Calibration of Time-Series Forecasting: Detecting and Adapting Context-Driven Distribution Shift

Mouxiang Chen*
Zhejiang University
Hangzhou, China
chenmx@zju.edu.cn

Lefei Shen*
Zhejiang University
Hangzhou, China
lefeishen@zju.edu.cn

Han Fu
Zhejiang University
Hangzhou, China
11821003@zju.edu.cn

Zhuo Li†
State Street Technology (Zhejiang) Ltd.
Hangzhou, China
lizhuo@zju.edu.cn

Jianling Sun
Zhejiang University
Hangzhou, China
sunjl@zju.edu.cn

Chenghao Liu†
Salesforce Research Asia
Singapore
chenghao.liu@salesforce.com

ABSTRACT

Recent years have witnessed the success of introducing deep learning models to time series forecasting. From a data generation perspective, we illustrate that existing models are susceptible to distribution shifts driven by temporal contexts, whether observed or unobserved. Such context-driven distribution shift (CDS) introduces biases in predictions within specific contexts and poses challenges for conventional training paradigms. In this paper, we introduce a universal calibration methodology for the detection and adaptation of CDS with a trained model. To this end, we propose a novel CDS detector, termed the "residual-based CDS detector" or "Reconditionor", which quantifies the model's vulnerability to CDS by evaluating the mutual information between prediction residuals and their corresponding contexts. A high Reconditionor score indicates a severe susceptibility, thereby necessitating model adaptation. In this circumstance, we put forth a straightforward yet potent adapter framework for model calibration, termed the "sample-level contextualized adapter" or "SOLID". This framework involves the curation of a contextually similar dataset to the provided test sample and the subsequent fine-tuning of the model's prediction layer with a limited number of steps. Our theoretical analysis demonstrates that this adaptation strategy can achieve an optimal bias-variance trade-off. Notably, our proposed Reconditionor and SOLID are model-agnostic and readily adaptable to a wide range of models. Extensive experiments show that SOLID consistently enhances the performance of current forecasting models on real-world datasets, especially on cases with substantial CDS detected by the proposed Reconditionor, thus validating the effectiveness of the calibration approach.

*Both authors contributed equally to this research.
†Corresponding authors.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
KDD '24, August 25–29, 2024, Barcelona, Spain.
© 2024 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 979-8-4007-0490-1/24/08
https://doi.org/10.1145/3637528.3671926

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**; • **Information systems** → **Data mining**.

KEYWORDS

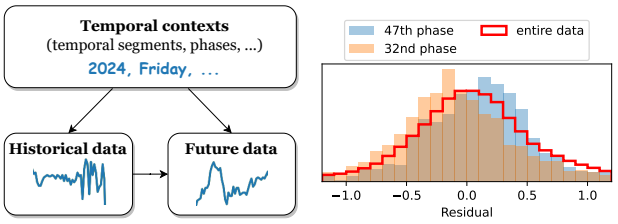
time series forecasting, distribution shift, context-driven distribution shift

ACM Reference Format:
Mouxiang Chen, Lefei Shen, Han Fu, Zhuo Li, Jianling Sun, and Chenghao Liu. 2024. Calibration of Time-Series Forecasting: Detecting and Adapting Context-Driven Distribution Shift. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24), August 25–29, 2024, Barcelona, Spain.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3637528.3671926

1 INTRODUCTION

Time Series Forecasting (TSF) plays a pivotal role in numerous real-world applications, including energy consumption planning [5, 6], weather forecasting [2, 10], financial risk assessment [1, 18], and web recommendation [9, 21, 22]. Recent years have witnessed the progress of introducing time series forecasting model [24, 26, 28–30] to better capture the temporal dependencies by extracting and stacking multi-level features. Despite the remarkable architecture design, the distribution shift [13] has become an unavoidable yet highly challenging issue, which engenders suboptimal performance and hampers generalization with a fluctuating distribution.

Generally, distribution shift signifies variations in the underlying data generation process, which is typically driven by some temporal observed or unobserved factors, namely *contexts*. In this paper, we reveal two significant observed contexts within time series data: **temporal segments** (*i.e.*, different temporal stages as the time evolution), and **periodic phases** (*i.e.*, the fraction of the period covered up to the current time), along with other **unobserved contexts**. For example, in the scenario of electricity consumption, factors like economic trends over the years (temporal segments) and seasonal fluctuations (periodic phases) can affect electricity usage. We visualized the impact of these two contexts on data distribution in Appendix C.2. Moreover, sudden policy changes (unobserved contexts) can also affect the usage. We refer to this phenomenon as *context-driven distribution shift*, or *CDS*.



(a) causal graph (b) (conditional) residual distribution

Figure 1: (a) Causal graph in the presence of context-driven distribution shift. (b) Impact of CDS: Autoformer's residual distribution on the entire Illness dataset, along with the residual distributions conditioned on two different periodic phases.

In the presence of CDS, TSF models remain constrained owing to their ignorance of contexts. Firstly, the training and testing datasets are often generated under distinct contexts (*i.e.*, temporal segments). This deviation from the conventional assumption of consistent dataset distributions between training and testing data can lead to suboptimal prediction results. Secondly, even within the training set, these contexts essentially function as confounders [19] — factors that simultaneously influence the historical and future data, as demonstrated in Figure 1(a). Such confounders lead trained models to capture spurious correlations, causing them to struggle with generalizing to data from new distributions.

To present the impact of CDS in practice, we trained an Autoformer [26] on *Illness* and assessed its ability to fit sub-data in different periodic phases. Residuals, the difference between a model's prediction and the ground truth that can reflect the goodness of fitting, were analyzed for the 47th and 32nd periodic phases, and the entire dataset. The residual results are visualized in Figure 1(b). Notably, it can be observed that the model provides an *unbiased* estimation for the entire training dataset (*i.e.*, the mean of residuals is zero). However, the estimation within two specific contexts is *biased* since both their means of residuals deviate from zero. This observation underscores the model's limitations in fitting sub-data within a context, as it is susceptible to data from other contexts and learns spurious correlations. It motivates us to calibrate the model to achieve more accurate estimation within each context.

Present work. In this paper, we introduce a general calibration approach to detect and adapt to CDS with a trained model. Specifically, we first propose a metric to measure the severity of model's susceptibility to CDS within the training data, namely **Residual-based context-driven distribution shift detector** or Reconditionor, by measuring the mutual information between residuals and observed contexts to quantify the impact of contexts on model's prediction.

A higher value from Reconditionor signifies a stronger CDS. Under this circumstance, we further propose a simple yet effective adapter framework for further calibration. Given the inherent variability in contexts across data samples, a one-size-fits-all adaptation of the model is inherently unfeasible. Hence, we posit the need to fine-tune the model at the individual *sample-level*. Figure 2 illustrates the comparison between the traditional method and our proposed sample-level adaptation framework. Notably, for each

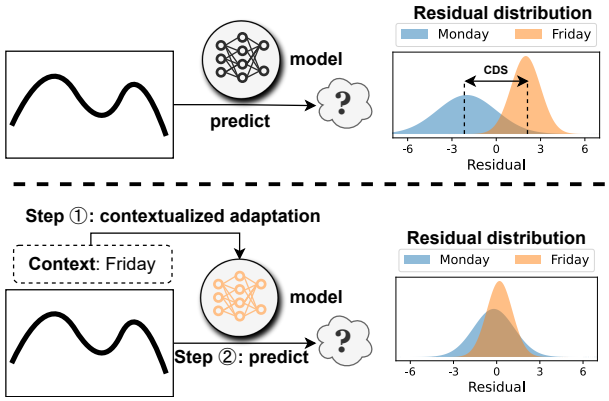


Figure 2: Illustrations of the traditional framework (top) and the proposed framework (bottom). By calibrating the model via contextualized adaptation before making each prediction, the context-driven distribution shift (CDS) can be alleviated.

test sample, adapting the model solely based on that single instance is intractable. As an alternative, we initiate a data augmentation process by curating a dataset comprising preceding samples characterized by akin contexts. Since the chosen samples can introduce significant variance during the adaptation process, we restrict the fine-tuning to the model's prediction layer with a limited number of steps. Our theoretical findings substantiate that this approach can attain an optimal bias-variance trade-off. We refer to this framework as **Sample-level cOntextuaLIzed aDapter**, or SOLID.

Extensive experiments indicate that our proposed calibration approach consistently enhances the performance of 7 forecasting models across 8 real-world datasets. Notably, our Reconditionor reliably identifies cases requiring CDS adaptation with a high accuracy of 89.3%. Furthermore, our proposed SOLID yields an average improvement ranging from 8.7% to 15.1% when addressing significant CDS situations as detected by Reconditionor. Even in cases with less pronounced CDS, SOLID still achieves an average improvement ranging from 0.3% to 6.3%. From an efficiency perspective, our method introduces a 20% additional time overhead, a gap smaller than the effects of dataset variability. Crucially, Reconditionor's metric closely aligns with SOLID's performance gains. These findings provide robust validation of the effectiveness of our calibration approach.

The main contributions of this work are summarized as follows:

- We propose the concept of context-driven distribution shift (CDS) by studying the driving factors of distribution shifts and investigating two observed contexts (*temporal segments* and *periodic phases*), as well as unobserved contexts.
- We propose an end-to-end calibration approach, including Reconditionor, a detector to measure the severity of the model's susceptibility to CDS, and SOLID, an adapter to calibrate models for enhancing performance under severe CDS.
- Extensive experiments over various datasets demonstrate that Reconditionor detects CDS of forecasting models on the training dataset accurately, and SOLID significantly enhances current models without substantially compromising time efficiency.

2 RELATED WORK

2.1 Time series forecasting models

With the successful rise of deep learning approaches, many recent work utilizes deep learning to better explore the non-linearity and multiple patterns of the time series and empirically show better performance. Some research introduces the Transformer to capture temporal dependencies with the attention mechanism. Specifically, Informer [29] proposes ProbSparse self-attention. Autoformer [26] introduces Auto-correlation attention based on seasonal-trend decomposition. FEDformer [30] proposes Fourier frequency enhanced attention. ETSformer [24] leverages exponential smoothing attention. Crossformer [28] utilizes a two-stage attention to capture both cross-time and cross-dimension dependency. PatchTST [16] proposes patching and channel independence techniques for better prediction. Different from the Transformer architecture, DLinear [27] leverages linear models with decomposition or normalization for TSF tasks. In this paper, our proposed pipeline can be easily applied to these TSF models, regardless of the model architecture.

2.2 Distribution shift in time series

Distribution shift in time series refers to the phenomenon that statistical properties and data distribution continuously vary over time. To better detect such distribution shifts, various methods have been proposed. Stephan *et al.* [20] employs dimension reduction and proposes a two-sample hypothesis testing. Sean *et al.* [12] proposes an expected conditional distance test statistic and localizes the exact feature where the distribution shift appears. Lipton *et al.* [15] utilizes hypothesis testing based on their proposed Black Box Shift Estimation, thereby detecting the shift. However, these detection methods ignore the important context, and are mostly not appropriate for time series data. In contrast, our proposed residual-based detector has sufficient ability to detect the influence of underlying context on distribution shift.

Meanwhile, addressing distribution shifts in time series forecasting is also crucial. One approach is to employ normalization techniques to stationarize the data. For example, DAIN [17] employs nonlinear networks to adaptively normalize time series. RevIN [11] proposes a reversible instance normalization to alleviate series shift. AdaRNN [7] introduces an Adaptive RNN to solve the problem. Dish-TS [8] proposes a Dual-Coefficient Net framework to separately learn the distribution of input and output space, thus capturing their divergence. Other approaches combine statistical methods with deep networks. Syml [23] applies Exponential smoothing on RNN, to concurrently fit seasonality and smoothing coefficients with RNN weights. SAF [3] integrates a self-supervised learning stage on test samples to train the model before making the prediction. However, these methods are coupled to model architecture or require modification during or before the training process, which limits their application. In contrast, our proposed approach can efficiently adapt the given trained models solely during test time and at the sample level.

3 PRELIMINARIES

For a multivariate time series with M variables, let $\boldsymbol{x}_t \in \mathbb{R}^M$ represent a sample at t-th timestep. Given a historical sequence: $\boldsymbol{X}_{t-L:t} = [\boldsymbol{x}_{t-L}, \cdots, \boldsymbol{x}_{t-1}] \in \mathbb{R}^{L\times M}$, where L is the look-back window size, the task is to predict future values with T forecasting window size: $\hat{\boldsymbol{X}}_{t:t+T} = [\boldsymbol{x}_t, \cdots, \boldsymbol{x}_{t+T-1}] \in \mathbb{R}^{T\times M}$. The training objective of a model f is to find the best mapping from input to output sequence, *i.e.* $\hat{\boldsymbol{X}}_{t:t+T} = f(\boldsymbol{X}_{t-L:t})$. In this work, we assume f is a deep neural network composed of two parts: a *feature extractor* $g_\Phi : \mathbb{R}^{L\times M} \to \mathbb{R}^d$ mapping the historical values to a d-dimensional latent representation, and a linear top h_θ named as *prediction layer* mapping the representation to predicted future values. The specific definitions of g_Φ and h_θ for different model architectures will be introduced in Appendix B.4.

