Performative Time-Series Forecasting

Zhiyuan Zhao
leozhao1997@gatech.edu
Georgia Institute of Technology
Atlanta, GA, USA

Haoxin Liu
hliu763@gatech.edu
Georgia Institute of Technology
Atlanta, GA, USA

Alexander Rodríguez
alrodri@umich.edu
University of Michigan
Ann Arbor, MI, USA

B. Aditya Prakash
badityap@cc.gatech.edu
Georgia Institute of Technology
Atlanta, GA, USA

Abstract

Time-series forecasting is a critical challenge in various domains and has witnessed substantial progress in recent years. Many real-life scenarios, such as public health, economics, and social applications, involve feedback loops where predictive models can trigger actions that influence the outcome they aim to predict, subsequently altering the target variable's distribution. This phenomenon, known as *performativity*, introduces the potential for 'self-negating' or 'self-fulfilling' predictions. Despite extensive studies on performativity in classification problems across domains, this phenomenon remains largely unexplored in the context of time-series forecasting from a machine-learning perspective.

In this paper, we formalize *Performative Time-Series Forecasting* (PeTS), addressing the challenge for predictions when performativity-induced distribution shifts are possible. We propose a novel solution to PeTS, *Feature Performative-Shifting* (FPS), which leverages the concept of delayed response to anticipate distribution shifts and subsequently predicts targets. We provide theoretical insights suggesting that FPS potentially leads to reduced generalization error. Extensive experiment results demonstrate that FPS consistently outperforms conventional time-series forecasting and concept drift methods, highlighting its efficacy in handling performativity-induced challenges.

CCS Concepts

• **Computing methodologies** → **Neural networks**.

Keywords

Time-Series Forecasting, Performativity, Distribution Shift

ACM Reference Format:
Zhiyuan Zhao, Haoxin Liu, Alexander Rodríguez, and B. Aditya Prakash. 2025. Performative Time-Series Forecasting. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2 (KDD '25), August 3–7, 2025, Toronto, ON, Canada.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3711896.3737078



This work is licensed under a Creative Commons Attribution 4.0 International License.
KDD '25, Toronto, ON, Canada
© 2025 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-1454-2/2025/08
https://doi.org/10.1145/3711896.3737078

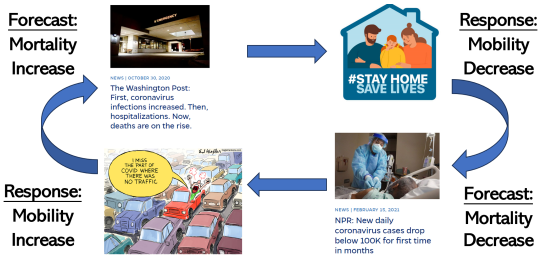


Figure 1: Example of performative feedback loop for COVID-19 mortality and mobility.

1 Introduction

Time-series forecasting is a fundamental task in time-series analysis that finds applications in various domains such as economics, urban computing, and epidemiology [30, 42, 50, 52]. These applications involve predicting future trends or events based on historical time-series data. For example, economists use forecasts to make financial and marketing plans, while sociologists use them to allocate resources and formulate policies for traffic or disease control.

The advent of deep learning methods has revolutionized time-series forecasting [20, 37, 44, 46, 51]. Despite existing models have achieved state-of-the-art performance by capturing long-term dependencies using well-designed attentions, these models are developed without considering feedback loops between input features and targets. These models neglect the fact that predictions can influence the outcomes they aim to predict, a phenomenon referred to as *performativity* [39]. While performativity has been extensively studied in literature such as strategic classification and reinforcement learning [3, 4, 27], it remains largely unexplored in time-series forecasting problems.

Performativity shows as 'self-negating' or 'self-fulfilling' predictions in real-world scenarios. For instance, traffic predictions can influence traffic patterns, and disease forecasts can affect behavior interventions, impacting future predictions. Figure 1 illustrates a performativity feedback loop between COVID-19 mortality predictions and mobility features. In this feedback loop, forecasting during a severe pandemic leads to more stringent interventions, resulting in reduced mobility and subsequently reduced mortality, and vice versa. Realizing the prevalence of performativity prompts studying its effects on time-series forecasting. Despite recent theoretical work of performativity in supervised learning for classification

problems [4, 29, 31, 39], performativity in time-series forecasting remains understudied both theoretically and practically. This paper formally defines the Performative Time-Series (PeTS) forecasting problem, which aims to make robust predictions in the presence of performativity-induced distribution shifts. Our main contributions are:

(1) **Novel Problem:** We introduce the novel performative time-series forecasting (PeTS) problem, which focuses on providing robust forecasts under the influence of performativity [39]. This refers to changes in the distribution of the target variable caused by the predictions themselves due to feedback loops.
(2) **New Methodology:** We propose a natural solution to PeTS, Feature Performative-Shifting (FPS), which anticipates and incorporates responses resulting from performativity into the forecasting process. We provide theoretical insights into how using delayed responses contributes to better forecasting performance. Importantly, FPS is a model-agnostic framework that can be integrated with existing time-series forecasting models.
(3) **Extensive Experiments:** We evaluate FPS on various time-series models and datasets for both standard and real-time forecasting settings. Extensive evaluation results show benefits of FPS for solving PeTS, as well as outperformance than existing concept drift methods, in forecasting accuracy and trend-change capturing ability.

2 Related Work

Time-Series Forecasting. Classical time series models such as ARIMA [12] focus on univariate time series which predicts each time series independently. Other statistical methods treat time-series forecasting as standard regression problems with time-varying parameters, solving with conventional regression techniques, such as kernel regression, support vector regression, and Gaussian processes [35, 45, 48].

Recent works in deep learning have achieved notable achievements in time-series forecasting, such as LSTNet, N-BEATS, and S4 [9, 20, 38]. Other deep learning methods have built upon the successes of self-attention mechanisms [47] with transformer-based architectures, such as Informer, Autoformer, PatchTST, and more [13, 26, 37, 49, 51]. These transformer-based models exhibit accurate forecasting accuracy through capturing long-range dependencies.

Concept Drift. Learning under non-stationary distributions, where the conditional distribution from exogenous covariates to the target changes with time, known as *concept drift*, has attracted attention from the learning theory community [3, 16]. Concept shift methods [1, 2, 15, 40, 43] assume instances sampled from diverse environments and propose to identify and utilize invariant predictors across environments. In time series data, concept shift represents a common yet challenging issue, given the temporal changes of mapping relationships and the challenges in acquiring adequate environment labels under the conventional time-series forecasting setups [22]. Other recent studies have explored multimodal time series analysis, which enhances numerical forecasting by incorporating additional contextual information [25]. Such analyses introduce new reasoning capabilities that hold promise for improved causal inference and mitigating concept drift in time series [23**?** , 24].

Performative Prediction. Performativity severs a particular case of concept drifts that states the phenomenon that the predictive model influences the distribution of future covariates through responses taken based on the model's predictions. This phenomenon has been well-studied in domains including social sciences and economics [11, 28]. In contrast to considering arbitrary shifts in general concept drifts, performative prediction problems, such as strategic classification [5, 31, 39], elaborates concept drifts with specific orientations of distribution shifts and learning objectives.

3 Preliminaries

Performative prediction involves making predictions where the predictive model θ affects the underlying data distribution $\mathcal{D}(\theta)$. Prior statistical works focus on a specific case problem of performative prediction called strategic classification [8, 10, 29, 31, 39]. In strategic classification, individuals strategically modify their input features before undergoing classification. The key challenge lies in anticipating the strategic responses to the classifier, drawn from the performativity-induced distribution $\mathcal{D}(\theta)$. This problem can be framed as an optimization problem:

$$\theta^* = \arg\min_{\theta} \mathop{\mathbb{E}}_{\mathcal{Z}\sim\mathcal{D}(\theta)} \mathcal{L}(\mathcal{Z};\theta) \tag{1}$$

Thus, the fundamental difficulty in strategic classification lies in capturing the distributional shift $\mathcal{D}(\cdot)$ induced by θ. Existing works [4, 39] address this challenge by modeling strategic behavior as an optimization problem: $\boldsymbol{x}_{br} = \arg\max_{\boldsymbol{x}'} u(\boldsymbol{x}', \theta) - c(\boldsymbol{x}', \boldsymbol{x})$, where $\boldsymbol{x}$ is the original feature, and $\boldsymbol{x}_{br}$ is the optimal strategic response (often termed as "best response"). $u(\cdot)$ and $c(\cdot)$ represent designed utility and cost functions respectively. The utility function quantifies the benefit for agents in modifying their distribution from $\mathcal{D}$ to $\mathcal{D}(\theta)$, while the cost function captures the associated loss due to this shift. Once $\boldsymbol{x}_{br}$ is determined, it can be used to recalibrate parameters θ, initiating a new best response and leading to a feedback loop. An iterative approach, as demonstrated in [39], simulates this feedback loop to solve the strategic classification problem. This iterative process ultimately converges to a stable optimum where θ effectively classifies samples drawn from $\mathcal{D}(\theta)$.

A classic example is a bank employing a classifier to predict the creditworthiness of loan applicants. Individuals react to the bank's classifier by adjusting features (e.g., credit line utilization) strategically, invoking the performative distribution $\mathcal{D}(\theta)$ to influence favorable classification outcomes.

