Goal-Oriented Time-Series Forecasting: Foundation Framework Design

Luca-Andrei Fechete[1*], Mohamed Sana[2], Fadhel Ayed[2], Nicola Piovesan[2], Wenjie Li[2], Antonio De Domenico[2], Tareq Si Salem[2†]

[1]École Polytechnique, Palaiseau, France
[2]Paris Research Center, Huawei Technologies, Boulogne-Billancourt, France

Abstract

Conventional time-series forecasting methods typically aim to minimize overall prediction error, without accounting for the varying importance of different forecast ranges in downstream applications. We propose a training methodology that enables forecasting models to adapt their focus to application-specific regions of interest at inference time, without retraining. The approach partitions the prediction space into fine-grained segments during training, which are dynamically reweighted and aggregated to emphasize the target range specified by the application. Unlike prior methods that predefine these ranges, our framework supports flexible, on-demand adjustments. Experiments on standard benchmarks and a newly collected wireless communication dataset demonstrate that our method not only improves forecast accuracy within regions of interest but also yields measurable gains in downstream task performance. These results highlight the potential for closer integration between predictive modeling and decision-making in real-world systems.

Code — https://github.com/netop-team/gotsf
Datasets — https://hf.co/datasets/netop/gotsf-ds
Extended version — https://arxiv.org/pdf/2504.17493

1 Introduction

Time-series forecasting (TSF) represents a significant area of study within machine learning (ML), with practical applications demonstrable in various domains, including but not limited to, economics (Granger and Newbold 2014), energy resource management (Martín et al. 2010; Qian et al. 2019), transportation optimization (Chen et al. 2001), meteorological prediction (Wu et al. 2023), inventory optimization (Li et al. 2022), and healthcare (Ghassemi et al. 2015; Ray et al. 2025). At its core, TSF is concerned with constructing predictive models for time-dependent sequential data. This involves leveraging historical patterns and relationships within observations to forecast future data points. The methodologies employed in TSF are diverse, ranging from classical statistical approaches, such as Autoregressive Integrated Moving Average (ARIMA) (Hyndman and Athanasopoulos 2015) and Exponential Smoothing (ETS) (Brown 1956), to deep learning (DL) approaches, including Multi-Layer Perceptrons (MLPs) (Zeng et al. 2023), Recurrent Neural Networks (RNNs) (Zhang and Man 1998), Long Short-Term Memory (LSTMs) (Siami-Namini, Tavakoli, and Namin 2019), Temporal Convolutional Networks (TCNs) (He and Zhao 2019), and Transformer architectures (Nie et al. 2022; Liu et al. 2023). More recently, the field has witnessed the appearance of foundation Large Time-Series Models (LTSMs), pre-trained on extensive time-series datasets to enable zero-shot forecasting. These models, including Timer (Liu et al. 2024b), Moirai (Woo et al. 2024), TimesFM (Das et al. 2024), Chronos (Ansari et al. 2024), Moment (Goswami et al. 2024), and Toto (Cohen et al. 2024) predominantly utilize variations of the Transformer architecture.

Generally, TSF methodologies prioritize the minimization of predictive error, often neglecting the integration of predicted outputs within subsequent downstream processes. In numerous downstream applications of forecasting, the practical significance of forecast errors is not uniform, and this treatment introduces suboptimal model performance with respect to the ultimately desired objective. This is effectively illustrated by forecasting challenges *IEEE-CIS Technical Challenge on Predict+Optimize for Renewable Energy Scheduling* (Bergmeir et al. 2022) and the *M5 Accuracy Competition* (Makridakis, Spiliotis, and Assimakopoulos 2022), where evaluations based solely on forecast accuracy substantially mismatches with evaluations based on the eventual optimization objective, such as energy minimization. This necessitates the development of TSF models that explicitly incorporate downstream task objectives during development and evaluation. This integration is essential given the widespread of real world analytical systems combining predictive and optimization components. For example, Bertsimas and Kallus (Bertsimas and Kallus 2020) propose learning weight functions from data for integration into optimization objectives. Similarly, Elmachtoub and Grigas (Elmachtoub and Grigas 2022) proposed a "Predict-then-Optimize" framework within linear programming, directly leveraging optimization structure to inform loss function design. Furthermore, the ML community has witnessed a significant trend towards end-to-end (E2E) learning paradigms, applied across domains such as

[*]Work done while an intern at Huawei.
[†]Lead researcher for this study (corresponding author).

finance (Bengio 1997), image recognition (Wang, Babenko, and Belongie 2011), robotic manipulation (Levine et al. 2016), and inventory-management (Qi et al. 2023) highlighting the potential for this approach in TSF.

A major limitation within existing literature (Bertsimas and Kallus 2020; Elmachtoub and Grigas 2022; Qi et al. 2023) is the assumption of pre-defined task specifications. Specifically, these methodologies presuppose that regions of forecasting importance are both provided and static. However, numerous real-world TSF applications, such as wireless traffic prediction, necessitate adaptive approaches due to unknown and dynamically shifting importance regions. For instance, in energy efficiency policies, low-traffic periods are more important for base-station deactivation, while high-traffic periods are less relevant. Conversely, power allocation strategies may require sensitivity to both extremes, with lower importance on intermediate values. In practice, these thresholds are often not known a priori and potentially time-varying. Consequently, there is a pressing need for TSF frameworks that enable post-hoc configuration to give importance to specific regions of interest during inference. This work takes the first step, to the best of our knowledge, to address the observed gap. A motivating problem that can be tackled by our approach is illustrated in Figure 1.

Contributions. In this work, we address this gap by: (1) proposing a training methodology that extends multivariate TSF models to adapt to multiple downstream tasks at inference without retraining, (2) conducting extensive experiments on synthetic and real-world traces, including comprehensive baseline comparisons and ablation studies, to validate its effectiveness, and (3) introducing a new wireless mobile network dataset to support further research.

Outline of Paper. This paper proceeds as follows. Section 2 reviews the related literature. Section 3 formalizes the forecasting problem. Section 4 details the methodology employed in this research. Section 5 describes the experiments we performed to investigate the performance of our training methodology. Finally, Section 6 concludes the paper and outlines avenues for future research.

2 Literature Review

Time-Series Forecasting. Traditionally, statistical approaches like ARIMA (Hyndman and Athanasopoulos 2015) and ETS (Brown 1956) were predominant in TSF. However, the surge in computational power and data availability has spurred the adoption of deep learning techniques. This transition encompasses MLPs (Zeng et al. 2023), RNNs (Zhang and Man 1998), LSTMs (Siami-Namini, Tavakoli, and Namin 2019), TCNs (He and Zhao 2019), and Transformer architectures (Nie et al. 2022; Liu et al. 2023; Wu et al. 2021a; Zhou et al. 2021, 2022). Adapting Transformers for TSF is now a key research area, with efforts focused on: (1) refining internal components, particularly attention mechanisms (Wu et al. 2021a; Zhou et al. 2021, 2022); (2) transforming input token representations using techniques like stationarization (Liu et al. 2022), channel independence (Nie et al. 2022), and patching (Nie et al. 2022; Liu et al. 2023); and (3) broadly modifying the Transformer architecture and its modules (Zhang and Yan 2023). Recent advances include foundation LTSMs like Timer (Liu et al. 2024b), Moirai (Woo et al. 2024), TimesFM (Das et al. 2024), Chronos (Ansari et al. 2024), Moment (Goswami et al. 2024), and Toto (Cohen et al. 2024) which leverage pre-training and Transformer variants for zero-shot forecasting. This research extends state-of-the-art transformer architectures by integrating a task-specific configuration mechanism, enabling dynamic adaptation of model predictions during inference.