4 CONTEXT-DRIVEN DISTRIBUTION SHIFT

In this section, we introduce the limitation of traditional TSF models. As mentioned before, the data generation in time series is typically influenced by temporal external factors (*i.e.*, context c), such as *temporal segments* and *periodic phases*. Let X, Y and C denote the variables of historical data $\boldsymbol{X}_{t-L:t}$, future data $\boldsymbol{X}_{t:t+T}$ and context c_t at time step t, respectively. The generation of Y is dependent on X and C, characterized as $P(Y \mid X, C)$.

Due to the ignorance of contexts, the model f trained on the dataset learns a marginal distribution $P(Y \mid X) = \sum_c P(Y \mid X, C = c)P(C = c)$. This introduces a confounding bias [19] since context C usually influences both the historical data X and the future data Y, which is evident in Appendix C.2. To illustrate this concept, consider a simplified example in the domain of recommendation. In the winter season (context C), users who purchase hot cocoa (historical data X) also tend to buy coats (future data Y). A model may "memorize" the correlation between X and Y (a spurious correlation), and mistakenly recommend coats to a user who purchases hot cocoa in summer. This confounding bias leads to suboptimal product recommendations. In our subsequent theoretical analysis (§ 6), we detail that such context consistently adds a bias term to the model. The following defines this phenomenon formally.

Definition 1 (Context-driven Distribution Shift). *If there exists a variable C (related to time t) that influences the data generation process from historical X to future Y, i.e.,*

$$P(Y \mid X) \neq P(Y \mid X, C),$$

*then this time series data is said to exhibit a **context-driven distribution shift**, or CDS. The variable C is termed the **context**.*

5 CALIBRATION FRAMEWORK FOR CDS

In this section, we introduce a general calibration methodology for detecting (in § 5.1) and adapting (in § 5.2) to CDS in conjunction with the model. The pipeline of this calibration framework is illustrated in Figure 3.

5.1 Residual-based CDS detector

Our primary focus lies in assessing the model's susceptibility to CDS. Our evaluation predominantly centers around observed contexts, as the analysis of unobserved contexts is computationally infeasible. Fortunately, for our empirical investigation, the utilization of observed contexts proves to be sufficient and effective.

As visually demonstrated in Figure 1(b), the presence of contexts introduces a bias to the model estimation, causing variations in

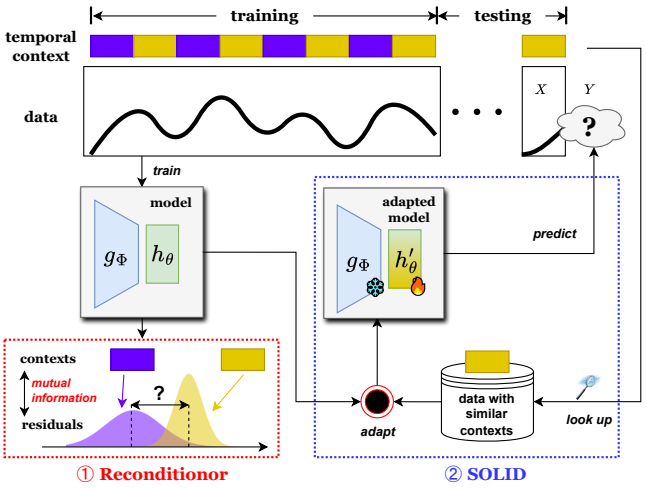


Figure 3: Pipeline of our calibration framework to detect and adapt to context-driven distribution shift (CDS). We leverage ① residual-based context-driven distribution shift detector (Reconditionor) to assess how susceptible a trained model is to CDS. If we detect a significant susceptibility, we employ ② sample-level contextualized adapter (SOLID) to adapt the model for each test sample using preceding data that share similar contexts.

residual distributions across different contexts. Based on it, we propose a novel detector, namely **Residual-based context-driven distribution shift detector** (or Reconditionor), by measuring the *mutual information* (MI) between prediction residuals and their corresponding contexts. The MI quantifies the extent of information acquired regarding the residuals when observing the context, which serves as a metric for evaluating the influence of contexts on the model. MI can be computed by:

$$\delta = \mathrm{MI}(\Delta Y; C) = \mathbb{E}_C \left[D_{\mathrm{KL}} \left(P(\Delta Y \mid C) \parallel P(\Delta Y) \right) \right], \tag{1}$$

where $\Delta Y = f(X) - Y$ is the residuals of model f, and $D_{\mathrm{KL}}(P \| Q)$ is Kullback–Leibler (KL) divergence between distributions P and Q.

We reuse Figure 1(b) to illustrate the concept behind Reconditionor when detecting the distribution shift based on the context of periodic phases. The marginal residual distribution $P(\Delta Y)$ typically exhibits a mean close to zero after training. However, the residual distributions conditioned on different contexts (*e.g.*, the 47th phase and the 32nd phase) $P(\Delta Y \mid C)$ clearly show non-zero mean values. This increases the KL divergence between the two distributions, consequently elevating δ and indicating a strong CDS. Additionally, a non-zero mean in the conditional residual distribution suggests the model f fails to adequately fit the data within each context, signaling the need for further adaptation. In summary, a high value of δ for a model f implies the necessity for adapting f, which is also empirically verified in § 7.2.3.

In practice, we assume that the residuals follow Gaussian distributions. This assumption is based on the utilization of MSE loss, which implicitly presupposes that the residuals adhere to additive Gaussian noise. This characteristic is also evident in Figure 1(b). The adoption of this assumption expedites the calculation of KL divergence because Gaussian distributions offer a straightforward analytical solution for it.

Algorithm 1: Algorithm for Reconditionor

Input: Model f, training data with K contexts $\mathcal{D}^{\text{train}} = \{(\boldsymbol{X}_{t-L:t}, \boldsymbol{X}_{t:t+T}, c_t) : t < t_{\text{train}}, c_t \in [K]\}$.
Output: $\delta \in [0, 1]$ indicating f's susceptibility to CDS.

1 $R \leftarrow \emptyset$;
2 $R_1, \cdots, R_K \leftarrow \emptyset, \cdots, \emptyset$;
3 **for** $L \le t < t_{train}$ **do**
4 $r \leftarrow f(\boldsymbol{X}_{t-L:t}) - \boldsymbol{X}_{t:t+T}$;
5 $R \leftarrow R \cup r$;
6 $R_{c_t} \leftarrow R_{c_t} \cup r$;
7 **end**
8 $\mu, \sigma \leftarrow \mathrm{Mean}(R), \mathrm{Standard\text{-}Deviation}(R)$;
9 $\delta \leftarrow 0$;
10 **for** $c \in [K]$ **do**
11 $\mu_c, \sigma_c \leftarrow \mathrm{Mean}(R_c), \mathrm{Standard\text{-}Deviation}(R_c)$;
12 $\delta \leftarrow \delta + \frac{|R_c|}{|R|} \mathrm{KL}(\mathcal{N}(\mu_c, \sigma_c^2) \parallel \mathcal{N}(\mu, \sigma^2))$;
13 **end**
14 **return** δ;

We illustrate the full algorithm for Reconditinor in Algorithm 1. In lines 1-2, we initialize the sets of residuals for both the marginal distribution $P(\Delta Y)$ and the conditional distributions $P(\Delta Y \mid C)$ for each $C \in [K]$. In lines 3-7, we update these sets with the residuals computed by f. We compute the mean and standard deviation values for $P(\Delta Y)$ in line 8 and perform a similar computation for $P(\Delta Y \mid C)$ in line 11. Finally, in line 12, we compute and average the KL divergences between $P(\Delta Y)$ and $P(\Delta Y \mid C)$ to obtain the detector score δ.

5.2 Sample-level contextualized adapter

A higher metric δ from Reconditionor signifies a stronger impact of CDS on a model. Given the nature of CDS, our primary concept is to adjust the model to align with the conditional distribution $P(Y|X, C)$ instead of the marginal distribution $P(Y|X)$. However, noticing that the context C is consistently changing at each time step, a one-size-fits-all adaptation of the model is inherently unfeasible. Therefore, we propose to carry out adaptations at the individual *sample-level*.

For each test sample $X_{t-L:t}$, it is not viable to adapt the model solely relying on the input $X_{t-L:t}$. Therefore, we commence by implementing data augmentation through the creation of a dataset derived from this specific sample, formulated as:

$$\mathcal{D}_{\text{ctx}} = \textsc{Select}(\{(\boldsymbol{X}_{t'-L:t'}, \boldsymbol{X}_{t':t'+T}) : t' + T \le t\}), \tag{2}$$

where SELECT operation involves the selection of preceding samples that share a similar context with the provided sample $\boldsymbol{X}_{t-L:t}$, and we will provide further elaboration on this operation in § 5.3. We denote the resulting dataset as the *contextualized dataset* ($\mathcal{D}_{\text{ctx}}$). Specifically, before making prediction to the test sample $X_{t-L:t}$, we employ $\mathcal{D}_{\text{ctx}}$ to adapt the model to alleviate the influence of CDS. We refer to this step as **Sample-level cOntextuaLIzed aDapter**, or SOLID. It's worth noting that since adaptation takes place during

the testing phase, we propose to modify the prediction layer h_θ while keeping the feature extractor g_Φ unchanged for efficiency. In our empirical analysis (§ 7.3.2), we observed that fine-tuning solely the prediction layer not only improves test phase efficiency but also consistently delivers better performance by mitigating the risk of overfitting.

Additionally, as we outline in the subsequent theoretical analysis (§ 6), the fine-tuning process is fundamentally a *bias-variance trade-off*: fine-tuning reduces the bias caused by CDS, but introduces an additional variance from the noise of contextualized dataset due to the data scarcity, potentially impacting model performance negatively. Thus, optimally tuning the fine-tuning steps to balance the trade-off is essential.

5.3 Contextualized dataset selection

As we mentioned previously, the core of adaptation involves creating the contextualized dataset $\mathcal{D}_{\text{ctx}}$ for the sample at t. In this section, we introduce the SELECT operation in Eq.(2). Note that due to the unavailability of the true context governing the data generation process, it is not feasible to select samples with precisely the same context. To address this issue, we design a comprehensive strategy based on the observable contexts (temporal segments and periodic phases), and employ sample similarity as a proxy for unobserved contexts.

5.3.1 Temporal segments. The data generation process typically evolves over time [13]. Consequently, we claim that the **temporal segment** is a critical context. Therefore, we focus on samples that are closely aligned with the test samples in the temporal dimension, formally,

$$\{(\boldsymbol{X}_{t'-L:t'}, \boldsymbol{X}_{t':t'+T}) : t - \lambda_T \le t' \le t - T\},$$

where λ_T controls the time range for selection. When samples are too distant from t, we conclude that they are in distinct temporal segments and consequently exclude them from selection.

5.3.2 Periodic phases. Furthermore, it's worth mentioning that time series data often exhibit periodic characteristics. The data generation process can vary across different phases. Therefore, we claim that the **periodic phase** constitutes another critical context.

To find the samples with similar phases, we need to detect the underlying periods. Specifically, we follow ETSformer [24] and TimesNet [25] to employ the widely-used Fast Fourier Transform (FFT) on the training dataset $\boldsymbol{X} \in \mathbb{R}^{t_{\text{train}} \times M}$ with length-t_{train} and M variables, formulated as:

$$T^* = \left\lfloor t_{\text{train}} \Big/ \left\{ \arg\max_{k \in \{2,\dots,\lfloor t_{\text{train}}/2 \rfloor\}} \sum_{i=1}^{M} \text{Ampl}(\text{FFT}(\boldsymbol{X}^i))_k \right\} \right\rfloor. \quad (3)$$

Here, $\boldsymbol{X}^i$ is the sequence of the i-th variables in $\boldsymbol{X}$, $\text{FFT}(\cdot)$ and $\text{Ampl}(\cdot)$ denote the Fast Fourier Transform (FFT) and amplitude respectively. To determine the most dominant frequency, we sum the amplitudes across all M channels and select the highest value, which is converted to the periodic length T^*.