4 Problem Formulation

Time-Series Forecasting. Time-series forecasting involves predicting future values of one or more dependent time series based on historical data, potentially augmented with exogenous covariate features. We denote the target time series as $\mathbf{Y}$ and its associated exogenous covariate features as $\mathbf{X}$. At any time step t, time-series forecasting aims to predict $\mathbf{Y}_t^H = [y_{t+1}, y_{t+2}, \ldots, y_{t+H}] \in \mathbf{Y}$ using historical data $(\mathbf{X}_t^L, \mathbf{Y}_t^L)$, where L represents the length of the historical data window, known as the *lookback window*, and H denotes the forecasting time steps, known as the *horizon window*. Here, $\mathbf{X}_t^L =$

$[x_{t-L+1}, x_{t-L+2}, \ldots, x_t] \in \mathbf{X}$ and $\mathbf{Y}_t^L = [y_{t-L+1}, y_{t-L+2}, \ldots, y_t] \in \mathbf{Y}$. For simplicity, we denote $\mathbf{Y}^H = \{\mathbf{Y}_t^H\}$ for $\forall t$ as the collection of horizon time-series of all time steps, similar for $\mathbf{Y}^L$ and $\mathbf{X}^L$. Conventional time-series forecasting aims to learn a model θ to obtain $\theta : (\mathbf{X}^L, \mathbf{Y}^L) \rightarrow \mathbf{Y}^H$ that models the distribution $P(\mathbf{Y}^H) = P(\mathbf{Y}^H|\mathbf{Y}^L)P(\mathbf{Y}^L) + P(\mathbf{Y}^H|\mathbf{X}^L)P(\mathbf{X}^L)$. This optimization is achieved through the empirical risk minimization (ERM):

$$\theta^* = \arg\min_{\theta} \mathop{\mathbb{E}}_{\mathcal{Z}\sim\mathcal{D}} \mathcal{L}(\mathcal{Z}; \theta) \tag{2}$$

where $\mathcal{D}$ denotes the distribution of training data, and $\mathcal{Z}$ are the samples drawn from the training data, i.e., $[(\mathbf{X}_t^L, \mathbf{Y}_t^L), \mathbf{Y}_t^H]$ for a time step t.

Performative Time-Series Forecasting. In the performative setup of time-series forecasting, predictions can influence future data points and subsequent decision-making. Unlike traditional forecasting, which assumes data is drawn from a fixed distribution $\mathcal{D}$, performative forecasting considers samples drawn from a dynamic distribution that exogenous features can response to the forecasting model θ. We refer to this evolving distribution as the performative distribution.

Nevertheless, time-series forecasting features distinct differences compared to strategic classification problems. While predictive models in time series can influence actions, such as COVID-19 mortality and mobility forecasts as shown in Figure 1, these actions require a response time after predictions are made. This contrasts with the core assumption in strategic classification that have been broadly studied, where individuals can strategically modify their features before the predictive model is applied. That is, in the COVID-19 mortality and mobility forecasts example, individuals can only respond after the model predicts an outbreak, rather than preemptively altering their behavior before the prediction occurs.

Thus, unlike the hypothetical gaming in strategic classification, performative time-series forecasting involves predicting targets under a performative distribution that evolves over time. Instead of drawing samples from a dynamic distribution solely dependent on θ, as in strategic classification, performative time-series forecasting requires optimizing under a co-evolving distribution that depends on both θ and time t, i.e., $\mathcal{D}(\theta, t)$. As such, we define the performative time-series forecasting as:

Definition (PeTS). Performative time-series forecasting (PeTS) involves making predictions under dynamic distribution shifts, where the distribution shift is induced by the model parameters θ and evolves over time t, by optimizing:

$$\theta^*_{\text{PeTS}} = \arg\min_{\theta} \mathop{\mathbb{E}}_{\mathcal{Z}(t)\sim\mathcal{D}(\theta,t)} \mathcal{L}(\mathcal{Z}(t); \theta) \tag{3}$$

Here, we denote the samples $\mathcal{Z}(t)$ that is dependent on t, as $\mathcal{D}(\theta, t)$ evolves with t and the samples are drawn from different time steps. We illustrate the difference between previous performative prediction studies and our performative time-series forecasting problems as shown in Figure 2.

Since the distribution $\mathcal{D}(\theta, t)$ depends on both θ and t, and θ aims to minimize the empirical loss using data drawn from the dynamic distribution itself, a feedback loop—referred to as the performative feedback loop—naturally arises in PeTS over time. This feedback loop can be described as the interactions between target time series and exogenous time series over time, using the multiplication of a series of conditional distributions, following the formulation:

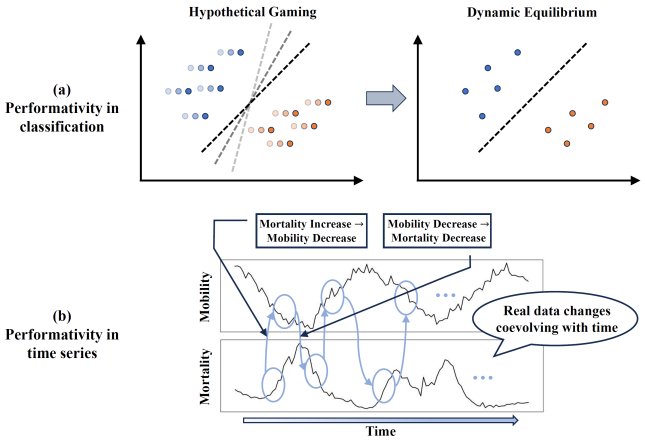


Figure 2: Comparison of performativity in strategic classification (synthetic example) and time-series forecasting (real COVID-19 example): (a) Strategic classification involves a hypothetical interaction between features and decisions, aiming to find a dynamic equilibrium. (b) Performative time-series forecasting, in contrast, deals with real data changes that evolve over time t rather than hypothetical adaptations.

$$\begin{aligned} P(\mathbf{Y}_t^H) =& P(\mathbf{Y}_t^H|\mathbf{X}_{t+k_i}^L)P(\mathbf{X}_{t+k_i}^L|\mathbf{Y}_{t+k_{i-1}}^L)\ldots \\ & P(\mathbf{Y}_{t+k_2}^L|\mathbf{X}_{t+k_1}^L)P(\mathbf{X}_{t+k_1}^L|\mathbf{Y}_t^L)P(\mathbf{Y}_t^L), \\ & where\ \ 0 \le k_1 \le k_2 \le \ldots k_i \le H \in \mathbb{N} \end{aligned} \tag{4}$$

Intuitively, Equation 4 illustrates how performativity influences the evolution of distributions over time, forming a performative feedback loop in time-series data. A decision made at time t, e.g. $P(\mathbf{Y}_t^L)$, can trigger a response from individuals after a time delay k_1, leading to a shift in the distribution of $P(\mathbf{X}_{t+k_1}^L)$. This altered distribution of exogenous features can subsequently impact back to the target time series after an additional delay k_2, resulting in a further distribution shift of $P(\mathbf{Y}_{t+k_2}^L)$, so on and so forth. The Figure 2.(b) showcases the feedback loop with the COVID-19 mortality and mobility example.

To address performative time-series forecasting, a natural solution involves both identifying the temporal correlations between the target and exogenous features—such as determining appropriate delays k_1 through k_i—and modeling the conditional distributions in Equation 4 to accurately capture the performative feedback loop.

Performative vs. Non-performative Features. In practice, the performative distribution shift may not happen to all features. For instance, in COVID-19 forecasting, social mobility may shift in response to current forecasting results, while age distribution cannot shift whatever forecasting results are made. Following the preliminary setups in [39], we distinguish performative features and non-performative features in multivariate time series by defining:

Definition (Performative Feature). The performative feature is a class of features whose distribution can actively shift as a response to the model predictions.

In general, multivariate time-series data may contain both performative and non-performative features. PETS studies the distribution shift only over performative features. Forecasting models of solving PETS hence take inputs from the concatenation shifted performative features and non-shifted non-performative features to conduct the target predictions.

Performativity vs. Concept Drift. Both performativity and concept drift delve into learning when the distributions of data change over time. While performativity specifically addresses the shift in the distribution of performative features due to predictive models, concept drift encompasses broader sources of distributional shift. To clarify, performativity studies the distribution shifts with specific shift orientation within the wider concept drift framework. Concept drift, in its broader scope, considers any arbitrary sources causing distributional changes.

5 Methodology

While performativity plays a role in both strategic classification and time-series forecasting, solving performative time-series forecasting presents unique challenges. First, as shown in the comparison between Equation 1 and Equation 3, PETS involves a performative distribution shift that is not only induced by θ but also evolves with time t. This temporal dependent dynamic violates the fundamental assumption of strategic classification, making existing frameworks unsuitable for PETS. Second, the temporally dependent feedback loop, as shown in Equation 4, involves multiple interactions between target and exogenous features, making it particularly challenging to explicitly anticipate the distribution shift by modeling all associated conditional distributions.