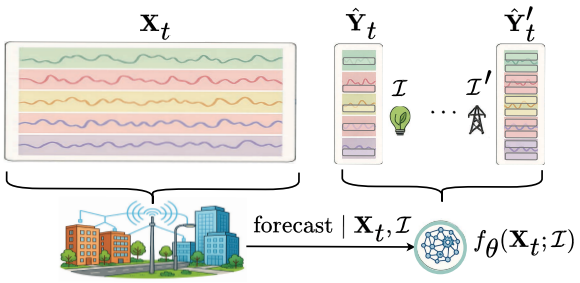

Figure 1: The figure illustrates time-series forecasting in a wireless network, showing the stages of data collection, model training, and deployment to downstream tasks such as energy efficiency, where accurate prediction of low traffic periods is critical, and power allocation, which requires precise forecasts across two traffic bands. Traditional forecasting methods typically ignore these task-specific requirements, whereas the proposed approach allows a single model to adapt to diverse downstream tasks at inference time and provides greater flexibility than task-specific end-to-end systems.

Task-Aware Forecasting. Prior research has explored customizations of the objective function for TSF, through loss reshaping or re-weighing (Park et al. 2023; Xue et al. 2023; Cheng, Huang, and Zheng 2023; Hounie, Porras-Valenzuela, and Ribeiro 2024). These works primarily aim to integrate uncertainty quantification or fairness considerations, diverging from the objective of tailoring models to specific downstream tasks. Conversely, other research have concentrated on developing frameworks that derive loss weights from downstream task characteristics, thereby guiding the training process of forecasting models (Bergmeir et al. 2022; Elmachtoub and Grigas 2022; Grabocka, Scholz, and Schmidt-Thieme 2019). While our methodology shares conceptual similarities with this latter line of work, specifically in the incorporation of loss shaping during training, it distinguishes itself by proposing a training methodology that enables the same model to accommodate diverse downstream task requirements at inference-time.

3 The Forecasting Problem

Time-Series. A multivariate time-series is a sequence of vector-valued observations ordered temporally. Let $\mathbf{x}_t \in \mathbb{R}^n$ denote a n-dimensional real-valued vector observed at time t, where $t \in [T]$ and $n \geq 1$. The full time-series is rep-

resented as the ordered set $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}$, where each $\mathbf{x}_t = (x_{t,i})_{i\in[n]}$ corresponds to measurements of n variables at time t. The temporal dependency between observations is intrinsic to the series, with $\mathbf{x}_t$ potentially influenced by past values $\mathbf{x}_{t-k}$ for some lag $k > 0$. We assume each series $i \in [n]$ lies within a bounded set $\mathcal{X} \subset \mathbb{R}$, where $|\mathcal{X}| < \infty$.

The Learning Problem. Time-series forecasting is reformulated as a supervised regression task by constructing input-output pairs from sliding windows over the series. Let $\tau \in \mathbb{N}$ be the forecast horizon and $w \in \mathbb{N}$ be the window size. In practice, the window size w is treated as a hyperparameter. For each time t, the input $\mathbf{X}_t = \mathbf{x}_{t-w:t-1} \in \mathcal{X}^{w\times n}$ consists of a history window $\{\mathbf{x}_{t-w}, \ldots, \mathbf{x}_{t-1}\}$, and the target $\mathbf{Y}_t = \mathbf{x}_{t:t+\tau-1} \in \mathcal{X}^{\tau\times n}$ is the future window $\{\mathbf{x}_t, \ldots, \mathbf{x}_{t+\tau-1}\}$. The mapping $f_{\boldsymbol{\theta}} : \mathcal{X}^{w\times n} \to \mathcal{X}^{\tau\times n}$, parameterized by $\boldsymbol{\theta}$, is learned to approximate the underlying dynamic $\mathbf{Y}_t = f(\mathbf{X}_t) + \boldsymbol{\epsilon}_t$, where $\boldsymbol{\epsilon}_t \in \mathbb{R}^{\tau\times n}$ is noise and f is some unknown dynamic. This generates a dataset $D = \{(\mathbf{X}_t, \mathbf{Y}_t)\}_{t=w}^{T-\tau}$. The goal is to learn the model $\boldsymbol{\theta}_\star \in \mathbb{R}^d$ for $d \geq 1$ that minimizes the expected loss over the data distribution $\mathcal{D}(\mathbf{X}, \mathbf{Y})$:

$$\boldsymbol{\theta}_\star \in \arg\min_{\boldsymbol{\theta}\in\mathbb{R}^d} \mathbb{E}_{(\mathbf{X},\mathbf{Y})\sim\mathcal{D}} \left[l(f_{\boldsymbol{\theta}}(\mathbf{X}), \mathbf{Y}) \right], \tag{1}$$

where $l : \mathbb{R}^{\tau\times n} \times \mathbb{R}^{\tau\times n} \to \mathbb{R}$ is a differentiable loss function. In practice, this expectation is approximated by the empirical risk over D: $\tilde{\boldsymbol{\theta}}_\star \in \arg\min_\theta \frac{1}{|D|} \sum_{(\mathbf{X}_t,\mathbf{Y}_t)\in D} l(f_{\boldsymbol{\theta}}(\mathbf{X}_t), \mathbf{Y}_t) + R(\boldsymbol{\theta})$. The regularization term $R : \mathbb{R}^d \to \mathbb{R}$ (e.g., the Euclidean distance $R(\boldsymbol{\theta}) = \lambda\|\boldsymbol{\theta}\|_2^2$) is incorporated to control the complexity of the model parameters $\boldsymbol{\theta}$, and it aims to prevent the memorization of noise within the training data, enhancing the model's ability to generalize to novel data (Shalev-Shwartz and Ben-David 2014). Among the most commonly used loss functions are the Mean Squared Error (MSE) and the Mean Absolute Error (MAE), each of which serves a distinct statistical purpose (i.e., conditional expectation and the conditional median, respectively).

4 Methodology

Our methodological investigation proceeds by first establishing a baseline policy (`B-Policy`), representative of standard forecasting model training that overlooks specific prediction intervals. We then define an end-to-end policy (`E2E-Policy`), which focuses on training a model tailored to a particular interval of interest. Subsequently, we introduce a naive policy (`C-Policy`) designed to train a foundation model capable of adapting to various intervals at inference time by exposing it to all possible intervals during training. Following this, we describe a policy ($\texttt{D}_L\texttt{-Policy}$) that explores a finite subset of relevant intervals, strategically chosen such that their combination covers the entire forecasting space. Finally, we propose a policy ($\texttt{D}_L^\star\texttt{-Policy}$) that discretizes the forecasting space and, when integrated with a patching technique, can enable the training of more effective foundation models.