Next, we employ the periodic length to select samples with closely aligned phases. Particularly, for the given test sample at time step t', we select samples that display minimal difference in the phases, formulated as

$$\left\{ (\boldsymbol{X}_{t'-L:t'}, \boldsymbol{X}_{t':t'+T}) : \left| \frac{t \bmod T^* - t' \bmod T^*}{T^*} \right| < \lambda_P \right\},$$

where $t \bmod T^*$ and $t' \bmod T^*$ are the phases of the test sample and preceding samples, respectively. λ_P is a hyperparameter controlling the threshold for the acceptable phase difference. If the difference exceeds a certain threshold, the preceding samples will not be considered to share the same context as the test sample.

5.3.3 Address unobserved contexts through sample similarity. Even though the strategies introduced in § 5.3.1 and § 5.3.2 efficiently identify potential samples with similar contexts, we cannot guarantee a consistent mapping relationship $X \mapsto Y$ for these samples due to the existence of *unobserved contexts*. To further enhance the quality of selection and address this issue, it's essential to recognize that context typically influences input data through a causal effect $C \mapsto X$, which suggests a correlation between contexts and inputs.

Inspired by this insight, we assume that when samples have similar inputs X, they are more likely to share a similar context C. Consequently, we incorporate **sample similarity** as a proxy of unobserved contexts. The calculation of the similarity can be any measurement of interest, and we employ the Euclidean distance as the chosen metric. Specifically, we select the top-λ_N similar samples, where λ_N serves as a hyperparameter governing the number of samples to be chosen.

5.3.4 Full algorithm for SOLID . Finally, we combine the above strategies for the SELECT operation, by first filtering the temporal segments and periodic phases (§ 5.3.1 and § 5.3.2) and then selecting top-λ_N samples based on sample similarity (§ 5.3.3).

We illustrate the full algorithm for SOLID in Algorithm 2. We first compute the periodic length on the training dataset (Eq.(3)). During the testing stage, for a given test sample, we perform the following steps: In lines 1-7, we filter the time steps based on the

Algorithm 2: Algorithm for SOLID

Input: Model $f = (g_\Phi, h_\theta)$, test sample $\boldsymbol{X}_{t-L:t}$, preceding data $\{(\boldsymbol{X}_{t'-L:t'}, \boldsymbol{X}_{t':t'+T}) : t' + T \le t\}$, similarity metric $S(\cdot,\cdot)$, periodic length T^* computed by Eq.(3), hyperparameters λ_T, λ_P, λ_N and lr.

Output: Prediction for the test sample: $\hat{\boldsymbol{X}}_{t:t+T}$

1 $\mathcal{T} \leftarrow \varnothing$;
2 **for** $t - \lambda_T \le t' \le t - T$ **do**
3 $\Delta_P \leftarrow \left| \frac{t \bmod T^* - t' \bmod T^*}{T^*} \right|$;
4 **if** $\Delta_P < \lambda_P$ **then**
5 $\mathcal{T} \leftarrow \mathcal{T} \cup \{t'\}$;
6 **end**
7 **end**
8 $\mathcal{T}_{\text{ctx}} \leftarrow \underset{t' \in \mathcal{T}}{\text{argTop-}\lambda_N}(S(\boldsymbol{X}_{t'-L:t'}, \boldsymbol{X}_{t-L:t}))$;
9 $\mathcal{D}_{\text{ctx}} \leftarrow \{(\boldsymbol{X}_{t'-L:t'}, \boldsymbol{X}_{t':t'+T}) \mid t \in \mathcal{T}_{\text{ctx}}\}$;
10 $h'_\theta \leftarrow$ fine-tune h_θ using $\mathcal{D}_{\text{ctx}}$ with a learning rate lr;
11 $\hat{\boldsymbol{X}}_{t:t+T} \leftarrow h'_\theta(g_\Phi(\boldsymbol{X}_{t-L:t}))$;
12 **return** $\hat{\boldsymbol{X}}_{t:t+T}$;

observed contexts, temporal segments (§ 5.3.1), and periodic phases (§ 5.3.2). In lines 8-9, we further select samples based on similarity (§ 5.3.3). In lines 10-11, we fine-tune the prediction layer (§ 5.2) and use it for making predictions.

6 THEORETICAL ANALYSIS

In this section, we provide a formal theoretical analysis to estimate the generalization error, before and after considering context, to illustrate the influence from CDS (§ 4), as well as the bias-variance trade-off during the fine-tuning process (§ 5.2). Given our fine-tuning targets the prediction layer with the feature extractor remaining frozen, our theoretical analysis centers on the latent representation space $g_\Phi(X)$ rather than the raw data X. To start with, we first make the following assumption for the generation process of Y:

Assumption 1 (Contextualized generation process). *Assume that the input latent representations $g_\Phi(X)$ on the training set can be divided into K context groups based on K different contexts: $(X_1, \cdots, X_K)$, where $X_i \in \mathbb{R}^{n_i \times d}$ and n_i is the number of data points in the i-th group. For each i, there exists a parameter vector $\theta_i \in \mathbb{R}^d$ such that the output Y_i follows:*

$$Y_i = X_i\theta_i + \epsilon_i,$$

where ϵ_i is an independent random noise, which satisfies $\mathbb{E}[\epsilon_i] = 0$, $\mathbb{VAR}[\epsilon_i] = \sigma^2$. Here we assume that Y_i are scalars for simplicity, although it can be readily extended to a multi-dimensional scenario.

This assumption extends the widely used *fixed design setting* [4, Chapter 3.5] to multi-context scenarios, which posits that the data generation parameters, θ_i, differ across various contexts i. The prediction layer h_θ directly trained without considering the contexts can be seen as a *global linear regressor* (GLR), as follows:

Definition 2 (Global linear regressor). *A global linear regressor (GLR) $h_{\hat{\theta}}$ parameterized by $\hat{\theta}$ is given by:*

$$\hat{\theta} = \arg\min_\theta \sum_{i=1}^{K} ||Y_i - X_i\theta||_2^2.$$

Define $\mathcal{R}(\alpha_1, \cdots, \alpha_K) = \mathbb{E}\left[\sum_{i=1}^K ||Y_i - X_i\alpha_i||_2^2\right]$ as the expected risk when using a parameter α_i to predict X_i ($i \in [K]$), and let $\mathcal{R}^*$ denote the minimum value of this risk. The following theorem computes the expected risk when we use the globally shared parameter for prediction, with its proof delegated to Appendix A.

Theorem 1 (Expected risk for GLR). *For $\hat{\theta}$ in Definition 2:*

$$\mathcal{R}(\underbrace{\hat{\theta}, \cdots, \hat{\theta}}_{K}) - \mathcal{R}^* = \underbrace{\sum_{i=1}^{K} \left\|\overline{\theta} - \theta_i\right\|_{\psi_i}^2}_{\text{bias part}} + \underbrace{\sigma^2 d}_{\text{variance part}},$$

where $\psi_i = X_i^\top X_i$, and $\overline{\theta} = (\sum_{i=1}^K \psi_i)^{-1}(\sum_{i=1}^K \psi_i\theta_i)$. The quantity $||\cdot||_{\psi_i}$ is the Mahalanobis distance norm, defined as $\|\theta\|_{\psi_i}^2 = \theta^\top\psi_i\theta$.

The bias part in Theorem 1 indicates that GLR is unbiased only when the data generation parameters θ_i are identical across all groups. However, if they differ due to the influence of contexts, the regressor is biased regardless of the amount of data, *i.e.*, CDS.

In the next, we explore a straightforward approach to address CDS: discarding the existing biased regressor and training a new individual regressor $\hat{\theta}_i$ for each context group X_i ($i \in [K]$) to eliminate the bias. We refer to this ensemble of regressors as *contextualized linear regressors* (CLR), as follows:

Definition 3 (Contextualized linear regressor). *A set of contextualized linear regressors (CLR) $h_{\hat{\theta}_i}$ parameterized by $\hat{\theta}_i$ ($i \in [K]$) are given by:*

$$\hat{\theta}_i = \arg\min_\theta ||Y_i - X_i\theta||_2^2, \quad \forall i \in \{1, 2, ..., K\}.$$

Using the same risk notations, the following theorem computes the expected risk for CLR (the proof is detailed in Appendix A).

Theorem 2 (Expected risk for CLR). *For $\hat{\theta}_i$ in Definition 3:*

$$\mathcal{R}(\hat{\theta}_1, \cdots, \hat{\theta}_K) - \mathcal{R}^* = \underbrace{0}_{\text{bias part}} + \underbrace{K\sigma^2 d}_{\text{variance part}}.$$

Comparing Theorem 1 and Theorem 2, we observe that CLR is always *unbiased* since it addresses the CDS. However, it suffers from a *larger variance* compared to GLR. Specifically, the variance of CLR is K-times larger than that of GLR, indicating that more detailed contexts result in higher variance. This makes sense because as K increases, the number of data available for training each CLR diminishes, consequently elevating the variance.

Based on the above findings, we argue that it's crucial to combine GLR and CLR to balance the bias and variance. This can be implemented through a standard pre-training / fine-tuning paradigm. In the pre-training stage, contexts are disregarded and a GLR h_θ is trained on the training dataset. This pre-training stage mirrors the conventional standard training process. In the fine-tuning stage, for a given new sample at t, we employ the dataset with the same context as this sample to fine-tune the learned GLR for limited steps. This stage mirrors the usage of the proposed SOLID, which brings GLR closer to CLR and reduces bias. In cases where the influence of CDS is substantial (*i.e.*, the bias part in Theorem 1 is large), it is advisable to increase the learning rate and the number of fine-tuning steps to mitigate bias. In the converse case, the fine-tuning process should be more limited to reduce variance, as CLR has greater variance (Theorem 2). In practice, it is recommended to tune the learning hyperparameters of SOLID to achieve an optimal trade-off between bias and variance.

7 EXPERIMENTS

In this section, we describe the experimental settings and provide extensive results and analysis. [1]

7.1 Experiment settings

Datasets. We conduct the experiments on 8 popular datasets for TSF: Electricity, Traffic, Illness, Weather [14], and 4 ETT datasets (ETTh1, ETTh2, ETTm1, ETTm2) [29]. We follow the standard preprocessing protocol [26, 29] and partition the datasets into train/validation/test sets by the ratio of 6:2:2 for ETT and 7:1:2 for the other datasets. Appendix B.1 contains more dataset details.

[1] Code is available at https://github.com/HALF111/calibration_CDS.

Table 1: Performance comparison. "24 / 96": prediction length is 24 (Illness) or 96 (other datasets), applicable similarly to "36 / 192", *etc.* "↑": average improvements achieved by *SOLID* compared to the baseline. "δ": metrics given by *Reconditioner* in two observed contexts: periodic phases (δ_P) and temporal segments (δ_T), in the form of $\log_{10}\delta_P$ & $\log_{10}\delta_T$. RED highlights a *strong* CDS in periodic phases (*i.e.*, $\log_{10}\delta_P \geq -3.2$), while BLUE highlights a *weak* CDS in periodic phases (*i.e.*, $\log_{10}\delta_P < -3.2$).