Methodology Overview. Given the difficulty of both adapting existing frameworks and modeling all distribution shifts. We thereby propose an alternative shortcut to address PETS. On realizing the difficulty explicitly modeling all conditional distributions as in Equation 4, we propose a shortcut that models:

$$\mathrm{P}(\mathbf{Y}_t^H) = \mathrm{P}(\mathbf{Y}_t^H|\mathbf{X}_{t+k_i}^L)\mathrm{P}(\mathbf{X}_{t+k_i}^L|\mathbf{X}_t^L)\mathrm{P}(\mathbf{X}_t^L) \tag{5}$$

The intuition behind Equation 5 is to simplify the complex performative feedback loop from Equation 4 into a two-stage forecasting process: an autoregressive process $\mathrm{P}(\mathbf{X}_{t+k_i}^L|\mathbf{X}_t^L)$ and a forecasting process $\mathrm{P}(\mathbf{Y}_t^H|\mathbf{X}_{t+k_i}^L)$. The intuition of this simplification is in twofold: First, since the ultimate objective of time-series forecasting is to accurately predict the target $\mathrm{P}(\mathbf{Y}_t^H)$, the most critical task is to model the deterministic conditional distribution $P(\mathbf{Y}_t^H|\mathbf{X}_{t+k_i}^L)$, given that $\mathbf{X}_{t+k_i}^L$ directly influences $\mathbf{Y}_t^H$ in the performative feedback loop. Second, while modeling $P(\mathbf{Y}_t^H|\mathbf{X}_{t+k_i}^L)$ can be achieved by conventional time-series forecasting models, it also presents two subsequent challenges: obtaining $\mathbf{X}_{t+k_i}^L$ and estimating it effectively. To estimate, a natural approach for estimating $\mathbf{X}_{t+k_i}^L$ is through an autoregressive process by modeling $\mathrm{P}(\mathbf{X}_{t+k_i}^L|\mathbf{X}_t^L)$. To determine $\mathbf{X}_{t+k_i}^L$, we propose identifying the lagging effects of $\mathbf{X}$ on $\mathbf{Y}$ by maximizing their similarity over time. By finding this lag time, we determine k_i and thus obtain $\mathbf{X}_{t+k_i}^L$. For more straightforward intuitivity, we refer to k_i as the delay τ and define $\mathbf{X}_{t+k_i}^L$ as the "**delayed response**", denoted as $\mathbf{X}^{DR} = \{\mathbf{X}_{t+\tau}^L\}$.

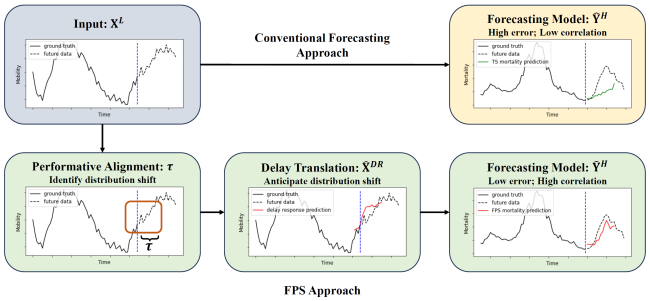


Figure 3: Comparison between the conventional forecasting approach and FPS. FPS utilizes a two-stage forecasting framework, which firstly estimates the delayed responses and then uses it to forecast the target variable (exampled on COVID-19 mobility and mortality data, trend changing of mobility is ahead of mortality).

By simplifying the performative feedback loop into this two-stage forecasting problem, we develop a more practical approach compared to directly solving Equation 4. For clearer notation and more intuitive understanding, we reformulate the two-step process in Equation 5 using the "delayed response" notation: The first step estimates the "delayed response" $\mathbf{X}^{DR}$ from historical data $\mathbf{X}^L$, while the second step forecasts the target $\mathbf{Y}^H$ using the estimated $\mathbf{X}^{DR}$. This is equivalent to:

$$\mathrm{P}(\mathbf{Y}^H) = \mathrm{P}(\mathbf{Y}^H|\mathbf{X}^{DR})\mathrm{P}(\mathbf{X}^{DR}|\mathbf{X}^L)\mathrm{P}(\mathbf{X}^L) \tag{6}$$

Based on the above intuition and Equation 6, we then propose *Feature Performative Shifting* (FPS) to address PETS. FPS consists of three components, shown in Figure 3: (1) Performative Time-Series Alignment, that determines the delay time τ such that $\mathbf{X}^{DR}$ causes $\mathbf{Y}^H$; (2) Delay Translation Module, that models the translation of $\mathrm{P}(\mathbf{X}^{DR}|\mathbf{X}^L)$; and (3) Forecasting Module, that forecasts target $\mathbf{Y}^H$ through modeling $\mathrm{P}(\mathbf{Y}^H|\mathbf{X}^{DR})$.

5.1 Performative Time-Series Alignment

To address the challenge of anticipating the distribution shift in PETS, it is crucial to determine the delay time τ where the shifted distribution $\mathbf{X}^{DR}$ directly causes $\mathbf{Y}^H$. Consider the COVID-19 forecasting example at a specific time step t: the performative feedback loop can be viewed as a two-step process. First, a prediction of reduced mortality can lead to increased mobility; second, this increased mobility can cause another outbreak of the pandemic. Viewing the prediction of the pandemic outbreak as the target $\mathbf{Y}_t^H$, the increase in mobility is the direct cause of this target, making $\mathbf{X}_t^{DR} \equiv \mathbf{X}_{t+\tau}^L$ share a similar trend with the target series.

Thus, the key idea of Performative Time-Series Alignment is to determine the delay time τ by maximizing the similarity between the target series and a sliding window of the exogenous features on the full training data, defined as:

$$\begin{aligned}\tau = \arg\max_i |\mathrm{Similarity}([x_i, \ldots, x_{T-H+i}], \\ [y_H, \ldots, y_T])|, \quad i \in [0, H]\end{aligned} \tag{7}$$

The concept of assessing the similarity between delayed responses and target predictions can be extended to general time-series forecasting tasks. If an input feature positively correlates with the target prediction, a trend change in the delayed response will induce a similar trend change in the target prediction, maximizing the similarity. Conversely, if the input feature negatively correlates with the target prediction, a trend change in the delayed response will lead to an opposite trend change in the target prediction, minimizing the similarity while maximizing its absolute value. This alignment can be achieved by calculating similarity measures between input and target time series at various time shifts, similar to the widely-used Dynamic Time Warping technique [34]. We employ cosine similarity, defined as $\mathrm{CS}(\mathbf{X}, \mathbf{Y}) = \frac{\mathbf{X}\cdot\mathbf{Y}}{|\mathbf{X}||\mathbf{Y}|}$, to measure the similarity between input performative features and target time series [36]. When input performative features are multivariate, each dimension is independently aligned with the target. Different metrics can be used to obtain τ, such as maximizing Pearson Correlation or minimizing Euclidean Distance [6, 18].

5.2 Delay Translation Module

To address the challenge of unobserved future data for delayed responses $\mathbf{X}^{DR}$ during testing as $t + \tau$ can possibly larger than T, we introduce the Delay Translation Module to model the translation $\mathrm{P}(\mathbf{X}^{DR}|\mathbf{X}^{L})$. This module accurately predicts the delayed response based on the present input feature, implicitly anticipating the distribution shift from $\mathcal{D}$ to $\mathcal{D}(\theta, t)$ in Equation 6 in Equation 3. Consequently, the task of anticipating the distribution shift can be reframed as an autoregressive problem, which has been well-established [12]. The Delay Translation Module, denoted as f_τ, addresses this autoregressive task using sequence-to-sequence mapping neural networks, characterizing better approximation capability and generalization ability. We employ Recurrent Neural Networks (RNNs), a common and simple approach for sequence predictions, which is optimized by minimizing the empirical mean squared error (MSE) risk:

$$\mathcal{L}_{DT}(f_\tau) = \mathrm{MSE}(\ \mathbf{X}^{DR},\ \hat{\mathbf{X}}^{DR} = f_\tau(\mathbf{X}^{L})\ ) \tag{8}$$

In practice, while this process may introduce additional errors, it generally captures trend changes in advance, providing sufficient support for target prediction. Meanwhile, although simple RNNs have consistently outperformed other models in empirical evaluations, more advanced architectures, such as DeepAR [44] or Transformer-based models, can also be employed to further enhance performance in autoregressive delay translation tasks.

5.3 Forecasting Module

After obtaining the estimated delayed response $\hat{\mathbf{X}}^{DR} = f_\tau(\mathbf{X}^{L})$ from the Delay Translation Module, the problem turns to a conventional time-series forecasting involving multivariate time-series data. The role of the forecasting module is to predict the targets $\mathbf{Y}^{H}$ using $\hat{\mathbf{X}}^{DR}$ along with non-performative features. For ease of reference, we represent $\hat{\mathbf{X}}^{DR}$ as the concatenation of the estimated delayed response and non-performative features. Recognizing the delayed responses $\mathbf{X}^{DR}$ are often unavailable as future time series, the forecasting module employs the estimated $\hat{\mathbf{X}}^{DR}$ to forecast the targets during training, validation, and testing.

The objective of the time-series forecasting module, denoted as g_τ, is to minimize the risk of the designed loss function, which is usually mean squared error (MSE):

$$\mathcal{L}_{TS}(g_\tau) = \mathrm{MSE}(\ \mathbf{Y}^{H},\ \hat{\mathbf{Y}}^{H} = g_\tau(\hat{\mathbf{X}}^{DR}, \mathbf{Y}^{L})\ ) \tag{9}$$

The key difference between the forecasting module in FPS and conventional time-series forecasting is that FPS employs estimated delayed responses $\hat{\mathbf{X}}^{DR}$ in the horizon window, which models the direct cause to target predictions. In contrast, conventional time-series forecasting simply uses $\mathbf{X}^{L}$ from the lookback window, which does not consider performativity and may model incorrect causal effects among time steps, as $\mathbf{X}^{L}$ may not directly cause $\mathbf{Y}^{H}$ at many cases.

