with deep learning. *Management Science*, 69(2):759–773, 2023.
- [31] Zheng Qian, Yan Pei, Hamidreza Zareipour, and Niya Chen. A review and discussion of decomposition-based hybrid models for wind energy forecasting applications. *Applied energy*, 235:939–953, 2019.
- [32] Evan L Ray, Yijin Wang, Russell D Wolfinger, and Nicholas G Reich. Flusion: Integrating multiple data sources for accurate influenza predictions. *Epidemics*, 50:100810, 2025.
- [33] UCI Machine Learning Repository. Electricityloadaddiagrams20112014. <https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014>, 2014.
- [34] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.
- [35] Sima Siami-Namini, Neda Tavakoli, and Akbar Siami Namin. The performance of lstm and bilstm in forecasting time series. In *2019 IEEE International conference on big data (Big Data)*, pages 3285–3292. IEEE, 2019.
- [36] 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, 2024.

- 221 [37] Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo.
222 Unified training of universal time series forecasting transformers. 2024.
- 223 [38] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition trans-
224 formers with auto-correlation for long-term series forecasting. *Advances in neural information*
225 *processing systems*, 34:22419–22430, 2021.
- 226 [39] Haixu Wu, Hang Zhou, Mingsheng Long, and Jianmin Wang. Interpretable weather forecasting
227 for worldwide stations with a unified deep model. *Nature Machine Intelligence*, 5(6):602–611,
228 2023.
- 229 [40] Qiong Wu, Xu Chen, Zhi Zhou, Liang Chen, and Junshan Zhang. Deep reinforcement learning
230 with spatio-temporal traffic forecasting for data-driven base station sleep control. *IEEE/ACM*
231 *transactions on networking*, 29(2):935–948, 2021.
- 232 [41] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series
233 forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages
234 11121–11128, 2023.
- 235 [42] Jun Zhang and Kim-Fung Man. Time series prediction using rnn in multi-dimension embedding
236 phase space. In *SMC’98 conference proceedings. 1998 IEEE international conference on*
237 *systems, man, and cybernetics (cat. no. 98CH36218)*, volume 2, pages 1868–1873. IEEE, 1998.

A Detailed Experimental Setup

Time-Series Forecasting Models. We evaluate four state-of-the-art TSF models: iTransformer [21], DLinear [41], PatchTST [28], and TimeMixer [36]. To incorporate target interval information \mathcal{I} , we modify each architecture accordingly. Given an interval of interest $\mathcal{I} \subseteq \mathcal{X}$, the vectorized form of \mathcal{I} is concatenated with the temporal encoding in iTransformer, while for DLinear, PatchTST, and TimeMixer, it is introduced as two additional temporal channels. We further extend these regression models to support a dual-task objective—forecasting and classification—by adding a classification head. This is implemented by doubling the output dimension of the final projection layer (equal to the forecasting horizon τ) and partitioning it into regression outputs (first τ dimensions) and classification logits (remaining τ dimensions). This enables the models to both predict future values and estimate the probability that these forecasts fall within the target interval \mathcal{I} .

Benchmarking Datasets. In our experimental evaluation, we employ several benchmark datasets to substantiate our claims:

1. SynthDS is a synthetic dataset designed to highlight the various components of our proposed methodology. The trace is constructed by combining an input signal

$$\left(\sin\left(\frac{\pi n}{2D}\right)\right)_{n \in [w]}, \quad (3)$$

where $w = 24$ denotes the signal length, with an output signal selected uniformly at random (u.a.r.) from the set

$$\left\{\frac{1}{4}\left(\sin\left(\frac{\pi}{2w}n + \frac{\pi}{2}\right) + k\right)_{n \in [w]} : k \in [4]\right\}. \quad (4)$$

To produce a trace spanning $T = 3.1 \times 10^3$ timesteps, the same randomly selected signal is concatenated multiple times. Gaussian noise with mean and standard deviation 0.05 is then added to the trace. A noise-free version, without the addition of Gaussian noise, is shown in Figure 4a.

2. BLW-TrafficDS [1] is released to the public domain as part of this study. We consider a subset of the DLPRB modality, which contains wireless beam-level measurements for 100 beams over 10^3 timesteps.
3. Traffic [7] contains hourly road occupancy rates (ranging from 0 to 1) collected by sensors on San Francisco Bay Area freeways between 2015 and 2016, covering a total of 48 months.
4. Electricity [33] records hourly electricity consumption (in kWh) for 321 clients from 2012 to 2014.
5. Weather [38] contains 21 meteorological variables, including air temperature and humidity, recorded every 10 minutes throughout 2020.