Dataset		Illness				Electricity				Traffic				ETTh1				ETTh2			
Method		/		+SOLID		/		+SOLID		/		+SOLID		/		+SOLID		/		+SOLID	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Informer	24 / 96	5.096	1.533	**2.874**	**1.150**	0.321	0.407	**0.245**	**0.355**	0.731	0.406	**0.628**	**0.393**	0.948	0.774	**0.684**	**0.586**	2.992	1.362	**1.659**	**0.988**
	36 / 192	5.078	1.535	**3.299**	**1.243**	0.351	0.434	**0.256**	**0.363**	0.739	0.414	**0.653**	**0.413**	1.009	0.786	**0.759**	**0.624**	6.256	2.091	**3.564**	**1.553**
	48 / 336	5.144	1.567	**2.879**	**1.169**	0.349	0.432	**0.279**	**0.381**	0.850	0.476	**0.758**	**0.466**	1.035	0.783	**0.776**	**0.640**	5.265	1.954	**1.845**	**1.079**
	60 / 720	5.243	1.582	**3.773**	**1.394**	0.385	0.493	**0.327**	**0.416**	0.945	0.530	**0.861**	**0.524**	1.153	0.845	**0.948**	**0.726**	4.038	1.673	**2.020**	**1.104**
	↑			**37.62%**	**20.29%**			**21.28%**	**14.21%**			**11.19%**	**1.65%**			**23.60%**	**19.18%**			**51.01%**	**33.29%**
	δ			**-1.096 & -1.148**				**-2.975 & -2.593**				**-2.762 & -2.238**				**-3.83 & -1.883**				**-3.021 & -1.513**	
Autoformer	24 / 96	3.314	1.245	**2.737**	**1.118**	0.207	0.324	**0.189**	**0.304**	0.621	0.391	**0.565**	**0.376**	0.440	0.444	**0.430**	**0.442**	0.363	0.405	**0.362**	**0.404**
	36 / 192	2.733	1.078	**2.540**	**1.015**	0.221	0.334	**0.205**	**0.316**	0.666	0.415	**0.599**	**0.379**	0.487	0.472	**0.480**	**0.470**	0.450	0.447	**0.449**	**0.446**
	48 / 336	2.651	1.075	**2.455**	**1.042**	0.244	0.350	**0.232**	**0.337**	0.649	0.405	**0.595**	**0.390**	0.471	0.475	**0.467**	**0.473**	0.470	0.474	**0.468**	**0.471**
	60 / 720	2.848	1.126	**2.689**	**1.082**	0.285	0.381	**0.273**	**0.370**	0.684	0.422	**0.635**	**0.411**	0.543	0.528	**0.542**	**0.527**	0.484	0.491	**0.483**	**0.489**
	↑			**9.75%**	**5.90%**			**6.16%**	**4.44%**			**8.61%**	**4.80%**			**1.08%**	**0.35%**			**0.25%**	**0.35%**
	δ			**-1.166 & -1.016**				**-2.408 & -2.134**				**-2.507 & -2.304**				**-3.572 & -2.321**				**-3.967 & -1.921**	
FEDformer	24 / 96	3.241	1.252	**2.707**	**1.123**	0.188	0.304	**0.172**	**0.284**	0.574	0.356	**0.513**	**0.344**	0.375	0.414	**0.370**	**0.410**	0.341	0.385	**0.339**	**0.383**
	36 / 192	2.576	1.048	**2.365**	**0.990**	0.197	0.311	**0.180**	**0.290**	0.612	0.379	**0.549**	**0.361**	0.427	0.448	**0.420**	**0.443**	0.433	0.441	**0.432**	**0.440**
	48 / 336	2.546	1.058	**2.435**	**1.023**	0.213	0.328	**0.195**	**0.307**	0.618	0.379	**0.557**	**0.363**	0.458	0.465	**0.454**	**0.462**	0.503	0.494	**0.501**	**0.491**
	60 / 720	2.784	1.136	**2.677**	**1.118**	0.243	0.352	**0.228**	**0.336**	0.629	0.382	**0.577**	**0.371**	0.482	0.495	**0.478**	**0.492**	0.479	0.485	**0.478**	**0.484**
	↑			**8.64%**	**5.34%**			**7.88%**	**5.95%**			**9.77%**	**3.70%**			**1.10%**	**0.74%**			**0.42%**	**0.40%**
	δ			**-1.099 & -0.917**				**-2.967 & -2.545**				**-2.254 & -2.265**				**-3.52 & -2.463**				**-3.733 & -1.943**	
ETSformer	24 / 96	2.397	0.993	**2.262**	**0.955**	0.187	0.304	**0.171**	**0.285**	0.599	0.386	**0.487**	**0.354**	0.495	0.480	**0.491**	**0.478**	0.346	0.401	**0.344**	**0.399**
	36 / 192	2.504	0.970	**2.301**	**0.934**	0.198	0.313	**0.184**	**0.298**	0.611	0.391	**0.492**	**0.354**	0.543	0.505	**0.538**	**0.503**	0.437	0.447	**0.430**	**0.443**
	48 / 336	2.488	0.999	**2.320**	**0.961**	0.210	0.326	**0.197**	**0.312**	0.619	0.393	**0.501**	**0.359**	0.581	0.521	**0.574**	**0.518**	0.478	0.479	**0.467**	**0.472**
	60 / 720	2.494	1.011	**2.358**	**1.002**	0.249	0.356	**0.234**	**0.341**	0.629	0.391	**0.529**	**0.374**	0.569	0.534	**0.562**	**0.530**	0.488	0.492	**0.474**	**0.482**
	↑			**6.50%**	**3.03%**			**6.89%**	**4.75%**			**18.28%**	**7.57%**			**1.08%**	**0.56%**			**1.93%**	**1.31%**
	δ			**-1.544 & -1.001**				**-2.559 & -2.724**				**-2.488 & -2.201**				**-3.207 & -3.019**				**-3.067 & -1.079**	
Crossformer	24 / 96	3.329	1.275	**2.353**	**0.986**	0.184	0.297	**0.182**	**0.295**	0.521	0.297	**0.473**	**0.277**	0.411	0.432	**0.382**	**0.415**	0.641	0.555	**0.527**	**0.544**
	36 / 192	3.392	1.185	**2.527**	**1.042**	0.219	0.317	**0.216**	**0.314**	0.523	0.298	**0.475**	**0.280**	0.419	0.444	**0.396**	**0.422**	1.262	0.814	**0.834**	**0.725**
	48 / 336	3.481	1.228	**2.499**	**1.063**	0.238	0.348	**0.235**	**0.343**	0.530	0.300	**0.481**	**0.286**	0.439	0.459	**0.418**	**0.443**	1.486	0.896	**1.048**	**0.835**
	60 / 720	3.571	1.234	**3.103**	**1.199**	0.274	0.373	**0.269**	**0.368**	0.573	0.313	**0.498**	**0.298**	0.504	0.514	**0.473**	**0.503**	1.220	0.848	**0.906**	**0.758**
	↑			**23.89%**	**12.84%**			**1.42%**	**1.11%**			**10.25%**	**5.55%**			**5.94%**	**3.53%**			**28.05%**	**8.06%**
	δ			**-1.038 & -0.917**				**-2.333 & -2.42**				**-2.879 & -2.171**				**-3.111 & -2.767**				**-2.451 & -1.149**	
DLinear	24 / 96	1.947	0.985	**1.843**	**0.938**	0.142	0.238	**0.140**	**0.237**	0.412	0.283	**0.404**	**0.277**	0.375	0.397	**0.368**	**0.391**	0.284	0.349	**0.280**	**0.346**
	36 / 192	2.182	1.036	**1.692**	**0.898**	0.153	0.250	**0.152**	**0.249**	0.423	0.287	**0.420**	**0.285**	0.418	0.429	**0.405**	**0.416**	0.389	0.422	**0.360**	**0.398**
	48 / 336	2.256	1.060	**1.694**	**0.916**	0.169	0.268	**0.168**	**0.266**	0.436	0.295	**0.432**	**0.291**	0.451	0.452	**0.436**	**0.435**	0.422	0.447	**0.400**	**0.430**
	60 / 720	2.381	1.102	**2.139**	**1.012**	0.204	0.301	**0.203**	**0.300**	0.466	0.315	**0.459**	**0.310**	0.624	0.593	**0.553**	**0.547**	0.698	0.594	**0.415**	**0.451**
	↑			**15.95%**	**10.02%**			**0.75%**	**0.47%**			**1.27%**	**1.44%**			**5.67%**	**4.38%**			**18.85%**	**10.32%**
	δ			**-1.821 & -1.301**				**-3.303 & -2.645**				**-2.883 & -2.589**				**-2.981 & -2.321**				**-3.023 & -1.891**	
PatchTST	24 / 96	1.301	0.734	**1.253**	**0.710**	0.134	0.227	**0.132**	**0.226**	0.385	0.263	**0.383**	**0.262**	0.375	0.400	**0.368**	**0.393**	0.274	0.336	**0.273**	**0.336**
	36 / 192	1.483	0.841	**1.449**	**0.823**	0.151	0.243	**0.150**	**0.242**	0.393	0.265	**0.392**	**0.264**	0.408	0.411	**0.402**	**0.407**	0.340	0.380	**0.339**	**0.379**
	48 / 336	1.652	0.845	**1.624**	**0.831**	0.168	0.262	**0.167**	**0.262**	0.403	0.273	**0.402**	**0.271**	0.431	0.430	**0.428**	**0.428**	0.332	0.383	**0.331**	**0.382**
	60 / 720	1.731	0.886	**1.661**	**0.843**	0.201	0.292	**0.200**	**0.291**	0.439	0.295	**0.438**	**0.295**	0.442	0.452	**0.433**	**0.447**	0.378	0.420	**0.377**	**0.418**
	↑			**2.92%**	**2.99%**			**0.76%**	**0.29%**			**0.31%**	**0.36%**			**1.51%**	**1.06%**			**0.30%**	**0.26%**
	δ			**-1.172 & -1.054**				**-3.834 & -2.878**				**-3.677 & -2.901**				**-3.087 & -2.529**				**-3.299 & -1.851**	

Results of *ETTm1*, *ETTm2* and *Weather* datasets are included in Table 8 of Appendix C.1, due to space limit.

Baseline models. As aforementioned, our proposed approach is a general framework that can calibrate many deep forecasting models. To verify the effectiveness, we utilize several forecasting models for the detection and adaptation, including Informer [29], Autoformer [26], FEDformer [30], ETSformer [24], Crossformer [28], DLinear [27] and PatchTST [16]. Appendix B.2 contains more details about the baselines.

Experimental details. For a fair comparison, we set the prediction length T of {24,36,48,60} for the Illness dataset, and {96,192,336,720} for the others, which aligns with the common setting for TSF tasks [26, 30]. Additionally, for other hyper-parameters, we follow the primary settings proposed in each respective paper [24, 26, 28–30]. For our SOLID, we employ gradient descent to fine-tune the prediction layer on $\mathcal{D}_{\text{ctx}}$ for one epoch. Appendix B.3 contains more details about the hyper-parameters.

Evaluation metrics. Consistent with previous research [26, 29, 30], we compare the performance via two widely-used metrics: Mean Squared Error (MSE) and Mean Absolute Error (MAE). Smaller MSE and MAE indicate better forecasting performance. We also compute the proposed Reconditionor as a metric for detecting the extent of models' susceptibility to CDS based on the two observed contexts, periodic phase (δ_P) and temporal segment (δ_T). For δ_P, the number of contexts equals to the periodic length, detailed in Eq.(3). For δ_T, we partition the training set into five equal segments, with each one representing a distinct temporal segment. Larger δ_P and δ_T indicate a stronger CDS for the given model and dataset.

7.2 Main results analysis

Table 1 shows the main results of our proposed SOLID alongside the scores of the proposed Reconditionor. We conduct a thorough analysis of the data from both perspectives.

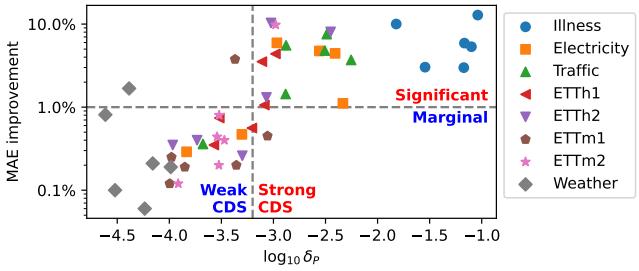


Figure 4: The relationship of Reconditionor metric $\log_{10} \delta_P$ (X-axis) and MAE improvements achieved by SOLID (Y-axis) for 8 datasets and 6 models.

7.2.1 Effectiveness of SOLID.

- SOLID enhances the performance of all models across various datasets by effectively addressing the CDS issue. Particularly, Informer, which is relatively less powerful, experiences a significant enhancement of 10%-60% in MSE. One explanation is that CDS has a more detrimental effect on weaker models, as they struggle to learn diverse patterns under different contexts.
- For datasets and models showing significant CDS under the context of periodic phases (*i.e.*, $\log_{10} \delta_P \geq -3.2$), SOLID achieves a considerable improvement. Specifically, it yields an average improvement of 15.1% for Illness, 9.9% for Traffic, and 8.7% for Electricity in terms of MSE on these cases. This highlights the effectiveness of SOLID in addressing severe CDS issues.
- On the cases of weak CDS under the context of periodic phases (*i.e.*, $\log_{10} \delta_P$ below the -3.2 threshold), SOLID still achieves a consistent improvement ranging from 0.3% to 6.7% across different datasets. This observation underscores the widespread presence of CDS in time series.

7.2.2 Effectiveness of Reconditionor.

- Our Reconditionor effectively assesses the magnitude of CDS and aligns with the enhancement achieved by SOLID. When employing the -3.2 threshold for $\log_{10} \delta_P$ to ascertain whether MAE improvement surpasses 1%, the classification accuracy reaches 89.3% (50 out of 56). This demonstrates that this detector is universally applicable and unaffected by different datasets and models.
- One exception is that Informer consistently achieves an improvement above 1%, irrespective of Reconditionor metrics. As aforementioned, one explanation is that Informer is relatively weaker, making it more amenable to improvement by SOLID. When the results of Informer are excluded, the classification accuracy for Reconditionor rises to 93.8% (45 out of 48).
- In addition, δ_P explains the performance improvement better than δ_T. We will give a possible conjecture in the following subsection to elucidate this observation.