The choice of the forecasting module is independent of the delay translation module, allowing for the flexible selection of any state-of-the-art time-series forecasting model. The loss objective for the forecasting module may include additional penalties, depending on the chosen model.

5.4 Feature Performative-Shifting - FPS

We integrate all the aforementioned modules to formulate the solution for PETS: *Feature Performative-Shifting* (FPS). In FPS, we first determine the time delay τ through performative time-series alignment. We then co-optimize the delay translation module $f_\tau(\cdot)$ and the forecasting module $g_\tau(\cdot)$. The complete model of FPS is denoted as $\theta_{\mathrm{FPS}} = g_\tau(f_\tau(\cdot))$. The joint optimization of the delay translation module and the forecasting module is accomplished by minimizing the combined loss:

$$\mathcal{L}_{\mathrm{FPS}} = \lambda_1 \mathcal{L}_{DT}(f_\tau) + \lambda_2 \mathcal{L}_{TS}(g_\tau) \tag{10}$$

Here, λ_1 and λ_2 act as hyperparameters, allowing the weighting of loss terms without loss of generality; both can be set to 1 with proven effectiveness in practice. The detailed training process of FPS is outlined in Algorithm 1. Intuitively, FPS is a two-stage forward algorithm: first, it predicts the intermediate target—the delayed responses due to the performative distribution shift—and then forecasts the desired target using the predicted delayed responses.

While FPS concurrently trains both the delay translation and forecasting modules, it is also viable to train these two modules independently in practice. Specifically, the delay translation module can be pre-trained or updated with lower frequency, providing FPS with enhanced efficiency and flexibility in real-time forecasting tasks.

During evaluation, we assume that the delay time τ determined during training also holds for the validation and testing data. Based on this assumption, FPS processes the input features through the delay translation module and subsequently through the forecasting module. For given inputs, the final forecasting out of FPS aligns same with conventional time-series methods optimized through ERM.

Analysis of FPS: Potential for tightening PAC bounds. To better understand why FPS may improve forecasting accuracy, we propose the following two propositions to explain the performance disparities between conventional forecasting approaches and FPS:

PROPOSITION 5.1. *Assume $\mathbf{X}^{DR}$ is known for both training and testing, and $\mathbf{Y}^{H}$ is known for training yet not in testing, employing*

Algorithm 1 Feature Performative-Shifting (FPS)

Input: time-series dataset $\mathcal{Z} = \{\mathbf{X}, \mathbf{Y}\}$, max time T, lookback L, horizon H, epoch E, initial model θ_0
Output: θ_{FPS} for solving PeTS
Initialize $\tau = 0$, $\text{CS}_{max} = 0$
for $i = 0, \ldots, H$ **do**
  Compute $\text{CS}([x_i, \ldots, x_{T-H+i}], [y_H, \ldots, y_T])$
  if $|\text{CS}| \geq \text{CS}_{max}$ **then**
    $\tau = i$, $\text{CS}_{max} = \text{CS}$
  end if
end for
Sample training data $\mathbf{X}^L$, $\mathbf{X}^{DR} = \{\mathbf{X}^L_{t+\tau}\}$, $\mathbf{Y}^L$, $\mathbf{Y}^H$
for epoch $= 1, \cdots, E$ **do**
  Compute $\hat{\mathbf{X}}^{DR} = f_\tau(\mathbf{X}^L)$
  Compute $\hat{\mathbf{Y}}^H = g_\tau(\hat{\mathbf{X}}^{DR}, \mathbf{Y}^L)$
  Compute $\mathcal{L}_{\text{FPS}}$ and update $\theta_i = g_\tau(f_\tau(\cdot))$ with gradient-based method.
end for
return $\theta_{\text{FPS}} = \theta_E$

$\mathbf{X}^{DR}$ to forecast $\mathbf{Y}^H$ yields a tighter PAC bound compared to using original input sequences.

Proposition 5.2. *Assume both $\mathbf{X}^{DR}$ and $\mathbf{Y}^H$ are known for training yet not testing, which requires the delay translation module to forecast the estimated delayed responses $\hat{\mathbf{X}}^{DR}$ during the testing stage, FPS reduces empirical loss by increasing model complexity of the PAC bound compared to conventional forecasting methods.*

The overall idea for proving the propositions is to show that the PAC bound is upper bounded by three terms: the empirical loss, the model complexity, and the non-vanishing errors. This is achieved by generalizing and adapting the theorems from previous studies—[33] for stationary cases and [19] for non-stationary cases. We then show that forecasting with $\mathbf{X}^{DR}$ results in a lower empirical error but a higher model complexity, as it involves a two-stage forecasting method. The detailed proof is provided in Appendix I.

Intuitively, Proposition 5.1 describes an ideal setting where $\mathbf{X}^{DR}$, which contains partial future values of exogenous features, is assumed to be known in advance. In this case, forecasting with $\mathbf{X}^{DR}$ results in a lower empirical error, while preserving the same model complexity as forecasting with $\mathbf{X}^L$. This leads to a tighter PAC bound when forecasting with $\mathbf{X}^{DR}$, this is further shown empically through Table 4.

Proposition 5.2 describes the practical setting, where the future values of $\mathbf{X}^{DR}$ are unknown. In this case, while FPS achieves lower empirical error by forecasting with $\hat{\mathbf{X}}^{DR}$, the introduction of the delayed translation module increases model complexity, which introduces a trade-off between lower empircal loss and higher model complexity in the PAC bound.

Indeed, quantatively determining the trade-off between the increase in model complexity and the reduction in empirical error is non-trivial. However, empirical evaluations demonstrate that the reduction in empirical loss outweighs the increase in model complexity in practice. As a result, leveraging forecasted delayed responses, even within FPS, proves beneficial for forecasting performance. This intuition is further shown by Table 1 and Table 2.

6 Experiments

6.1 Setup

Datasets. Our empirical evaluations use the COVID-19 [41] and METR-LA traffic [21] datasets, both of which offer clearly identifiable performative features, such as mobility data in the COVID-19 dataset and traffic volumes from neighboring sensors in the METR-LA dataset. Our forecasting tasks are to predict national and state-wise weekly mortality for the next 8 weeks using the COVID-19 dataset and to forecast traffic volume every 5 minutes for the next 2 hours using the METR-LA dataset.

We evaluate FPS under two settings: standard setting [51] and real-time setting [41]. In standard setting, the time series is divided into 60% for training, 20% for validation, and 20% for testing. For real-time setting, we make weekly re-training and predictions from October 2021 to March 2022 for the COVID-19 dataset and hourly predictions throughout the day for the METR-LA dataset. The first 20% of real-time steps are used for validation. This setup is consistent with prior research [21, 41]. Additional experiment details are available in Appendix II.

Baselines. Our primary objective is to showcase the advantages of FPS compared to existing concept drift and non-stationary approaches in addressing PeTS. We include five baselines (referred to as 'Method' in the result tables): Empirical Risk Minimization (ERM) as defined in Equation 2, two general concept drift methods, GroupDRO [43] and IRM [2], and two non-stationary methods specifically designed for time-series forecasting, RevIN [14] and Dish-TS [7]. We exclude other concept drift baselines, such as SD [40] and IB-ERM [1], as they do not show significant empirical benefits over ERM for time-series forecasting tasks.

Since all methods are model-agnostic frameworks, we include five forecasting models (referred to as 'Model' in the result tables): RNNs, LSTNet [20], Transformer [47], Informer [51], and Autoformer [49]. This inclusion demonstrates that the benefits of FPS are consistent and independent of the specific forecasting model employed.

Evaluation. We measure performance disparities using Normalized Root Mean Squared Error (NRMSE) and Normalized Mean Absolute Error (NMAE) for standard time-series forecasting tasks, and additionally Pearson Correlation (PC) for real-time forecasting tasks. NRMSE and NMAE measure forecasting accuracy, while PC evaluates the models' abilities to capture trends. The detailed metrics' formulations are:

$$\text{NMAE} = \frac{\sum_{N,H} |y_{n,h} - \hat{y}_{n,h}|}{\sum_{N,H} |y_{n,h}|}$$

$$\text{NRMSE} = \frac{\sqrt{\frac{1}{NH} \sum_{N,H} (y_{n,h} - \hat{y}_{n,h})^2}}{\frac{1}{NH} \sum_{N,H} |y_{n,h}|}$$

$$\text{PC} = \frac{\sum_H (y_h - \bar{y}_h)(\hat{y}_h - \bar{\hat{y}}_h)}{\sqrt{\sum_H (y_h - \bar{y}_h)^2 (\hat{y}_h - \bar{\hat{y}}_h)^2}}$$

Here, N and n denote the total number and the index of testing sequences, respectively, while H and h represent the horizon length and the time steps within each test sequence.

Reproducibility. All models are trained on NVIDIA Tesla V100 GPUs. All data and code are available at: https://github.com/AdityaLab/pets.

Table 1: Performance comparison between baselines and FPS for standard time-series forecasting. FPS shows better performance in 19 out of 20 evaluations compared to all baselines across various forecasting models and evaluation metrics.

