Time-o1: Time-Series Forecasting Needs Transformed Label Alignment

Hao Wang[1*] **Licheng Pan**[1*] **Zhichao Chen**[2] **Xu Chen**[3†]
Qingyang Dai[4] **Lei Wang**[3] **Haoxuan Li**[5†] **Zhouchen Lin**[2,6,7†]
[1]Xiaohongshu Inc.
[2]State Key Lab of General AI, School of Intelligence Science and Technology, Peking University
[3]Gaoling School of Artificial Intelligence, Renmin University of China
[4]Department of Control Science and Engineering, Zhejiang University
[5]Center for Data Science, Peking University
[6]Institute for Artificial Intelligence, Peking University
[7]Pazhou Laboratory (Huangpu), Guangzhou, Guangdong, China

Abstract

Training time-series forecast models presents unique challenges in designing effective learning objectives. Existing methods predominantly utilize the temporal mean squared error, which faces two critical challenges: (1) label autocorrelation, which leads to bias from the label sequence likelihood; (2) excessive amount of tasks, which increases with the forecast horizon and complicates optimization. To address these challenges, we propose `Time-o1`, a transformation-augmented learning objective for training time-series forecasting models. The central idea is to transform the label sequence into decorrelated components with discriminated significance. Models are then trained to align the most significant components, thereby effectively mitigating label autocorrelation and reducing task amount. Extensive experiments demonstrate that Time-o1 achieves state-of-the-art performance and is compatible with various forecast models. Code is available at `https://github.com/Master-PLC/Time-o1`.

1 Introduction

Time-series forecasting involves predicting future data from historical observations [60, 23] and has been applied across diverse domains, such as air quality prediction in meteorology [28], user behavior analysis in e-commerce [3], and process monitoring in manufacturing [47, 50]. To build effective forecast models, there are two questions that warrant investigation: *(1) How to design a neural network architecture to encode historical observations, and (2) How to devise a learning objective to train the neural network.* Both are critical for model performance.

Recent research has primarily focused on developing neural network architectures [54, 59]. The key challenge lies in exploiting the autocorrelation in the historical sequences. To this end, various architectures have been proposed [24, 40, 51, 30], such as recurrent neural networks [10, 9, 53], convolutional neural networks [55, 52, 26], and graph neural networks [62, 2, 12]. The current progress is marked by a debate between Transformers and simple linear models. Transformers, equipped with self-attention mechanisms, offer superior scalability [27, 34, 29]. In contrast, linear models, which encapsulate temporal dynamics using linear layers, are straightforward to implement and often demonstrate strong performance [66, 42, 64, 61]. These advancements showcase the rapid evolution in neural architecture design for time-series forecasting.

*This work was done in the internship at Xiaohongshu Inc. Both authors have equal contribution.
†Corresponding author.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

In contrast, the design of learning objectives has received less attention [49, 39, 17, 22]. Most existing methods employ the temporal mean squared error (TMSE) as the learning objective, which measures the step-wise discrepancy between forecast and label sequences [27, 34]. While being effective for various scenarios, it has two critical limitations. First, it is biased against the true likelihood of label sequence due to the presence of autocorrelation in the label sequence. Second, the number of prediction tasks increases with the forecast horizon, which complicates the optimization process since multitask learning is known to be challenging given excessive tasks [68, 25]. These limitations highlight the importance of developing more principled learning objectives for time-series forecasting.

To handle these challenges, we propose a transformation-augmented learning objective tailored for time-series forecasting. The key idea is to transform the label sequence into decorrelated components ranked by significance. By aligning the most significant decorrelated components, Time-o1 mitigates label autocorrelation and reduces the number of tasks.

Our main contributions are summarized as follows:

- We formulate two critical challenges in designing objectives for time-series forecasting: label autocorrelation that induces bias, and the excessive number of tasks that impedes optimization.
- We propose Time-o1, which transforms labels into decorrelated components with discriminated significance. By aligning the significant components, it addresses the two challenges above.
- We conduct comprehensive experiments to demonstrate Time-o1's efficacy, consistently boosting the performance of state-of-the-art forecast models across diverse datasets.

2 Preliminaries

This paper focuses on the time-series forecasting problem [38]. By the way of preface, uppercase letters (e.g., Y) denote random variables, and bolded letters (e.g., $\mathbf{Y}$) denote matrices containing data or parameters. One key distinction warrants emphasis: we are concentrating on the design of learning objectives for training forecast models [49, 39], rather than on the design of neural network architectures to implement the forecast models [27, 66, 58].

Suppose X is a time-series dataset with D covariates, where X_n denotes the observation at the n-th step. At an arbitrary n-th step, the historical sequence is defined as $L = [X_{n-\mathrm{H}+1}, \ldots, X_n] \in \mathbb{R}^{\mathrm{H}\times\mathrm{D}}$, the label sequence is defined as $Y = [X_{n+1}, \ldots, X_{n+\mathrm{T}}] \in \mathbb{R}^{\mathrm{T}\times\mathrm{D}}$, where H is the historical length and T is the forecast horizon. The target of time-series forecasting is to train a model $g : \mathbb{R}^{\mathrm{H}\times\mathrm{D}} \rightarrow \mathbb{R}^{\mathrm{T}\times\mathrm{D}}$ that generates accurate prediction sequence $\hat{Y}$ approximating the label sequence.

There are two aspects to building forecast models: (1) neural network architectures that effectively encode historical sequences, and (2) learning objectives for training these neural networks. While this paper focuses on the learning objective, we provide a brief review of both aspects for contextualization.

2.1 Model architectures for time-series forecasting

Neural networks are widely employed for encoding historical sequences [31, 57] due to their ability to automatically model feature interactions and capture complex nonlinear autocorrelation effects [30, 13, 11]. Notable examples include recurrent neural networks (e.g., S4 [10], Mamba [9]), convolutional neural networks (e.g., SCINet [26], TimesNet [55], MICN [52]), and graph neural networks (e.g., MTGNN [33], StemGNN [2]), each tailored to encode the dynamics within input sequences. The current progress centers on the comparison between Transformer-based architectures and linear architectures. Transformers (e.g., PatchTST [34], iTransformer [27], FreeFormer [65]) exhibit substantial scalability with increasing data size but entail high computational costs. In contrast, linear architectures (e.g., DLinear [66], RLinear [42], OLinear [64], TimeBase [14]) are generally more efficient but less scalable with larger datasets and struggle to handle varying input lengths.

2.2 Learning objectives for time-series forecasting

Modern time-series models predominantly adopt the direct forecast paradigm, generating T-step forecasts simultaneously using a multi-output head [21, 27, 66]. The learning objective is typically the temporal mean squared error (TMSE) between the forecast and label sequences, given by:

$$\mathcal{L}_{\mathrm{tmse}} = \sum_{t=1}^{\mathrm{T}} \left(Y_t - \hat{Y}_t\right)^2, \tag{1}$$

which is widely employed in recent studies (e.g., FreTS [63], iTransformer [27], FredFormer [36], DUET [40]). However, this objective has been shown to be biased due to the autocorrelation present in the label sequence [49]. To address this bias, one line of research advocates for shape alignment between the forecast and label sequences to exploit autocorrelation (e.g., Dilate [19] and Soft-DTW [4]). However, these methods lack rigorous theoretical guarantees for unbiased objective and empirical evidence of improved performance. Another notable approach involves computing the forecast error in the frequency domain [49, 22], which reduces bias with theoretical guarantees [49].

3 Methodology

3.1 Motivation

The learning objective is a fundamental component in training effective forecast models, yet its importance remains underexplored. Existing approaches predominantly employ the TMSE in (1) as the objective [27, 34, 64]. This practice, however, encounters two fundamental limitations rooted in the characteristics of time-series forecasting task.

First, TMSE introduces bias due to autocorrelation. In time-series forecasting, observations exhibit strong dependencies on their past values [66], resulting in step-wise correlation in the label sequence. In contrast, TMSE treats the forecast of each step as an independent task, thereby neglecting these correlations. This mismatch makes TMSE biased with respect to the true likelihood of the label sequence, as presented in Theorem 3.1.

Theorem 3.1 (Autocorrelation bias)**.** *Given label sequence Y where $\Sigma \in \mathbb{R}^{\mathrm{T}\times\mathrm{T}}$ denotes the step-wise correlation coefficient, the TMSE in* (1) *is biased compared to the negative log-likelihood of the label sequence, which is given by:*

$$\mathrm{Bias} = \left\|Y - \hat{Y}\right\|_{\Sigma^{-1}}^2 - \left\|Y - \hat{Y}\right\|^2 - \frac{1}{2}\log|\Sigma|. \tag{2}$$

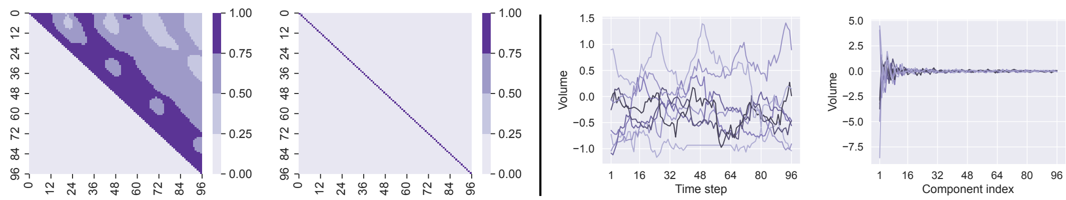
where $\|v\|_{\Sigma^{-1}}^2 = v^\top \Sigma^{-1} v$. The bias vanishes if different steps in Y are decorrelated.[3]

Second, TMSE poses optimization difficulties as the forecast horizon grows. The large forecast horizon is crucial for applications such as manufacturing (enabling comprehensive production planning [48, 46]) and transportation (enabling proactive traffic management [67, 32]). As TMSE treats each forecasted step as an independent task, a large horizon results in excessive tasks. However, optimization is known to be difficult given excessive tasks [68, 25], as gradients from different tasks often conflict [68, 25], impeding convergence and leading to suboptimal model performance.

Designing effective learning objectives to handle the two limitations is challenging. The previous work **FreDF** [49] proposes a frequency loss, which transforms the label and forecast sequences into frequency components and aligns them in the frequency domain. This approach is motivated by Theorem 3.1: bias vanishes if different components are decorrelated. However, the decorrelation of frequency components holds only when the forecast horizon $\mathrm{T} \to \infty$ (see Theorem 3.3 in [49]). In real-world settings with finite horizon, frequency components remain correlated, rendering FreDF ineffective in eliminating bias. Additionally, the optimization difficulty remains, since transforming to the frequency domain retains the label length. *Consequently, FreDF does not fully address the autocorrelation bias and the optimization difficulty.*

Given the critical role of objective in training forecast models and the limitations of existing methods, it is compelling to develop an innovative objective to address the limitations and advance forecast performance. Importantly, there are two questions that warrant investigation. *How to devise an objective that eliminates autocorrelation and reduces task amount? Does it improve forecast performance?*

[3]The pioneering work [49] identifies the bias under the first-order Markov assumption on the label sequence. This study generalizes this bias without the first-order Markov assumption.



(a) Correlation in the label sequence and components. (b) 8 label sequences and their components.

Figure 1: Comparison of label sequence and associated components. (a) shows the correlation volume within the label sequence (left panel) and components (right panel). (b) visualizes 8 label sequences randomly from ETTh1 (left panel) and the associated components (right panel).

3.2 Transforming label sequence with optimized projection matrix

In this section, we present a method for transforming label sequences into latent components to eliminate autocorrelation and distinguish significant components. Suppose $\mathbf{Y} \in \mathbb{R}^{\mathrm{m}\times\mathrm{T}}$ contains normalized label sequences for m samples, $\mathbf{P} = [\mathbf{P}_1, \mathbf{P}_2, ..., \mathbf{P}_\mathrm{T}]$ is the projection matrix, and the components are produced as $\mathbf{Z} = \mathbf{Y}\mathbf{P}$. The target is for $\mathbf{Z}$ to be decorrelated and ranked by significance. For example, FreDF specifies $\mathbf{P}$ as a Fourier matrix, which does not adapt to specific data properties and thus fails to decorrelate the components and distinguish the significant components[4].

A natural approach to obtaining the projection matrix $\mathbf{P}$ is solving optimization problem with constraints to ensure the desired properties. To find the p-th component, the projection vector can be calculated by solving the following problem:

$$\begin{aligned} \mathbf{P}_p^* = \underset{\mathbf{P}_p}{\text{argmax}} \qquad & (\mathbf{Y}\mathbf{P}_p)^\top (\mathbf{Y}\mathbf{P}_p) \\ \text{subject to} \qquad & \begin{cases} \|\mathbf{P}_p\|^2 = 1 \\ \mathbf{P}_p^\top \mathbf{P}_j = 0, \ \forall\, j < p \quad \text{if } p > 1 \end{cases} \end{aligned} \tag{3}$$

where $\mathbf{Z}_p = \mathbf{Y}\mathbf{P}_p$ is the p-th component, the normalization constraint $\|\mathbf{P}_p\|^2 = 1$ is imposed to avoid trivial solution: $\mathbf{P}_p \to \infty$. The optimization target is to maximize the variance of $\mathbf{Z}_p$, *which is equivalent to maximizing its significance, as components with larger variance contain richer information*. For $p > 1$, the projection axis is required to be orthogonal to the previous axes to avoid redundancy. By solving the optimizations above from $p = 1$ to T sequentially, we obtain the projection matrix $\mathbf{P}^* = [\mathbf{P}_1^*, ..., \mathbf{P}_\mathrm{T}^*]$. The components are then produced as $\mathbf{Z} = \mathbf{Y}\mathbf{P}^*$.

Lemma 3.2 (Decorrelated components)**.** *Suppose $\mathbf{Y} \in \mathbb{R}^{\mathrm{m}\times\mathrm{T}}$ contains normalized label sequences for m samples, $\mathbf{Z} = [\mathbf{Z}_1, ..., \mathbf{Z}_\mathrm{T}]$ are the obtained components; for any $p \neq p'$, we have $\mathbf{Z}_p^\top \mathbf{Z}_{p'} = 0$.*

Lemma 3.3. *The projection matrix $\mathbf{P}^*$ can be obtained via singular value decomposition (SVD): $\mathbf{Y} = \mathbf{U}\mathbf{\Lambda}(\mathbf{P}^*)^\top$, where $\mathbf{U} \in \mathbb{R}^{\mathrm{m}\times\mathrm{m}}$ and $\mathbf{P}^* \in \mathbb{R}^{\mathrm{T}\times\mathrm{T}}$ consist of singular vectors, and the diagonal of $\mathbf{\Lambda} \in \mathbb{R}^{\mathrm{m}\times\mathrm{T}}$ consists of singular values.*

Theoretical Justification. According to Theorem 3.1, the bias vanishes as the correlations between labels to be aligned are eliminated. The obtained components are decorrelated (Lemma 3.2), thereby mitigating autocorrelation-induced bias. Moreover, component significance decreases from $\mathbf{Z}_1$ to $\mathbf{Z}_\mathrm{T}$ as they are derived by maximizing significance under sequentially augmented constraints. Furthermore, $\mathbf{P}^*$ can be computed via SVD (Lemma 3.3), offering an efficient alternative to sequentially solving the constrained optimization problems in (3).