7.2.3 Correlation of Reconditionor and SOLID. To further investigate the correlation between Reconditionor and the improvements achieved by SOLID, we plot δ_P and MAE improvements in Figure 4 based on the results of Table 1 and Table 8 in Appendix C.1. To depict the trend more effectively, we have excluded the data of Informer as previously explained. Notably, we have observed a pronounced and consistent upward trend between δ_P and MAE improvement in Figure 4. This trend is highly evident, with a Spearman correlation coefficient of 0.7998. This finding indicates that δ_P can serve as a valuable metric for estimating and explaining performance improvements. We also elaborate on the relationship between δ_T and MAE improvement in detail in Appendix C.3.

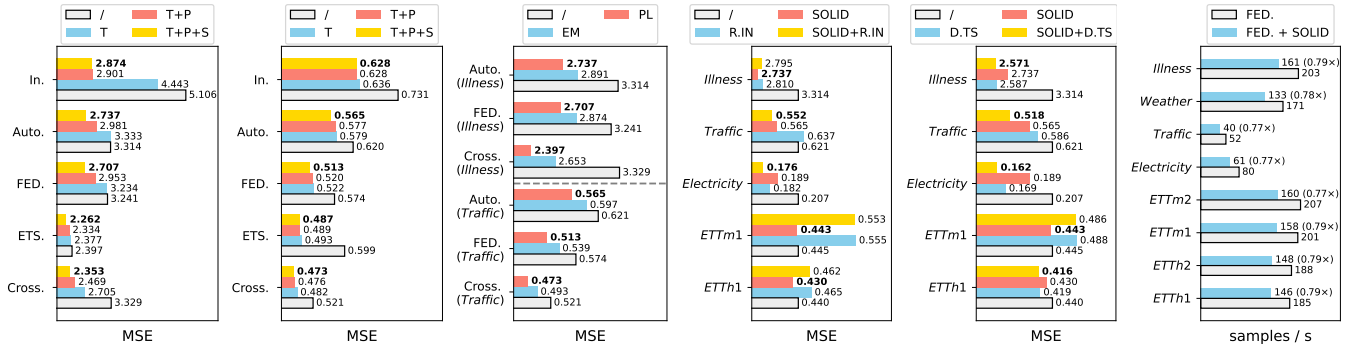
7.3 Further analysis for SOLID

In this section, we conducted additional experiments to further analyze SOLID. Figure 5 presents partial results, with the full results reported in Appendix C.1.

7.3.1 Ablation studies. We conducted ablation studies to investigate the relationship between performance and each component in the selection strategy, particularly each observable or unobservable context we propose in § 5.3. The investigation involves a progression of strategy components. Specifically, we commence from original forecasts that ignore any contexts, denoted as "/". We then create contextualized datasets with various selection variations. For a fair comparison, all datasets are constructed by selecting λ_N samples from λ_T nearest ones. We first consider **Temporal segment** context by selecting the nearest λ_N samples, denoted as **"T"**. Then we take **Periodic phase** into consideration, and select λ_N nearest samples with phase differences less than λ_P, denoted as **"T+P"**. Finally, we add **sample Similarity** and select the top-λ_N similar samples from nearest λ_T samples with phase differences less than λ_P. This strategy mirrors SOLID and is denoted as **"T+P+S"**. Based on the results in Figure 5(a) and 5(b), we showcase that each component leads to a consistent improvement in both MSE and MAE. These findings provide compelling evidence supporting the effectiveness of all components utilized in our selection strategy as well as the SOLID method.

7.3.2 Comparative studies on adapting only prediction layer or entire model. All our previous experiments are performed solely by adapting the prediction layer while keeping the crucial parameters of bottom feature extractor layers unchanged. Nevertheless, it is imperative to conduct comparative experiments to validate this perspective. Specifically, we compare the results achieved by solely adapting the prediction layer against adapting the entire model. The experimental results are presented in Figure 5(c). It indicates that solely adapting the prediction layer consistently yields superior performance. One explanation is that it reduces the risk of overfitting. Moreover, solely adapting the prediction layer significantly faster the speed of inference.

7.3.3 Comparison studies against other approaches addressing distribution shifts. In this section, we compared two widely used baselines, RevIN [11] and Dish-TS [8] which also address distribution shifts in time series forecasting. It is worth mentioning that they are strategies that work during training, which differs from our SOLID acting as a post-calibration method during the test process. Also, both RevIN and Dish-TS adapt the *input data* to better suit the model, whereas SOLID adapts the *model* to match the data distribution of the current context. Therefore, these methods are complementary and not exclusive. We explored their joint use on Autoformer, shown in Figures 5(d) and 5(e). We can observe that SOLID consistently improves performance, regardless of whether models were trained with RevIN or Dish-TS. Moreover, in many



(a) Ablation (illness) (b) Ablation (traffic) (c) Tuning strategies (d) RevIN (e) Dish-TS (f) Speed

Figure 5: Further analysis of SOLID. (a)(b) Ablation studies for three contexts, temporal segment (T), periodic phase (P), and sample similarity (S). (c) Studies on tuning strategies to explore adaptation on prediction layer (PL) only vs. entire model (EM). (d)(e) Comparison studies against RevIN (R.IN) and Dish-TS (D.TS). (f) Efficiency studies on the prediction speed.

cases, combining them achieves the best performance, suggesting the orthogonality of Context-driven DS (CDS) with other types of DS addressed by them.

7.3.4 Efficiency analysis. Furthermore, we analyze the practical efficiency of the proposed framework. For Reconditionor, it conducts a one-time analysis of residuals during the training process, thus not incurring any additional overhead during deployment. For SOLID, it incurs two steps of overhead for each prediction sample: (1) constructing the contextualized dataset, and (2) fine-tuning the model. Step (1) can be optimized using some engineering techniques, such as pre-caching intermediate embeddings and building efficient vector indexing. For step (2), since only the prediction layer is fine-tuned, the computational cost will not be too high. To further assess SOLID's efficiency, we report the additional time overhead introduced by SOLID to FEDFormer across various datasets in Figure 5(f). The results indicate that SOLID's inference speed is about 80% of traditional methods, a gap smaller than the variability introduced by different datasets, which suggests that our method does not significantly reduce inference speed.

7.3.5 Parameter sensitivity analysis and case study. We adjusted the hyperparameters within SOLID across multiple datasets and models to assess their impact on performance and conducted a case study to visualize the improvement. From the results, we discover that SOLID is insensitive to λ_T, which controls the time range of preceding data for selection, and λ_P, which governs the acceptable threshold for periodic phase difference. But concerning λ_N, which determines the number of similar samples to be selected for model adaptation, and lr, which regulates the extent of adaptation on the prediction layer for the models, they exist an optimal value and should be well selected. The detailed results are reported in Appendix C.4 and Appendix C.5.

8 CONCLUSION

In this paper, we introduce context-driven distribution shift (CDS) problem in TSF and identify two significant observed contexts, including temporal segments and periodic phases, along with unobserved contexts. To address the issue, we propose a general calibration framework, including a detector, Reconditionor, to evaluate the degree of a model's susceptibility to CDS and the necessity for model adaptation; and an adaptation framework, SOLID, for calibrating models and enhancing their performance under severe CDS. We conduct extensive experiments on 8 real-world datasets and 7 models, demonstrating the accuracy of Reconditionor in detecting CDS and the effectiveness of SOLID in adapting TSF models without substantially compromising time efficiency. This adaptation consistently leads to improved performance, which can be well explained by Reconditionor.

Limitations and future work. (1) Despite demonstrating consistent performance improvements, SOLID introduces an approximate additional 20% time overhead during the testing phase. Employing further engineering optimizations to narrow this efficiency gap will be investigated in future work. (2) While Reconditionor proves effective in leveraging observed contexts to ascertain the impact of CDS, whether unobserved context can be used for detection is an interesting open question. (3) The selection of contextually similar samples relies on heuristic rules in this paper. Investigating further contexts or developing more generalized selection criteria represents a promising area of research.

ACKNOWLEDGMENTS

Research work mentioned in this paper is supported by State Street Zhejiang University Technology Center. We would also like to thank Yusu Hong for the valuable discussion of the theory.

REFERENCES

[1] Mohsen Ahmadi, Saeid Ghoushchi, Rahim Taghizadeh, and Abbas Sharifi. 2019. Presentation of a new hybrid approach for forecasting economic growth using artificial intelligence approaches. *Neural Computing and Applications* 31 (12 2019), 8661–8680. https://doi.org/10.1007/s00521-019-04417-0

[2] Rafal Angryk, Petrus Martens, Berkay Aydin, Dustin Kempton, Sushant Mahajan, Sunitha Basodi, Azim Ahmadzadeh, Xumin Cai, Soukaina Filali Boubrahimi, Shah Muhammad Hamdi, Michael Schuh, and Manolis Georgoulis. 2020. Multivariate time series dataset for space weather data analytics. *Scientific Data* 7 (07 2020), 227. https://doi.org/10.1038/s41597-020-0548-x

[3] Sercan O. Arik, Nathanael C. Yoder, and Tomas Pfister. 2022. Self-Adaptive Forecasting for Improved Deep Learning on Non-Stationary Time-Series. arXiv:2202.02403 [cs.LG]

[4] Francis Bach. 2023. Learning theory from first principles. (2023).

[5] Chirag Deb, Fan Zhang, Junjing Yang, Siew Eang Lee, and Kwok Wei Shah. 2017. A review on time series forecasting techniques for building energy consumption. *Renewable and Sustainable Energy Reviews* 74 (2017), 902–924. https://doi.org/10.1016/j.rser.2017.02.085
[6] Funda Demirel, Selim Zaim, Ahmet Caliskan, and Pinar Gokcin Ozuyar. 2012. Forecasting natural gas consumption in Istanbul using neural networks and multivariate time series methods. *Turkish Journal of Electrical Engineering and Computer Sciences* 20 (01 2012), 695–711. https://doi.org/10.3906/elk-1101-1029
[7] Yuntao Du, Jindong Wang, Wenjie Feng, Sinno Pan, Tao Qin, Renjun Xu, and Chongjun Wang. 2021. AdaRNN: Adaptive Learning and Forecasting of Time Series. In *CIKM*. Association for Computing Machinery, New York, NY, USA, 402–411. https://doi.org/10.1145/3459637.3482315
[8] 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 *AAAI*, Vol. 37. 7522–7529.
[9] Yan Hu, Qimin Peng, Xiaohui Hu, and Rong Yang. 2015. Web Service Recommendation Based on Time Series Forecasting and Collaborative Filtering. In *2015 IEEE International Conference on Web Services*. 233–240. https://doi.org/10.1109/ICWS.2015.40
[10] Zahra Karevan and Johan A.K. Suykens. 2020. Transductive LSTM for time-series prediction: An application to weather forecasting. *Neural Networks* 125 (2020), 1–9. https://doi.org/10.1016/j.neunet.2019.12.030
[11] Taesung Kim, Jinhee Kim, Yunwon Tae, Cheonbok Park, Jang-Ho Choi, and Jaegul Choo. 2022. Reversible Instance Normalization for Accurate Time-Series Forecasting against Distribution Shift. In *ICLR*. https://openreview.net/forum?id=cGDAkQo1C0p
[12] Sean Kulinski, Saurabh Bagchi, and David I Inouye. 2020. Feature shift detection: Localizing which features have shifted via conditional distribution tests. *NeurIPS* 33 (2020), 19523–19533.
[13] Vitaly Kuznetsov and Mehryar Mohri. 2015. Learning theory and algorithms for forecasting non-stationary time series. *NeurIPS* 28 (2015).
[14] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. 2018. Modeling long-and short-term temporal patterns with deep neural networks. In *SIGIR*. 95–104.
[15] Zachary Lipton, Yu-Xiang Wang, and Alexander Smola. 2018. Detecting and correcting for label shift with black box predictors. In *ICML*. PMLR, 3122–3130.
[16] Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. 2022. A Time Series is Worth 64 Words: Long-term Forecasting with Transformers. In *ICLR*.
[17] Nikolaos Passalis, Anastasios Tefas, Juho Kanniainen, Moncef Gabbouj, and Alexandros Iosifidis. 2019. Deep adaptive input normalization for time series forecasting. *IEEE transactions on neural networks and learning systems* 31, 9 (2019), 3760–3765.
[18] Andrew Patton. 2013. Chapter 16 - Copula Methods for Forecasting Multivariate Time Series. In *Handbook of Economic Forecasting*, Graham Elliott and Allan Timmermann (Eds.). Handbook of Economic Forecasting, Vol. 2. Elsevier, 899–960. https://doi.org/10.1016/B978-0-444-62731-5.00016-6
[19] Judea Pearl. 2009. *Causality*. Cambridge university press.
[20] Stephan Rabanser, Stephan Günnemann, and Zachary Lipton. 2019. Failing loudly: An empirical study of methods for detecting dataset shift. *NeurIPS* 32 (2019).
[21] Vijendra Pratap Singh, Manish Kumar Pandey, Pangambam Sendash Singh, and Subbiah Karthikeyan. 2020. An Econometric Time Series Forecasting Framework for Web Services Recommendation. *Procedia Computer Science* 167 (2020), 1615–1625. https://doi.org/10.1016/j.procs.2020.03.372 International Conference on Computational Intelligence and Data Science.
[22] Vijendra Pratap Singh, Manish Kumar Pandey, Pangambam Sendash Singh, and Subbiah Karthikeyan. 2020. Neural Net Time Series Forecasting Framework for Time-Aware Web Services Recommendation. *Procedia Computer Science* 171 (2020), 1313–1322. https://doi.org/10.1016/j.procs.2020.04.140 Third International Conference on Computing and Network Communications (CoCoNet'19).
[23] Slawek Smyl. 2019. A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting. *International Journal of Forecasting* 36 (07 2019). https://doi.org/10.1016/j.ijforecast.2019.03.017
[24] Gerald Woo, Chenghao Liu, Doyen Sahoo, Akshat Kumar, and Steven C. H. Hoi. 2022. ETSformer: Exponential Smoothing Transformers for Time-series Forecasting. *CoRR* abs/2202.01381 (2022). arXiv:2202.01381 https://arxiv.org/abs/2202.01381
[25] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. 2023. TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis. In *ICLR*. https://openreview.net/forum?id=ju_Uqw384Oq
[26] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. 2021. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *NeurIPS* 34 (2021), 22419–22430.
[27] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. 2022. Are Transformers Effective for Time Series Forecasting? arXiv:2205.13504 [cs.AI]
[28] Yunhao Zhang and Junchi Yan. 2023. Crossformer: Transformer Utilizing Cross-Dimension Dependency for Multivariate Time Series Forecasting. In *ICLR*. https://openreview.net/forum?id=vSVLM2j9eie
[29] 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 *AAAI*, Vol. 35. 11106–11115.
[30] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. 2022. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *ICML*. PMLR, 27268–27286.