Baseline Policy (`B-Policy`). This policy follows the baseline training procedure detailed in Section 3. This training approach does not incorporate interval sensitivity, and the loss is computed with respect to the forecasting target over the domain $\mathcal{X}$.

Task-specific Policy (`E2E-Policy`). Given a target interval $\mathcal{I} \subseteq \mathcal{X}$, the task-specific policy only considers a forecasting target within this range. In particular, this policy can be considered as a variation of the baseline policy limited to consider a forecasting loss solely over the intervals of interest. Our goal is to learn the model $\boldsymbol{\theta}_\star \in \mathbb{R}^d$ that minimizes the interval-specific expected loss over a data distribution $\mathcal{D}(\mathbf{X}, \mathbf{Y})$ given as

$$\boldsymbol{\theta}_\star \in \arg\min_{\boldsymbol{\theta}\in\mathbb{R}^d} \mathbb{E}_{(\mathbf{X},\mathbf{Y})\sim\mathcal{D}} \left[l\left(f_{\boldsymbol{\theta}}(\mathbf{X}), \mathbf{Y}\right) \mathbb{1}\left(\mathbf{Y} \in \mathcal{I}^{\tau\times n}\right) \right],$$

where $\mathbb{1}(\chi) \in \{0, 1\}$ is the indicator function equal to 1 when condition χ is true. Again, the expectation can be approximated by the interval-specific empirical risk over some dataset D consisting of samples from $\mathcal{D}$ with a regularization term.

Continuous-Interval Training Policy (`C-Policy`). To allow the forecasting model to differentiate between any target intervals within $\mathcal{X}$, we incorporate the interval as a covariate. A covariate interval $\mathcal{I} \subseteq \mathcal{X}$ is represented as a two-dimensional vector in $\mathcal{X}^2$ containing its boundary values (e.g., a vector encoding $(I_{\min}, I_{\max}) \in \mathcal{X}^2$, for some interval $[I_{\min}, I_{\max}] \subseteq \mathcal{X}$). This enables the model to learn the relationship between varying intervals and its predictive outputs. Specifically, the learned mapping $f_{\boldsymbol{\theta}}$ becomes a function of both the input time-series $\mathbf{X}$ and the interval $\mathcal{I}$, i.e., the mapping $f_{\boldsymbol{\theta}} : \mathcal{X}^{w\times n+2} \to \mathcal{X}^{\tau\times n}$. During training, we sample intervals uniformly at random over the set $\mathcal{U}_\delta = \{\mathcal{I} \subset \mathcal{X} : |\mathcal{I}| \geq \delta\}$, where δ represents the minimum length of the sampled intervals. The minimal distance constraint is introduced to make the training more stable, because small intervals contain less samples introducing high variance. Consequently, for a data sample $(\mathbf{X}_t, \mathbf{Y}_t) \sim \mathcal{D}$ and its corresponding sampled interval $\mathcal{I}_t$, our custom loss function is defined as:

$$L(f_{\boldsymbol{\theta}}(\mathbf{X}_t,\mathcal{I}_t),\mathbf{Y}_t,\mathcal{I}_t) = l\left(f_{\boldsymbol{\theta}}(\mathbf{X}_t,\mathcal{I}_t),\mathbf{Y}_t\right) \mathbb{1}(\mathbf{Y}_t \in \mathcal{I}_t^{\tau\times n}). \tag{2}$$

Thus, our training objective is to learn the model parameters $\boldsymbol{\theta}^\star_{\text{Cont}} \in \mathbb{R}^d$ ($d \geq 1$) that minimize the interval-uniform expected loss over the data distribution $\mathcal{D}(\mathbf{X}, \mathbf{Y})$ and the interval distribution $\mathcal{U}_\delta$:

$$\boldsymbol{\theta}_\star \in \arg\min_{\boldsymbol{\theta}\in\mathbb{R}^d} \mathbb{E}_{(\mathbf{X},\mathbf{Y})\sim\mathcal{D},\mathcal{I}\sim\mathcal{U}_\delta} \left[L(f_{\boldsymbol{\theta}}(\mathbf{X}, \mathcal{I}), \mathbf{Y}, \mathcal{I}) \right]. \tag{3}$$

Similar to the previous policies, the above problem is approximated by minimizing the empirical risk over the dataset D and the interval distribution $\mathcal{U}_\delta$: $\tilde{\boldsymbol{\theta}}_\star \in \arg\min_{\boldsymbol{\theta}\in\mathbb{R}^d} \frac{1}{|D|} \sum_{(\mathbf{X}_t,\mathbf{Y}_t)\in D} L(f_{\boldsymbol{\theta}}(\mathbf{X}_t, \mathcal{I}_t), \mathbf{Y}_t, \mathcal{I}_t) + R(\boldsymbol{\theta})$, where $\mathcal{I}_t$ is the sampled interval from $\mathcal{U}_\delta$ for each $(\mathbf{X}_t, \mathbf{Y}_t) \in D$.

Discretized-Interval Training Policy ($\texttt{D}_L\texttt{-Policy}$). This policy closely resembles the `C-Policy`, except that

the support space of the distribution of possible interval is severely reduced. In particular, the interval selection process now involves sampling an interval from a discrete distribution, denoted by $\mathcal{C}_L$, defined over a set of L disjoint intervals that collectively cover the interval $\mathcal{X}$, i.e., for a given L the support $\text{supp}\,(\mathcal{C}_L)$ satisfies $\dot{\bigcup}_{\mathcal{I}\in\text{supp}(\mathcal{C}_L)} = \mathcal{X}$ where $\dot{\bigcup}$ denotes the union of disjoint sets.

Patching-Augmented Discretized Training Policy ($\texttt{D}^\star_L\texttt{-Policy}$). The premise behind the design of this policy is to consider the $\texttt{D}_L\texttt{-Policy}$ configured with a sufficiently large L corresponding to a finer discretization of the domain $\mathcal{X}$. At inference time when requested with an unknown interval $\mathcal{I}$, the policy combines the constituent intervals to make the predictions. This is made possible with two main modifications. First, the loss function incorporates a decay function that modulates the contribution of each sample to the loss based on its distance from the center of its associated interval.[1] This decay mechanism is defined as

$$d_\nu(y,\mathcal{I}) = \exp\left(-\nu\cdot\max\left(0,|y-\Delta_{\text{avg}}|-\Delta_{\text{diff}}\right)\right), \quad (4)$$

where ν is the decay rate, and $\Delta_{\text{avg}} = (\max(\mathcal{I})+\min(\mathcal{I}))/2$ is the midpoint of an interval and $\Delta_{\text{diff}} = (\max(\mathcal{I})-\min(\mathcal{I}))/2$ is half of the length of the interval.