Dataset	covid									
Model	RNN		LSTNet		Transformer		Informer		Autoformer	
Method \ Metric	NMAE	NRMSE	NMAE	NRMSE	NMAE	NRMSE	NMAE	NRMSE	NMAE	NRMSE
ERM	0.288	0.491	0.223	0.479	0.249	0.491	0.305	0.626	0.810	1.772
GroupDRO	0.263	0.424	0.233	0.381	0.409	0.765	0.352	0.659	0.524	1.008
IRM	0.303	0.501	0.271	0.487	0.548	1.048	0.517	0.995	0.449	0.852
RevIN	0.337	0.669	0.379	0.738	0.478	0.898	0.394	0.744	0.521	1.011
Dish-TS	0.388	0.736	0.291	0.516	0.322	0.530	0.338	0.569	0.442	0.821
FPS (Ours)	**0.194**	**0.321**	**0.185**	**0.311**	**0.179**	**0.259**	**0.231**	**0.396**	**0.370**	**0.678**
Dataset	traffic									
Model	RNN		LSTNet		Transformer		Informer		Autoformer	
Method \ Metric	NMAE	NRMSE	NMAE	NRMSE	NMAE	NRMSE	NMAE	NRMSE	NMAE	NRMSE
ERM	0.141	0.208	0.135	0.196	0.145	0.216	0.141	0.198	0.166	0.217
GroupDRO	0.132	0.194	0.127	0.181	0.151	0.216	0.130	0.187	0.179	0.243
IRM	0.135	0.196	0.125	0.175	0.147	0.218	0.131	0.184	0.191	0.259
RevIN	0.166	0.232	0.163	0.226	0.164	0.225	0.170	0.234	0.196	0.266
Dish-TS	0.135	**0.185**	0.148	0.198	0.135	0.187	0.140	0.190	0.172	0.222
FPS (Ours)	**0.131**	0.194	**0.119**	**0.171**	**0.129**	**0.179**	**0.114**	**0.166**	**0.141**	**0.194**

6.2 Results

Standard Time-Series Forecasting. We conducted experiments on standard time-series forecasting tasks, dividing the data into 60% for training, 20% for validation, and 20% for testing. The evaluation compares the conventional approach (Empirical Risk Minimization, ERM) with two concept drift methods (GroupDRO and IRM), two non-stationary methods (RevIN and Dish-TS), and FPS. The comprehensive results, detailed in Table1, led to three key conclusions:
(1). *Introducing performativity to ERM enhances forecasting accuracy.* FPS can be conceptualized as a two-stage process. In the first stage, it anticipates performative distribution shifts by estimating delayed responses, and in the second stage, it forecasts the target based on these estimations. Both stages of FPS employ ERM, making FPS a two-stage ERM with performativity introduced. Evaluation results consistently demonstrate an improvement in FPS over ERM, with average error reductions of 37.1% and 12.8% for COVID-19 and traffic forecasting tasks, respectively. This improvement highlights the advantage of introducing performativity to ERM in addressing PeTS with FPS.
(2). *FPS better addresses distribution shifts in scenarios with performativity compared to other general distribution shift methods.* The key difference between FPS and existing distribution drift methods lies in the manner of handling distribution shifts. While general distribution drift methods focus on mitigating the effects of distribution shifts by learning environment-invariant patterns or utilizing instance normalization, FPS anticipates distribution shifts due to performativity, thereby enhancing forecasting outcomes. Evaluation results demonstrate that FPS consistently achieves lower forecasting errors than popular concept drift methods on COVID-19 and traffic datasets, whose performativity impacts are clearly presented, outperforming in 19 out of 20 comparisons with time-series non-stationary methods. These findings highlight the benefits of anticipating the impact of distribution shifts in time-series forecasting tasks, rather than merely mitigating or normalizing them. Notably, time-series-specific distribution shift methods, such as RevIN, do not significantly improve short-term forecasting performance, as their instance normalization processes limit potential trend changes during prediction. A detailed analysis, including case studies, can be found in Appendix III.
(3). *Empirical evaluation of FPS validates theoretical analysis.* Recall that Proposition 5.2 introduces the trade-off of FPS to the PAC bound, where FPS yields lower empirical training error but higher model complexity. Our empirical evaluations confirm that the reduction in empirical loss outweighs the increase in model complexity when comparing ERM and FPS, thus leading to lower testing errors. Additionally, experiments directly utilizing $\mathbf{X}^{DR}$ (i.e., future time-series from the horizon window) for traffic predictions show a 56.3% reduction in average forecasting errors except when using Autoformer (Table 4 in Appendix III). These additional results validate Proposition 5.1, demonstrating that forecasting with $\mathbf{X}^{DR}$ yields a lower error bound.
Real-Time Forecasting. We explore real-time forecasting, a critical task with practical applications such as CDC requiring weekly updates on COVID-19 predictions. Real-time forecasting involves sequentially predicting imminent values or events as the most current data becomes available. Practical real-time data often exhibit dynamic patterns, including seasonal variations, interventions, sudden anomalies, and trend changes, posing distribution shift challenges and necessitating adaptable forecasting models capable of adjusting to evolving patterns or abrupt shifts in data.

To evaluate the efficacy of FPS in addressing PeTS within real-time forecasting, we compare its performance with that of ERM in practical real-time scenarios, as detailed in Table 2. We include

Table 2: Performance comparison between ERM and FPS for real-time forecasting tasks. FPS uniformly outperforms ERM on both datasets and all metrics (⇓ means lower is better, ⇑ means higher is better)

		covid			traffic		
Model	Method	NMAE(⇓)	NRMSE(⇓)	PC(⇑)	NMAE(⇓)	NRMSE(⇓)	PC(⇑)
RNNs	ERM	0.454	1.061	0.526	0.168	0.235	0.272
	FPS	**0.405**	**0.877**	**0.661**	**0.138**	**0.198**	**0.562**
LSTNet	ERM	0.324	0.597	0.656	0.177	0.251	0.550
	FPS	**0.261**	**0.570**	**0.818**	**0.130**	**0.190**	**0.723**
Transformer	ERM	0.551	1.048	0.159	0.202	0.277	0.257
	FPS	**0.442**	**0.898**	**0.401**	**0.133**	**0.202**	**0.571**
Informer	ERM	0.392	0.958	0.693	0.142	0.201	0.487
	FPS	**0.340**	**0.883**	**0.791**	**0.138**	**0.200**	**0.651**
Autoformer	ERM	0.435	0.833	0.535	0.183	0.234	0.339
	FPS	**0.312**	**0.568**	**0.697**	**0.140**	**0.194**	**0.482**

Pearson Correlation to highlight differences in trend correlations between FPS and the conventional forecasting method. Concept drift and non-stationary methods are excluded for two reasons: (a) these methods are designed to mitigate distribution shifts for stable predictions, which contrasts with capturing dynamic patterns and rapid trend changes in real-time forecasting, and (b) we have demonstrated that FPS outperforms distribution shift baselines in earlier evaluations.

The evaluation demonstrates that FPS consistently outperforms conventional approaches across all testing metrics. Specifically, FPS achieves an average 16.3% error reduction and a 47.9% improvement in correlations for COVID-19 forecasting, and an average 18.9% error reduction and a 67.1% improvement in correlations for traffic forecasting. The importance of real-time forecasting, as evidenced by the CDC's requirement for timely updates on COVID-19 and flu predictions, underscores the value of FPS. Its adaptive nature makes it a crucial tool for decision-makers who rely on precise and timely forecasts, particularly in domains where timely insights have a significant impact.

Case Study: Capturing Trend Changes with Minimal Delay. To illustrate the benefits of FPS in anticipating performative distribution shifts and effectively learning dynamic patterns, we present a case study on real-time forecasting of COVID-19 trends. This case study uses data up to Week 2022-03 to predict national mortality rates for Weeks 2022-04 through 2022-11. Leading up to Week 2022-03, the national mortality rate had risen steadily due to the Omicron outbreak beginning in December 2021. In response to the foreseen outbreak, public mobility decreased, leading to a subsequent decline in the mortality rate starting from Week 2022-04. This case study highlights how FPS captures the performative shifts associated with the Omicron outbreak, public response, and the subsequent decline in the pandemic.

The case study showcases real-time forecasting results of with and without FPS, with qualitative and quantitative results shown in Figure 4 and Table 3. Qualitative, FPS consistently provides a more accurate prediction of trend changes compared to ERM. Specifically, FPS captures steady and precise decreasing trends, while ERM exhibits delayed decreases. Quantitatively, predictions derived from the FPS model consistently exhibit lower forecasting errors and higher correlations, highlighting its effectiveness in capturing trend changes with minimal time delay. These findings underscore FPS's advantages in detecting rapid and abrupt shifts in trends.

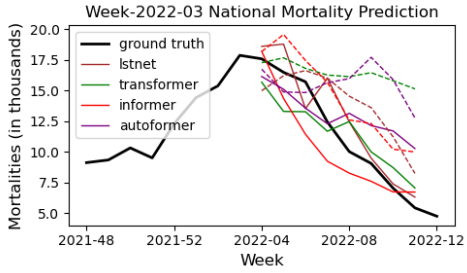


Figure 4: Case Study: Plots of national mortality prediction at the trend turning point, with and without FPS. (Solid and dashed plots for with and without FPS, respectively)

Table 3: Case Study: Comparison between conventional forecasting models and FPS for real-time forecasting at Week 2022-03. Our method FPS uniformly outperforms conventional forecasting models on all metrics.