Case study. To showcase the implications of the obtained components, a case study was conducted on the ETTh1 dataset. Implementation details are provided in Appendix A. The results are illustrated in Fig. 1, with key observations summarized as follows:

- **Decorrelation effect:** Fig. 1 (a) compares the correlation volume in the label sequence and the generated components. In the left panel, the value at row i and column j represents the correlation

[4] In the subsequent paragraphs, we use the univariate case with $\mathrm{D} = 1$ for clarity. In the multivariate case, different variates can be treated separately to produce decorrelated components.

between the i-th and j-th steps. A large number of non-diagonal elements exhibit substantial values, with approximately 50.5% exceeding 0.25, indicating notable autocorrelation in the label sequence. In contrast, the right panel shows negligible values for the non-diagonal elements. This demonstrates that transforming the label sequence into components effectively eliminates correlation, thereby corroborating Theorem 3.2.

- **Significance discrimination:** Fig. 1 (b) compares the variance of the label sequence and associated components. In the left panel, the variance of different steps in the label sequence is relatively uniform, ranging from -1.5 to 1.5, suggesting that all steps are equally significant. In the right panel, however, only a few components exhibit large variance, while the others fluctuate within a narrow range. This indicates that the significance of different components can be clearly discerned, allowing for a trade-off between a slight loss of information and reduced optimization complexity by focusing on the most significant components.

The transformation is highly inspired by principal component analysis (PCA) [35]. However, one key distinction warrants emphasis. Existing works dominantly employ principal component analysis on *input features* for obtaining informative representations [8, 6], in contrast, we adapt it to *label sequence*, specifically aiming to reduce autocorrelation bias and simplify optimization for time-series forecasting. To our knowledge, this remains a technically innovative strategy.

3.3 Model implementation

In this section, we present the implementation details of Time-o1. The approach centers on extracting the latent components from the label sequence, then optimizing the forecast model using the most significant components.

Algorithm 1 The workflow of Time-o1.

Input: $\hat{\mathbf{Y}}$: forecast sequences, $\mathbf{Y}$: label sequences.
Parameter: α: the relative weight of the transformed loss, γ: the ratio of involved significant components.
Output: $\mathcal{L}_{\alpha,\gamma}$: the obtained learning objective.

1: $\mathbf{Y} \leftarrow \mathrm{Standardize}(\mathbf{Y})$.
2: $\mathbf{P}^* \leftarrow \mathrm{SVD}(\mathbf{Y})$
3: $\mathbf{Z} \leftarrow \mathbf{Y}\mathbf{P}^*, \hat{\mathbf{Z}} \leftarrow \hat{\mathbf{Y}}\mathbf{P}^*$
4: $\mathrm{K} \leftarrow \mathrm{round}(\gamma \cdot \mathrm{T})$
5: $\mathcal{L}_{\mathrm{trans},\gamma} \leftarrow \|\hat{\mathbf{Z}}_{\cdot,1:\mathrm{K}} - \mathbf{Z}_{\cdot,1:\mathrm{K}}\|_1$
6: $\mathcal{L}_{\mathrm{tmp}} \leftarrow \|\hat{\mathbf{Y}} - \mathbf{Y}\|_2^2$
7: $\mathcal{L}_{\alpha,\gamma} := \alpha \cdot \mathcal{L}_{\mathrm{trans},\gamma} + (1-\alpha) \cdot \mathcal{L}_{\mathrm{tmse}}$.

Given an input historical sequence, the forecast model predicts a sequence $\hat{\mathbf{Y}}$. In line with prevailing preprocessing practices [27, 66, 36], label sequences are first standardized (step 1), which facilitates the decorrelation prerequisite specified in Lemma 3.2. Next, following Lemma 3.3, we compute the optimal projection by applying SVD to the label sequence. The matrix $\mathbf{P}^*$, composed of the right singular vectors, provides the required projections described in (3). Both forecasted and label sequences are then projected into the latent component space (step 3), where the first column carries the largest significance, successively diminishing across columns.

Suppose K is the number of retained components, the training objective using them is given by:

$$\mathcal{L}_{\mathrm{trans},\gamma} := \left\|\hat{\mathbf{Z}}_{\cdot,1:\mathrm{K}} - \mathbf{Z}_{\cdot,1:\mathrm{K}}\right\|_1, \tag{4}$$

where $\mathrm{K} = \mathrm{round}(\gamma \cdot \mathrm{T})$, with γ controlling the involution ratio, the ℓ_1 norm $\|\cdot\|_1$ computes the sum of element-wise absolute differences. Typically, we use the ℓ_1 norm instead of the squared norm following [49], considering that latent components typically vary greatly in scale (Fig. 1), which makes the squared norm unstable in practice. The ℓ_1 norm yields more stable and robust optimization.

Finally, the two objectives ($\mathcal{L}_{\mathrm{tmse}}$ and $\mathcal{L}_{\mathrm{trans},\gamma}$) are fused following [49], with $0 \leq \alpha \leq 1$ controlling the relative contribution:

$$\mathcal{L}_{\alpha,\gamma} := \alpha \cdot \mathcal{L}_{\mathrm{trans},\gamma} + (1-\alpha) \cdot \mathcal{L}_{\mathrm{tmse}}. \tag{5}$$

By projecting both forecasts and labels into decorrelated components, Time-o1 effectively reduces autocorrelation bias. By focusing exclusively on the most significant components, Time-o1 reduces optimization difficulty with minimal information loss. Time-o1 is model-agnostic, offering practitioners the flexibility to employ the most suitable forecast model for each specific scenario.

4 Experiments

To demonstrate the efficacy of Time-o1, there are six aspects empirically investigated:

Table 1: Long-term forecasting performance.

Models	**Time-o1 (Ours)**		Fredformer (2024)		iTransformer (2024)		FreTS (2023)		TimesNet (2023)		MICN (2023)		TiDE (2023)		DLinear (2023)		FEDformer (2022)		Autoformer (2021)		Transformer (2017)	
Metrics	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm1	**0.380**	**0.393**	0.387	0.398	0.411	0.414	0.414	0.421	0.438	0.430	0.396	0.421	0.413	0.407	0.403	0.407	0.442	0.457	0.526	0.491	0.799	0.648
ETTm2	**0.272**	**0.317**	0.280	0.324	0.295	0.336	0.316	0.365	0.302	0.334	0.308	0.364	0.286	0.328	0.342	0.392	0.308	0.354	0.315	0.358	1.662	0.917
ETTh1	**0.431**	**0.429**	0.447	0.434	0.452	0.448	0.489	0.474	0.472	0.463	0.533	0.519	0.448	0.435	0.456	0.453	0.447	0.470	0.477	0.483	0.983	0.774
ETTh2	**0.359**	**0.388**	0.377	0.402	0.386	0.407	0.524	0.496	0.409	0.420	0.620	0.546	0.378	0.401	0.529	0.499	0.452	0.461	0.448	0.460	2.688	1.291
ECL	**0.170**	**0.260**	0.191	0.284	0.179	0.270	0.199	0.288	0.212	0.306	0.192	0.302	0.215	0.292	0.212	0.301	0.214	0.328	0.249	0.354	0.265	0.358
Traffic	**0.419**	**0.280**	0.486	0.336	0.426	0.285	0.538	0.330	0.631	0.338	0.529	0.312	0.624	0.373	0.625	0.384	0.640	0.398	0.662	0.416	0.692	0.379
Weather	**0.241**	**0.280**	0.261	0.282	0.269	0.289	0.249	0.293	0.271	0.295	0.264	0.321	0.272	0.291	0.265	0.317	0.326	0.372	0.319	0.365	0.699	0.601
PEMS03	**0.097**	**0.208**	0.146	0.260	0.122	0.233	0.149	0.261	0.126	0.230	0.106	0.223	0.316	0.370	0.216	0.322	0.152	0.275	0.411	0.475	0.122	0.226
PEMS08	**0.141**	**0.237**	0.171	0.271	0.149	0.247	0.174	0.275	0.152	0.243	0.153	0.258	0.318	0.378	0.249	0.332	0.226	0.312	0.422	0.456	0.240	0.261

Note: We fix the input length as 96 following [27]. **Bold** and underlined denote best and second-best results, respectively. *Avg* indicates average results over forecast horizons: T=96, 192, 336 and 720. Time-o1 employs the top-performing baseline on each dataset as its underlying forecast model.

1. **Performance:** *Does Time-o1 work?* We compare Time-o1 with state-of-the-art baselines using public datasets on long-term forecasting in Section 4.2 and short-term forecasting tasks in Appendix E.1. Moreover, we compare Time-o1 with other learning objectives in Section 4.3.
2. **Gain:** *How does it work?* Section 4.4 offers an ablative study to dissect the contributions of the individual factors of Time-o1, elucidating their roles in enhancing forecast accuracy.
3. **Generality:** *Does it support other forecast models?* Section 4.5 verifies the adaptability of Time-o1 across different forecast models, with additional results in Appendix E.4.
4. **Flexibility:** *Does it support alternative transformations?* Section 4.5 also investigates generating latent components with other transformations to showcase flexibility of implementation.
5. **Sensitivity:** *Does it require careful fine-tuning?* Section 4.6 presents a sensitivity analysis of the hyperparameter α, where Time-o1 maintains efficacy across a broad range of parameter values.
6. **Efficiency:** *Is Time-o1 computationally expensive?* Section D investigates the running cost of Time-o1 in diverse settings.

4.1 Setup

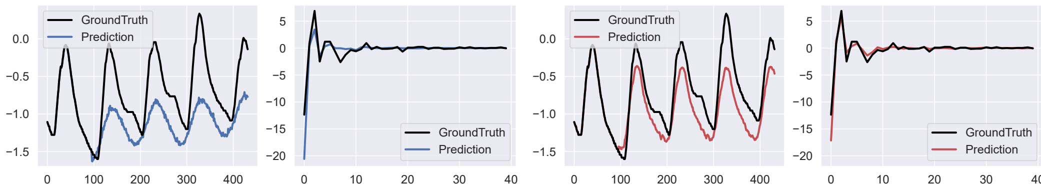
Datasets. In this work, we conduct experiments on ETT (4 subsets), ECL, Traffic, Weather, and PEMS [27] for long-term forecasting task, and M4 for short-term forecasting task [55]. All datasets are split chronologically into training, validation, and testing sets following our prior work [49].

We compare Time-o1 with a range of established models, including Transformer [43], Autoformer [56], FEDformer [69], iTransformer [27], Fredformer [36], DLinear [66], TiDE [5], FreTS [63], TimesNet [55], and MICN [52]. As a learning objective, Time-o1 is model-agnostic and can be integrated with any model architecture. For fair comparison, Time-o1 employs the best-performing baseline model on each dataset as its underlying model architecture.

Implementation. The baseline models are reproduced using the scripts provided by Fredformer [36]. [37]. They are trained using the Adam [15] optimizer to minimize the TMSE loss. Datasets are split chronologically into training, validation, and test sets. Following the prestigious benchmark [37], the dropping-last trick is disabled during the test phase. When integrating Time-o1 to enhance an established model, we adhere to the associated hyperparameter settings in the public benchmark [36, 27], only tuning α, γ and learning rate conservatively. Experiments are conducted on Intel(R) Xeon(R) Platinum 8383C CPUs and NVIDIA RTX H100 GPUs.

4.2 Overall performance

Table 1 presents the long-term forecasting results. Time-o1 consistently improves base model performance. For example, on ETTh1, it reduces Fredformer's MSE by 0.016. Similar gains



(a) The generated forecast with DF. (b) The generated forecast with Time-o1.

Figure 2: The visualization of forecast sequence generated by DF and Time-o1. The left panels in (a) and (b) present label and forecast sequences, the right panels present the associated components.

Table 2: Comparable results with other objectives for time-series forecast.

Loss		**Time-o1**		FreDF		Koopman		Dilate		Soft-DTW		DPTA		DF	
Metrics		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Fredformer	ETTm1	**0.379**	**0.393**	0.384	0.394	0.389	0.400	0.389	0.400	0.397	0.402	0.396	0.402	0.387	0.398
	ETTh1	**0.431**	**0.429**	0.438	0.434	0.452	0.443	0.453	0.442	0.460	0.449	0.460	0.449	0.447	0.434
	ECL	**0.178**	**0.270**	0.179	0.272	0.190	0.282	0.187	0.280	0.206	0.298	0.202	0.294	0.191	0.284
	Weather	**0.255**	**0.276**	0.256	0.277	0.257	0.279	0.258	0.280	0.261	0.280	0.260	0.280	0.261	0.282
iTransformer	ETTm1	**0.395**	**0.401**	0.405	0.405	0.413	0.416	0.407	0.412	0.417	0.415	0.416	0.415	0.411	0.414
	ETTh1	**0.438**	**0.434**	0.442	0.437	0.455	0.451	0.452	0.448	0.470	0.457	0.463	0.454	0.452	0.448
	ECL	**0.170**	**0.260**	0.176	0.264	0.178	0.269	0.178	0.269	0.175	0.266	0.177	0.267	0.179	0.270
	Weather	**0.251**	**0.272**	0.257	0.276	0.289	0.313	0.286	0.309	0.292	0.316	0.291	0.313	0.269	0.289

Note: **Bold** and underlined denote best and second-best results, respectively. The reported results are averaged over forecast horizons: T=96, 192, 336 and 720.

across other datasets further validate its effectiveness. These results suggest that modifying the learning objective can yield improvements comparable to, or even exceeding, those from architectural advancements. We attribute this to two key aspects of Time-o1: its decorrelation effect for debiased training, and its discrimination on significant components, which simplifies optimization.

Showcases. We visualize the forecast sequences and the generated components to showcase the improvements of Time-o1 in forecast quality. A snapshot on ETTm2 with historical window $\mathrm{H} = 96$ and forecast horizon $\mathrm{T} = 336$ is depicted in Fig. 2. Although the model trained using canonical DF captures general trends, its forecast struggles with large variations (e.g., peaks within steps 100-400). This reflects its difficulty in modeling significant, high-variance components. In contrast, Time-o1, by explicitly discriminating and aligning these significant components, generates a forecast that accurately captures these large variations, including the peaks within steps 100-400.

4.3 Learning objective comparison

Table 2 compares Time-o1 against other time-series learning objectives: FreDF [49], Koopman [18], Dilate [19], Soft-DTW [4], and DPTA [41]. For fair evaluation, we integrated their official implementations into both Fredformer and iTransformer.

Overall, shape alignment objectives (Dilate, Soft-DTW, DPTA) offer little performance gain over canonical DF (using TMSE loss), consistent with the findings in [19]. This phenomenon is rationalized by the fact that they do not mitigate the label autocorrelation nor reduce task amounts for simplifying optimization. FreDF improves performance by partly addressing autocorrelation bias. However, as discussed in Section 3.1, FreDF does not fully eliminate this bias, nor does it distinguish significant components to simplify the optimization landscape. Time-o1 directly addresses these two limitations of FreDF, leading to its superior overall performance.