Training Configuration. To ensure the reproducibility of our experiments, we provide a comprehensive description of the training setup. The datasets were divided into training, validation, and testing subsets. A 66-17-17 split was used for the SynthDS, Traffic, Electricity, and Weather datasets, while the BLW-TrafficDS dataset used a 70-10-20 split. Four state-of-the-art time series forecasting models—iTransformer, DLinear, PatchTST, and TimeMixer—were evaluated across the designated tasks.

The BLW-TrafficDS, Traffic, Electricity, and Weather datasets were processed using a multivariate-to-multivariate configuration, whereas the SynthDS dataset used a univariate-to-univariate setup. Input sequence lengths and forecasting horizons were tailored to each dataset: 48/24 for SynthDS, 96/24 for BLW-TrafficDS, and 168/48 for Traffic, Electricity, and Weather. The number of input channels was cropped to 100 for all datasets except Weather, which used 21 channels due to its limited dimensionality.

Model configurations across all experiments included 3 layers with a model dimension of 256. TimeMixer was configured with a channel dimension of 100. All models were trained with a batch size of 32 for 50 epochs. The AdamW optimizer [23] was used with an initial learning rate of 10^{-3} and a cosine annealing schedule with $\eta_{\min} = 10^{-5}$, defined as: $\eta_t = \eta_{\min} + 0.5(\eta_{\max} - \eta_{\min})(1 + \cos(\pi \cdot t/n_{\text{epochs}}))$.

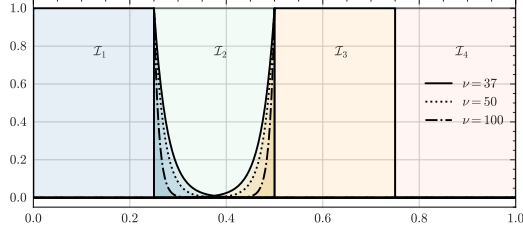
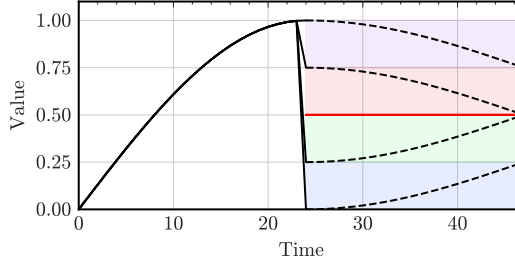
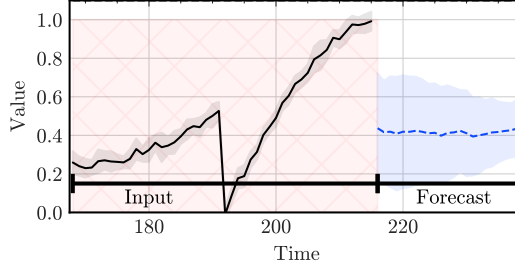


Figure 3: Impact of decay on the overlap of four intervals spanning the range $[0, 1]$. The figure illustrates the weighting associated with the intervals $\mathcal{I}_1 = [0, 0.25]$ and \mathcal{I}_3 for various decay rates $\nu \in \{37, 50, 100\}$. Note how increasing the decay rate reduces the overlap between the intervals. For $\nu = 37$, the weight value reaches 1% at the midpoint of the adjacent interval.



(a) Noise-free hypotheses in SynthDS



(b) Averaging tendencies of B-Policy

Figure 4: Subfigure (a) depicts the expected hypotheses used to construct the synthetic trace SynthDS. Subfigure (b) depicts the inability of the baseline policy B-Policy to distinguish between the different patterns in SynthDS dataset, as it predicts the average hypothesis (red line at 0.5). The model used is a trained iTransformer. Each subplot shows a time-shift of six steps. The black dotted line indicates the true values. The blue line shows the model's predictions, over 10 random seeds.

285 The MAE loss was used for regression tasks, while Binary Cross Entropy loss was applied for
 286 classification tasks when applicable. Early stopping with a patience of 5 epochs and checkpoint
 287 saving mechanisms were employed to reduce overfitting. Validation loss was computed as the average
 288 loss over each training interval.

289 The D_L^* -Policy was evaluated using 4 and 8 intervals for the SynthDS, Traffic, Electricity,
 290 and Weather datasets, and 8 and 16 intervals for the BLW-TrafficDS dataset. Interval endpoint
 291 sampling was performed per sample within each batch and incorporated into the forward pass as
 292 detailed in Section 3.