As the decay rate ν becomes large ($\nu\to\infty$), the weight $\prod_{i\in[n],t\in[\tau]} d_\nu(y_i,\mathcal{I}) \in [0,1]$ converges to the indicator function $\mathbb{1}(\mathbf{Y}\in\mathcal{I}^{\tau\times n})\in\{0,1\}$ in Eq. (2). This weighting allows learning at interval boundaries and introduces a soft overlap between intervals, which is beneficial for reducing patching errors caused by boundary uncertainties. Additionally, a classification head $f^c_{\boldsymbol{\theta}}: \mathcal{X}^{w\times n+2}\to[0,1]^{\tau\times n}$ (parameterized by a shared $\boldsymbol{\theta}$) is appended to predict if the true value falls within $\mathcal{I}_t$, using the same notation as before. The modified regression loss with the soft boundary is $L_\nu(f_{\boldsymbol{\theta}}(\mathbf{X}_t,\mathcal{I}_t),\mathbf{Y}_t,\mathcal{I}_t) \triangleq l(f_{\boldsymbol{\theta}}(\mathbf{X}_t,\mathcal{I}_t),\mathbf{Y}_t)\cdot\prod_{i\in[n],t\in[\tau]} d_\nu(y_{t,i},\mathcal{I}_t)$, and the modified classification loss is $L'_\nu(f^c_{\boldsymbol{\theta}}(\mathbf{X}_t,\mathcal{I}_t),\mathbb{1}(\mathbf{Y}_t\in\mathcal{I}_t),\mathcal{I}_t)\triangleq l'_\nu(f^c_{\boldsymbol{\theta}}(\mathbf{X}_t,\mathcal{I}_t),\mathbb{1}(\mathbf{Y}_t\in\mathcal{I}_t))\cdot\prod_{i\in[n],t\in[\tau]} d_\nu(y_{t,i},\mathcal{I}_t)$, where l'_ν is a classification loss like cross entropy. The learning objective for the augmented model parameters (including the classifier) is defined as follows:

$$\boldsymbol{\theta}_\star \in \underset{\boldsymbol{\theta}\in\mathbb{R}^d}{\arg\min}\ \mathbb{E}_{(\mathbf{X},\mathbf{Y})\sim\mathcal{D},\mathcal{I}\sim\mathcal{C}_L}\big[L_\nu(f_{\boldsymbol{\theta}}(\mathbf{X},\mathcal{I}),\mathbf{Y},\mathcal{I}) + \phi\cdot L'_\nu(f^c_{\boldsymbol{\theta}}(\mathbf{X},\mathcal{I}),\mathbb{1}(\mathbf{Y}\in\mathcal{I}),\mathcal{I})\big], \quad (5)$$

where $\phi\in[0,1]$ is a hyperparameter introduced to balance the importance of the regression and classification tasks. This objective is approximated empirically using the dataset D and the interval distribution $\mathcal{C}_L$:

$$\tilde{\boldsymbol{\theta}}_\star \in \underset{\boldsymbol{\theta}\in\mathbb{R}^d}{\arg\min}\frac{1}{|D|}\sum_{(\mathbf{X}_t,\mathbf{Y}_t)\in D}\big(L_\nu(f_{\boldsymbol{\theta}}(\mathbf{X}_t,\mathcal{I}_t),\mathbf{Y}_t,\mathcal{I}_t) + \phi\cdot L'_\nu(f^c_{\boldsymbol{\theta}}(\mathbf{X}_t,\mathcal{I}_t),\mathbb{1}(\mathbf{Y}_t\in\mathcal{I}_t),\mathcal{I}_t)\big)+R(\boldsymbol{\theta}),$$

where $\mathcal{I}_t$ is the sampled interval from $\mathcal{C}_t$ for each $(\mathbf{X}_t,\mathbf{Y}_t)\in D$.

[1]This function is depicted in the *extended* version of this paper.

Patching Mechanism. Given an arbitrary interval $\mathcal{I}$, we first identify the subset of training intervals that intersect with $\mathcal{I}$, given by the mapping

$$\Xi_L(\mathcal{I}) \triangleq \{\mathcal{I}'\in\text{supp}\,(\mathcal{C}_L) : \mathcal{I}'\cap\mathcal{I}\neq\emptyset\}, \quad (6)$$

where $\text{supp}\,(\mathcal{C}_L)$ is the support of the distribution $\mathcal{C}_L$ of size L. The logic behind this is that when L is large enough, the union of intervals in $\Xi_L(\mathcal{I})$ is a good approximation of $\mathcal{I}$. For an input series $\mathbf{X}$ and target interval $\mathcal{I}$ the patching mechanism strategies are defined as follows:

- *Averaging Strategy (1-strategy).* This strategy computes a weighted average of the regression predictions for all intersecting training intervals, weighted by their respective classification probabilities for the given input and the target interval $\mathcal{I}$. Formally, defined as

$$\hat{\mathbf{Y}} = \frac{\sum_{\mathcal{I}'\in\Xi_L(\mathcal{I})} f^c_\theta(\mathbf{X},\mathcal{I}')f_\theta(\mathbf{X},\mathcal{I}')}{\sum_{\mathcal{I}'\in\Xi_L(\mathcal{I})} f^c_\theta(\mathbf{X},\mathcal{I}')}. \quad (7)$$

- *Maximum Confidence Strategy (∞-strategy).* This strategy selects the regression prediction corresponding to the training interval with the highest classification probability for the given input and the target interval $\mathcal{I}$. Formally, defined as

$$\hat{\mathbf{Y}} = f_\theta\big(\mathbf{X},\ \underset{\mathcal{I}'\in\Xi_L(\mathcal{I})}{\text{argmax}}\ f^c_\theta(\mathbf{X},\mathcal{I}')\big). \quad (8)$$

5 Experiments

This section presents the numerical results of our training methodology and compares its forecasting performance with several baselines. We start by outlining the forecasting models, training policies, benchmark datasets, and training configurations used. Following this, we provide a quantitative analysis of the different training strategies applied to these models.

5.1 Experimental Setup

Time-Series Forecasting Models. We evaluate four leading time series forecasting models: iTransformer (Liu et al. 2024a), DLinear (Zeng et al. 2023), PatchTST (Nie et al. 2023), and TimeMixer (Wang et al. 2024). To incorporate target interval information $\mathcal{I}$, we modified these architectures. Given an interval of interest $\mathcal{I}\subseteq\mathcal{X}$, for iTransformer, the vectorized form of $\mathcal{I}$ was concatenated to the temporal encoding. For DLinear, PatchTST, and TimeMixer, $\mathcal{I}$'s vectorized encoding was added as two extra temporal channels. Furthermore, we adapted these regression models for a dual-task objective (regression and classification) by adding a classification head. This involved doubling the final projection layer's output size (τ, the forecasting horizon) and splitting it into regression forecasts (first τ dimensions) and classification logits (remaining τ dimensions). This modification allows the models to predict future values and simultaneously assess the probability of these forecasts falling within the target interval $\mathcal{I}$.