4.4 Ablation studies

Table 3 presents an ablation study dissecting the contributions of critical factors in Time-o1: the decorrelation effect and the task reduction effect. The main findings are summarized as follows.

Table 3: Ablation study results.

Model	Decorrelation	Reduction	Data	T=96		T=192		T=336		T=720		Avg	
				MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
DF	✗	✗	ETTm1	0.326	0.361	0.365	0.382	0.396	0.404	0.459	0.444	0.387	0.398
			ETTh1	0.377	0.396	0.437	0.425	0.486	0.449	0.488	0.467	0.447	0.434
			ECL	0.150	0.242	0.168	0.259	0.182	0.274	0.214	0.304	0.179	0.270
			Weather	0.174	0.228	0.213	0.266	0.270	0.316	0.337	0.362	0.249	0.293
Time-o1†	✗	✓	ETTm1	0.338	0.366	0.369	0.383	0.397	0.403	0.458	0.441	0.391	0.398
			ETTh1	0.376	0.395	0.437	0.430	0.478	0.450	0.469	0.467	0.440	0.436
			ECL	0.150	0.239	0.164	0.253	0.178	0.268	0.210	0.296	0.175	0.264
			Weather	0.170	0.216	0.213	0.259	0.262	0.300	0.332	0.351	0.244	0.281
Time-o1‡	✓	✗	ETTm1	0.324	0.359	0.362	0.379	0.390	0.400	0.451	0.438	0.382	0.394
			ETTh1	0.373	0.395	0.433	0.423	0.476	0.445	0.474	0.463	0.439	0.431
			ECL	0.147	0.238	0.162	0.252	0.174	0.267	0.205	0.294	0.172	0.263
			Weather	0.172	0.220	0.211	0.259	0.261	0.301	0.331	0.353	0.244	0.283
Time-o1	✓	✓	ETTm1	**0.321**	**0.357**	**0.360**	**0.378**	**0.389**	**0.400**	**0.447**	**0.435**	**0.379**	**0.393**
			ETTh1	**0.368**	**0.391**	**0.424**	**0.422**	**0.467**	**0.441**	**0.465**	**0.463**	**0.431**	**0.429**
			ECL	**0.145**	**0.235**	**0.159**	**0.249**	**0.173**	**0.264**	**0.203**	**0.292**	**0.170**	**0.260**
			Weather	**0.169**	**0.219**	**0.210**	**0.258**	**0.259**	**0.297**	**0.327**	**0.349**	**0.241**	**0.280**

Note: **Bold** and underlined denote best and second-best results, respectively.

Table 4: Varying transformations results.

	ECL				Weather			
Transformation	MSE	Δ	MAE	Δ	MSE	Δ	MAE	Δ
None	0.179	-	0.270	-	0.249	-	0.293	-
RPCA	0.171	4.31% ↓	0.261	3.16% ↓	0.244	1.78% ↓	0.286	2.38% ↓
SVD	0.175	2.24% ↓	0.264	2.18% ↓	0.248	0.34% ↓	0.290	0.93% ↓
FA	0.175	2.35% ↓	0.265	1.82% ↓	0.245	1.35% ↓	0.287	1.97% ↓
Ours	**0.170**	**4.86%↓**	**0.260**	**3.57%↓**	**0.241**	**2.94%↓**	**0.280**	**4.28%↓**

Note: Δ refers to the relative error reduction compared to the baseline (None). **Bold** and underlined denote best and second-best results.

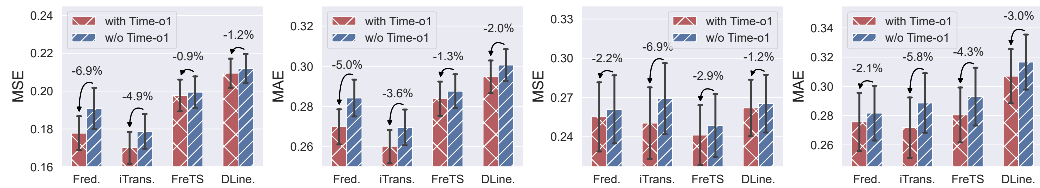
- Time-o1† improves DF by reducing the number of tasks to optimize. To this end, it employs a randomized matrix as the projection matrix to generate components and aligns only a subset of the obtained components. The involution ratio γ is finetuned on the validation set. It consistently improves over DF (e.g., -0.012 MAE on Weather). This demonstrates that reducing tasks with a minimal loss of label information can reduce optimization difficulty and improve performance.
- Time-o1‡ improves DF by aligning decorrelated components. To this end, the objective is calculated in (5) with $\gamma = 1$. It also outperforms DF, achieving the second-best results overall. This demonstrates aligning decorrelated label components to mitigate bias benefits forecast performance.
- Time-o1 integrates both factors above by aligning the most significant decorrelated components, which achieves the best performance, demonstrating the synergistic effect of these two factors.

4.5 Generalization studies

In this section, we investigate the utility of Time-o1 with different transformation strategies and forecast models, to showcase the generality of Time-o1. In the bar-plots, the forecast errors are averaged over forecast lengths (96, 192, 336, 720), with error bars as 50% confidence intervals.

Varying transformations. We select alternative approaches to transform the label sequence into latent components and report the forecast performance in Table 4. The selected transformation methods include robust principal component analysis (RPCA) [1], SVD [7], and factor analysis [16]. Noting that the output of SVD yields components here, not a projection matrix as in Section 3.2. Implementation details are in Appendix C. Overall, all these transformation methods outperform canonical DF without transformation. However, the components obtained by these methods, including RPCA, cannot be guaranteed to be decorrelated. Consequently, autocorrelation bias may persist. In contrast, our approach ensures full decorrelation of the derived components (see Lemma 3.2), effectively addressing autocorrelation bias and leading to the best overall performance.

Varying forecast models. We explore the versatility of Time-o1 in augmenting representative forecast models: Fredformer [36], iTransformer [27], FreTS [63], and DLinear [66]. As illustrated in Fig. 3, Time-o1 improves forecast performance in all cases. For instance, on the Weather dataset, iTransformer and FreTS with Time-o1 achieve substantial reductions in MSE—up to 6.9% and 2.9%,



(a) ECL with MSE (b) ECL with MAE (c) Weather with MSE (d) Weather with MAE

Figure 3: Improvement of Time-o1 applied to different forecast models, shown with colored bars for means over forecast lengths (96, 192, 336, 720) and error bars for 50% confidence intervals.

Table 5: Hyperparameter results on α.

	ETTm1		ETTh2		Weather	
α	MSE	MAE	MSE	MAE	MSE	MAE
0	0.3867	0.3979	0.3766	0.4019	0.2486	0.2930
0.3	0.3871	0.3983	0.3742	0.3982	0.2439	0.2851
0.5	0.3864	0.3976	0.3703	0.3964	**0.2432**	**0.2833**
0.7	**0.3831**	0.3959	0.3674	**0.3943**	0.2433	0.2849
1	0.3850	**0.3933**	**0.3606**	0.3890	0.2753	0.3209

Note: **Bold** and underlined denote best and second-best results.

Table 6: Hyperparameter results on γ.

	ETTm1		ETTh2		Weather	
γ	MSE	MAE	MSE	MAE	MSE	MAE
0.1	0.3915	0.4002	0.3816	0.4029	0.2437	0.2845
0.3	0.3849	0.3964	0.3694	0.3961	**0.2424**	**0.2825**
0.5	0.3817	0.3943	0.3651	0.3923	0.2466	0.2877
0.7	**0.3798**	**0.3930**	**0.3603**	**0.3886**	0.2443	0.2861
1	0.3814	0.3940	0.3624	0.3903	0.2491	0.2924

Note: **Bold** and underlined denote best and second-best results.

respectively. Further evidence of Time-o1's versatility can be found in Appendix E.4. These results confirm Time-o1's potential as a plug-and-play strategy to enhance adiverse forecast models.

4.6 Hyperparameter sensitivity

In this section, we examine the impact of critical hyperparameters on the performance of Time-o1. The results are presented in Table 5 and Table 6. Additional trends across different datasets and forecast lengths are provided in Appendix E.5. The primary observations are summarized as follows:

- The coefficient α determines the relative importance of the transformed objective in (5). When α is set to 1, Time-o1 exclusively uses the transformed objective. We observe that increasing α from 0 to 1 generally leads to improved forecasting accuracy, with the best results typically achieved when α is close to 1. The performance improvement is significant, e.g., MSE reduction on ETTh2 by 0.016, showcasing the utility of the transformed objective to improve forecast performance.
- The coefficient γ determines the ratio of involved components for alignment. When γ is set to 1, Time-o1 aligns all obtained components for model training. The results demonstrate that setting γ to 1, with all label information preserved, does not necessarily yield optimal performance. Instead, the best results are often obtained at $\gamma < 1$, rendering some loss of label information. For instance, $\gamma = 0.7$ yields the best results on ETTm1 and ETTh2, while $\gamma = 0.3$ is optimal for the Weather dataset. The rationale is that focusing on aligning the most significant components can reduce the task amount, thereby simplifying optimization. Since the majority of the information is contained in the most significant components, the information loss is minimal. Collectively, these factors contribute to improved forecast performance.

5 Conclusion

In this study, we highlight the importance of designing effective objectives for time-series forecasting. Two critical challenges are formulated: label autocorrelation, which induces bias, and the excessive number of tasks, which impedes optimization. To address these challenges, we introduce a model-agnostic learning objective called Time-o1. This method transforms the label sequence into decorrelated components with discernible significance. Forecast models are trained to align the most significant components, which effectively mitigates label autocorrelation due to the decorrelation between components and reduces task amount by discarding non-significant components. Experiments demonstrate that Time-o1 improves the performance of forecast models across diverse datasets.

Limitations & future works. In this work, we investigate the challenges of label autocorrelation and excessive number of tasks in time-series forecasting. Nevertheless, these issues also manifest in areas such as speech generation, target recognition, and dense image prediction. Applying Time-o1 in these contexts is a promising avenue for future research. Additionally, historical sequence also exhibits autocorrelation and contains redundancy. Transforming inputs to derive decorrelated, compact representations could offer additional performance gains and also warrants investigation.

Acknowledgments

Z. Lin was supported by the NSF China (No. 62276004) and the State Key Laboratory of General Artificial Intelligence. H. Li was supported by National Natural Science Foundation of China (623B2002).

References

[1] HanQin Cai, Jian-Feng Cai, and Ke Wei. Accelerated alternating projections for robust principal component analysis. *Journal of Machine Learning Research*, 20(20):1–33, 2019.

[2] Defu Cao, Yujing Wang, Juanyong Duan, Ce Zhang, Xia Zhu, Congrui Huang, Yunhai Tong, Bixiong Xu, Jing Bai, Jie Tong, et al. Spectral temporal graph neural network for multivariate time-series forecasting. In *Proc. Adv. Neural Inf. Process. Syst.*, volume 33, pages 17766–17778, 2020.

[3] Zhichao Chen, Leilei Ding, Zhixuan Chu, Yucheng Qi, Jianmin Huang, and Hao Wang. Monotonic neural ordinary differential equation: Time-series forecasting for cumulative data. In *Proc. ACM Int. Conf. Inf. Knowl. Manag.*, pages 4523–4529, 2023.

[4] Marco Cuturi and Mathieu Blondel. Soft-dtw: a differentiable loss function for time-series. In *Proc. Int. Conf. Mach. Learn.*, pages 894–903. PMLR, 2017.

[5] Abhimanyu Das, Weihao Kong, Andrew Leach, Rajat Sen, and Rose Yu. Long-term forecasting with tide: Time-series dense encoder. *Trans. Mach. Learn. Res.*, 2023.

[6] Urška Demšar, Paul Harris, Chris Brunsdon, A Stewart Fotheringham, and Sean McLoone. Principal component analysis on spatial data: an overview. *Ann. Assoc. Am. Geogr.*, 103(1):106–128, 2013.

[7] Andrew Gibiansky. Cool linear algebra: Singular value decomposition. *Andrew Gibiansky Blog*, 29, 2013.

[8] Michael Greenacre, Patrick JF Groenen, Trevor Hastie, Alfonso Iodice d'Enza, Angelos Markos, and Elena Tuzhilina. Principal component analysis. *Nat. Rev. Methods Primers*, 2(1):100, 2022.

[9] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *Proc. Conf. Lang. Model.*, 2023.

[10] Albert Gu, Karan Goel, and Christopher Re. Efficiently modeling long sequences with structured state spaces. In *Proc. Int. Conf. Learn. Represent.*, 2021.

[11] Qihe Huang, Lei Shen, Ruixin Zhang, Jiahuan Cheng, Shouhong Ding, Zhengyang Zhou, and Yang Wang. Hdmixer: Hierarchical dependency with extendable patch for multivariate time series forecasting. In *Proc. AAAI Conf. Artif. Intell.*, volume 38, pages 12608–12616, 2024.

[12] Qihe Huang, Lei Shen, Ruixin Zhang, Shouhong Ding, Binwu Wang, Zhengyang Zhou, and Yang Wang. Crossgnn: Confronting noisy multivariate time series via cross interaction refinement. *Proc. Adv. Neural Inf. Process. Syst.*, 36:46885–46902, 2023.

[13] Qihe Huang, Zhengyang Zhou, , Yangze Li, Kuo Yang, Binwu Wang, and Yang Wang. Many minds, one goal: Time series forecasting via sub-task specialization and inter-agent cooperation. In *Proc. Adv. Neural Inf. Process. Syst.*, 2025.

[14] Qihe Huang, Zhengyang Zhou, Kuo Yang, Zhongchao Yi, Xu Wang, and Yang Wang. Timebase: The power of minimalism in efficient long-term time series forecasting. In *Proc. Int. Conf. Mach. Learn.*, 2025.

[15] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proc. Int. Conf. Learn. Represent.*, pages 1–9, 2015.

[16] Paul Kline. *An easy guide to factor analysis*. Routledge, 2014.

[17] Dilfira Kudrat, Zongxia Xie, Yanru Sun, Tianyu Jia, and Qinghua Hu. Patch-wise structural loss for time series forecasting. In *Proc. Int. Conf. Mach. Learn.*, 2025.

[18] Henning Lange, Steven L Brunton, and J Nathan Kutz. From fourier to koopman: Spectral methods for long-term time series prediction. *Journal of Machine Learning Research*, 22(41):1–38, 2021.

[19] Vincent Le Guen and Nicolas Thome. Shape and time distortion loss for training deep time series forecasting models. In *Proc. Adv. Neural Inf. Process. Syst.*, volume 32, 2019.

[20] Haoxuan Li, Chunyuan Zheng, Shuyi Wang, Kunhan Wu, Eric Wang, Peng Wu, Zhi Geng, Xu Chen, and Xiao-Hua Zhou. Relaxing the accurate imputation assumption in doubly robust learning for debiased collaborative filtering. In *Proc. Int. Conf. Mach. Learn.*, volume 235, pages 29448–29460, 2024.

[21] Jianxin Li, Xiong Hui, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proc. AAAI Conf. Artif. Intell.*, 2021.