APPENDIX

A PROOFS FOR THE THEORETICAL RESULTS

In this section, we give the proofs for the expected risks in Theorem 1 and Theorem 2. To start with, we give the following lemma to perform the usual bias/variance decomposition. The risk notations $\mathcal{R}$ and $\mathcal{R}^*$ are defined in § 6.

Lemma A.1 (Risk decomposition). *For a set of random variables $\alpha_1, \cdots, \alpha_K$ used as parameters for each group, we have:*

$$\mathcal{R}(\alpha_1, \cdots, \alpha_K) - \mathcal{R}^* = \underbrace{\sum_{i=1}^{K} \|\mathbb{E}[\alpha_i] - \theta_i\|_{\psi_i}^2}_{\textit{bias part}} + \underbrace{\sum_{i=1}^{K} \mathbb{E}\left[\|\alpha_i - \mathbb{E}[\alpha_i]\|_{\psi_i}^2\right]}_{\textit{variance part}},$$

where $\psi_i = X_i^\top X_i$, $\overline{\theta} = (\sum_{i=1}^K \psi_i)^{-1}(\sum_{i=1}^K \psi_i \theta_i)$, and $\|\theta\|_{\psi_i}^2 = \theta^\top \psi_i \theta$.

Proof. This is a direct corollary of [4, Proposition 3.3]. □

Based on the risk decomposition, we give the computation for GLR and CLR respectively, as follows.

A.1 Proof for Theorem 1

Set $\alpha_1 = \cdots = \alpha_K = \hat{\theta}$ in Lemma A.1, we obtain:

$$\underbrace{\mathcal{R}\left(\hat{\theta}, \cdots, \hat{\theta}\right)}_{K} - \mathcal{R}^* = \underbrace{\sum_{i=1}^{K} \left\|\mathbb{E}[\hat{\theta}] - \theta_i\right\|_{\psi_i}^2}_{\text{bias part}} + \underbrace{\sum_{i=1}^{K} \mathbb{E}\left[\left\|\alpha_i - \mathbb{E}[\hat{\theta}]\right\|_{\psi_i}^2\right]}_{\text{variance part}}.$$

We first compute the expectation in the bias part. Recall that,

$$\hat{\theta} = \arg\min_{\theta} \sum_{i=1}^{K} \|Y_i - X_i\theta\|_2^2,$$

which has a closed-form solution, as follows:

$$\hat{\theta} = \left(\sum_{i=1}^{K} \psi_i\right)^{-1} \left(\sum_{i=1}^{K} X_i^\top Y_i\right) = \left(\sum_{i=1}^{K} \psi_i\right)^{-1} \left(\sum_{i=1}^{K} X_i^\top (X_i\theta_i + \epsilon_i)\right). \tag{A.1}$$

Therefore, we can obtain the expectation:

$$\begin{aligned}\mathbb{E}[\hat{\theta}] &= \left(\sum_{i=1}^{K} \psi_i\right)^{-1} \left(\sum_{i=1}^{K} X_i^\top (X_i\theta_i + \mathbb{E}[\epsilon_i])\right) \\ &= \left(\sum_{i=1}^{K} \psi_i\right)^{-1} \left(\sum_{i=1}^{K} \psi_i\theta_i\right) := \overline{\theta},\end{aligned} \tag{A.2}$$

which implies the bias part. Then we compute the variance part V:

$$V = \sum_{i=1}^{K} \mathbb{E}\left[\left\|\hat{\theta} - \mathbb{E}[\hat{\theta}]\right\|_{\psi_i}^2\right]$$

$$
\begin{aligned}
&= \sum_{i=1}^{K} \mathbb{E}\left[\left\|\left(\sum_{j=1}^{K}\psi_j\right)^{-1}\left(\sum_{j=1}^{K}X_j^\top\epsilon_j\right)\right\|_{\psi_i}^2\right] \cdots \text{by Eq.(A.1) and Eq.(A.2)} \\
&= \sum_{i=1}^{K} \mathbb{E}\left[\left(\sum_{j=1}^{K}X_j^\top\epsilon_j\right)^\top \underbrace{\left(\sum_{j=1}^{K}\psi_j\right)^{-1}\psi_i\left(\sum_{j=1}^{K}\psi_j\right)^{-1}}_{\text{denoted by } P_i}\left(\sum_{j=1}^{K}X_j^\top\epsilon_j\right)\right] \\
&= \sum_{i=1}^{K} \mathbb{E}\left[\left(\sum_{j=1}^{K}X_j^\top\epsilon_j\right)^\top P_i\left(\sum_{j=1}^{K}X_j^\top\epsilon_j\right)\right] \\
&= \sum_{i=1}^{K} \mathbb{E}\left[\left(\sum_{j=1}^{K}\epsilon_j^\top X_j\right)P_i\left(\sum_{j=1}^{K}X_j^\top\epsilon_j\right)\right] \\
&= \sum_{i=1}^{K} \mathbb{E}\left[\sum_{j=1}^{K}\sum_{k=1}^{K}\epsilon_j^\top X_j P_i X_k^\top\epsilon_k\right] \\
&= \sum_{i=1}^{K}\sum_{j=1}^{K}\sum_{k=1}^{K} \mathbb{E}\left[\epsilon_j^\top X_j P_i X_k^\top\epsilon_k\right] \\
&= \sum_{i=1}^{K}\sum_{j=1}^{K} \mathbb{E}\left[\epsilon_j^\top X_j P_i X_j^\top\epsilon_j\right] \cdots \text{by } \mathbb{E}[\epsilon_j^\top X_j P_i X_k^\top\epsilon_k] = 0 \text{ if } j \neq k \\
&= \sum_{i=1}^{K}\sum_{j=1}^{K} \mathbb{E}\left[\mathrm{tr}(X_j P_i X_j^\top\epsilon_j\epsilon_j^\top)\right] \cdots \text{by } \mathrm{tr}(AB) = \mathrm{tr}(BA) \\
&= \sigma^2\sum_{i=1}^{K}\sum_{j=1}^{K}\mathrm{tr}(X_j P_i X_j^\top) \cdots \text{by } \mathbb{E}[\epsilon_j\epsilon_j^\top] = \sigma^2 I_n \\
&= \sigma^2\sum_{i=1}^{K}\sum_{j=1}^{K}\mathrm{tr}\left(X_j\left(\sum_{k=1}^{K}\psi_k\right)^{-1}\psi_i\left(\sum_{k=1}^{K}\psi_k\right)^{-1}X_j^\top\right) \cdots \text{by plugging } P_i \\
&= \sigma^2\sum_{i=1}^{K}\sum_{j=1}^{K}\mathrm{tr}\left(\psi_j\left(\sum_{k=1}^{K}\psi_k\right)^{-1}\psi_i\left(\sum_{k=1}^{K}\psi_k\right)^{-1}\right) \cdots \text{by } \psi_j = X_j^\top X_j \\
&= \sigma^2\mathrm{tr}\left(\sum_{i=1}^{K}\sum_{j=1}^{K}\psi_j\left(\sum_{k=1}^{K}\psi_k\right)^{-1}\psi_i\left(\sum_{k=1}^{K}\psi_k\right)^{-1}\right) \\
&= \sigma^2\mathrm{tr}\left(\left(\sum_{j=1}^{K}\psi_j\right)\left(\sum_{k=1}^{K}\psi_k\right)^{-1}\left(\sum_{i=1}^{K}\psi_i\right)\left(\sum_{k=1}^{K}\psi_k\right)^{-1}\right) \\
&= \sigma^2\mathrm{tr}(I\cdot I) \\
&= \sigma^2 d.
\end{aligned}
$$

A.2 Proof for Theorem 2

Set $\alpha_i = \hat{\theta}_i$ in Lemma A.1 for $i \in [K]$, we obtain:

$$
\mathcal{R}\left(\hat{\theta}_1, \cdots, \hat{\theta}_K\right) - \mathcal{R}^* = \underbrace{\sum_{i=1}^{K}\left\|\mathbb{E}[\hat{\theta}_i] - \theta_i\right\|_{\psi_i}^2}_{\text{bias part}} + \underbrace{\sum_{i=1}^{K}\mathbb{E}\left[\left\|\hat{\theta}_i - \mathbb{E}[\hat{\theta}_i]\right\|_{\psi_i}^2\right]}_{\text{variance part}}. \tag{A.3}
$$

From Proposition 3.4 and Proposition 3.5 in [4], for each $i \in [K]$:

$$
\left\|\mathbb{E}[\hat{\theta}_i] - \theta_i\right\|_{\psi_i}^2 = 0, \quad \mathbb{E}\left[\left\|\hat{\theta}_i - \mathbb{E}[\hat{\theta}_i]\right\|_{\psi_i}^2\right] = \sigma^2 d. \tag{A.4}
$$

Combining Eq.(A.3) and Eq.(A.4), we obtain the desired result.

B DETAILS OF EXPERIMENTS

B.1 Datasets details

We conduct our experiment on 8 popular datasets following previous researches [26, 29]. The statistics of these datasets are summarized in Table 2, and publicly available at https://github.com/zhouhaoyi/Informer2020 and https://github.com/thuml/Autoformer.

(1) ETTh1/ETTh2/ETTm1/ETTm2. ETT dataset contains 7 indicators collected from electricity transformers from July 2016 to July 2018, including useful load, oil temperature, etc. Data points are recorded hourly for ETTh1 and ETTh2, while recorded every 15 minutes for ETTm1 and ETTm2.

(2) Electricity. Electricity dataset contains the hourly electricity consumption (in KWh) of 321 customers from 2012 to 2014.

(3) Traffic. Traffic dataset contains hourly road occupancy rate data measured by different sensors on San Francisco Bay area freeways in 2 years. The data is from the California Department of Transportation.

(4) Illness. Illness dataset includes 7 weekly recorded indicators of influenza-like illness patients data from Centers for Disease Control and Prevention of the United States between 2002 and 2021.

(5) Weather. Weather dataset contains 21 meteorological indicators, like temperature, humidity, etc. The dataset is recorded every 10 minutes for the 2020 whole year.

B.2 Baseline models

We briefly describe our selected baseline models:

(1) Informer [29] utilizes ProbSparse self-attention and distillation operation to capture cross-time dependency.

(2) Autoformer [26] utilizes series decomposition block architecture with Auto-Correlation to capture cross-time dependency.

(3) FEDformer [30] presents a sparse representation within the frequency domain and proposes frequency-enhanced blocks to capture the cross-time dependency.

(4) ETSformer [24] exploits the principle of exponential smoothing and leverages exponential smoothing attention and frequency attention.

(5) Crossformer [28] utilizes a dimension-segment-wise embedding and introduces a two-stage attention layer to capture the cross-time and cross-dimension dependency.

(6) DLinear [27] proposes a linear forecasting model, along with seasonal-trend decomposition or normalization operations.