Training Policies. We evaluate the training policies introduced in Section 4, namely $\texttt{B-Policy}$, $\texttt{E2E-Policy}$, $\texttt{C-Policy}$, $\texttt{D}_L\texttt{-Policy}$, and $\texttt{D}^\star_L\texttt{-Policy}$. The $\star$ in the latter denotes either average (1) or maximum (∞) patching.

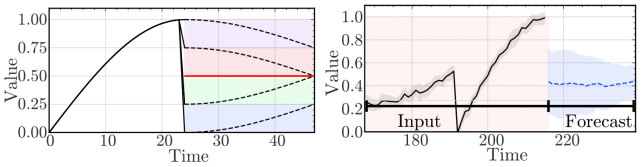

Figure 2: Subfigure (a) depicts the expected hypotheses used to construct the synthetic trace SynthDS. Subfigure (b) depicts the inability of the baseline policy B-Policy to distinguish between the different patterns in SynthDS dataset, as it predicts the average hypothesis (red line at 0.5). The model 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.

Benchmarking Datasets. In our experimental evaluation, we rely on several benchmark datasets to support our claims. (1) SynthDS is a synthetic dataset designed to illustrate the components of our proposed methodology. The trace is constructed bu concatenating an input signal $\mathbf{s}_i = \left(\sin\left(\frac{\pi(n-1)}{w-1}\right)\right)_{n\in[w]}$, where $w = 24$ denotes the signal length, and a corresponding output signal $\mathbf{s}_o = \left(s \cdot 0.25\cos\left(\frac{\pi(n-1)}{2(w-1)}\right) + b\right)_{n\in[w]}$, where the pair (s, b) is sampled uniformly at random from $\{(1, 0.75), (1, 0.5), (-1, 0.25), (-1, 0.5)\}$. To generate a trace spanning $T = 3.1 \times 10^3$ timesteps, we concatenate the same randomly selected signal multiple times. Subsequently, we introduce Gaussian noise to the trace, where the noise has a mean of 0 and a standard deviation of 0.05. A noise-free version of this trace is depicted in Figure 2a. (2) BLW-TrafficDS is released to the public domain as part of this study. The dataset comprises a wireless beam-level traffic volume containing 100 beams and 10^3 timesteps. (3) Traffic (Caltrans 2016) contains hourly road occupancy rates (ranging from 0 to 1) recorded by sensors on San Francisco Bay area freeways from 2015 to 2016, totaling 48 months of data. (4) Electricity (Repository 2014) includes hourly electricity consumption (in kWh) records of 321 clients from 2012 to 2014. (5) Weather (Wu et al. 2021a) contains 21 meteorological indicators, such as air temperature and humidity, recorded every 10 minutes throughout the entire year of 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

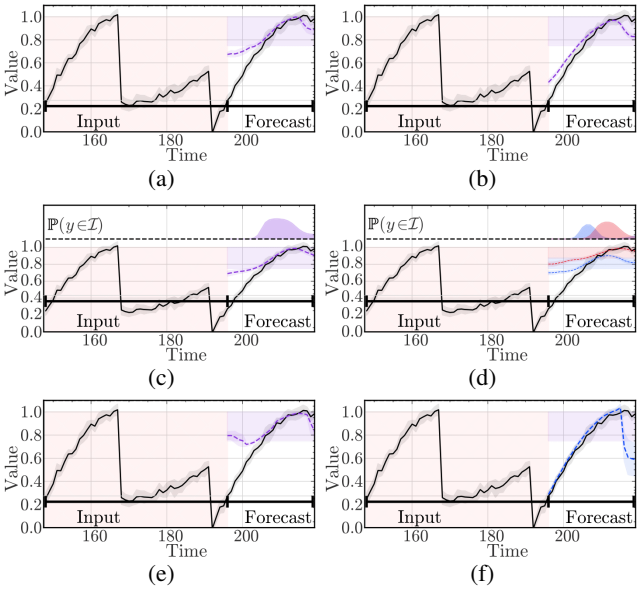

Figure 3: Subfigures (a)–(f) present the performance of the iTransformer model trained on the SynthDS trace. Each subfigure corresponds to a different policy: E2E, C, $\mathtt{D}_4$, $\mathtt{D}_8$, $\mathtt{D}_8^1$, and B, respectively. The purple region marks the interval of interest, $\mathcal{I} = [0.75, 1.0]$. 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 $\mathtt{D}_8^1$ 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.

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 (Loshchilov and Hutter 2017) was used with an initial learning rate of 10^{-3} and a cosine annealing schedule with $\eta_{\min} = 10^{-5}$, defined as:$\eta_t = \eta_{\min} + 0.5(\eta_{\max} - \eta_{\min})(1 + \cos(\pi \cdot t/n_{\text{epochs}}))$.

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

The $\mathtt{D}_L^\star$-Policy was evaluated using 4 and 8 intervals for the SynthDS, Traffic, Electricity, and Weather datasets, and 8 and 16 intervals for the

BLW-TrafficDS dataset. Interval endpoint sampling was performed per sample within each batch and incorporated into the forward pass as detailed in Section 4.

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

5.2 Numerical Results

Qualitative Comparison of Training Schemes. To build intuition about how different training policies behave when downstream tasks specify an interval of interest, thereby prioritizing accuracy within that interval, we begin by evaluating the iTransformer model on the SynthDS trace. This dataset is chosen for its simplicity, which facilitates both result interpretation and qualitative performance assessment.

Baseline. We first examine the averaging behavior of the baseline policy when training iTransformer on SynthDS. As shown in Figure 2b, the model converges to an averaged prediction centered at the trace's midpoint ($1/2$), reflecting its insensitivity to specific intervals of interest. This is undesirable in scenarios where the conditional mean within the interval of interest differs from the global mean as the model fails to adapt its predictions to the interval-specific distribution.

End-to-End Training Approach. Figure 3a shows the performance of E2E-Policy with the iTransformer model, targeting the specific interval $\mathcal{I} = [0.75, 1]$. This corresponds to an end-to-end strategy in which the forecasting loss is adapted to emphasize task-relevant intervals. The selected interval represents a distinct hypothesis, and this policy serves as an optimal benchmark, illustrating the best-case performance achievable by a method that can adapt to any interval and be configured at inference time.

Continuous Exploration of Intervals. We also evaluate C-Policy using the iTransformer model on SynthDS. The results in Figure 3b show that this approach struggles to effectively learn the relationship between intervals of interest and their corresponding hypotheses, underscoring the difficulty of the learning task even in a synthetic setting. In this configuration, the model is tasked with learning a mapping from every possible interval $\mathcal{I} \subset \mathcal{X}$ to its associated hypothesis. We further explore, in the *extended version* of this paper, the impact of restricting the support space of training intervals by enforcing a minimum separation $|\mathcal{I}| \geq \delta$. While performance improves with a reduced sampling space, beyond a certain threshold δ', it deteriorates as the model overshoots the true interval lengths, introducing bias.