[22] Xinyu Li, Yuchen Luo, Hao Wang, Haoxuan Li, Liuhua Peng, Feng Liu, Yandong Guo, Kun Zhang, and Mingming Gong. Towards accurate time series forecasting via implicit decoding. *Proc. Adv. Neural Inf. Process. Syst.*, 2025.

[23] Zhe Li, Xiangfei Qiu, Peng Chen, Yihang Wang, Hanyin Cheng, Yang Shu, Jilin Hu, Chenjuan Guo, Aoying Zhou, Christian S Jensen, et al. Tsfm-bench: A comprehensive and unified benchmark of foundation models for time series forecasting. In *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, pages 5595–5606, 2025.

[24] Zhe Li, Xiangfei Qiu, Peng Chen, Yihang Wang, Hanyin Cheng, Yang Shu, Jilin Hu, Chenjuan Guo, Aoying Zhou, Qingsong Wen, et al. Foundts: Comprehensive and unified benchmarking of foundation models for time series forecasting. *arXiv preprint arXiv:2410.11802*, 2024.

[25] Bo Liu, Xingchao Liu, Xiaojie Jin, Peter Stone, and Qiang Liu. Conflict-averse gradient descent for multi-task learning. *Proc. Adv. Neural Inf. Process. Syst.*, 34:18878–18890, 2021.

[26] Minhao Liu, Ailing Zeng, Muxi Chen, Zhijian Xu, Qiuxia Lai, Lingna Ma, and Qiang Xu. Scinet: time series modeling and forecasting with sample convolution and interaction. In *Proc. Adv. Neural Inf. Process. Syst.*, 2022.

[27] Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. In *Proc. Int. Conf. Learn. Represent.*, 2024.

[28] Jiaming Ma, Zhiqing Cui, Binwu Wang, Pengkun Wang, Zhengyang Zhou, Zhe Zhao, and Yang Wang. Causal learning meet covariates: Empowering lightweight and effective nationwide air quality forecasting. *Proc. Int. Joint Conf. Artif. Intell.*, 2025.

[29] Jiaming Ma, Binwu Wang, Qihe Huang, Guanjun Wang, Pengkun Wang, Zhengyang Zhou, and Yang Wang. Mofo: Empowering long-term time series forecasting with periodic pattern modeling. In *Proc. Adv. Neural Inf. Process. Syst.*, 2025.

[30] Jiaming Ma, Binwu Wang, Guanjun Wang, Kuo Yang, Zhengyang Zhou, Pengkun Wang, Xu Wang, and Yang Wang. Less but more: Linear adaptive graph learning empowering spatiotemporal forecasting. In *Proc. Adv. Neural Inf. Process. Syst.*, 2025.

[31] Jiaming Ma, Binwu Wang, Pengkun Wang, Zhengyang Zhou, Xu Wang, and Yang Wang. Bist: A lightweight and efficient bi-directional model for spatiotemporal prediction. *Proc. VLDB Endow.*, 18(6):1663–1676, 2025.

[32] Jiaming Ma, Binwu Wang, Pengkun Wang, Zhengyang Zhou, Yudong Zhang, Xu Wang, and Yang Wang. Mobimixer: A multi-scale spatiotemporal mixing model for mobile traffic prediction. *IEEE Trans. Mob. Comput.*, 2025.

[33] Gonzalo Mateos, Santiago Segarra, Antonio G. Marques, and Alejandro Ribeiro. Connecting the dots: Identifying network structure via graph signal processing. *IEEE Signal Process. Mag.*, 36(3):16–43, 2019.

[34] Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *Proc. Int. Conf. Learn. Represent.*, 2023.

[35] Karl Pearson. Principal components analysis. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 6(2):559, 1901.

[36] Xihao Piao, Zheng Chen, Taichi Murayama, Yasuko Matsubara, and Yasushi Sakurai. Fredformer: Frequency debiased transformer for time series forecasting. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 2400–2410, 2024.

[37] Xiangfei Qiu, Jilin Hu, Lekui Zhou, Xingjian Wu, Junyang Du, Buang Zhang, Chenjuan Guo, Aoying Zhou, Christian S. Jensen, Zhenli Sheng, and Bin Yang. Tfb: Towards comprehensive and fair benchmarking of time series forecasting methods. *Proc. VLDB Endow.*, 17(9):2363–2377, 2024.

[38] Xiangfei Qiu, Xiuwen Li, Ruiyang Pang, Zhicheng Pan, Xingjian Wu, Liu Yang, Jilin Hu, Yang Shu, Xuesong Lu, Chengcheng Yang, Chenjuan Guo, Aoying Zhou, Christian S. Jensen, and Bin Yang. Easytime: Time series forecasting made easy. In *Proc. IEEE Int. Conf. Data Eng.*, 2025.

[39] Xiangfei Qiu, Xingjian Wu, Hanyin Cheng, Xvyuan Liu, Chenjuan Guo, Jilin Hu, and Bin Yang. Dbloss: Decomposition-based loss function for time series forecasting. *Proc. Adv. Neural Inf. Process. Syst.*, 2025.

[40] Xiangfei Qiu, Xingjian Wu, Yan Lin, Chenjuan Guo, Jilin Hu, and Bin Yang. Duet: Dual clustering enhanced multivariate time series forecasting. In *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2025.

[41] Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Trans. Signal Process.*, 26(1):43–49, 2003.

[42] William Toner and Luke Nicholas Darlow. An analysis of linear time series forecasting models. In *Proc. Int. Conf. Mach. Learn.*, pages 48404–48427. PMLR, 2024.

[43] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proc. Adv. Neural Inf. Process. Syst.*, 2017.

[44] Hao Wang, Zhichao Chen, Jiajun Fan, Haoxuan Li, Tianqiao Liu, Weiming Liu, Quanyu Dai, Yichao Wang, Zhenhua Dong, and Ruiming Tang. Optimal transport for treatment effect estimation. In *Proc. Adv. Neural Inf. Process. Syst.*, volume 36, pages 5404–5418, 2023.

[45] Hao Wang, Zhichao Chen, Zhaoran Liu, Xu Chen, Haoxuan Li, and Zhouchen Lin. Proximity matters: Local proximity enhanced balancing for treatment effect estimation. *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2025.

[46] Hao Wang, Zhichao Chen, Zhaoran Liu, Licheng Pan, Hu Xu, Yilin Liao, Haozhe Li, and Xinggao Liu. Spot-i: Similarity preserved optimal transport for industrial iot data imputation. *IEEE Trans. Ind. Informat.*, 20(12):14421–14429, 2024.

[47] Hao Wang, Zhichao Chen, Yuan Shen, Hui Zheng, Degui Yang, Dangjun Zhao, and Buge Liang. Unbiased recommender learning from implicit feedback via weakly supervised learning. In *IEEE Trans. Neural Netw. Learn. Syst.*, 2025.

[48] Hao Wang, Xinggao Liu, Zhaoran Liu, Haozhe Li, Yilin Liao, Yuxin Huang, and Zhichao Chen. Lspt-d: Local similarity preserved transport for direct industrial data imputation. *IEEE Trans. Autom. Sci. Eng.*, 22:9438–9448, 2025.

[49] Hao Wang, Licheng Pan, Yuan Shen, Zhichao Chen, Degui Yang, Yifei Yang, Sen Zhang, Xinggao Liu, Haoxuan Li, and Dacheng Tao. Fredf: Learning to forecast in the frequency domain. In *Proc. Int. Conf. Learn. Represent.*, pages 1–9, 2025.

[50] Hao Wang, Zhiyu Wang, Yunlong Niu, Zhaoran Liu, Haozhe Li, Yilin Liao, Yuxin Huang, and Xinggao Liu. An accurate and interpretable framework for trustworthy process monitoring. *IEEE Trans. Artif. Intell*, 5(5):2241–2252, 2023.

[51] Haotian Wang, Haoxuan Li, Hao Zou, Haoang Chi, Long Lan, Wanrong Huang, and Wenjing Yang. Effective and efficient time-varying counterfactual prediction with state-space models. In *Proc. Int. Conf. Learn. Represent.*, 2025.

[52] Huiqiang Wang, Jian Peng, Feihu Huang, Jince Wang, Junhui Chen, and Yifei Xiao. Micn: Multi-scale local and global context modeling for long-term series forecasting. In *Proc. Int. Conf. Learn. Represent.*, 2023.

[53] Zihan Wang, Fanheng Kong, Shi Feng, Ming Wang, Xiaocui Yang, Han Zhao, Daling Wang, and Yifei Zhang. Is mamba effective for time series forecasting? *Neurocomputing*, 619:129178, 2025.

[54] Qingsong Wen, Tian Zhou, Chaoli Zhang, Weiqi Chen, Ziqing Ma, Junchi Yan, and Liang Sun. Transformers in time series: a survey. In *Proc. Int. Joint Conf. Artif. Intell.*, pages 6778–6786, 2023.

[55] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. In *Proc. Int. Conf. Learn. Represent.*, 2023.

[56] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with Auto-Correlation for long-term series forecasting. In *Proc. Adv. Neural Inf. Process. Syst.*, 2021.

[57] Xingjian Wu, Zhengyu Li, Hanyin Cheng, Xiangfei Qiu, Jilin Hu, Chenjuan Guo, and Bin Yang. Unlocking the power of mixture-of-experts for task-aware time series analytics. *arXiv preprint arXiv:2509.22279*, 2025.

[58] Xingjian Wu, Xiangfei Qiu, Hanyin Cheng, Zhengyu Li, Jilin Hu, Chenjuan Guo, and Bin Yang. Enhancing time series forecasting through selective representation spaces: A patch perspective. In *Proc. Adv. Neural Inf. Process. Syst.*, 2025.

[59] Xingjian Wu, Xiangfei Qiu, Hongfan Gao, Jilin Hu, Bin Yang, and Chenjuan Guo. K2vae: A koopman-kalman enhanced variational autoencoder for probabilistic time series forecasting. In *Proc. Int. Conf. Mach. Learn.*, 2025.

[60] Xinle Wu, Xingjian Wu, Dalin Zhang, Miao Zhang, Chenjuan Guo, Bin Yang, and Christian S Jensen. Fully automated correlated time series forecasting in minutes. *Proc. VLDB Endow.*, 18(2):144–157, 2024.

[61] Zhijian Xu, Ailing Zeng, and Qiang Xu. Fits: Modeling time series with $10k$ parameters. In *Proc. Int. Conf. Learn. Represent.*, 2024.

[62] Kun Yi, Qi Zhang, Wei Fan, Hui He, Liang Hu, Pengyang Wang, Ning An, Longbing Cao, and Zhendong Niu. Fouriergnn: Rethinking multivariate time series forecasting from a pure graph perspective. In *Proc. Adv. Neural Inf. Process. Syst.*, 2023.

[63] Kun Yi, Qi Zhang, Wei Fan, Shoujin Wang, Pengyang Wang, Hui He, Ning An, Defu Lian, Longbing Cao, and Zhendong Niu. Frequency-domain mlps are more effective learners in time series forecasting. In *Proc. Adv. Neural Inf. Process. Syst.*, 2023.

[64] Wenzhen Yue, Yong Liu, Haoxuan Li, Hao Wang, Xianghua Ying, Ruohao Guo, Bowei Xing, and Ji Shi. Olinear: A linear model for time series forecasting in orthogonally transformed domain. *Proc. Adv. Neural Inf. Process. Syst.*, 2025.

[65] Wenzhen Yue, Yong Liu, Xianghua Ying, Bowei Xing, Ruohao Guo, and Ji Shi. Freeformer: Frequency enhanced transformer for multivariate time series forecasting. *arXiv preprint arXiv:2501.13989*, 2025.

[66] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *Proc. AAAI Conf. Artif. Intell.*, 2023.

[67] Weijia Zhang, Le Zhang, Jindong Han, Hao Liu, Yanjie Fu, Jingbo Zhou, Yu Mei, and Hui Xiong. Irregular traffic time series forecasting based on asynchronous spatio-temporal graph convolutional networks. In *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, pages 4302–4313, 2024.

[68] Yu Zhang and Qiang Yang. A survey on multi-task learning. *IEEE Trans. Knowl. Data Eng.*, 34(12):5586–5609, 2021.

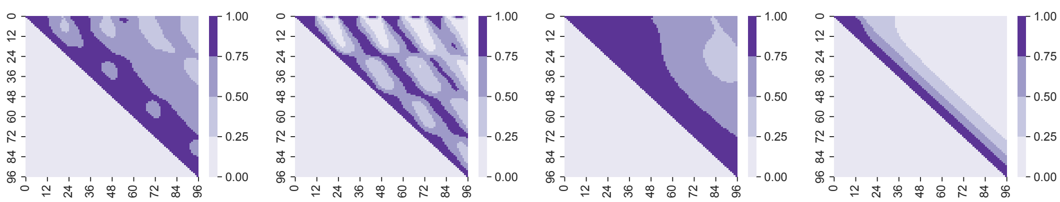
[69] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *Proc. Int. Conf. Mach. Learn.*, 2022.

A On the Implementation Details of Label Correlation Estimation

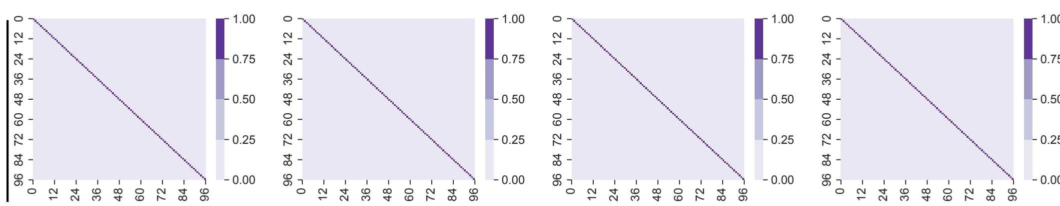
In this section, we introduce the motivation and implementation details of the label autocorrelation estimation techniques in Fig. 1. Measuring label autocorrelation $Y_t \rightarrow Y_{t'}$ is indeed challenging due to the presence of confounding effect [45, 44, 20]. Specifically, the fork structure $Y_t \leftarrow L \rightarrow Y_{t'}$ introduces spurious correlations between Y_t and $Y_{t'}$, thereby distorting the true strength of the label autocorrelation $Y_t \rightarrow Y_{t'}$ of interest. This structural confounding undermines the validity of traditional measures such as Pearson correlation for quantifying label autocorrelation.

The previous work [49] involved the double machine learning (DML) method to estimate the ground-truth correlation while mitigating the influence of the fork structure. We adopt this in our experiments. DML is a statistical technique designed to estimate the causal effect of a treatment on an outcome while controlling for fork variables. Specifically, suppose we have a treatment variable $\mathcal{T}$, an outcome variable $\mathcal{Y}$, and a set of fork variables $\mathcal{X}$. The goal is to estimate the causal effect of $\mathcal{T}$ on $\mathcal{Y}$ while controlling for the influence of $\mathcal{X}$. To this end, DML first orthogonalizes both the treatment and outcome with respect to the fork variables. Two parametric models are employed to predict the treatment and outcome based on the fork variables. These predictions capture the impact of $\mathcal{X}$ on $\mathcal{Y}$ and $\mathcal{T}$. Subsequently, such impact of $\mathcal{X}$ is eliminated by calculating the residuals. Finally, the DML method regresses the outcome residuals on the treatment residuals, thereby measuring the causal effect of T on Y while removing the influence of the fork variables.