(7) PatchTST [16] utilizes Transformer encoders as model backbone, and proposes patching and channel independence techniques for better prediction.

Table 2: The statistics of 8 datasets, including the number of variates, total timesteps, and the frequency of data sampling.

Dataset	ETTh1	ETTh2	ETTm1	ETTm2	Electricity	Traffic	Weather	Illness
Variates	7	7	7	7	321	862	21	7
Timesteps	17,420	17,420	69,680	69,680	26,304	17,544	52,695	966
Frequency	1hour	1hour	15min	15min	1hour	1hour	10min	1week

Table 3: Hyper-parameters for SOLID.

Dataset	λ_T	λ_P	λ_N	lr/lr_{train}
ETTh1	{500, 1000, 2000}	{0.02, 0.05, 0.1}	{5, 10, 20}	{5, 10, 20, 50}
ETTh2				
ETTm1				
ETTm2				
Electricity				{500, 1000, 1500, 2000}
Traffic				{1000, 1500, 2000, 3000}
Weather				{5, 10, 20, 50}
Illness	{100, 200, 300}		{2, 3, 5}	{10, 20, 50, 100}

B.3 Hyper-parameters

For all experiments, during the training process, we use the same hyper-parameters as reported in the corresponding papers [24, 26–30], *e.g.*, encoder/decoder layers, model hidden dimensions, head numbers of multi-head attention and batch size.

As for hyper-parameters for adaptation process, there exists 4 major hyper-parameters for SOLID, including λ_T, λ_P, λ_N, lr (We report the ratio lr/lr_{train} between adaptation learning rate lr and the training learning rate lr_{train} as an alternative). We select the setting which performs the best on the validation set. The search range for the parameters is presented in Table 3.

B.4 Specific structures of g_Φ and h_θ

We divide our baseline TSF models into three categories, including Encode-decoder-based, Encoder-based, and Linear-based architectures for explanation.

- Encode-decoder-based (*e.g.*, Informer, Autoformer, FEDformer, ETSformer, and Crossformer): h_θ refers to the linear projection layer at the top of decoders, and the other parts of decoders and the entire encoders correspond to g_Φ.
- Encoder-based (*e.g.*, PatchTST): h_θ refers to the top linear prediction layer at the top of the encoders, and the other parts (including self-attention and embedding layers) correspond to g_Φ.
- Linear-based (*e.g.*, DLinear): Since Linear models only include linear layers for modeling, h_θ refers to the linear layers, and there is no g_Φ.

C FURTHER EXPERIMENT RESULTS

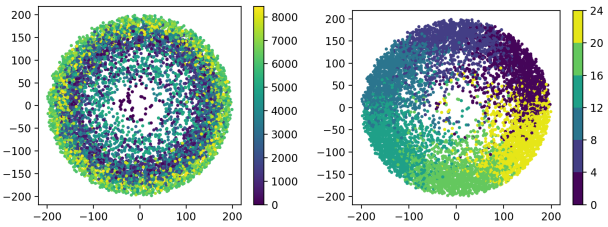
C.1 Full results

In this section, we report the full results of § 7.2 and § 7.3.

- Table 4 details the ablation study's results across five models and three datasets.

Table 4: Extended table of Figures 5(a)-5(b): Ablation study on SOLID.

Dataset		Electricity		Traffic		Illness	
Metric		MSE	MAE	MSE	MAE	MSE	MAE
Informer	/	0.321	0.407	0.731	0.406	5.106	1.534
	T	0.252	0.362	0.636	0.396	4.443	1.427
	T+P	0.248	0.358	**0.628**	**0.393**	2.901	1.186
	T+P+S	**0.245**	**0.355**	**0.628**	**0.393**	**2.874**	**1.150**
Autoformer	/	0.207	0.324	0.620	0.391	3.314	1.245
	T	0.193	0.309	0.579	0.380	3.333	1.251
	T+P	0.196	0.309	0.577	0.380	2.981	1.201
	T+P+S	**0.189**	**0.304**	**0.565**	**0.377**	**2.737**	**1.118**
FEDformer	/	0.188	0.304	0.574	0.356	3.241	1.252
	T	0.173	0.284	0.522	0.346	3.234	1.253
	T+P	**0.172**	**0.283**	0.520	0.345	2.953	1.189
	T+P+S	**0.172**	0.284	**0.513**	**0.344**	**2.707**	**1.123**
ETSformer	/	0.187	0.304	0.599	0.386	2.397	0.993
	T	0.174	0.289	0.493	0.356	2.377	0.989
	T+P	0.173	0.287	0.489	0.356	2.334	0.977
	T+P+S	**0.171**	**0.285**	**0.487**	**0.354**	**2.262**	**0.955**
Crossformer	/	0.184	0.297	0.521	0.297	3.329	1.275
	T	0.188	0.301	0.482	0.273	2.705	1.098
	T+P	0.184	0.297	0.476	0.270	2.469	0.993
	T+P+S	**0.182**	**0.295**	**0.473**	**0.267**	**2.353**	**0.986**



(a) context: temporal segments (b) context: periodic phases

Figure 6: Illustration for two contexts on ETTh1 training dataset, which has a period of 24 and length of 8760. Different colors represent different values of contexts.

- Tables 5 and 6 compare the effectiveness of our approach against RevIN and Dish-TS in addressing distribution shifts.
- Table 7 compares the results of two tuning strategies, *i.e.*, full parameter fine-tuning vs. prediction layer-only fine-tuning, applied to five models and three datasets. The prediction lengths are 24 (for Illness) and 96 (for Traffic and ETTh1).
- Table 8 details the benchmark results for ETTm1, ETTm2, and Weather datasets, where our proposed Reconditioner identified relatively lower metrics.

Table 5: Extended table of Figure 5(d): Comparison experiments between our proposed SOLID and RevIN for tackling distribution shift. The TSF model is Autoformer.

Method		+SOLID		+RevIN		+RevIN +SOLID	
Metric		MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	**0.430**	**0.442**	0.465	0.453	0.462	0.450
ETTh2	96	**0.362**	0.404	0.396	0.401	0.393	**0.398**
ETTm1	96	**0.443**	**0.448**	0.555	0.479	0.553	0.477
ETTm2	96	0.312	0.345	0.232	0.307	**0.230**	**0.305**
Electricity	96	0.189	0.304	0.182	0.289	**0.176**	**0.284**
Traffic	96	0.565	0.376	0.637	0.392	**0.552**	**0.357**
Weather	96	0.259	0.334	0.209	0.254	**0.208**	**0.253**
Illness	24	**2.737**	1.118	2.810	1.119	2.795	**1.107**

Table 6: Extended table of Figure 5(e): Comparison experiments between our proposed SOLID and Dish-TS for tackling distribution shift. The TSF model is Autoformer.

Method		+SOLID		+Dish-TS		+Dish-TS +SOLID	
Metric		MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.430	0.442	0.419	0.430	**0.416**	**0.428**
ETTh2	96	**0.362**	0.404	0.369	0.394	0.366	**0.393**
ETTm1	96	**0.443**	0.448	0.488	0.452	0.486	**0.449**
ETTm2	96	0.312	0.345	0.213	0.295	**0.209**	**0.291**
Electricity	96	0.189	0.304	0.169	0.279	**0.162**	**0.273**
Traffic	96	0.565	0.376	0.586	0.381	**0.518**	**0.345**
Weather	96	0.259	0.334	0.194	0.244	**0.192**	**0.243**
Illness	24	2.737	1.118	2.587	1.079	**2.571**	**1.063**

Table 7: Extended table of Figure 5(c): Performance comparison between adaptation on prediction layer (PL) versus adaptation on entire model (EM).

Dataset		ETTh1		Traffic		Illness	
Metric		MSE	MAE	MSE	MAE	MSE	MAE
Informer	/	0.948	0.774	0.731	0.406	5.096	1.533
	+SOLID-PL	**0.684**	**0.586**	**0.628**	**0.393**	**2.874**	**1.150**
	+SOLID-EM	0.817	0.656	0.692	0.403	2.991	1.208
Autoformer	/	0.440	0.444	0.621	0.391	3.314	1.245
	+SOLID-PL	**0.430**	**0.442**	**0.565**	**0.376**	**2.737**	**1.118**
	+SOLID-EM	0.436	0.443	0.597	0.385	2.891	1.150
FEDformer	/	0.375	0.414	0.574	0.356	3.241	1.252
	+SOLID-PL	**0.370**	**0.410**	**0.513**	**0.344**	**2.707**	**1.123**
	+SOLID-EM	0.373	0.413	0.539	0.348	2.874	1.201
ETSformer	/	0.495	0.480	0.599	0.386	2.397	0.993
	+SOLID-PL	**0.491**	**0.478**	**0.487**	**0.354**	**2.262**	**0.955**
	+SOLID-EM	0.492	0.479	0.517	0.368	2.353	0.986
Crossformer	/	0.411	0.432	0.521	0.297	3.329	1.275
	+SOLID-PL	**0.382**	**0.415**	**0.473**	**0.267**	**2.397**	**1.052**
	+SOLID-EM	0.401	0.427	0.493	0.273	2.653	1.074

C.2 Visualizing the impact of contexts on data distribution

To demonstrate the influence of context on data distribution, we utilized the Autoformer [26] to extract latent representations from the ETTh1 dataset and applied PCA for dimensionality reduction and visualization. This ensures the spatial positions of the data points in the figure represent their original distribution.

We marked two observed contexts – temporal segments and periodic phases – on the figures. Unobserved contexts, being difficult to visualize, are not displayed. Figure 6(a) reveals a progressive outward shift in data distribution with increasing temporal segments. Similarly, Figure 6(b) shows that changes in the periodic phase lead to a rotational shift in data distribution. Note that if CDS doesn't exist, different colors (denoting contexts) should be scattered and randomly mixed in the figure since colors are independent of spatial positions, which is not the case shown in Figure 6. Therefore, it is evident that these contexts markedly affect data distribution.

Table 8: Extended table of Table 1. "↑": average improvements achieved by *SOLID* compared to the original forecasting results. "δ": metrics given by *Reconditionor* in two observed contexts: periodic phases (δ_P) and temporal segments (δ_T), presented in the form of $\log_{10}\delta_P$ & $\log_{10}\delta_T$. RED highlights a *strong* CDS in periodic phases (*i.e.*, $\log_{10}\delta_P \geq -3.2$), while BLUE highlights a *weak* CDS in periodic phases (*i.e.*, $\log_{10}\delta_P < -3.2$).