Targeted Exploration of Intervals. Building on the observed limitations of continuous exploration, Figure 3c examines a discrete policy (D_L-Policy) on SynthDS. This policy uniformly samples from a predefined, finite, and relatively small set of intervals of interest during training. We first evaluate it using the optimal E2E-Policy, which can distinguish signal patterns. The results show that D_L-Policy achieves performance comparable to E2E-Policy while retaining the advantage of inference-time adaptability to diverse downstream tasks. However, this approach assumes prior knowledge of the relevant interval set (or its size), motivating the subsequent experiment in which we intentionally increase the number of considered intervals to assess the reconstructability of the true signal.

Adaptive Interval Exploration with Patching Techniques. Figures 3d and 3e evaluate the effectiveness of our patching strategy, described in Section 4, in capturing the underlying dynamics of SynthDS. These experiments consider scenarios in which the number of intervals is deliberately overestimated during training and subsequently patched back to an unknown target interval at inference time. This overestimation is intended to qualitatively assess the robustness of the patching schemes. While the schemes exhibit visually similar performance, a quantitative analysis is required for further comparison. Notably, both approaches distinguish the four distinct underlying hypotheses used to construct the synthetic trace compared to C-Policy, thereby motivating our design choice: train a model on a large set of intervals that can be flexibly combined at inference time to produce focused forecasts for a specific interval of interest.

Quantitative Comparison of Training Schemes. Following our qualitative evaluation of the proposed training policies, Table 1 presents the numerical results obtained on the BLW-TrafficDS, Traffic, Weather, Electricity, and SynthDS datasets. The table reports the MAE values for the four models, each trained under the four distinct policies.

Table 1 reports the results of training the different models on the SynthDS trace. We observe comparable performance between the iTransformer and PatchTST models. Notably, B-Policy, which does not support inference-time adaptation of the interval of interest, yields the poorest performance. The C-Policy shows a modest MAE improvement, suggesting that training toward specific intervals provides some benefit. However, sampling all possible intervals in C-Policy hinders the model's ability to learn distinct underlying hypotheses, as evidenced by the superior performance of D_L-Policy, configured here with four specific intervals. Introducing the patching mechanism in $D_L^\star$-Policy leads to some performance degradation for certain intervals; nonetheless, its overall MAE remains comparable to, and occasionally exceeds, that of D_L-Policy, potentially due to an ensemble-like noise reduction effect.

These conclusions generally extend to the real-world traces. Interestingly, in some cases patching outperforms the vanilla discrete approach, with averaging-based patching achieving the best performance for the PatchTST model. Results differ for weaker models such as DLinear, where C-Policy appears to dominate; however, given its substantially lower overall accuracy compared to the other models, the ordering of improvements may be influenced by random factors. We also note that D_L-Policy is optimally configured here with intervals that precisely align with the true underlying hypotheses under the SynthDS trace—a level of prior knowledge unlikely in practical applications.

		DLinear						TimeMixer						PatchTST						iTransformer					
		$\mathtt{D}_L$	$\mathtt{D}^1_{2L}$	$\mathtt{D}^\infty_{2L}$	B	$\mathtt{C}_{0.2}$	↑	$\mathtt{D}_L$	$\mathtt{D}^1_{2L}$	$\mathtt{D}^\infty_{2L}$	B	$\mathtt{C}_{0.2}$	↑	$\mathtt{D}_L$	$\mathtt{D}^1_{2L}$	$\mathtt{D}^\infty_{2L}$	B	$\mathtt{C}_{0.2}$	↑	$\mathtt{D}_L$	$\mathtt{D}^1_{2L}$	$\mathtt{D}^\infty_{2L}$	B	$\mathtt{C}_{0.2}$	↑
BLW-TrafficDS	$\mathcal{I}_1$	167.1	172.3	164.9	**163.8**	218.0	0.0%	127.5	148.1	138.0	**125.9**	143.7	0.0%	$\underline{\mathbf{102.8}}$	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	$\underline{\mathbf{30.4}}$	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	$\underline{\mathbf{17.2}}$	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	$\underline{\mathbf{10.4}}$	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	$\underline{\mathbf{6.5}}$	7.9	34.9	9.8	81.4%	31.4	32.1	31.9	35.5	**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	$\underline{\mathbf{4.0}}$	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	$\underline{\mathbf{2.4}}$	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	$\underline{\mathbf{1.5}}$	1.8	11.7	2.6	87.2%	**11.2**	11.2	11.2	11.8	11.4	5.1%
	$\overline{\mathcal{I}_1\text{–}\mathcal{I}_8}$	**54.0**	54.8	54.3	57.6	61.6	6.3%	**29.4**	31.4	30.9	52.7	34.9	44.2%	$\underline{\mathbf{22.4}}$	28.8	25.5	50.7	26.7	55.8%	**47.8**	48.6	48.6	51.5	48.5	7.2%
Traffic	$\mathcal{I}_1$	1.70	**1.53**	1.67	1.79	1.82	14.5%	1.90	**1.71**	1.73	2.44	2.17	29.9%	$\underline{\mathbf{1.08}}$	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	$\underline{\mathbf{0.89}}$	37.3%	**1.19**	1.22	1.24	1.50	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%	$\underline{\mathbf{0.32}}$	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	$\underline{\mathbf{0.16}}$	0.19	0.53	0.16	69.8%	0.35	**0.36**	0.37	0.51	0.35	31.4%
	$\overline{\mathcal{I}_1\text{–}\mathcal{I}_4}$	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	$\underline{\mathbf{0.59}}$	0.63	1.01	0.62	41.6%	**0.84**	0.85	0.87	1.02	0.84	17.6%
Weather	$\mathcal{I}_1$	1.05	0.76	**0.75**	0.78	1.32	3.9%	2.25	**0.73**	0.71	1.00	1.94	27.0%	$\underline{\mathbf{0.40}}$	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%	$\underline{\mathbf{0.22}}$	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%	$\underline{\mathbf{0.08}}$	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%	$\underline{\mathbf{0.10}}$	0.13	0.14	0.23	0.11	56.5%	0.19	**0.19**	0.19	0.21	0.38	9.5%
	$\overline{\mathcal{I}_1\text{–}\mathcal{I}_4}$	0.43	**0.31**	0.34	0.40	0.55	22.5%	0.75	**0.29**	0.30	0.52	0.72	44.2%	$\underline{\mathbf{0.20}}$	0.28	0.26	0.37	0.40	46.0%	**0.30**	0.29	0.29	0.38	0.74	23.7%
Electricity	$\mathcal{I}_1$	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	$\underline{\mathbf{3.26}}$	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%	$\underline{\mathbf{0.919}}$	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%	$\underline{\mathbf{0.671}}$	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%	$\underline{\mathbf{0.238}}$	0.248	0.272	0.370	0.243	35.7%	0.567	0.509	0.522	0.372	**0.322**	13.4%
	$\overline{\mathcal{I}_1\text{–}\mathcal{I}_4}$	**1.56**	1.57	1.57	1.57	1.58	0.7%	**1.51**	1.52	1.54	1.86	1.90	18.8%	$\underline{\mathbf{1.29}}$	1.75	1.81	1.39	1.42	6.93%	5.02	4.09	4.21	**1.39**	1.46	0.0%
SynthDS	$\mathcal{I}_1$	15.0	**14.5**	18.5	50.9	24.8	71.5%	16.2	**15.7**	17.9	45.5	17.5	65.5%	$\underline{\mathbf{13.2}}$	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%	$\underline{\mathbf{7.96}}$	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%	$\underline{\mathbf{7.74}}$	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	$\underline{\mathbf{12.4}}$	13.6	33.7	13.4	63.2%
	$\overline{\mathcal{I}_1\text{–}\mathcal{I}_4}$	13.8	**13.6**	15.1	32.3	17.1	57.9%	**13.7**	14.2	16.5	28.3	14.6	51.7%	$\underline{\mathbf{10.7}}$	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 values (↓ is better) and improvement percentages (↑ is better) relative to the baseline policy are reported for four models: `DLinear`, `TimeMixer`, `iTransformer`, and `PatchTST`, trained using five distinct policies: $\mathtt{D}_L$`-Policy`, $\mathtt{D}^1_{2L}$`-Policy`, $\mathtt{D}^\infty_{2L}$`-Policy`, `B-Policy`, and $\mathtt{C}_\delta$`-Policy` ($\delta = 0.2$). The experiments use $L = 8$ target intervals for the `BLW-TrafficDS` trace and $L = 4$ for all other datasets. MAE values are scaled for readability: $\times 10^3$ for `SynthDS` and `BLW-TrafficDS`, $\times 500$ for `Traffic`, $\times 2$ for `Weather`, and $\times 10^{-1}$ for `Electricity`. For `BLW-TrafficDS`, the number of intervals is increased to 8 and 16 for patching-based strategies. Bold values indicate the best MAE per model within each row, while underlined values denote the overall lowest MAE across all models. Model improvements are evaluated by comparing the best-performing policy to the baseline. The intervals of interest, denoted as $\mathcal{I}_1$–$\mathcal{I}_4$, correspond to a partition of the time series domain $\mathcal{X}$ into four equal-length contiguous sub-intervals. Similarly, $\mathcal{I}_1$–$\mathcal{I}_8$ indicates eight such sub-intervals. Notation $\overline{\mathcal{I}_1\text{–}\mathcal{I}_L}$ denotes the average performance over all intervals.