In our experiments, we measure label autocorrelation by treating the input sequence L as the fork variable and different steps of the label sequences Y_t and $Y_{t'}$ as the treatment and outcome variables, respectively. Then, we estimate the treatment effect of Y_t on $Y_{t'}$ controlling L. Similarly, when measuring the correlation between different components, we use different components Z_k and $Z_{k'}$ as the treatment and outcome variables. Linear regression model is employed as the parametric model for both the treatment and outcome variables for efficiency, which is consistent to [49].



(a) Correlation between different steps in the label sequence.



(b) Correlation between different extracted components.

Figure 4: The label autocorrelation in the original label sequence and the extracted components. The datasets are ETTh1, ETTh2, ETTm1, and Weather from left to right. The forecast length is set to 96.

To further complement the case study in Fig. 1, we analyzed the correlation matrices of the label sequences and the extracted components across multiple datasets, with the results presented in Fig. 4. The main observations are summarized as follows.

- Panel (a) displays the correlation matrix of the label sequence, characterized by substantial non-diagonal elements, which highlight the strong autocorrelation among the labels. In contrast, panel (b) shows the correlation matrix of the extracted components, where the non-diagonal elements are nearly zero, indicating effective decorrelation.
- Compared to the results reported in [49], where some obtained components remain correlated, the non-diagonal elements in panel (b) are fully eliminated. This difference arises because the Fourier transform in [49] achieves decorrelation only when the original label sequence is nearly infinitely long (Tß∞), a condition that is not met in real-world applications with finite forecast horizons.

This limitation stems from the predefined nature of the projection matrix, which lacks adaptation to the specific properties of the data. In contrast, our method ensures decorrelation by solving a constrained optimization problem, without relying on an infinitely long forecast horizon, thereby providing a more reliable approach for handling autocorrelation bias.

B Theoretical Justification

Theorem B.1 (Autocorrelation bias, Theorem 3.1 in the main text)**.** *Given label sequence Y where $\Sigma \in \mathbb{R}^{\mathrm{T}\times\mathrm{T}}$ denotes the step-wise correlation coefficient, the TMSE in (1) is biased compared to the negative log-likelihood of the label sequence, which is given by:*

$$\mathrm{Bias} = \left\|Y - \hat{Y}\right\|_{\Sigma^{-1}}^{2} - \left\|Y - \hat{Y}\right\|^{2} - \frac{1}{2}\log|\Sigma| \,. \tag{6}$$

where $\|v\|_{\Sigma^{-1}}^2 = v^\top \Sigma^{-1} v$. The bias vanishes if different steps in Y are decorrelated.[5]

Proof. The proof follows our previous work [49] but relaxes the first-order Markov assumption.

Suppose the label sequence follows a multivariate normal distribution with mean vector $\mu = [\hat{Y}_1, \hat{Y}_2, \ldots, \hat{Y}_\mathrm{T}]$ and covariance matrix Σ, where the off-diagonal entries are $\Sigma_{ij} = \rho_{ij}\sigma^2$ for $i \neq j$. Here, ρ_{ij} denotes the partial correlation between Y_i and Y_j given the input sequence L. The log-likelihood of the label sequence Y can be expressed as:

$$\log p(Y) = \frac{1}{2}\left(\mathrm{T}\log(2\pi) + \log|\Sigma| + (Y - \hat{Y})^\top \Sigma^{-1}(Y - \hat{Y})\right).$$

Removing the constant terms unrelated to $\hat{Y}$, we obtain the practical negative log-likelihood (PNLL):

$$\mathrm{PNLL} = (Y - \hat{Y})^\top \Sigma^{-1}(Y - \hat{Y}).$$

On the other hand, the TMSE loss can be expressed as:

$$\mathrm{TMSE} = \left\|Y - \hat{Y}\right\|_2^2 = (Y - \hat{Y})^\top I^{-1}(Y - \hat{Y}).$$

where I is the identity matrix. The difference between TMSE and PNLL can be expressed as:

$$\mathrm{Bias} = \mathrm{PNLL} - \mathrm{TMSE} = (Y - \hat{Y})^\top \Sigma^{-1}(Y - \hat{Y}) - (Y - \hat{Y})^\top I (Y - \hat{Y}),$$

which immediately vanishes if the label sequence is decorrelated, i.e., $\Sigma = I$. The proof is completed. $\square$

Lemma B.2 (Lemma 3.3 in the main text)**.** *The projection matrix $\mathbf{P}^*$ can be obtained via singular value decomposition (SVD): $\mathbf{Y} = \mathbf{U}\mathbf{\Lambda}(\mathbf{P}^*)^\top$, where $\mathbf{U} \in \mathbb{R}^{\mathrm{m}\times\mathrm{m}}$ and $\mathbf{P}^* \in \mathbb{R}^{\mathrm{T}\times\mathrm{T}}$ consist of singular vectors, and the diagonal of $\mathbf{\Lambda} \in \mathbb{R}^{\mathrm{m}\times\mathrm{T}}$ consists of singular values.*

Proof. We first consider the case with $p = 1$, where the orthogonal constrains are not involved. The Lagrangian can be written as:

$$\mathcal{L}(\mathbf{P}_1, \lambda_1) = \mathbf{P}_1^\top \mathbf{S}\mathbf{P}_1 - \lambda_1(\mathbf{P}_1^\top \mathbf{P}_1 - 1), \tag{7}$$

where λ_1 is the Lagrangian multiplier. According to the first-order condition, the derivative with respect to $\mathbf{P}_1$ should be zero:

$$\left.\frac{\partial \mathcal{L}}{\partial \mathbf{P}_1}\right|_{\mathbf{P}_1 = \mathbf{P}_1^*} = 2\mathbf{S}\mathbf{P}_1^* - 2\lambda_1 \mathbf{P}_1^* = 0, \tag{8}$$

which immediately follows by $\mathbf{S}\mathbf{P}_1^* = \lambda_1 \mathbf{P}_1^*$. Apparently, $\mathbf{P}_1^*$ is an eigenvector of $\mathbf{S}$, with corresponding eigenvalue λ_1. Moreover, $\mathbf{P}_1^*$ is the leading eigenvector of $\mathbf{S}$ associated with the largest eigenvalue, which follows from the maximization objective is ${\mathbf{P}_1}^{*\top}\mathbf{S}\mathbf{P}_1^* = \lambda_1$,

[5] The pioneering work [49] identifies the bias under the first-order Markov assumption on the label sequence. This study generalizes this bias without the first-order Markov assumption.

We further consider the case with $p \geq 2$, impose orthogonality to all previous projection vectors. Defining Lagrangian multipliers λ_p and $\{\mu_j\}_{j=1}^{p-1}$, we write the Lagrangian as follow

$$\mathcal{L}(\mathbf{P}_p, \lambda_p, \{\mu_j\}) = \mathbf{P}_p^\top \mathbf{S} \mathbf{P}_p - \lambda_p(\mathbf{P}_p^\top \mathbf{P}_p - 1) - \sum_{j=1}^{p-1} \mu_j \mathbf{P}_p^\top \mathbf{P}_j. \tag{9}$$

According to the first-order condition, the derivative with respect to $\mathbf{P}_p$ should be zero:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{P}_p}\Big|_{\mathbf{P}_p=\mathbf{P}_p^*} = 2\mathbf{S}\mathbf{P}_p^* - 2\lambda_p \mathbf{P}_p^* - \sum_{j=1}^{p-1} \mu_j \mathbf{P}_j = 0. \tag{10}$$

To resolve $\{\mu_j\}$, we take the inner product of both sides using $\mathbf{P}_k^{*\top}$ with $k = 1, 2, ..., p-1$:

$$\mathbf{P}_k^{*\top} \mathbf{S} \mathbf{P}_p^* - \lambda_p \mathbf{P}_k^{*\top} \mathbf{P}_p^* - \frac{1}{2} \sum_{j<p} \mu_j \mathbf{P}_k^{*\top} \mathbf{P}_j = 0. \tag{11}$$

Since $\mathbf{P}_k^*$ is previously obtained that satisfies $\mathbf{S}\mathbf{P}_k^* = \lambda_k \mathbf{P}_k^*$ and $\mathbf{P}_k^{*\top}\mathbf{P}_k^* = 1$, we have $\mathbf{P}_k^{*\top}\mathbf{S}\mathbf{P}_p^* = \lambda_k$. Due to the orthogonal constraint, the current projection vector should be orthogonal to the previously derived ones, we have $\mathbf{P}_k^{*\top}\mathbf{P}_p^* = 0$ and $\mathbf{P}_k^{*\top}\mathbf{P}_j = \delta_{k,j}$, where $\delta_{k,j} = 1$ if $j = k$ and 0 otherwise. Putting together, the equation simplifies to: $\mu_k = 0$.

Since $\mu_k = 0$ holds for all $k = 1, 2, ..., p-1$, we have $\mu_1 = \mu_2 = ... = \mu_{p-1} = 0$. Plugging back to (10), the optimal condition becomes:

$$\mathbf{S}\mathbf{P}_p^* = \lambda_p \mathbf{P}_p^*, \tag{12}$$

with the additional restriction that $\mathbf{P}_p^*$ is orthogonal to all previous directions. Therefore, $\mathbf{P}_p^*$ must be the eigenvector of $\mathbf{S}$ corresponding to the p-th largest eigenvalue. That is, $\mathbf{P}$ can be derived by performing eigenvector decomposition on $\mathbf{S}$.

Moving forward, consider SVD: $\mathbf{Y} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{V}^\top$, we can represent $\mathbf{S}$ as

$$\mathbf{S} = \mathbf{Y}^\top\mathbf{Y} = \mathbf{V}\boldsymbol{\Lambda}^\top\mathbf{U}^\top\mathbf{U}\boldsymbol{\Lambda}\mathbf{V}^\top = \mathbf{V}\boldsymbol{\Lambda}^2\mathbf{V}^\top, \tag{13}$$

which implies that each column of $\mathbf{V}$ is the eigenvector of $\mathbf{S}$, i.e., $\mathbf{V} = \mathbf{P}^*$. Therefore, the projection matrix $\mathbf{P}^*$ can be obtained by performing SVD on $\mathbf{Y}$ as: $\mathbf{Y} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{V}^\top$, where $\mathbf{V}$ is exactly the optimum projection matrix $\mathbf{P}^*$. The proof is therefore completed. □

Lemma B.3 (Decorrelated components, Lemma 3.2 in the main text)**.** *Suppose* $\mathbf{Y} \in \mathbb{R}^{\mathrm{m}\times\mathrm{T}}$ *contains normalized label sequences for* m *samples,* $\mathbf{Z} = [\mathbf{Z}_1, ..., \mathbf{Z}_\mathrm{T}]$ *are the obtained components; for any* $p \neq p'$*, we have* $\mathbf{Z}_p^\top \mathbf{Z}_{p'} = 0$.

Proof. For any two latent components $\mathbf{Z}_p$ and $\mathbf{Z}_{p'}$ with $p \neq p'$, we have:

$$\mathbf{Z}_p^\top \mathbf{Z}_{p'} = (\mathbf{Y}\mathbf{P}_p)^\top(\mathbf{Y}\mathbf{P}_{p'}) = \mathbf{P}_p^\top \mathbf{Y}^\top \mathbf{Y} \mathbf{P}_{p'} \tag{14}$$

According to (12), $\mathbf{P}_p$ and $\mathbf{P}_{p'}$ are eigenvectors of $\mathbf{Y}^\top\mathbf{Y}$, we have

$$\mathbf{Y}^\top\mathbf{Y}\mathbf{P}_p = \lambda_p \mathbf{P}_p, \qquad \mathbf{Y}^\top\mathbf{Y}\mathbf{P}_{p'} = \lambda_{p'} \mathbf{P}_{p'}, \tag{15}$$

which immediately follows by $\mathbf{Z}_p^\top \mathbf{Z}_{p'} = \lambda_{p'} \mathbf{P}_p^\top \mathbf{P}_{p'}$. Recalling that different projection bases are constrained to orthogonal, i.e., $\mathbf{P}_p^\top \mathbf{P}_{p'} = 0$ for $p \neq p'$, we have

$$\mathbf{Z}_p^\top \mathbf{Z}_{p'} = 0 \qquad \text{for all } p \neq p'. \tag{16}$$

The proof is completed. □

C Generalized Orthogonalization Methods

In this section, we introduce alternative transform methods for obtaining latent components, each with distinct characteristics such as dimensionality reduction and noise isolation. We discuss their implications for transforming label sequences in time-series forecasting, with a comparative study detailed in Section 4.5.

RPCA. The robust principal component analysis decomposes the data into a low-rank informative component and a sparse noise component, effectively separating structured signals from noise. Specifically, given $\mathbf{Y} \in \mathbb{R}^{\mathrm{m}\times\mathrm{T}}$, it is achieved by solving:

$$\min_{\mathbf{V},\mathbf{S}} \|\mathbf{V}\|_* + \lambda\|\mathbf{S}\|_1, \quad \text{subject to} \quad \mathbf{Y} = \mathbf{V} + \mathbf{S}, \tag{17}$$

where $\|\cdot\|_*$ is the nuclear norm, $\|\cdot\|_1$ is the element-wise ℓ_1 norm, and λ is a regularization parameter. Afterwards, it performs the principal component analysis on the obtained informative component $\mathbf{V}$ to derive the projection matrix $\mathbf{P}$. The latent components are generated by $\mathbf{Z} = \mathbf{YP}$. While this approach enhances noise elimination, it does not guarantee decorrelation of the derived components, as the projection matrix $\mathbf{P}$ is derived from $\mathbf{V}$ instead of the original data matrix $\mathbf{Y}$.

SVD. The singular value decomposition provides a method to decompose the matrix into different components. Given $\mathbf{Y} \in \mathbb{R}^{\mathrm{m}\times\mathrm{T}}$, we have:

$$\mathbf{Y} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^\top, \tag{18}$$

where $\mathbf{U} \in \mathbb{R}^{\mathrm{m}\times\mathrm{r}}$ and $\mathbf{V} \in \mathbb{R}^{\mathrm{T}\times\mathrm{R}}$ are singular vectors, $\mathbf{\Lambda} \in \mathbb{R}^{\mathrm{r}\times\mathrm{r}}$ is diagonal with rank r. The right singular vector is used as the projection matrix, and the latent components are generated by $\mathbf{Z} = \mathbf{YV}$. One key distinction here needs to be highlighted. Unlike the workflow in the main text (Algorithm 1), the label sequence is not normalized after window generation before computing SVD here, resulting in non-decorrelated components.