Model	Method	NMAE(⇓)	NRMSE(⇓)	PC(⇑)
LSTNet	ERM	0.250	0.281	0.845
	FPS	**0.140**	**0.167**	**0.932**
Transformer	ERM	0.409	0.505	0.922
	FPS	**0.160**	**0.172**	**0.939**
Informer	ERM	0.236	0.255	0.911
	FPS	**0.160**	**0.191**	**0.976**
Autoformer	ERM	0.398	0.483	0.223
	FPS	**0.221**	**0.257**	**0.921**

7 Conclusion and Discussion

In this paper, we introduce a novel time-series problem PeTS that addresses the challenge of achieving accurate forecasts in the presence of performativity. We present a natural solution to PeTS, termed FPS, which effectively integrates the notion of delayed responses into time-series forecasting. By incorporating performativity, our proposed approach demonstrates notable improvements over conventional time-series forecasting models, as substantiated by both theoretical analyses and empirical investigations. Specifically, we have showcased the advantages of FPS through comprehensive experiments conducted on real-time COVID-19 and METR-LA traffic prediction tasks. Furthermore, our framework exhibits enhanced generalization capabilities and a heightened ability to capture underlying trends.

Beyond the context of FPS, it is crucial to recognize that delayed responses are not the only solution for anticipating distribution shifts in PeTS. Recent research avenues [17, 32] explore alternative interpretations of performativity through the lens of causality. Thus, there exists the exciting prospect of investigating how causal discovery techniques, such as causal discovery, can assist in identifying the performativity in time-series forecasting problems. We expect that future works delve into these open questions, both theoretically and practically, to enrich our understanding of PeTS.

8 Acknowledgment

This paper was supported in part by the NSF (Expeditions CCF-1918770, CAREER IIS-2028586, Medium IIS-1955883, Medium IIS-2106961, Medium IIS-2403240, PIPP CCF-2200269), CDC MInD program, Meta faculty gift, and funds/computing resources from Georgia Tech and GTRI.

References

[1] Kartik Ahuja, Ethan Caballero, Dinghuai Zhang, Jean-Christophe Gagnon-Audet, Yoshua Bengio, Ioannis Mitliagkas, and Irina Rish. 2021. Invariance principle meets information bottleneck for out-of-distribution generalization. *Advances in Neural Information Processing Systems* 34 (2021), 3438–3450.

[2] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. 2019. Invariant risk minimization. *arXiv preprint arXiv:1907.02893* (2019).

[3] Peter L Bartlett. 1992. Learning with a slowly changing distribution. In *Proceedings of the fifth annual workshop on Computational learning theory*. 243–252.

[4] Rupali Bhati, Jennifer Jones, David Langelier, Anthony Reiman, Jonathan Greenland, Kristin Campbell, and Audrey Durand. 2022. Performative Prediction in Time Series: A Case Study. In *NeurIPS 2022 Workshop on Learning from Time Series for Health*.

[5] Gavin Brown, Shlomi Hod, and Iden Kalemaj. 2022. Performative prediction in a stateful world. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 6045–6061.

[6] Kevin J Coakley and Paul Hale. 2001. Alignment of noisy signals. *IEEE Transactions on Instrumentation and Measurement* 50, 1 (2001), 141–149.

[7] Wei Fan, Pengyang Wang, Dongkun Wang, Dongjie Wang, Yuanchun Zhou, and Yanjie Fu. 2023. Dish-TS: A General Paradigm for Alleviating Distribution Shift in Time Series Forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 37. 7522–7529.

[8] Ganesh Ghalme, Vineet Nair, Itay Eilat, Inbal Talgam-Cohen, and Nir Rosenfeld. 2021. Strategic classification in the dark. In *International Conference on Machine Learning*. PMLR, 3672–3681.

[9] Albert Gu, Karan Goel, and Christopher Ré. 2022. Efficiently Modeling Long Sequences with Structured State Spaces. In *The International Conference on Learning Representations (ICLR)*.

[10] Moritz Hardt, Nimrod Megiddo, Christos Papadimitriou, and Mary Wootters. 2016. Strategic classification. In *Proceedings of the 2016 ACM conference on innovations in theoretical computer science*. 111–122.

[11] Kieran Healy. 2015. The performativity of networks. *European Journal of Sociology/Archives Européennes de Sociologie* 56, 2 (2015), 175–205.

[12] Rob J Hyndman and George Athanasopoulos. 2018. *Forecasting: principles and practice*. OTexts.

[13] Harshavardhan Kamarthi, Lingkai Kong, Alexander Rodríguez, Chao Zhang, and B Aditya Prakash. 2022. CAMul: Calibrated and Accurate Multi-view Time-Series Forecasting. In *Proceedings of the ACM Web Conference 2022*. 3174–3185.

[14] Taesung Kim, Jinhee Kim, Yunwon Tae, Cheonbok Park, Jang-Ho Choi, and Jaegul Choo. 2021. Reversible Instance Normalization for Accurate Time-Series Forecasting against Distribution Shift. In *International Conference on Learning Representations*. https://openreview.net/forum?id=cGDAkQo1C0p

[15] David Krueger, Ethan Caballero, Joern-Henrik Jacobsen, Amy Zhang, Jonathan Binas, Dinghuai Zhang, Remi Le Priol, and Aaron Courville. 2021. Out-of-distribution generalization via risk extrapolation (rex). In *International Conference on Machine Learning*. PMLR, 5815–5826.

[16] Anthony Kuh, Thomas Petsche, and Ronald Rivest. 1990. Learning time-varying concepts. *Advances in neural information processing systems* 3 (1990).

[17] Bogdan Kulynych. 2022. Causal prediction can induce performative stability. In *ICML 2022: Workshop on Spurious Correlations, Invariance and Stability*.

[18] K Raj Kumar and Feng Xue. 2010. An iterative algorithm for joint signal and interference alignment. In *2010 IEEE International Symposium on Information Theory*. IEEE, 2293–2297.

[19] Vitaly Kuznetsov and Mehryar Mohri. 2017. Generalization bounds for non-stationary mixing processes. *Machine Learning* 106, 1 (2017), 93–117.

[20] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. 2018. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st international ACM SIGIR conference on research & development in information retrieval*. 95–104.

[21] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. 2018. Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting. In *International Conference on Learning Representations (ICLR '18)*.

[22] Haoxin Liu, Harshavardhan Kamarthi, Lingkai Kong, Zhiyuan Zhao, Chao Zhang, and B Aditya Prakash. 2024. Time-series forecasting for out-of-distribution generalization using invariant learning. *arXiv preprint arXiv:2406.09130* (2024).

[23] Haoxin Liu, Harshavardhan Kamarthi, Zhiyuan Zhao, Shangqing Xu, Shiyu Wang, Qingsong Wen, Tom Hartvigsen, Fei Wang, and B Aditya Prakash. 2025. How can time series analysis benefit from multiple modalities? a survey and outlook. *arXiv preprint arXiv:2503.11835* (2025).

[24] Haoxin Liu, Chenghao Liu, and B Aditya Prakash. 2024. A picture is worth a thousand numbers: Enabling llms reason about time series via visualization. *arXiv preprint arXiv:2411.06018* (2024).

[25] Haoxin Liu, Shangqing Xu, Zhiyuan Zhao, Lingkai Kong, Harshavardhan Prabhakar Kamarthi, Aditya Sasanur, Megha Sharma, Jiaming Cui, Qingsong Wen, Chao Zhang, et al. 2024. Time-mmd: Multi-domain multimodal dataset for time series analysis. *Advances in Neural Information Processing Systems* 37 (2024), 77888–77933.

[26] Haoxin Liu, Zhiyuan Zhao, Jindong Wang, Harshavardhan Kamarthi, and B Aditya Prakash. 2024. Lstprompt: Large language models as zero-shot time series forecasters by long-short-term prompting. *arXiv preprint arXiv:2402.16132* (2024).

[27] Jie Lu, Anjin Liu, Fan Dong, Feng Gu, Joao Gama, and Guangquan Zhang. 2018. Learning under concept drift: A review. *IEEE transactions on knowledge and data engineering* 31, 12 (2018), 2346–2363.

[28] Donald A MacKenzie, Fabian Muniesa, Lucia Siu, et al. 2007. *Do economists make markets?: on the performativity of economics*. Princeton University Press.

[29] Debmalya Mandal, Stelios Triantafyllou, and Goran Radanovic. 2022. Performative Reinforcement Learning. *arXiv preprint arXiv:2207.00046* (2022).

[30] Sarabeth M Mathis, Alexander E Webber, Tomás M León, Erin L Murray, Monica Sun, Lauren A White, Logan C Brooks, Alden Green, Addison J Hu, Roni Rosenfeld, et al. 2024. Evaluation of FluSight influenza forecasting in the 2021–22 and 2022–23 seasons with a new target laboratory-confirmed influenza hospitalizations. *Nature communications* 15, 1 (2024), 6289.

[31] Celestine Mendler-Dünner, Juan Perdomo, Tijana Zrnic, and Moritz Hardt. 2020. Stochastic optimization for performative prediction. *Advances in Neural Information Processing Systems* 33 (2020), 4929–4939.

[32] John Miller, Smitha Milli, and Moritz Hardt. 2020. Strategic classification is causal modeling in disguise. In *International Conference on Machine Learning*. PMLR, 6917–6926.

[33] Mehryar Mohri and Afshin Rostamizadeh. 2008. Rademacher complexity bounds for non-iid processes. *Advances in Neural Information Processing Systems* 21 (2008).