Consequently, $\mathtt{D}^\star_L$`-Policy` offers a more pragmatic alternative, providing a viable strategy for training foundation models capable of adapting to varying intervals during inference.

Forecasting for Energy-Saving and Sensitivity Analysis. We assess our proposed method in a realistic wireless network energy-saving scenario. We consider a two-tier heterogeneous network consisting of a high-throughput *capacity cell* overlaid by a wide-area *coverage cell*. Dynamic small-cell control has become a central mechanism for improving spectral efficiency and reducing energy consumption in 5G and beyond (Celebi et al. 2019; Ju et al. 2022; Wu et al. 2021b; Lozano et al. 2025; 3GPP 2014), with DLPRB utilization shown to be an effective signal for real-time sleep-mode activation. The capacity cell can be selectively deactivated during low traffic and adopt a threshold-based policy based on *forecasted* DLPRB utilizatio (Celebi et al. 2019; Ju et al. 2022). Details are provided in the *extended version*.

Our method reduces sleep-duration error by a factor of three ($\times 3$), corresponding to an average of one hour saved per day. This yields an energy-saving error of only 337 W, compared to 0.950 kW under the baseline policy. We also report sensitivity results across hyperparameters including the number and length of intervals, patching strategy, and decay rate.

6 Conclusion

This work introduces policies that equips existing TSF models with the ability to adapt at inference time to different downstream tasks. We conduct an extensive empirical evaluation to demonstrate the effectiveness of the proposed framework. Future work may explore scaling the approach to additional domains and fine-tuning existing time-series foundation models (e.g., Timer (Liu et al. 2024b), Chronos (Ansari et al. 2024)) to support adaptation to arbitrary intervals as described here. Another promising direction is a theoretical analysis, potentially within a PAC-learning framework (Shalev-Shwartz and Ben-David 2014), to explain the performance gap observed between the `C-Policy` and $\mathtt{D}^\star_L$`-Policy` policies.

References

3GPP. 2014. 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.

Ansari, A. F.; Stella, L.; Turkmen, C.; Zhang, X.; Mercado, P.; Shen, H.; Shchur, O.; Rangapuram, S.; Arango, S. P.; Kapoor, S.; et al. 2024. Chronos: Learning the language of time series. *Transactions on Machine Learning Research*.

Bengio, Y. 1997. Using a financial training criterion rather than a prediction criterion. *International journal of neural systems*, 8(04): 433–443.

Bergmeir, C.; de Nijs, F.; Sriramulu, A.; Abolghasemi, M.; Bean, R.; Betts, J.; Bui, Q.; Dinh, N. T.; Einecke, N.; Esmaeilbeigi, R.; et al. 2022. Comparison and evaluation of methods for a predict+ optimize problem in renewable energy. *arXiv preprint arXiv:2212.10723*.

Bertsimas, D.; and Kallus, N. 2020. From predictive to prescriptive analytics. *Management Science*, 66(3): 1025–1044.

Brown, R. G. 1956. *Exponential smoothing for predicting demand*. Little.

Caltrans. 2016. PeMS Traffic Data. http://pems.dot.ca.gov.

Celebi, H.; Yapıcı, Y.; Güvenç, I.; and Schulzrinne, H. 2019. Load-based on/off scheduling for energy-efficient delay-tolerant 5g networks. *IEEE Transactions on Green Communications and Networking*, 3(4): 955–970.

Chen, C.; Petty, K.; Skabardonis, A.; Varaiya, P.; and Jia, Z. 2001. Freeway performance measurement system: mining loop detector data. *Transportation research record*, 1748(1): 96–102.

Cheng, J.; Huang, K.; and Zheng, Z. 2023. Fitting imbalanced uncertainties in multi-output time series forecasting. *ACM Transactions on Knowledge Discovery from Data*, 17(7): 1–23.

Cohen, B.; Khwaja, E.; Wang, K.; Masson, C.; Ramé, E.; Doubli, Y.; and Abou-Amal, O. 2024. Toto: Time series optimized transformer for observability. *arXiv preprint arXiv:2407.07874*.

Das, A.; Kong, W.; Sen, R.; and Zhou, Y. 2024. A decoder-only foundation model for time-series forecasting. In *Forty-first International Conference on Machine Learning*.

Elmachtoub, A. N.; and Grigas, P. 2022. Smart "predict, then optimize". *Management Science*, 68(1): 9–26.

Ghassemi, M.; Pimentel, M.; Naumann, T.; Brennan, T.; Clifton, D.; Szolovits, P.; and Feng, M. 2015. 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.

Goswami, M.; Szafer, K.; Choudhry, A.; Cai, Y.; Li, S.; and Dubrawski, A. 2024. Moment: A family of open time-series foundation models. *arXiv preprint arXiv:2402.03885*.