FA. Factor analysis models the observed data as linear combinations of a small number of latent factors plus noise, capturing the covariance structure through these unobserved factors. Specifically, given mean-centered $\mathbf{Y} \in \mathbb{R}^{\mathrm{m}\times\mathrm{T}}$, the model assumes:

$$\mathbf{Y} = \mathbf{V}\mathbf{F}^\top + \mathbf{E}, \tag{19}$$

where $\mathbf{V} \in \mathbb{R}^{\mathrm{m}\times\mathrm{K}}$ is the factor loading matrix, K is the number of latent factors ($\mathrm{K} \ll \mathrm{m}$), $\mathbf{F} \in \mathbb{R}^{\mathrm{T}\times\mathrm{K}}$ contains the latent factor scores for each sample, and $\mathbf{E} \in \mathbb{R}^{\mathrm{m}\times\mathrm{T}}$ is the noise matrix. The standard assumption is that each factor $f_i \sim \mathcal{N}(0, \mathbf{I})$ and noise $\epsilon_i \sim \mathcal{N}(0, \Psi)$, where Ψ is a diagonal covariance matrix. The loadings $\mathbf{V}$ and factor scores $\mathbf{F}$ are typically estimated via maximum likelihood. The latent components are given by the estimated factor scores, *i.e.*, $\mathbf{Z} = \mathbf{Y}\Psi^{-1}\mathbf{F}(\mathbf{I} + \mathbf{F}^\top\Psi^{-1}\mathbf{F})^{-1} := \mathbf{YP}$ [6]. This approach captures the covariance structure of $\mathbf{Y}$ via a small number of factors, but does not necessarily guarantee uncorrelated or noise-isolated components.

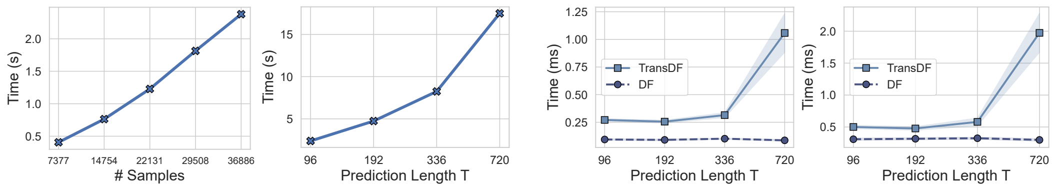
D Complexity Analysis

In this section, we analyze the running cost of Time-o1. The core computation of Time-o1 involves (a) calculating the projection matrix $\mathbf{P}^*$ via SVD, and (b) performing transformation on both predicted and label sequences, followed by calculating their point-wise MAE loss. Given the target matrix $\mathbf{Y} \in \mathbb{R}^{\mathrm{m}\times\mathrm{T}}$, the SVD step decomposes $\mathbf{Y}$ with an established complexity of $\mathcal{O}(\mathrm{mT}^2)$ (assuming $\mathrm{m} \geq \mathrm{T}$). For the sequence transformation, each sample (row) in $\mathbf{Y}$ is multiplied by the projection matrix $\mathbf{P}^* \in \mathbb{R}^{\mathrm{T}\times\mathrm{T}}$, resulting in a total complexity of $\mathcal{O}(\mathrm{mT}^2)$. The computation of point-wise MAE loss across all samples and forecast steps is $\mathcal{O}(\mathrm{mT})$, which is negligible compared to the complexity of previous steps. Thus, the overall complexity per batch is dominated by the SVD and projection operations, both scaling as $\mathcal{O}(\mathrm{mT}^2)$. The main findings from the empirical evaluations are as follows.

- Fig. 5 (a) presents the computational cost for calculating the projection matrix. Overall, it increases linearly with the sample size and quadratically with the prediction length, which aligns with the theoretical complexity. Importantly, this operation is performed only once before training begins, rendering the associated overhead acceptable.
- Fig. 5 (b) presents the computational cost for the sequence transformation. The cost increases quadratically with the prediction length, but remains below 2 ms. This cost is comparable to that of a linear projection. Furthermore, sequence transformation is not required during inference.

In conclusion, Time-o1 does not add complexity to model inference, and the additional complexity during the training stage is negligible.

[6] Adapted from source code of sklearn: `https://github.com/scikit-learn/scikit-learn/blob/98ed9dc73/sklearn/decomposition/`



(a) Running cost of SVD.   (b) Running cost of transformation.

Figure 5: Running cost for projection matrix calculation (left panel with varying number of samples, right panel with varying prediction length) and sequence transformation (left panel for forward pass, right panel for backward pass, with average and shaded areas for 95% confidence intervals).

E More Experimental Results

E.1 Long-term forecast performance

Additional results on long-term forecast performance are available in Table 7.

E.2 Short-term forecast performance

Additional results on short-term forecast performance are available in Table 8, where Fredformer [36] serves as the forecast model.

E.3 Showcases

Additional results on showcases are available in Fig. 6 and Fig. 7.

E.4 Generalization studies

Additional results on varying forecast models and transformations are available in Fig. 8 and Table 9.

E.5 Hyperparameter sensitivity

Additional results on hyperparameter sensitivity are available in Fig. 9 for α and Fig. 10 for γ.

E.6 Comparison with different learning objectives

Additional results on comparing different learning objectives are available in Table 10.

E.7 Varying input length results

Additional results on varying input lengths are available in Table 11—complementing the fixed length of 96 used in the main text.

E.8 Random seed sensitivity

Additional results on varying random seeds are available in Table 12.

Table 7: The comprehensive results on the long-term forecasting task.

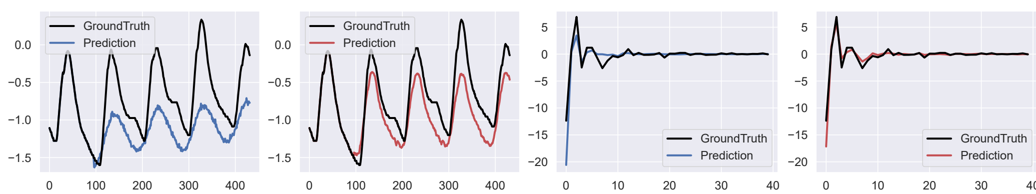
Models		**Time-o1 (Ours)**		Fredformer (2024)		iTransformer (2024)		FreTS (2023)		TimesNet (2023)		MICN (2023)		TiDE (2023)		DLinear (2023)		FEDformer (2022)		Autoformer (2021)		Transformer (2017)	
Metrics		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm1	96	0.321	**0.357**	0.326	0.361	0.338	0.372	0.342	0.375	0.368	0.394	**0.319**	0.366	0.353	0.374	0.346	0.373	0.401	0.434	0.485	0.468	0.503	0.482
	192	**0.360**	**0.378**	0.365	0.382	0.382	0.396	0.385	0.400	0.406	0.409	0.364	0.395	0.391	0.393	0.380	0.390	0.415	0.446	0.504	0.482	0.807	0.664
	336	**0.389**	**0.400**	0.396	0.404	0.427	0.424	0.416	0.421	0.454	0.444	0.395	0.425	0.423	0.414	0.413	0.414	0.432	0.450	0.520	0.489	0.847	0.678
	720	**0.447**	**0.435**	0.459	0.444	0.496	0.463	0.513	0.489	0.527	0.474	0.505	0.499	0.486	0.448	0.472	0.450	0.522	0.500	0.594	0.523	1.037	0.771
	Avg	**0.379**	**0.393**	0.387	0.398	0.411	0.414	0.414	0.421	0.438	0.430	0.396	0.421	0.413	0.407	0.403	0.407	0.442	0.457	0.526	0.491	0.799	0.648
ETTm2	96	**0.172**	**0.251**	0.177	0.260	0.182	0.265	0.188	0.279	0.184	0.262	0.178	0.277	0.182	0.265	0.188	0.283	0.205	0.289	0.218	0.300	0.386	0.441
	192	**0.235**	**0.294**	0.242	0.300	0.257	0.315	0.264	0.329	0.257	0.308	0.266	0.343	0.247	0.304	0.280	0.356	0.271	0.334	0.282	0.340	1.410	0.881
	336	**0.293**	**0.333**	0.302	0.340	0.320	0.354	0.322	0.369	0.315	0.345	0.299	0.354	0.307	0.343	0.375	0.420	0.327	0.366	0.335	0.370	1.940	1.070
	720	**0.388**	**0.389**	0.399	0.397	0.423	0.411	0.489	0.482	0.452	0.421	0.489	0.482	0.408	0.398	0.526	0.508	0.428	0.425	0.423	0.420	2.914	1.276
	Avg	**0.272**	**0.317**	0.280	0.324	0.295	0.336	0.316	0.365	0.302	0.334	0.308	0.364	0.286	0.328	0.342	0.392	0.308	0.354	0.315	0.358	1.662	0.917
ETTh1	96	**0.368**	**0.391**	0.377	0.396	0.385	0.405	0.398	0.409	0.399	0.418	0.381	0.416	0.387	0.395	0.389	0.404	0.391	0.433	0.449	0.465	1.028	0.778
	192	**0.424**	**0.422**	0.437	0.425	0.440	0.437	0.451	0.442	0.452	0.451	0.497	0.489	0.439	0.425	0.442	0.440	0.418	0.448	0.459	0.469	1.010	0.789
	336	**0.467**	**0.441**	0.486	0.449	0.480	0.457	0.501	0.472	0.488	0.469	0.589	0.555	0.482	0.447	0.488	0.467	0.487	0.484	0.511	0.500	0.908	0.743
	720	**0.465**	**0.463**	0.488	0.467	0.504	0.492	0.608	0.571	0.549	0.515	0.665	0.617	0.484	0.471	0.505	0.502	0.494	0.514	0.488	0.498	0.987	0.785
	Avg	**0.431**	**0.429**	0.447	0.434	0.452	0.448	0.489	0.474	0.472	0.463	0.533	0.519	0.448	0.435	0.456	0.453	0.447	0.470	0.477	0.483	0.983	0.774
ETTh2	96	**0.282**	**0.330**	0.293	0.344	0.301	0.349	0.315	0.374	0.321	0.358	0.351	0.398	0.291	0.340	0.330	0.383	0.351	0.391	0.355	0.397	1.485	0.959
	192	**0.359**	**0.381**	0.372	0.391	0.383	0.397	0.466	0.467	0.418	0.417	0.492	0.489	0.376	0.392	0.439	0.450	0.456	0.456	0.478	0.471	4.218	1.585
	336	**0.394**	**0.414**	0.420	0.433	0.425	0.432	0.522	0.502	0.464	0.454	0.656	0.582	0.417	0.427	0.589	0.538	0.477	0.492	0.459	0.469	2.775	1.361
	720	**0.400**	**0.427**	0.421	0.439	0.436	0.448	0.792	0.643	0.434	0.450	0.981	0.718	0.429	0.446	0.757	0.626	0.522	0.505	0.499	0.502	2.274	1.257
	Avg	**0.359**	**0.388**	0.377	0.402	0.386	0.407	0.524	0.496	0.409	0.420	0.620	0.546	0.378	0.401	0.529	0.499	0.452	0.461	0.448	0.460	2.688	1.291
ECL	96	**0.145**	**0.235**	0.161	0.258	0.150	0.242	0.180	0.266	0.170	0.272	0.170	0.281	0.197	0.274	0.197	0.282	0.187	0.302	0.189	0.304	0.253	0.350
	192	**0.159**	**0.249**	0.174	0.269	0.168	0.259	0.184	0.272	0.183	0.282	0.185	0.297	0.197	0.277	0.197	0.286	0.207	0.322	0.271	0.371	0.262	0.356
	336	**0.173**	**0.264**	0.194	0.290	0.182	0.274	0.199	0.290	0.203	0.302	0.190	0.298	0.212	0.292	0.209	0.301	0.211	0.326	0.243	0.352	0.269	0.363
	720	**0.203**	**0.292**	0.235	0.319	0.214	0.304	0.234	0.322	0.294	0.366	0.221	0.329	0.254	0.325	0.245	0.334	0.253	0.361	0.295	0.388	0.277	0.365
	Avg	**0.170**	**0.260**	0.191	0.284	0.179	0.270	0.199	0.288	0.212	0.306	0.192	0.302	0.215	0.292	0.212	0.301	0.214	0.328	0.249	0.354	0.265	0.358
Traffic	96	**0.393**	**0.265**	0.461	0.327	0.397	0.271	0.531	0.323	0.590	0.316	0.498	0.298	0.646	0.386	0.649	0.397	0.588	0.367	0.575	0.356	0.689	0.396
	192	**0.410**	**0.275**	0.470	0.326	0.416	0.279	0.519	0.321	0.624	0.336	0.521	0.309	0.599	0.362	0.598	0.371	0.613	0.377	0.647	0.394	0.710	0.388
	336	**0.421**	**0.280**	0.492	0.338	0.429	0.286	0.529	0.327	0.641	0.345	0.529	0.314	0.606	0.363	0.605	0.373	0.640	0.398	0.694	0.446	0.687	0.366
	720	**0.451**	**0.298**	0.521	0.353	0.462	0.303	0.573	0.346	0.670	0.356	0.567	0.326	0.643	0.383	0.646	0.395	0.718	0.450	0.731	0.468	0.681	0.366
	Avg	**0.419**	**0.280**	0.486	0.336	0.426	0.285	0.538	0.330	0.631	0.338	0.529	0.312	0.624	0.373	0.625	0.384	0.640	0.398	0.662	0.416	0.692	0.379
Weather	96	**0.169**	0.219	0.180	0.220	0.171	**0.210**	0.174	0.228	0.183	0.229	0.179	0.244	0.192	0.232	0.194	0.253	0.235	0.310	0.233	0.306	0.423	0.448
	192	**0.210**	**0.258**	0.222	0.258	0.246	0.278	0.213	0.266	0.242	0.276	0.242	0.310	0.240	0.270	0.238	0.296	0.295	0.353	0.286	0.347	0.664	0.585
	336	**0.259**	**0.297**	0.283	0.301	0.296	0.313	0.270	0.316	0.293	0.312	0.273	0.330	0.292	0.307	0.282	0.332	0.364	0.397	0.346	0.385	0.848	0.686
	720	**0.327**	0.349	0.358	**0.348**	0.362	0.353	0.337	0.362	0.366	0.361	0.360	0.399	0.364	0.353	0.347	0.385	0.411	0.429	0.412	0.420	0.861	0.685
	Avg	**0.241**	**0.280**	0.261	0.282	0.269	0.289	0.249	0.293	0.271	0.295	0.264	0.321	0.272	0.291	0.265	0.317	0.326	0.372	0.319	0.365	0.699	0.601
PEMS03	12	**0.070**	**0.176**	0.081	0.191	0.072	0.179	0.085	0.198	0.094	0.201	0.096	0.217	0.117	0.226	0.105	0.220	0.108	0.229	0.233	0.366	0.106	0.206
	24	**0.087**	**0.198**	0.121	0.240	0.104	0.217	0.129	0.244	0.116	0.221	0.095	0.210	0.233	0.322	0.183	0.297	0.131	0.255	0.405	0.485	0.117	0.221
	36	**0.105**	**0.219**	0.180	0.292	0.137	0.251	0.173	0.286	0.134	0.237	0.107	0.223	0.379	0.418	0.258	0.361	0.159	0.285	0.327	0.415	0.127	0.233
	48	**0.124**	**0.238**	0.201	0.316	0.174	0.285	0.207	0.315	0.161	0.262	0.125	0.242	0.535	0.516	0.319	0.410	0.209	0.331	0.679	0.634	0.139	0.245
	Avg	**0.097**	**0.208**	0.146	0.260	0.122	0.233	0.149	0.261	0.126	0.230	0.106	0.223	0.316	0.370	0.216	0.322	0.152	0.275	0.411	0.475	0.122	0.226
PEMS08	12	**0.081**	**0.183**	0.091	0.199	0.084	0.187	0.096	0.205	0.111	0.208	0.161	0.274	0.121	0.233	0.113	0.225	0.163	0.258	0.232	0.334	0.204	0.232
	24	**0.117**	**0.218**	0.138	0.245	0.123	0.227	0.151	0.258	0.139	0.232	0.127	0.237	0.232	0.325	0.199	0.302	0.197	0.288	0.545	0.550	0.232	0.251
	36	**0.157**	**0.253**	0.199	0.303	0.170	0.268	0.203	0.303	0.168	0.260	0.148	0.252	0.376	0.427	0.295	0.371	0.241	0.326	0.379	0.436	0.246	0.263
	48	0.207	0.294	0.255	0.338	0.218	0.306	0.247	0.334	**0.189**	**0.272**	0.175	0.270	0.543	0.527	0.389	0.429	0.302	0.375	0.531	0.502	0.278	0.297
	Avg	**0.141**	**0.237**	0.171	0.271	0.149	0.247	0.174	0.275	0.152	0.243	0.153	0.258	0.318	0.378	0.249	0.332	0.226	0.312	0.422	0.456	0.240	0.261
1^{st} Count		**43**	**42**	0	1	0	1	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0

Note: We fix the input length as 96 following [27]. **Bold** typeface highlights the top performance for each metric, while underlined text denotes the second-best results. *Avg* indicates the results averaged over forecasting lengths: T=96, 192, 336 and 720.