293 The dataset sizes are as follows: SynthDS—3,456 points, BLW-TrafficDS—1,025 points, Weather—
 294 52,696 points, Traffic—17,544 points, and Electricity—26,304 points. To set forecasting
 295 boundaries, the maximum values were assigned as $\max(\mathcal{X}) \in \{0.2, 500.0, 10000.0\}$ for Traffic,
 296 Weather, and Electricity, respectively. All experiments were executed on a system with 6 Tesla
 297 V100/16GB GPUs and an Intel(R) Xeon(R) Platinum 8164 CPU (104 cores).

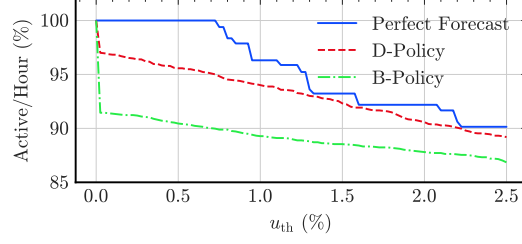


Figure 5: The impact of forecasting accuracy on wireless network energy saving as downstream application.

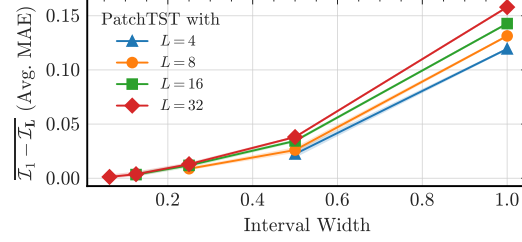


Figure 6: Comparison of the performance of the D_4 , D_8 , D_{16} and D_{32} policies with a fixed decay rate for the PatchTST model. The MAE is averaged across the different intervals.

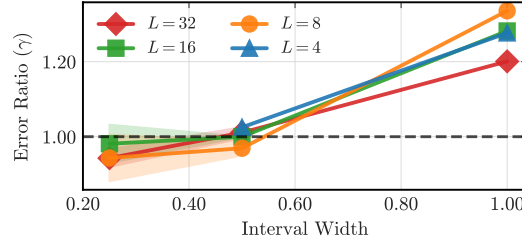


Figure 7: Comparison of the error ratio γ across discretizations. Values $\gamma < 1$ indicate that the ∞ -strategy performs better, while $\gamma > 1$ favors the 1-strategy.

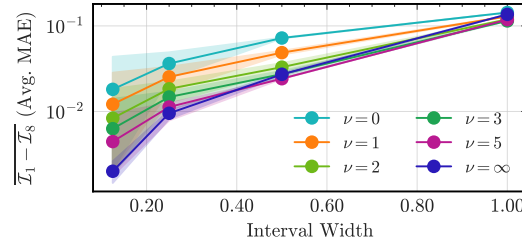


Figure 8: Comparison of the performance of the different decay rates for the PatchTST model trained with the D_8 -Policy.

298 B Implications for Downstream Task: Forecasting for Energy-Saving

299 We consider a heterogeneous wireless network comprising two tiers: a *capacity cell* (e.g., a small
300 or micro cell) providing high-throughput service within a localized area, and an overlaid *coverage*
301 *cell* (e.g., a macro cell) ensuring broad-area connectivity. Two-tier architectures with dynamic small
302 cell control have emerged as a key strategy for balancing spectral efficiency and energy savings in
303 5G and beyond [8, 18, 40, 24, 2]. In particular, recent work has shown that monitoring DLPRB
304 utilization provides an effective signal for real-time sleep mode activation in energy-aware networks.
305 We assume that the capacity cell can be selectively deactivated to reduce energy consumption when

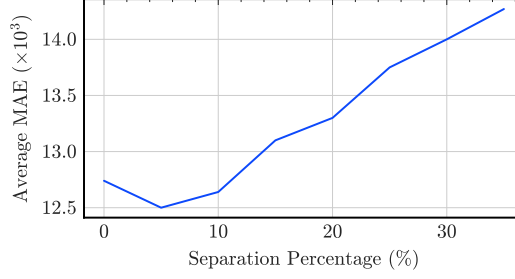


Figure 9: Interval-separation percentage (δ) impact on C-Policy.

306 traffic demand is low. To govern this behavior, we adopt a threshold-based energy-saving policy that
 307 operates based on the observed DLPRB utilization. This mirrors widely-used control strategies that
 308 compare traffic load against fixed or learned thresholds to trigger cell activation or sleep [8, 18].