Grabocka, J.; Scholz, R.; and Schmidt-Thieme, L. 2019. Learning surrogate losses. *arXiv preprint arXiv:1905.10108*.

Granger, C. W. J.; and Newbold, P. 2014. *Forecasting economic time series*. Academic press.

He, Y.; and Zhao, J. 2019. Temporal convolutional networks for anomaly detection in time series. In *Journal of Physics: Conference Series*, volume 1213, 042050. IOP Publishing.

Hounie, I.; Porras-Valenzuela, J.; and Ribeiro, A. 2024. Loss shaping constraints for long-term time series forecasting. In *Proceedings of the 41st International Conference on Machine Learning*, 19062–19084.

Hyndman, R. J.; and Athanasopoulos, G. 2015. 8.9 Seasonal ARIMA models. *Forecasting: principles and practice. oTexts. Retrieved*, 19.

Ju, H.; Kim, S.; Kim, Y.; and Shim, B. 2022. Energy-efficient ultra-dense network with deep reinforcement learning. *IEEE Transactions on Wireless Communications*, 21(8): 6539–6552.

Levine, S.; Finn, C.; Darrell, T.; and Abbeel, P. 2016. End-to-End Training of Deep Visuomotor Policies. arXiv. *URL: https://arxiv. org/abs/1504.00702*.

Li, N.; Arnold, D. M.; Down, D. G.; Barty, R.; Blake, J.; Chiang, F.; Courtney, T.; Waito, M.; Trifunov, R.; and Heddle, N. M. 2022. 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.

Liu, Y.; Hu, T.; Zhang, H.; Wu, H.; Wang, S.; Ma, L.; and Long, M. 2023. itransformer: Inverted transformers are effective for time series forecasting. *arXiv preprint arXiv:2310.06625*.

Liu, Y.; Hu, T.; Zhang, H.; Wu, H.; Wang, S.; Ma, L.; and Long, M. 2024a. iTransformer: Inverted Transformers Are Effective for Time Series Forecasting. arXiv:2310.06625.

Liu, Y.; Wu, H.; Wang, J.; and Long, M. 2022. Non-stationary transformers: Exploring the stationarity in time series forecasting. *Advances in neural information processing systems*, 35: 9881–9893.

Liu, Y.; Zhang, H.; Li, C.; Huang, X.; Wang, J.; and Long, M. 2024b. Timer: Transformers for time series analysis at scale. *arXiv e-prints*, arXiv–2402.

Loshchilov, I.; and Hutter, F. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Lozano, J. X. S.; Ayala-Romero, J. A.; Garcia-Saavedra, A.; and Costa-Perez, X. 2025. Kairos: Energy-efficient radio unit control for O-RAN via advanced sleep modes. In *IEEE INFOCOM 2025-IEEE Conference on Computer Communications*, 1–10. IEEE.

Makridakis, S.; Spiliotis, E.; and Assimakopoulos, V. 2022. M5 accuracy competition: Results, findings, and conclusions. *International Journal of Forecasting*, 38(4): 1346–1364.

Martín, L.; Zarzalejo, L. F.; Polo, J.; Navarro, A.; Marchante, R.; and Cony, M. 2010. 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.

Nie, Y.; Nguyen, N. H.; Sinthong, P.; and Kalagnanam, J. 2022. A time series is worth 64 words: Long-term forecasting with transformers. *arXiv preprint arXiv:2211.14730*.

Nie, Y.; Nguyen, N. H.; Sinthong, P.; and Kalagnanam, J. 2023. A Time Series is Worth 64 Words: Long-term Forecasting with Transformers. arXiv:2211.14730.

Park, J.; Lee, J.; Cho, Y.; Shin, W.; Kim, D.; Choo, J.; and Choi, E. 2023. Deep imbalanced time-series forecasting via local discrepancy density. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 139–155. Springer.

Qi, M.; Shi, Y.; Qi, Y.; Ma, C.; Yuan, R.; Wu, D.; and Shen, Z.-J. 2023. A practical end-to-end inventory management model with deep learning. *Management Science*, 69(2): 759–773.

Qian, Z.; Pei, Y.; Zareipour, H.; and Chen, N. 2019. A review and discussion of decomposition-based hybrid models for wind energy forecasting applications. *Applied energy*, 235: 939–953.

Ray, E. L.; Wang, Y.; Wolfinger, R. D.; and Reich, N. G. 2025. Flusion: Integrating multiple data sources for accurate influenza predictions. *Epidemics*, 50: 100810.

Repository, U. M. L. 2014. ElectricityLoadDiagrams20112014. https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014.

Shalev-Shwartz, S.; and Ben-David, S. 2014. *Understanding machine learning: From theory to algorithms*. Cambridge university press.

Siami-Namini, S.; Tavakoli, N.; and Namin, A. S. 2019. The performance of LSTM and BiLSTM in forecasting time series. In *2019 IEEE International conference on big data (Big Data)*, 3285–3292. IEEE.

Wang, K.; Babenko, B.; and Belongie, S. 2011. End-to-end scene text recognition. In *2011 International conference on computer vision*, 1457–1464. IEEE.

Wang, S.; Wu, H.; Shi, X.; Hu, T.; Luo, H.; Ma, L.; Zhang, J. Y.; and Zhou, J. 2024. TimeMixer: Decomposable Multiscale Mixing for Time Series Forecasting. arXiv:2405.14616.

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

Wu, H.; Xu, J.; Wang, J.; and Long, M. 2021a. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in neural information processing systems*, 34: 22419–22430.

Wu, H.; Zhou, H.; Long, M.; and Wang, J. 2023. Interpretable weather forecasting for worldwide stations with a unified deep model. *Nature Machine Intelligence*, 5(6): 602–611.

Wu, Q.; Chen, X.; Zhou, Z.; Chen, L.; and Zhang, J. 2021b. Deep reinforcement learning with spatio-temporal traffic forecasting for data-driven base station sleep control. *IEEE/ACM transactions on networking*, 29(2): 935–948.

Xue, W.; Zhou, T.; Wen, Q.; Gao, J.; Ding, B.; and Jin, R. 2023. Card: Channel aligned robust blend transformer for time series forecasting. *arXiv preprint arXiv:2305.12095*.

Zeng, A.; Chen, M.; Zhang, L.; and Xu, Q. 2023. Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, 11121–11128.

Zhang, J.; and Man, K.-F. 1998. Time series prediction using RNN in multi-dimension embedding phase space. In *SMC'98 conference proceedings. 1998 IEEE international conference on systems, man, and cybernetics (cat. no. 98CH36218)*, volume 2, 1868–1873. IEEE.

Zhang, Y.; and Yan, J. 2023. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *The eleventh international conference on learning representations*.

Zhou, H.; Zhang, S.; Peng, J.; Zhang, S.; Li, J.; Xiong, H.; and Zhang, W. 2021. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, 11106–11115.

Zhou, T.; Ma, Z.; Wen, Q.; Wang, X.; Sun, L.; and Jin, R. 2022. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International conference on machine learning*, 27268–27286. PMLR.