[34] Meinard Müller. 2007. Dynamic time warping. *Information retrieval for music and motion* (2007), 69–84.

[35] Elizbar A Nadaraya. 1964. On estimating regression. *Theory of Probability & Its Applications* 9, 1 (1964), 141–142.

[36] Tetsuya Nakamura, Keishi Taki, Hiroki Nomiya, Kazuhiro Seki, and Kuniaki Uehara. 2013. A shape-based similarity measure for time series data with ensemble learning. *Pattern Analysis and Applications* 16 (2013), 535–548.

[37] Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. 2022. A Time Series is Worth 64 Words: Long-term Forecasting with Transformers. *arXiv preprint arXiv:2211.14730* (2022).

[38] Boris N. Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. 2020. N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. In *International Conference on Learning Representations*. https://openreview.net/forum?id=r1ecqn4YwB
[39] Juan Perdomo, Tijana Zrnic, Celestine Mendler-Dünner, and Moritz Hardt. 2020. Performative prediction. In *International Conference on Machine Learning*. PMLR, 7599–7609.
[40] Mohammad Pezeshki, Oumar Kaba, Yoshua Bengio, Aaron C Courville, Doina Precup, and Guillaume Lajoie. 2021. Gradient starvation: A learning proclivity in neural networks. *Advances in Neural Information Processing Systems* 34 (2021), 1256–1272.
[41] Alexander Rodriguez, Jiaming Cui, Naren Ramakrishnan, Bijaya Adhikari, and B. Aditya Prakash. 2023. EINNs: Epidemiologically-Informed Neural Networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 37.
[42] Alexander Rodríguez, Harshavardhan Kamarthi, Pulak Agarwal, Javen Ho, Mira Patel, Suchet Sapre, and B Aditya Prakash. 2022. Data-centric epidemic forecasting: A survey. *arXiv preprint arXiv:2207.09370* (2022).
[43] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. 2019. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731* (2019).
[44] David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. 2020. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting* 36, 3 (2020), 1181–1191.
[45] Alex J Smola and Bernhard Schölkopf. 2004. A tutorial on support vector regression. *Statistics and computing* 14, 3 (2004), 199–222.
[46] José F Torres, Dalil Hadjout, Abderrazak Sebaa, Francisco Martínez-Álvarez, and Alicia Troncoso. 2021. Deep learning for time series forecasting: a survey. *Big Data* 9, 1 (2021), 3–21.
[47] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).
[48] Christopher Williams and Carl Rasmussen. 1995. Gaussian processes for regression. *Advances in neural information processing systems* 8 (1995).
[49] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. 2021. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in Neural Information Processing Systems* 34 (2021), 22419–22430.
[50] Yu Zheng, Licia Capra, Ouri Wolfson, and Hai Yang. 2014. Urban computing: concepts, methodologies, and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)* 5, 3 (2014), 1–55.
[51] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. 2021. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 11106–11115.
[52] Yunyue Zhu and Dennis Shasha. 2002. Statstream: Statistical monitoring of thousands of data streams in real time. In *VLDB'02: Proceedings of the 28th International Conference on Very Large Databases*. Elsevier, 358–369.

A Appendix I: FPS Analysis

To investigate the benefits of utilizing FPS in addressing PeTS compared to conventional forecasting models through optimizing ERM, we analyze the discrepancy in the tightness of the generalization error bounds between the two approaches. In summary, our analysis presents two primary propositions: Firstly, when the $\mathbf{X}^{DR}$ (typically refers to future data of features) is accessible, employing $\mathbf{X}^{DR}$ for forecasting yields a tighter error bound compared to using $\boldsymbol{x}$ (Proposition 5.1). Secondly, in scenarios where the $\mathbf{X}^{DR}$ is unavailable and forecasting using the estimated $\hat{\mathbf{X}}^{DR}$, there exists a trade-off in the error bound between reducing empirical loss and increasing model complexity as compared to forecasting with $\mathbf{X}^{L}$ (Proposition 5.2). We now present a formal and detailed analysis of these propositions as follows.

For deriving the generalization error bounds for time-series data, the common practice assumes a β-mixing distribution for the time series $\mathcal{Z}$ (e.g., $\mathcal{Z}$ as the general representation of time series $\mathbf{X}$ or $\mathbf{Y}$). This property states that two distinct sequences become more independent as their time gap increases, which is naturally reasonable for real-world time-series data. When this assumption holds, the generalization bound for probably approximately correct (PAC) for time-series models follows:

Theorem A.1. *[19, 33] Let Θ be the space of candidate predictors θ and let $\mathcal{H}$ be the space of induced losses:*

$$\mathcal{H} = \{h = \mathcal{L}(\theta) : \theta \in \Theta\}$$

for some loss function $0 \leq \mathcal{L}(\cdot) \leq M$. Then for any sequential data $\mathcal{Z}$ of size n drawn from a stationary β-mixing distribution, with probability at least $1-\eta$:

$$\mathbb{E}[\mathcal{L}(\theta)] \leq \mathcal{L}(\theta) + \delta(\mathcal{R}(\mathcal{H}), n, \eta)$$

where $\mathcal{R}(\cdot)$ measures the complexity of the model class Θ, such as Rademacher Complexity, $\delta(\cdot)$ is a function of the complexity $\mathcal{R}(\cdot)$, the confidence level η, and the number of observed data points n.

Proof of Theorem A.1. We first give the formal definition of stationary, β-mixing, and Rademacher complexity.

Definition (Stationary). A sequence of random variables $\mathcal{Z} = \{\mathcal{Z}_t\}_{t=-\infty}^{\infty}$ is is said to be stationary if for any t and and non-negative integers m, k, the random vectors $(\mathcal{Z}_t, \ldots, \mathcal{Z}_{t+m})$ and $(\mathcal{Z}_{t+k}, \ldots, \mathcal{Z}_{t+k+m})$ have the same distribution.

Definition (β-mixing). Let $\mathcal{Z} = \{\mathcal{Z}_t\}_{t=-\infty}^{\infty}$ be a sequence of random variables. For any $i, j \in \mathbb{Z}$, let σ_i^j denote the σ-algebra generated by the random variables $\mathcal{Z}_k$, $i \leq k \leq j$. Then for any positive integer k, the β-mixing coefficient of the stochastic process $\mathcal{Z}$ is defined as

$$\beta(k) = \sup_n \mathbb{E}_{B \in \sigma_{-\infty}^n}[\sup_{A \in \sigma_{n+k}^{\infty}} |P(A|B) - P(A)|] \tag{11}$$

Then $\mathcal{Z}$ is said to be β-mixing if $\beta(k) \to 0$ as $k \to \infty$.

Definition (Rademacher Complexity). Let $\mathcal{Z} = \{\mathcal{Z}_t\}_{t=1}^{n}$ be the samples drawn from the distribution Π, then the empirical Rademacher complexity is

$$\hat{\mathcal{R}}(\mathcal{H}) = \frac{2}{n}\mathbb{E}[\sup_{h \in \mathcal{H}} \left| \sum_{i=1}^{n} \sigma_i h(\mathcal{Z}_i) \right| | \mathcal{Z}] \tag{12}$$

where σ_i is a sequence of Rademacher random variables, independent of each other and everything else, and drawn randomly from $\{\pm 1\}$. $\mathcal{H}$ be the space of induced losses contains all possible h. The Rademacher complexity is

$$\mathcal{R}_n(\mathcal{H}) = \mathbb{E}_{\Pi}[\hat{\mathcal{R}}_n(\mathcal{H})] \tag{13}$$

We first start with the stationary assumption holds, in which we assume the time series are both stationary and β-mixing. Then the PAC bound holds by the following lemma:

Lemma A.2. *[33] Let Θ be the space of candidate predictors θ and let $\mathcal{H}$ be the space of induced losses:*

$$\mathcal{H} = \{h = \mathcal{L}(\theta) : \theta \in \Theta\}$$

for some loss function $0 \leq \mathcal{L}(\cdot) \leq M$. Then for any sequential data $\mathcal{Z}$ of size n drawn from a stationary β-mixing distribution, and for any $\mu, k > 0$, with $2\mu k = n$, and $\eta > 2(\mu - 1)\beta(k)$, where $\beta(k)$ is the mixing coefficient, with probability at least $1-\eta$:

$$\mathbb{E}[\mathcal{L}(\theta)] \leq \mathcal{L}(\theta) + \hat{\mathcal{R}}_{\mu}(\mathcal{H}) + 3M\sqrt{\frac{\ln 4/\eta'}{2\mu}}$$

where $\eta' = \eta - 4(\mu - 1)\beta(k)$ and $\hat{\mathcal{R}}(\cdot)$ is the empirical Rademacher complexity.

We then remove the stationary assumption, in which we only assume the time series are β-mixing. Then the PAC bound is loosened by the following lemma:

Lemma A.3. *[19] Let Θ be the space of candidate predictors θ and let $\mathcal{H}$ be the space of induced losses:*

$$\mathcal{H} = \{h = \mathcal{L}(\theta) : \theta \in \Theta\}$$

for some loss function $0 \leq \mathcal{L}(\cdot) \leq M$. Then for any sequential data $\mathcal{Z}$ of size n, drawn from a β-mixing distribution, and for any $\mu, k > 0$, with $2\mu k = n$, and $\eta > 2(\mu - 1)\beta(k)$, where $\beta(k)$ is the mixing coefficient, with probability at least $1 - \eta$:

$$\mathbb{E}[\mathcal{L}(\theta)] \leq \mathcal{L}(\theta) + \frac{2}{k}\hat{\mathcal{R}}_{\mu}(\mathcal{H}) + \frac{2}{n}\sum_{t=1}^{n}\bar{d}(t, n+s) + M\sqrt{\frac{\ln 2/\eta'}{8\mu}}$$

where $\hat{d}$ is the averaged discrepancy, s is the forecasting steps, $\eta' = \eta - 4(\mu - 1)\beta(k)$ and $\hat{\mathcal{R}}(\cdot)$ is the empirical Rademacher complexity.