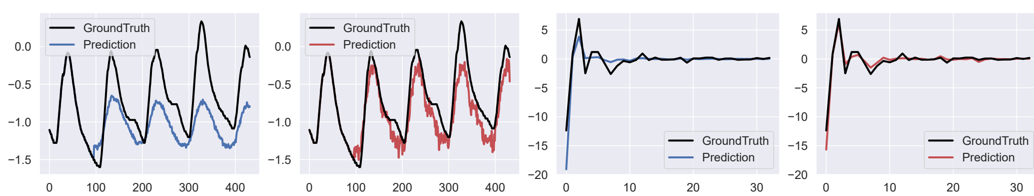
Table 8: The comprehensive results on the short-term forecasting task.

Models	Time-o1 (Ours)			Fredformer (2024)			iTransformer (2024)			FreTS (2023)			MICN (2023)			DLinear (2023)			Fedformer (2023)		
Metric	SMAPE	MASE	OWA	SMAPE	MASE	OWA	SMAPE	MASE	OWA	SMAPE	MASE	OWA	SMAPE	MASE	OWA	SMAPE	MASE	OWA	SMAPE	MASE	OWA
Yearly	**13.485**	**3.010**	**0.791**	13.509	3.028	0.794	13.797	3.143	0.818	13.576	3.068	0.801	14.594	3.392	0.873	14.307	3.094	0.827	13.648	3.089	0.806
Quarterly	**10.105**	**1.180**	**0.889**	10.140	1.185	0.893	10.503	1.248	0.932	10.361	1.223	0.916	11.417	1.385	1.023	10.500	1.237	0.928	10.612	1.246	0.936
Monthly	**12.649**	**0.930**	**0.875**	12.696	0.931	0.878	13.227	1.013	0.935	13.088	0.990	0.919	13.834	1.080	0.987	13.362	1.007	0.937	14.181	1.105	1.011
Others	4.852	3.274	1.027	4.848	**3.230**	**1.019**	5.101	3.419	1.076	5.563	3.71	1.17	6.137	4.201	1.308	5.12	3.649	1.114	**4.823**	3.243	1.019
Average	**11.841**	**1.585**	**0.851**	11.879	1.590	0.854	12.298	1.68	0.893	12.169	1.66	0.883	13.044	1.841	0.962	12.48	1.674	0.898	12.734	1.702	0.914
1st Count	**4**	**4**	**4**	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

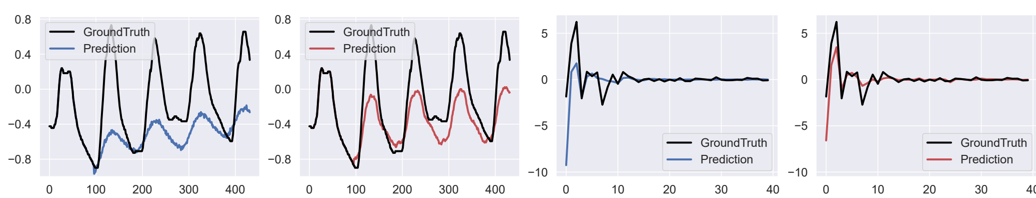
Note: **Bold** typeface highlights the top performance for each metric, while underlined text denotes the second-best results. *Avg* indicates the results averaged over forecasting lengths: yearly, quarterly, and monthly.



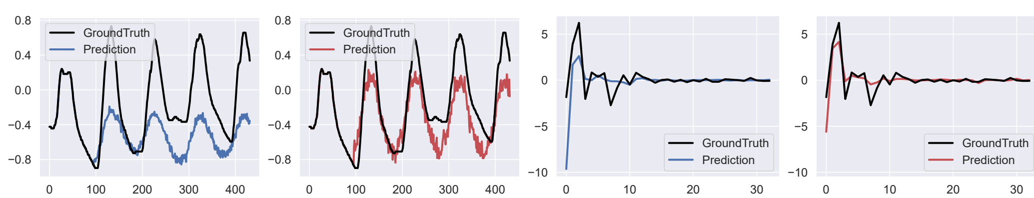
(a) Fredformer with ETTm2 case 1



(b) iTransformer with ETTm2 case 1

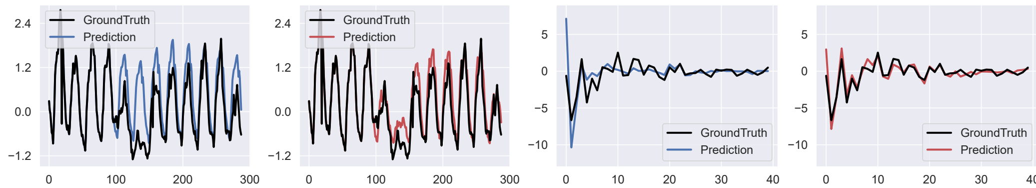


(c) Fredformer with ETTm2 case 2

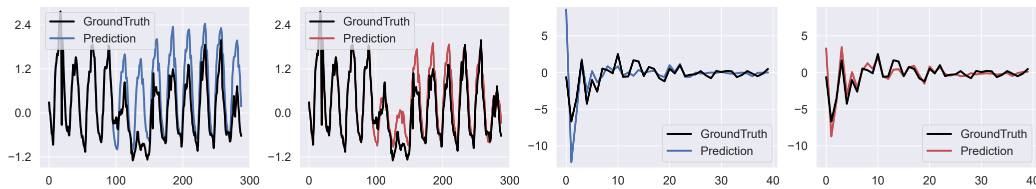


(d) iTransformer with ETTm2 case 2

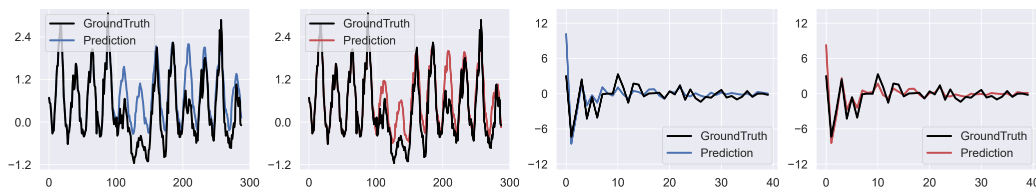
Figure 6: The forecast sequences generated with DF and Time-o1. The forecast length is set to 336 and the experiment is conducted on ETTm2.



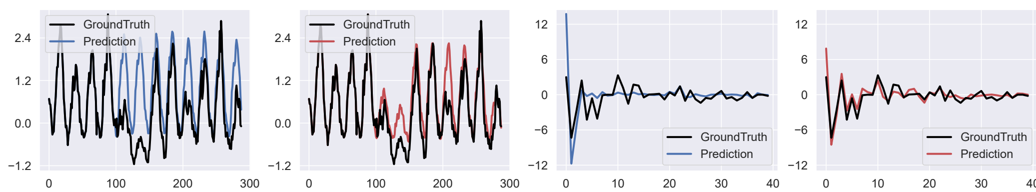
(a) Fredformer with ECL case 1



(b) iTransformer with ECL case 1

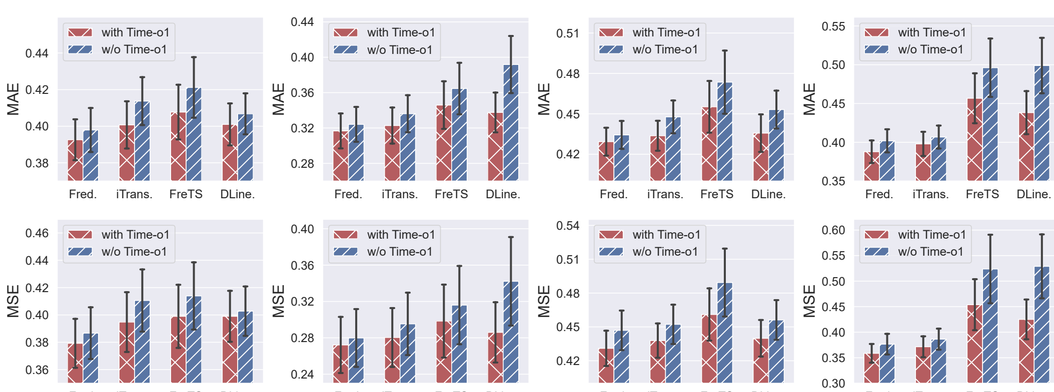


(c) Fredformer with ECL case 2



(d) iTransformer with ECL case 2

Figure 7: The forecast sequences generated with DF and Time-o1. The forecast length is set to 192 and the experiment is conducted on ECL.



(a) ETTm1 (b) ETTm2 (c) ETTh1 (d) ETTh2

Figure 8: Performance of different forecast models with and without Time-o1. The forecast errors are averaged over forecast lengths and the error bars represent 50% confidence intervals.

Table 9: Varying transformation results.

Trans		PCA		RPCA		SVD		FA		DF	
Metrics		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ECL	96	**0.1449**	**0.2348**	0.1450	0.2349	0.1450	0.2350	0.1478	0.2385	0.1500	0.2415
	192	**0.1592**	**0.2487**	0.1594	0.2487	0.1595	0.2490	0.1619	0.2517	0.1681	0.2591
	336	0.1731	0.2645	0.1732	0.2646	**0.1730**	**0.2643**	0.1789	0.2711	0.1823	0.2744
	720	**0.2033**	**0.2920**	0.2066	0.2960	0.2214	0.3066	0.2095	0.2975	0.2145	0.3035
	Avg	**0.1701**	**0.2600**	0.1710	0.2611	0.1747	0.2637	0.1745	0.2647	0.1787	0.2696
Weather	96	**0.1692**	**0.2185**	0.1715	0.2199	0.1723	0.2223	0.1717	0.2247	0.1737	0.2277
	192	**0.2102**	**0.2575**	0.2116	0.2590	0.2116	0.2597	0.2125	0.2636	0.2128	0.2661
	336	**0.2586**	**0.2971**	0.2631	0.3072	0.2676	0.3110	0.2613	0.2997	0.2705	0.3159
	720	**0.3271**	**0.3487**	0.3303	0.3581	0.3394	0.3681	0.3354	0.3610	0.3372	0.3623
	Avg	**0.2413**	**0.2805**	0.2441	0.2860	0.2477	0.2903	0.2452	0.2872	0.2486	0.2930

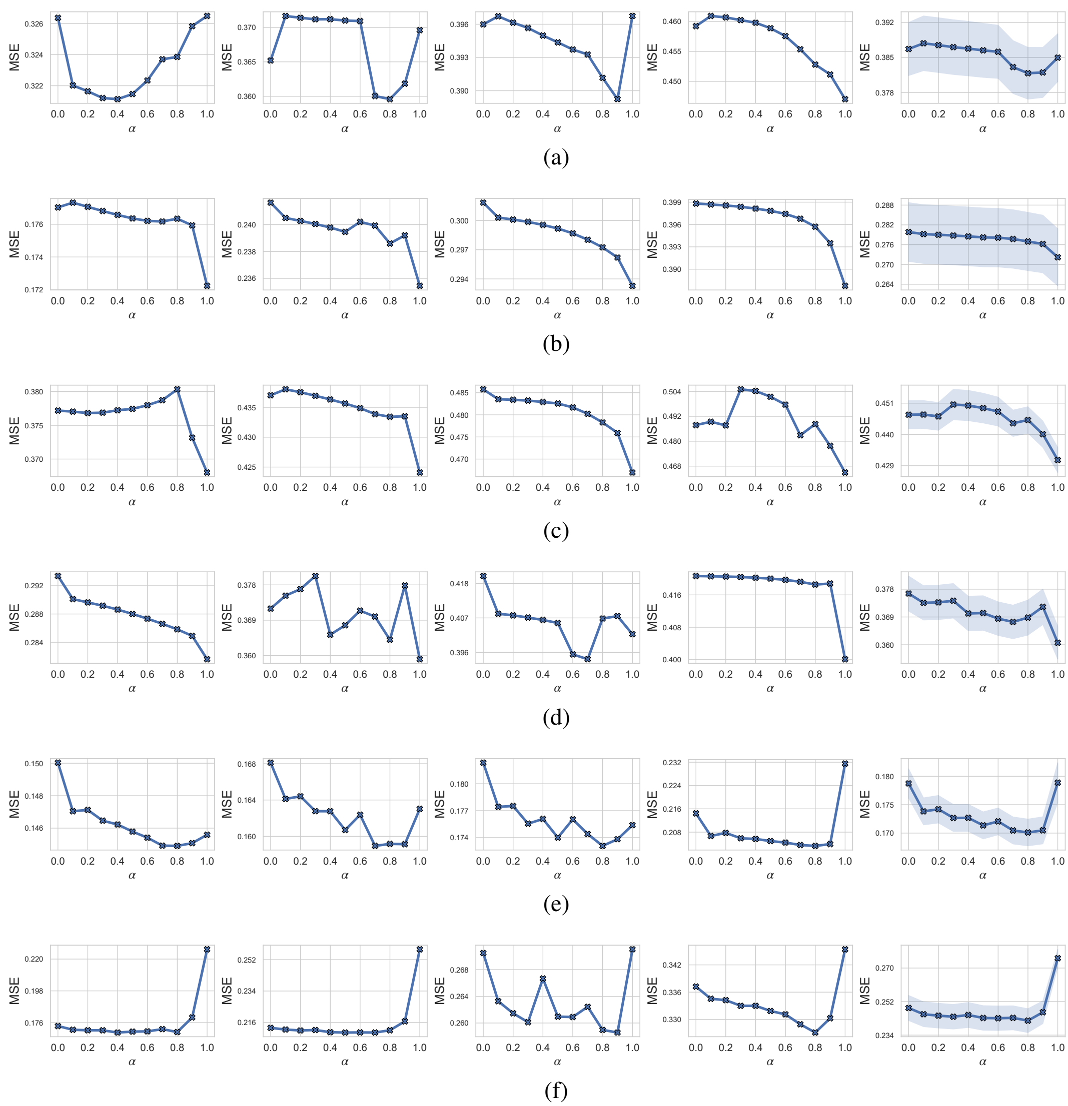
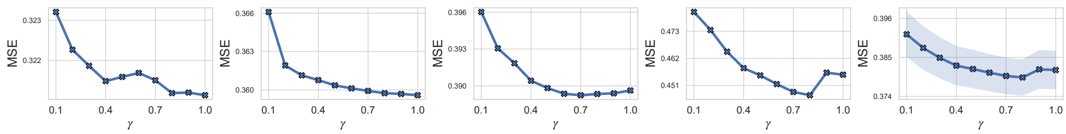
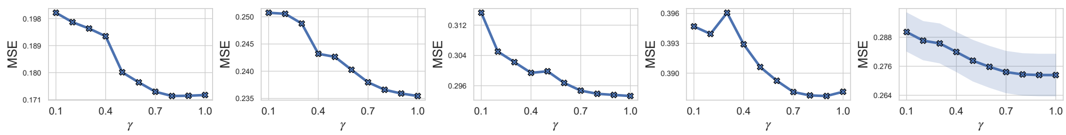