309 Let $u(t) \in [0, 1]$ denote the normalized DLPRB utilization at discrete time index $t \in \mathbb{Z}_{\geq 0}$, and let
 310 $u_{\text{th}} \in [0, 1]$ represent a fixed utilization threshold. The binary state variable $S(t) \in \{0, 1\}$ indicates
 311 whether the capacity cell is active ($S(t) = 1$) or deactivated ($S(t) = 0$). The policy is defined as:

$$S(t) = \begin{cases} 1, & \text{if } u(t) \geq u_{\text{th}}, \\ 0, & \text{otherwise.} \end{cases} \quad (5)$$

312 The instantaneous cell traffic load, denoted $L(t)$, is computed as the product of DLPRB utilization
 313 and the maximum throughput capacity of the capacity cell:

$$L(t) \triangleq u(t) \cdot C_{\text{cap}}, \quad (6)$$

314 where $C_{\text{cap}} > 0$ is the peak service rate (in Mbps) of the capacity cell.

315 **Realized Throughput.** The realized throughput, $R(t)$, is constrained by the physical limits of
 316 the serving infrastructure. When the capacity cell is active, it serves the traffic directly, up to its
 317 maximum capacity. If deactivated, the traffic is offloaded to the coverage cell, subject to its own
 318 capacity constraint and a possible degradation in service quality. This is formalized as:

$$R(t) = \begin{cases} \min(L(t), C_{\text{cap}}), & \text{if } S(t) = 1, \\ \alpha \cdot \min(L(t), C_{\text{cov}}), & \text{if } S(t) = 0, \end{cases} \quad (7)$$

319 where $C_{\text{cov}} > 0$ denotes the capacity of the coverage cell and $\alpha \in (0, 1]$ is a degradation factor that
 320 accounts for reduced performance when traffic is offloaded.

321 **Energy Consumption.** To evaluate the energy-performance trade-off, we associate an energy cost
 322 with each state. Let E_{on} and E_{off} represent the per-time-unit energy consumption when the capacity
 323 cell is active or inactive, respectively. The total energy consumption at time t is then:

$$E(t) = S(t) \cdot E_{\text{on}} + (1 - S(t)) \cdot E_{\text{off}}. \quad (8)$$

324 Over a time horizon of T intervals, the average throughput and average energy consumption are given
 325 by:

$$\bar{R} = \frac{1}{T} \sum_{t=1}^T R(t), \quad \bar{E} = \frac{1}{T} \sum_{t=1}^T E(t). \quad (9)$$

326 This model captures the core dynamics of a load-aware energy-saving policy in a two-tier network,
 327 balancing energy efficiency with service quality under realistic physical constraints.

328 **Downstream Application Optimization Problem.** We consider the following optimization prob-
 329 lem that balances average throughput and energy consumption:

$$\max_{u_{\text{th}} \in [0, 1]} (1 - \lambda) \bar{R}(u_{\text{th}}) - \lambda \bar{E}(u_{\text{th}}) \quad (10)$$

330 where $\lambda \in [0, 1]$ is a scalar trade-off parameter that governs the relative importance of energy
 331 efficiency versus throughput performance. Let $u_{\text{th}}^*(\lambda)$ denote the optimal threshold that solves
 332 problem (10) for a given value of λ .

Instantiation and Numerical Results. We instantiate the problem using representative parameter values from prior work [8, 18]. In particular, the capacity cell peak throughput was set to $C_{\text{cap}} = 100$ Mbps, while the coverage cell peak throughput was $C_{\text{cov}} = 30$ Mbps. An offloading degradation factor of $\alpha = 0.5$ was applied to model performance reduction during traffic redirection. The energy consumption of a capacity cell was configured as $E_{\text{on}} = 1266$ Wh when active and $E_{\text{off}} = 320$ Wh when inactive. These parameters define the operational conditions under which forecasting models were evaluated.

Specifically, we train the iTransformer model on the BLW-TrafficDS dataset under two distinct policies: the B-Policy and the D_L^* -Policy ($L = 4$ intervals), with a forecasting horizon length of $H = 24$, with a test set rolled over 96 hours. We then evaluate and compare the performance of these policies. Since the threshold of interest is not known *a priori*, we deliberately select a slightly broader interval, $[0, 0.5]$, to ensure coverage; this involves patching together two intervals for the D_L^* -Policy. Based on the generated forecasts, we fix a threshold value and observe the resulting sequence of decisions, denoted as $S(t)$ for $t \in [T]$, where T is the length of the forecast horizon. We analyze the discrepancies in sleep durations induced by the forecasts under the different policies and restrict our attention to a specific range of thresholds: $U_{\text{th}} \in [0, 0.025]$.

As shown in Figure 5, the task-specific policy corresponding to a selected threshold yields decisions that are more closely aligned with the optimal policy—that is, the policy a decision-maker would follow with perfect foresight of future DLPRB values. This alignment demonstrates that the task-specific forecasts lead to better downstream optimization performance. Quantitatively, the task-specific policy produces forecasts that reduce the sleep duration error by a factor of three ($\times 3$), corresponding to an average reduction of one hour in sleep duration error per day. This improvement translates into an energy saving error of only 337 watts, in contrast to a significantly higher mismatch of 0.950 kilowatts observed under the baseline policy per day.