Lemma A.3 provides a more general PAC bound of time-series data without the stationary assumption necessarily holding. Lemma A.3 reduces exactly to Lemma A.2 if the stationary assumption holds, where the expected discrepancy is supposed to be 0 for stationary time series.

By concluding Lemma A.2 and Lemma A.3, the general form of the PAC bound for time series is given by:

$$\mathbb{E}[\mathcal{L}(\theta)] \leq \mathcal{L}(\theta) + \delta(\mathcal{R}(\mathcal{H}), n, \eta)$$

Then Theorem A.1 is proved.

In a more intuitive sense, $\delta(\cdot)$ refers to the non-vanishing generalization errors caused by the model complexity. For models with unknown complexity, such as autoregressive moving average (ARMA), the stationary assumption is necessary for leading to Lemma A.2. However, for models with known complexity, such as neural networks, which are leveraged by FPS, the stationary assumption is not necessarily needed for the generalization bounds. Bridging Theorem A.1 to FPS, it is challenging to precisely quantify the exact model complexity for neural networks. However, analyzing disparities in empirical loss is still possible. In FPS, the delayed responses that maximize the similarity to the target are leveraged as forecasting inputs, which results in the following theorem:

Theorem A.4. *Let $i \in [0, H]$ be all possible time delay and $\Theta = \{\theta_i\}$ be the time-series predictors for the corresponding i, including τ. Let $\mathcal{L}$ be MSE-loss and $\mathcal{H} = \{h = \mathcal{L}(\theta_i) : \theta_i \in \Theta\}$ be the space of induced losses. Then $\mathcal{L}(\theta_\tau) \leq \mathcal{L}(\theta_i), \forall i \in [0, H]$.*

Here, θ_i reduces to conventional forecasting models when $\tau = 0$. Intuitively, Theorem A.4 signifies that utilizing sequences most similar to the targets as forecasting inputs yields the lowest empirical error.

Proof of Theorem A.4 Let $\mathbf{X}$ denote the input features and $\mathbf{Y}$ denote the target. Let g be the linear regressor such that $g : \mathbf{X} \rightarrow \mathbf{Y}$. Then the optimal solution

$$g^* = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y} = \frac{\mathbf{X}^T\mathbf{Y}}{\|\mathbf{X}\|^2}$$

Then the empirical mean-square-error is $\|\mathbf{Y} - \hat{\mathbf{Y}}\|^2 = \|\mathbf{Y} - \frac{\mathbf{X}^T\mathbf{Y}\mathbf{X}}{\|\mathbf{X}\|^2}\|^2$, denoted as $h(\mathbf{X})$. According to the fact that $< \mathbf{X}, \mathbf{Y} >= \|\mathbf{X}\|\|\mathbf{Y}\| \cos\theta$, then we can simplify $h(\mathbf{X})$ as

$$h(\mathbf{X}) = \|\mathbf{Y}\|^2(1 - \cos^2\theta) = \|\mathbf{Y}\|^2 \sin^2\theta := h(\theta)$$

Then we have the derivate of $h(\theta)$ w.r.t. θ as

$$\frac{dh(\theta)}{d\theta} == \|\mathbf{Y}\|^2 \sin 2\theta$$

Suppose $\theta \in [-\pi, \pi]$, then it is evident that $h(\theta)$ has the local minimums when $\theta = 0, \pm\pi$, that is, when the similarity is maximized. Hence, we can conclude that the MSE loss will reduce as the absolute similarity between the two signals increases, then Theorem A.4 is proved.

Combining Theorem A.1 and Theorem A.4, we can conclude Proposition 5.1 and Proposition 5.2.

B Appendix II: Evaluation Setup and Additional Results (Cont.)

Additional Experiment Setups. In our evaluation of both regular time-series forecasting and real-time forecasting tasks using the COVID-19 dataset, we employ ensemble results derived from four distinct random seeds. This is because of the large variability observed in COVID-19 predictions through different seeds, and utilizing ensemble results helps reduce the uncertainty. Conversely, for METR-LA traffic predictions, we rely on predictions generated from a fixed random seed, as these predictions tend to exhibit stability across different random seeds.

In the case of the COVID-19 dataset, evaluation results are exclusively based on data from California (CA), Texas (TX), New York (NY), Florida (FL), and national data. Despite the model being trained on samples from all U.S. states, this selection is made for statistical significance considerations. For instance, evaluations of other states, like Alaska, are less statistically significant due to their smaller population bases or irregular reporting frequencies, such as biweekly data updates.

In the context of regular time-series forecasting tasks, all experiments follow a two-stage process: initially, we fine-tune all hyperparameters on the validation set, where all models are trained based on the training set. We retain all tuned hyperparameters fixed and subsequently retrain the model using joint training and validation sets. The testing results are then recorded based on the retrained models. We use such training schemes to mitigate the difference of time delays τ on training, validation, and test sets.

Table 4: Performance comparison between forecasting using original inputs and exact X^{DR} (marked as ✘ and ✔ respectively). Forecasting using X^{DR} significantly improves the forecasting accuracy and correlations except with Autoformer.

		traffic		
Model	X^{DR}	NMAE(⇓)	NRMSE(⇓)	PC(⇑)
RNN	✘	0.141	0.208	0.681
	✔	**0.059**	**0.088**	**0.880**
LSTNet	✘	0.135	0.196	0.663
	✔	**0.060**	**0.087**	**0.895**
Transformer	✘	0.145	0.216	0.648
	✔	**0.062**	**0.091**	**0.898**
Informer	✘	0.141	0.198	0.673
	✔	**0.062**	**0.093**	**0.870**
Autoformer	✘	**0.166**	**0.217**	**0.530**
	✔	0.181	0.239	0.450

C Appendix III: Additional Evaluation Results

Forecasting with X^{DR} yeilds tighter error bound. We here show the additional results to validate Proposition 5.1, which compares the performance disparities between forecasting with original inputs and with exact X^{DR}. The results of evaluations on METR-LA traffic dataset are shown in Table 4. Here, we exclude the evaluations on the COVID-19 dataset, as the COVID-19 dataset includes both performative and non-performative features which may perturb testing the impacts of forecasting with X^{DR} from performative features.

The validation results show clear benefits of forecasting with the delayed response X^{DR} than the original inputs. Particularly, we notice an averaged 56.8% NMAE reduction, 56.0% NRMSE reduction, and 32.9% PC increase when forecasting with RNN, LSTNet, Transformer, and Informer. However, when employing Autoformer as the forecasting model, we observe a slight performance degradation. This is because of the mechanism of Autoformer which enforces the forecasting trend based on historical look-back windows for long-term stability, which limits its ability to learn rapid trend changes in the short term and aligns less with our focus on short-term forecasting tasks.

Non-stationarity methods can be detrimental to short-term forecasting. As shown in Table 1, forecasting results using non-stationary methods, such as RevIN and Dish-TS, often do not lead to performance improvements and may even degrade performance in most scenarios. The intuition behind this is that existing non-stationary methods focus on producing stable predictions, which contrasts with the design of FPS, which aims to capture rapid trend changes caused by performativity. Non-stationary methods typically employ normalization and de-normalization to ensure that predictions within the forecasting window follow a similar mean and variance as the lookback window. While these approaches help maintain stable forecasting results in the long term, they restrict the exploration of trend information and enforce stability in predictions. This interpretation is further supported by visualizations of predictions on the METR-LA dataset in Figure 5.

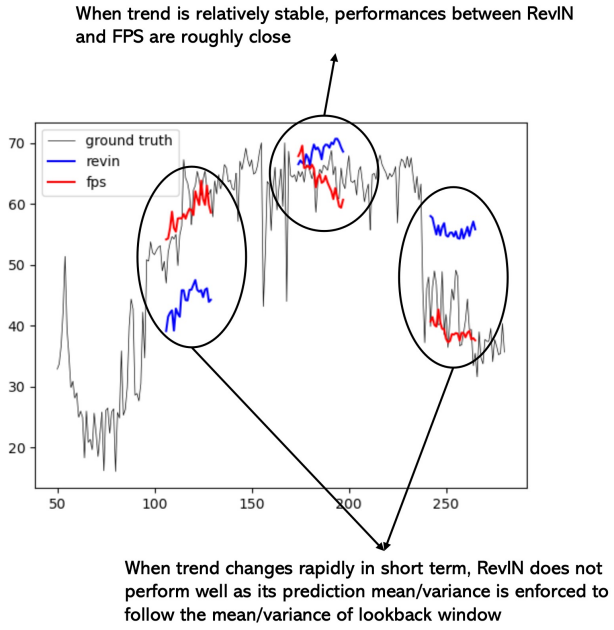


Figure 5: Visualizations of predictions from RevIN and FPS show that the normalization and de-normalization strategy in RevIN restricts the model's ability to capture the correct trend changes in the short term.