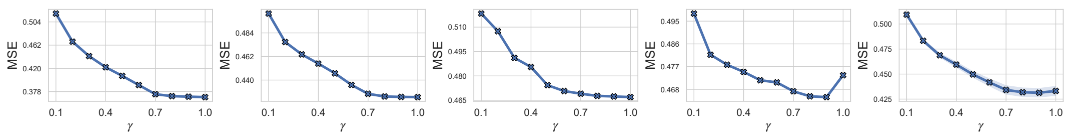
Figure 9: Time-o1 improves Fredformer performance given a wide range of transformed loss strength α. These experiments are conducted on ETTh1 (a), ETTh2 (b), ETTm1 (c), ETTm2 (d), ECL (e), Weather (f) datasets. Different columns correspond to different forecast lengths (from left to right: 96, 192, 336, 720, and their average with shaded areas being 15% confidence intervals).



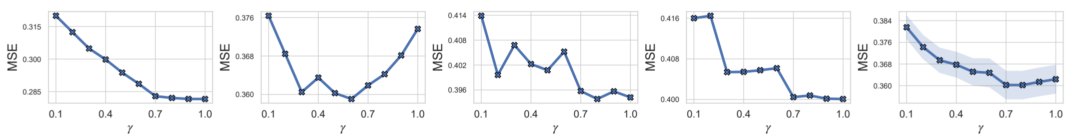
(a)



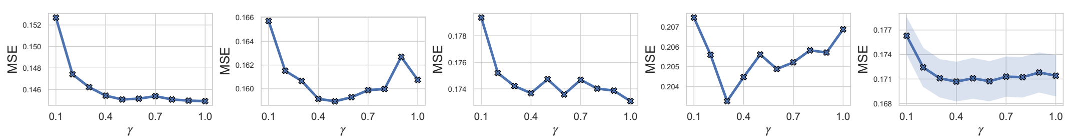
(b)



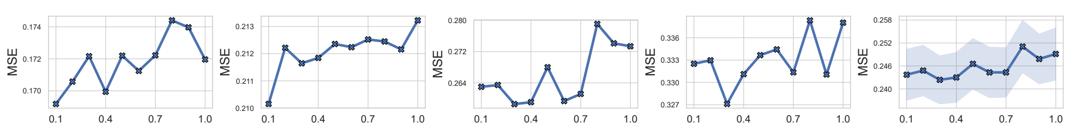
(c)



(d)



(e)



(f)

Figure 10: Time-o1 improves Fredformer performance given a wide range of rank ratio γ. These experiments are conducted on ETTh1 (a), ETTh2 (b), ETTm1 (c), ETTm2 (d), ECL (e), and Weather (f) datasets. Different columns correspond to different forecast lengths (from left to right: 96, 192, 336, 720, and their average with shaded areas being 15% confidence intervals).

Table 10: Comparable results with different learning objectives.

Loss		**Time-o1**		FreDF		Koopman		Dilate		Soft-DTW		DPTA		DF	
Metrics		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Forecast model: FredFormer															
ETTm1	96	0.321	0.357	0.326	0.355	0.335	0.368	0.337	0.367	0.332	0.363	0.332	0.364	0.326	0.361
	192	0.360	0.378	0.363	0.380	0.366	0.384	0.364	0.384	0.370	0.386	0.370	0.386	0.365	0.382
	336	0.389	0.400	0.392	0.400	0.399	0.408	0.397	0.406	0.406	0.409	0.409	0.410	0.396	0.404
	720	0.447	0.435	0.455	0.440	0.456	0.441	0.457	0.443	0.478	0.450	0.476	0.448	0.459	0.444
	Avg	0.379	0.393	0.384	0.394	0.389	0.400	0.389	0.400	0.397	0.402	0.396	0.402	0.387	0.398
ETTh1	96	0.368	0.391	0.370	0.392	0.375	0.397	0.378	0.399	0.376	0.398	0.378	0.399	0.377	0.396
	192	0.424	0.422	0.436	0.437	0.438	0.434	0.439	0.435	0.439	0.435	0.438	0.433	0.437	0.425
	336	0.467	0.441	0.473	0.443	0.473	0.455	0.481	0.453	0.484	0.455	0.486	0.455	0.486	0.449
	720	0.465	0.463	0.474	0.466	0.523	0.487	0.516	0.482	0.542	0.510	0.538	0.510	0.488	0.467
	Avg	0.431	0.429	0.438	0.434	0.452	0.443	0.453	0.442	0.460	0.449	0.460	0.449	0.447	0.434
ECL	96	0.151	0.245	0.152	0.247	0.166	0.263	0.158	0.253	0.168	0.266	0.158	0.253	0.161	0.258
	192	0.166	0.256	0.166	0.257	0.174	0.267	0.170	0.263	0.218	0.313	0.216	0.307	0.174	0.269
	336	0.181	0.274	0.183	0.278	0.188	0.280	0.190	0.286	0.197	0.291	0.199	0.295	0.194	0.290
	720	0.213	0.304	0.216	0.304	0.232	0.318	0.229	0.316	0.240	0.322	0.235	0.322	0.235	0.319
	Avg	0.178	0.270	0.179	0.272	0.190	0.282	0.187	0.280	0.206	0.298	0.202	0.294	0.191	0.284
Weather	96	0.171	0.208	0.174	0.213	0.174	0.214	0.173	0.214	0.173	0.213	0.179	0.219	0.180	0.220
	192	0.219	0.253	0.219	0.254	0.220	0.256	0.225	0.260	0.220	0.255	0.223	0.257	0.222	0.258
	336	0.277	0.295	0.278	0.296	0.280	0.298	0.280	0.299	0.281	0.296	0.281	0.298	0.283	0.301
	720	0.353	0.346	0.354	0.347	0.354	0.347	0.355	0.348	0.369	0.355	0.356	0.347	0.358	0.348
	Avg	0.255	0.276	0.256	0.277	0.257	0.279	0.258	0.280	0.261	0.280	0.260	0.280	0.261	0.282
Forecast model: itranFormer															
ETTm1	96	0.323	0.358	0.334	0.365	0.350	0.382	0.342	0.376	0.339	0.373	0.341	0.375	0.338	0.372
	192	0.371	0.388	0.381	0.390	0.389	0.400	0.381	0.396	0.383	0.395	0.383	0.395	0.382	0.396
	336	0.408	0.407	0.417	0.412	0.425	0.423	0.418	0.418	0.429	0.423	0.429	0.423	0.427	0.424
	720	0.477	0.450	0.489	0.453	0.489	0.458	0.487	0.457	0.516	0.469	0.512	0.467	0.496	0.463
	Avg	0.395	0.401	0.405	0.405	0.413	0.416	0.407	0.412	0.417	0.415	0.416	0.415	0.411	0.414
ETTh1	96	0.378	0.393	0.378	0.395	0.392	0.411	0.385	0.405	0.387	0.405	0.386	0.405	0.385	0.405
	192	0.428	0.423	0.428	0.423	0.446	0.442	0.440	0.437	0.443	0.439	0.441	0.439	0.440	0.437
	336	0.473	0.450	0.470	0.447	0.483	0.461	0.480	0.457	0.494	0.464	0.489	0.462	0.480	0.457
	720	0.473	0.469	0.490	0.484	0.501	0.491	0.504	0.492	0.557	0.520	0.538	0.509	0.504	0.492
	Avg	0.438	0.434	0.442	0.437	0.455	0.451	0.452	0.448	0.470	0.457	0.463	0.454	0.452	0.448
ECL	96	0.145	0.235	0.149	0.238	0.151	0.243	0.150	0.241	0.149	0.241	0.149	0.240	0.150	0.242
	192	0.159	0.249	0.163	0.251	0.167	0.257	0.168	0.259	0.164	0.255	0.166	0.257	0.168	0.259
	336	0.173	0.264	0.179	0.268	0.182	0.275	0.181	0.274	0.180	0.274	0.180	0.272	0.182	0.274
	720	0.203	0.292	0.212	0.297	0.212	0.300	0.212	0.300	0.207	0.296	0.212	0.300	0.214	0.304
	Avg	0.170	0.260	0.176	0.264	0.178	0.269	0.178	0.269	0.175	0.266	0.177	0.267	0.179	0.270
Weather	96	0.163	0.202	0.170	0.208	0.206	0.257	0.208	0.259	0.207	0.252	0.209	0.258	0.171	0.210
	192	0.214	0.248	0.219	0.252	0.264	0.300	0.252	0.285	0.264	0.303	0.258	0.291	0.246	0.278
	336	0.274	0.294	0.279	0.296	0.309	0.326	0.311	0.328	0.314	0.333	0.312	0.331	0.296	0.313
	720	0.351	0.344	0.358	0.347	0.377	0.369	0.374	0.364	0.384	0.377	0.383	0.373	0.362	0.353
	Avg	0.251	0.272	0.257	0.276	0.289	0.313	0.286	0.309	0.292	0.316	0.291	0.313	0.269	0.289

Table 11: Varying input sequence length results on the Weather dataset.

Models			**Time-o1**		iTransformer		**Time-o1**		PatchTST	
Metrics			MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Input sequence length	96	96	0.163	0.202	0.171	0.210	0.175	0.213	0.200	0.244
		192	0.214	0.248	0.246	0.278	0.224	0.257	0.229	0.263
		336	0.274	0.294	0.296	0.313	0.276	0.296	0.287	0.303
		720	0.351	0.344	0.362	0.353	0.353	0.346	0.363	0.353
		Avg	0.250	0.272	0.269	0.289	0.257	0.278	0.270	0.291
	192	96	0.163	0.205	0.168	0.215	0.158	0.199	0.164	0.208
		192	0.210	0.248	0.213	0.253	0.204	0.242	0.225	0.269
		336	0.259	0.287	0.265	0.294	0.257	0.286	0.287	0.308
		720	0.334	0.338	0.341	0.345	0.332	0.337	0.341	0.345
		Avg	0.241	0.270	0.247	0.277	0.238	0.266	0.254	0.283
	336	96	0.157	0.203	0.162	0.213	0.150	0.196	0.156	0.206
		192	0.199	0.246	0.211	0.256	0.196	0.241	0.222	0.277
		336	0.251	0.287	0.260	0.295	0.246	0.282	0.251	0.285
		720	0.324	0.338	0.332	0.341	0.320	0.333	0.327	0.338
		Avg	0.233	0.268	0.241	0.276	0.228	0.263	0.239	0.277
	720	96	0.161	0.213	0.172	0.225	0.152	0.201	0.154	0.207
		192	0.205	0.250	0.220	0.268	0.198	0.248	0.205	0.254
		336	0.254	0.292	0.282	0.311	0.248	0.284	0.248	0.288
		720	0.318	0.339	0.337	0.351	0.313	0.335	0.317	0.339
		Avg	0.235	0.274	0.253	0.289	0.228	0.267	0.231	0.272

Table 12: Experimental results ($\mathrm{mean}_{\pm\mathrm{std}}$) with varying seeds (2021-2025).

Dataset	ECL				Weather			
Models	**Time-o1**		DF		**Time-o1**		DF	
Metrics	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
96	$0.145_{\pm 0.000}$	$0.235_{\pm 0.000}$	$0.150_{\pm 0.001}$	$0.242_{\pm 0.001}$	$0.164_{\pm 0.001}$	$0.203_{\pm 0.001}$	$0.190_{\pm 0.012}$	$0.232_{\pm 0.014}$
192	$0.160_{\pm 0.001}$	$0.249_{\pm 0.001}$	$0.166_{\pm 0.002}$	$0.257_{\pm 0.002}$	$0.216_{\pm 0.002}$	$0.250_{\pm 0.001}$	$0.240_{\pm 0.011}$	$0.272_{\pm 0.010}$
336	$0.174_{\pm 0.002}$	$0.266_{\pm 0.002}$	$0.181_{\pm 0.001}$	$0.273_{\pm 0.001}$	$0.274_{\pm 0.001}$	$0.294_{\pm 0.001}$	$0.293_{\pm 0.003}$	$0.310_{\pm 0.003}$
720	$0.205_{\pm 0.001}$	$0.293_{\pm 0.001}$	$0.216_{\pm 0.004}$	$0.303_{\pm 0.003}$	$0.353_{\pm 0.002}$	$0.344_{\pm 0.001}$	$0.361_{\pm 0.002}$	$0.352_{\pm 0.001}$
Avg	$0.171_{\pm 0.001}$	$0.261_{\pm 0.001}$	$0.178_{\pm 0.001}$	$0.269_{\pm 0.001}$	$0.252_{\pm 0.001}$	$0.273_{\pm 0.001}$	$0.271_{\pm 0.003}$	$0.292_{\pm 0.003}$

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims made in the abstract and introduction accurately reflect our paper's contributions and scope.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Our limitations are discussed in conclusions.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: All proofs of theoretical results are involved in Appendix.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We detail our training and evaluation protocols in the experimental setting section.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We use open access data, and the code is provided in an anonymous link.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We detail our training and evaluation protocols in the experimental setting section.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: In Figure 3 which are crucial for evaluating the efficacy of TransDF, we report the error bars to make the results more rigorous and comprehensive.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We report the type of compute workers, memory in Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: Since it is an algorithm-oriented research, there is no societal impact of the work performed.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: This paper does not involve these issues.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The creators or original owners of assets (e.g., code, data, models), used in the paper, are properly credited. The license and terms of use are explicitly mentioned and properly respected.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: We do not introduce new datasets and benchmarks in the paper.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: We use open-access datasets and do not involve human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: We use open-access datasets and do not involve human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: We do not involve LLMs as any important, original, or non-standard components in this work.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.