C Hyperparameter Sensitivity Analysis

We conducted experiments on the specific hyperparameters introduced by our work, namely the number of intervals L in which we divided the time series domain \mathcal{X} , the decay rate $\nu \in [0, \infty]$ and the patching strategies (1-strategy and ∞ -strategy).

Interval Granularity (L). Among the hyperparameters introduced, the number of intervals L plays the most critical role. It determines the granularity of the discretization over the time series domain \mathcal{X} , directly influencing both model complexity and performance. Figure 6 illustrates the effect of varying L on model performance. To assess this, we trained the PatchTST model on the SynthDS, under four instances of D_L^* -Policy. Each policy corresponds to a partitioning of the domain \mathcal{X} into $L \in \{4, 8, 16, 32\}$ intervals, with a fixed decay rate ν across all runs. For evaluation, we measured the MAE over coarser partitions obtained via powers-of-two binning. The results demonstrate a clear trade-off: increasing the number of intervals improves granularity but may lead to decreased accuracy at coarser levels due to over-patching. This underscores the importance of selecting an appropriate value of L to balance detail sensitivity with robustness across scales. For practical applications, a preliminary sweep over L is recommended to identify the configuration that best controls patching frequency while maintaining predictive accuracy.

Patching Strategies Comparison (\star). To systematically investigate the performance trade-off between the 1-strategy and the ∞ -strategy, we formulated an evaluation strategy centered on the MAE across varying interval lengths. For this purpose, the PatchTST model, one of the best models from our empirical studies, was trained on the noiseless SynthDS dataset under four distinct policies: D_4 , D_8 , D_{16} , and D_{32} . A comparative evaluation of the 1-strategy and ∞ -strategy was then conducted for each policy. This evaluation was performed on partitions of the codomain of the time-series where the total number of intervals corresponded to a power of two. Figure 7 presents a comparative analysis of the two patching mechanisms where we divided the MAE of the ∞ -strategy by the MAE of the 1-strategy and we observed the results over different intervals length.

We observe that for smaller intervals where the hypotheses are well-defined and there is no need for averaging, the ∞ -strategy outperforms the 1-strategy. As the intervals are getting larger, the need for averaging grows, thus making the 1-strategy better.

385 This analysis shows that for a dataset, when performing inference on small intervals or intervals with
 386 clearly defined hypotheses, the ∞ -strategy is needed, while for larger intervals on which the B-Policy
 387 would average, the 1-strategy is needed.

388 **Decay Rate Analysis (ν).** In the same context as the experiment involving the number of intervals,
 389 we conducted an experiment related to the behaviors of the decay rate. We employed the D_{32} policy
 390 and studied decay rates in the set $\{0, 1, 2, 5, \infty\}$. The results of the experiment can be seen in Figure 8.

391 We observe that the decay rate ν plays a significant role in enhancing forecasting accuracy, particularly
 392 in scenarios where substantial patching is required, i.e., when the interval length is large. In the
 393 extreme case where all 32 training intervals are aggregated into a single target interval (resulting in
 394 an interval length of 1), the best performance is achieved when $\nu = \infty$, corresponding to a no-decay
 395 setting. However, as the intervals of interest become smaller, corresponding to finer-grained target
 396 partitions, we observe improved performance when a finite decay rate is applied. In these cases,
 397 the decay mechanism effectively introduces a soft overlap between training intervals, promoting
 398 smoother generalization across neighboring regions of the domain. This highlights the importance of
 399 tuning ν based on the level of aggregation or granularity used in the target interval structure.

400 **Interval Separation (δ).** As introduced in Section 3, the C-Policy incorporates a hyperparameter,
 401 the interval separation δ . In this section, we investigate its effect using the iTransformer model trained
 402 on the SynthDS trace. Specifically, we vary δ within the range $[0, 0.4]$ over the domain $\mathcal{X} = [0, 1]$.

403 Figure 9 illustrates the effect of constraining the sampling space of training intervals by enforcing
 404 a minimum separation constraint $\mathcal{I} \geq \delta$. Reducing the sampling space initially yields improved
 405 performance; however, beyond a critical threshold $\delta' \approx 5\%$, performance degrades as the model tends
 406 to overestimate true interval lengths, thereby introducing bias. It is worth noting that in SynthDS, the
 407 ground-truth hypotheses are separated by intervals of length $1/4$.