Dataset		ETTm1				ETTm2				Weather			
Method		/		+SOLID		/		+SOLID		/		+SOLID	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Informer	96	0.624	0.556	**0.426**	**0.436**	0.412	0.498	**0.279**	**0.379**	0.394	0.436	**0.321**	**0.377**
	192	0.727	0.620	**0.535**	**0.503**	0.821	0.710	**0.453**	**0.503**	0.501	0.491	**0.330**	**0.367**
	336	1.085	0.776	**0.696**	**0.601**	1.459	0.926	**0.664**	**0.612**	0.591	0.540	**0.400**	**0.421**
	720	1.200	0.814	**0.775**	**0.631**	3.870	1.461	**1.875**	**0.982**	1.070	0.755	**0.681**	**0.576**
	↑			**33.13%**	**21.52%**			**50.15%**	**31.11%**			**32.19%**	**21.71%**
	δ			**-3.357 & -2.229**				**-3.082 & -1.527**				**-4.495 & -1.402**	
Autoformer	96	0.445	0.449	**0.443**	**0.448**	0.313	0.346	**0.312**	**0.345**	0.260	0.334	**0.259**	**0.334**
	192	0.549	0.500	**0.547**	**0.499**	0.283	0.341	**0.282**	**0.340**	0.322	0.377	**0.321**	**0.376**
	336	0.633	0.533	**0.631**	**0.532**	0.327	0.367	**0.326**	**0.366**	0.367	0.398	**0.367**	**0.397**
	720	0.672	0.559	**0.670**	**0.558**	0.445	0.434	**0.444**	**0.433**	0.414	0.421	**0.414**	**0.421**
	↑			**0.30%**	**0.19%**			**0.24%**	**0.20%**			**0.06%**	**0.06%**
	δ			**-3.851 & -2.851**				**-3.524 & -1.264**				**-4.236 & -1.714**	
FEDformer	96	0.362	0.412	**0.360**	**0.410**	0.190	0.283	**0.189**	**0.283**	0.225	0.307	**0.225**	**0.307**
	192	0.399	0.428	**0.398**	**0.427**	0.256	0.324	**0.255**	**0.324**	0.318	0.374	**0.317**	**0.374**
	336	0.441	0.456	**0.440**	**0.455**	0.327	0.365	**0.326**	**0.364**	0.347	0.385	**0.346**	**0.385**
	720	0.486	0.477	**0.485**	**0.477**	0.434	0.425	**0.434**	**0.424**	0.408	0.424	**0.407**	**0.422**
	↑			**0.31%**	**0.25%**			**0.14%**	**0.12%**			**0.22%**	**0.21%**
	δ			**-3.979 & -2.896**				**-3.914 & -1.133**				**-4.159 & -1.569**	
ETSformer	96	0.371	0.394	**0.369**	**0.392**	0.187	0.280	**0.186**	**0.279**	0.216	0.298	**0.213**	**0.295**
	192	0.402	0.405	**0.402**	**0.405**	0.251	0.319	**0.251**	**0.318**	0.253	0.329	**0.251**	**0.326**
	336	0.429	0.423	**0.429**	**0.423**	0.313	0.356	**0.312**	**0.355**	0.289	0.352	**0.286**	**0.348**
	720	0.429	0.455	**0.428**	**0.455**	0.414	0.413	**0.410**	**0.410**	0.355	0.401	**0.353**	**0.399**
	↑			**0.14%**	**0.12%**			**0.56%**	**0.44%**			**0.93%**	**0.81%**
	δ			**-3.996 & -2.900**				**-3.541 & -1.188**				**-4.614 & -1.793**	
Crossformer	96	0.312	0.367	**0.311**	**0.367**	0.770	0.599	**0.689**	**0.580**	0.151	0.219	**0.151**	**0.219**
	192	0.350	0.391	**0.348**	**0.390**	0.567	0.516	**0.450**	**0.512**	0.196	0.264	**0.196**	**0.264**
	336	0.407	0.427	**0.403**	**0.425**	0.830	0.637	**0.616**	**0.613**	0.246	0.307	**0.246**	**0.306**
	720	0.648	0.580	**0.538**	**0.515**	1.754	1.010	**1.037**	**0.786**	0.311	0.357	**0.311**	**0.357**
	↑			**6.81%**	**3.77%**			**28.75%**	**9.81%**			**0.11%**	**0.10%**
	δ			**-3.369 & -2.148**				**-2.979 & -1.767**				**-4.521 & -1.423**	
DLinear	96	0.310	0.353	**0.306**	**0.349**	0.169	0.262	**0.168**	**0.261**	0.176	0.238	**0.175**	**0.232**
	192	0.340	0.369	**0.339**	**0.368**	0.232	0.308	**0.230**	**0.306**	0.218	0.276	**0.216**	**0.272**
	336	0.374	0.390	**0.373**	**0.389**	0.299	0.360	**0.296**	**0.356**	0.262	0.312	**0.261**	**0.308**
	720	0.440	0.435	**0.437**	**0.434**	0.439	0.451	**0.434**	**0.447**	0.326	0.365	**0.324**	**0.359**
	↑			**0.61%**	**0.45%**			**0.97%**	**0.80%**			**0.61%**	**1.68%**
	δ			**-3.057 & -2.786**				**-3.521 & -1.697**				**-4.386 & -1.793**	
PatchTST	96	0.292	0.345	**0.290**	**0.344**	0.166	0.257	**0.165**	**0.255**	0.152	0.200	**0.151**	**0.199**
	192	0.333	0.370	**0.332**	**0.370**	0.220	0.293	**0.219**	**0.292**	0.197	0.243	**0.196**	**0.242**
	336	0.366	0.390	**0.365**	**0.389**	0.275	0.329	**0.274**	**0.328**	0.250	0.285	**0.250**	**0.285**
	720	0.420	0.424	**0.419**	**0.423**	0.366	0.385	**0.365**	**0.384**	0.316	0.334	**0.316**	**0.334**
	↑			**0.35%**	**0.20%**			**0.39%**	**0.40%**			**0.22%**	**0.19%**
	δ			**-3.356 & -2.879**				**-3.471 & -1.454**				**-3.986 & -1.573**	

C.3 Correlation of δ_T and MAE improvement

In § 7.2.3, we found that the correlation between δ_P and MAE improvement is very strong. We display the correlation between δ_T and MAE improvement in Figure 7 as well. Unlike δ_P, the relation between δ_T and MAE improvement is not as straightforward.

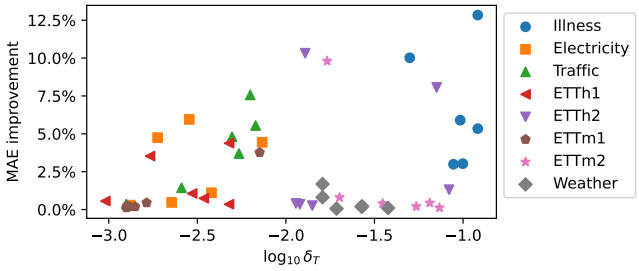


Figure 7: The relationship $\log_{10} \delta_T$ (X-axis) and MAE improvements achieved by SOLID (Y-axis) for 8 datasets and 6 models.

Despite showing an increasing trend in Figure 7, the Spearman correlation coefficient is merely 0.2129. This implies that, while there is evidence of CDS stemming from temporal segments as detected by Reconditionor, SOLID is comparatively less effective at mitigating it when compared to CDS caused by periodic phases. One possible explanation is that data generated within the same phase tends to follow a more predictable pattern, while data within the same temporal segment exhibits greater diversity and uncertainty, which may limit the utility of selecting data from the same segment to address CDS caused by temporal segments. We leave further investigation of it to future work.

C.4 Parameter sensitivity analysis

Within our proposed approach, several crucial parameters are presented, including λ_T, which controls the time range of preceding data for selection; λ_P, which governs the acceptable threshold for periodic phase difference; λ_N, which determines the number of similar samples to be selected for model adaptation; And lr, which regulates the extent of adaptation on the prediction layer for the models. The search range for these parameters is presented in Table 3. The results of parameter sensitivity analysis are visually presented in Figure 8.

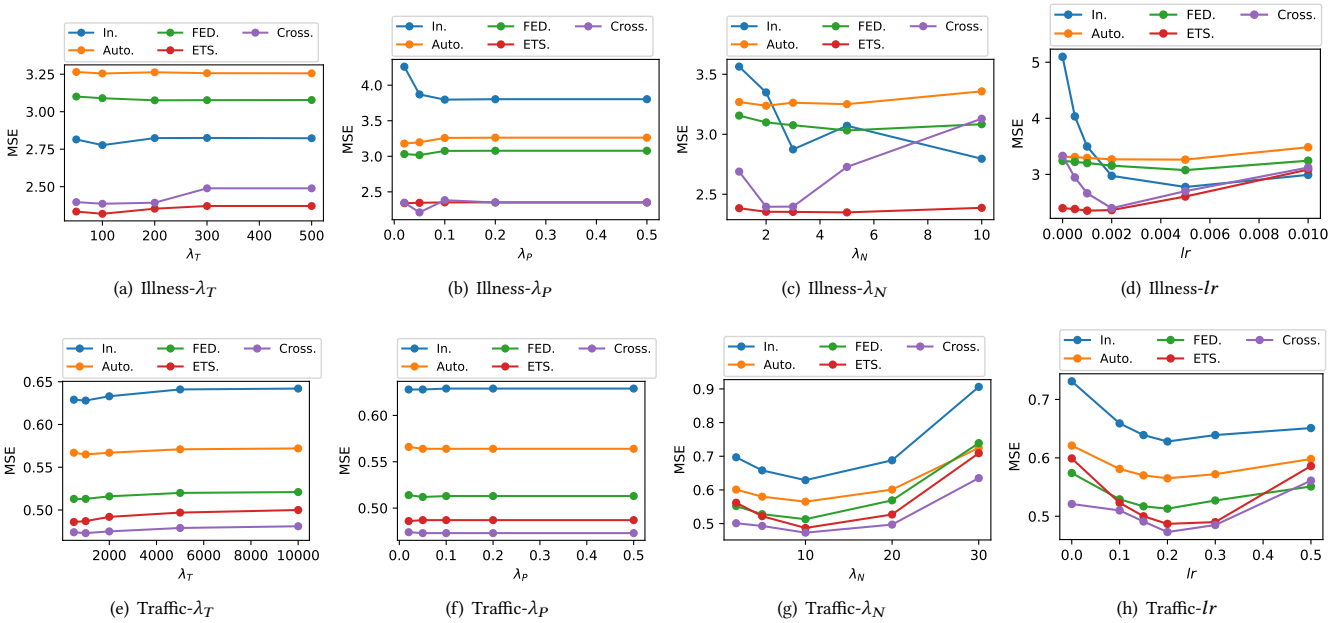
Firstly, we discover that our proposed SOLID is insensitive to λ_T and λ_P parameters, based on the results obtained. Regarding the λ_N parameter, the selection of insufficient samples would increase the variance during adaptation due to the data shortage. Conversely, selecting excessive samples carries the risk of including samples with unobserved irrelevant contexts, thereby deteriorating the model's performance. Lastly, For parameter lr, a very small lr leads to inadequate model adaptation, preventing the model from effectively addressing CDS and resulting in a bias towards the test sample (Theorem 1). Conversely, a too large value for lr can lead to excessive adaptation, which also risks bringing in substantial variance to the model (Theorem 2). Therefore, a well-selected learning rate will contribute to an optimal trade-off between bias and variance.

C.5 Case study

In addition, we conduct a case study and visualize our proposed method on various cases across different datasets and models, as presented in Figure 9.

Specifically, we plot the figures on these combinations: Illness (timestep 120 & 160, variate 7) on Crossformer, Traffic (timestep 200 & 550, variate 862) on FEDformer, Electricity (timestep 1500 & 3000, variate 321) on Autoformer, and ETTh1 (timestep 250 & 650, variate 7) on Informer, corresponding to Figure 9 (a)-(h). The visualization vividly illustrates the effectiveness of our approach in improving forecasting performance.

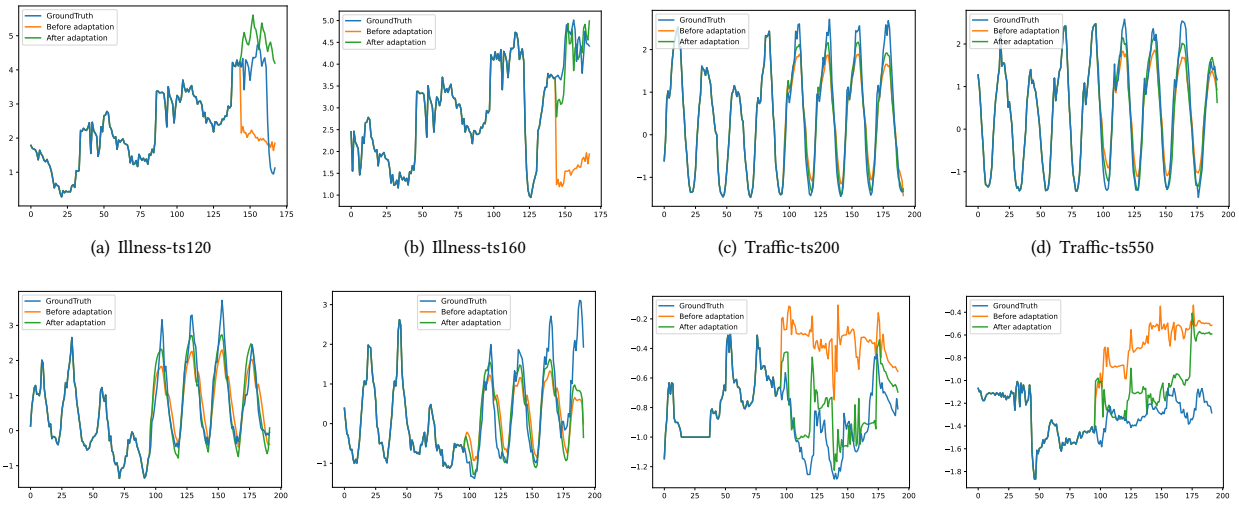
Figure 8: Parameter sensitivity results for λ_T, λ_P, λ_N and lr, on Illness and Traffic datasets.



(a) Illness-λ_T (b) Illness-λ_P (c) Illness-λ_N (d) Illness-lr

(e) Traffic-λ_T (f) Traffic-λ_P (g) Traffic-λ_N (h) Traffic-lr

Figure 9: Case study and visualization of our proposed methods on various cases across different datasets and models, where BLUE lines represent ground-truth, ORANGE lines represent original forecasting results, and GREEN lines represent the forecasts after employing our approach. Besides, $ts120$ denotes this case is sampled at timestep-120 in the test set, and so on.



(a) Illness-ts120 (b) Illness-ts160 (c) Traffic-ts200 (d) Traffic-ts550

(e) Electricity-ts1500 (f) Electricity-ts3000 (g) ETTh1-ts250 (h) ETTh1-ts650