TransDF: Time-Series Forecasting Needs Transformed Label Alignment

Anonymous Author(s)
Affiliation
Address
email

Abstract

Training time-series forecasting 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 Transform-enhanced Direct Forecast (TransDF), which transforms the label sequence into decorrelated components with discriminated significance. Models are trained to align the most significant components, thereby effectively mitigating label autocorrelation and reducing task amount. Extensive experiments demonstrate that TransDF achieves state-of-the-art performance and is compatible with various forecasting models. Code is available at `https://anonymous.4open.science/r/TransDF-88CF`.

1 Introduction

Time-series forecasting involves predicting future data from historical observations and has been applied across diverse domains, such as weather forecasting in meteorology [1], traffic flow prediction in transportation [42], and process monitoring in manufacturing [33]. To build effective forecast models, two questions 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. The key challenge lies in exploiting the autocorrelation in the historical sequences. To this end, various architectures have been proposed [28, 20], such as recurrent neural networks [10], convolutional neural networks [37, 35], and graph neural networks [39]. The current progress is marked by a debate between Transformers and simple linear models. Transformers, equipped with self-attention mechanisms, offer superior scalability [23, 25, 27]. In contrast, linear models, which encapsulate temporal dynamics using linear layers, are straightforward to implement and often demonstrate strong performance [41]. These advancements showcase the rapid evolution in neural architecture design for time-series forecasting.

In contrast, the design of learning objectives has received less attention. Most existing methods follow the standard direct forecast (DF) paradigm, using the temporal mean squared error (TMSE) between forecast and label sequences as the learning objective [23, 25]. While effective for many tasks, this approach has two critical issues in the context of time-series forecasting. First, the TMSE objective is biased against the true label sequence likelihood 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 [43, 21]. These challenges present unique challenges in designing learning objectives for time-series forecasting.

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

To handle these challenges, we propose the *Transform-enhanced Direct Forecast* (TransDF), a simple yet effective refinement of the direct forecast paradigm. The key idea is to transform the label sequence into decorrelated components ranked by significance. By aligning the most significant decorrelated components, TransDF mitigates label autocorrelation and reduces the number of tasks, while retaining the efficiency and implementation simplicity of DF.

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 TransDF, 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 TransDF's efficacy, consistently boosting the performance of state-of-the-art forecasting models across diverse datasets.

2 Preliminaries

This paper focuses on the time-series forecasting problem. 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 forecasting models [34, 16, 15], rather than on the design of neural network architectures to implement the forecast models [23, 41].

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 $g : \mathbb{R}^{\mathrm{H}\times\mathrm{D}} \to \mathbb{R}^{\mathrm{T}\times\mathrm{D}}$ 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 have been pervasive in encoding historical sequences for their capability of automating feature interactions and capturing nonlinear correlations. Notable examples include RNNs (e.g., S4 [10], Mamba), CNNs (e.g., TimesNet [37]), and GNNs (e.g., MTGNN [24]), each tailored to encode the dynamics within input sequences. The current progress centers on the comparison between Transformer-based and MLP-based architectures. Transformers (e.g., PatchTST [25], iTransformer [23]) exhibit substantial scalability with increasing data size but entail high computational costs. In contrast, MLPs (e.g., DLinear [41], TimeMixer [36]) 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 [19, 41, 23]. The learning objective is typically the temporal mean squared error (TMSE) between the forecast and label sequences, given by:

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

which is widely employed in recent studies (e.g., FreTS [40], iTransformer [23], FredFormer [27]). However, this objective has been shown to be biased due to the autocorrelation present in the label sequence [34]. To address this bias, one line of research advocates for shape alignment between the forecast and label sequences to exploit autocorrelation (e.g., Dilate [15], Soft-DTW [5] and DPTA [29]). 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, which reduces bias with theoretical guarantees [34].

3 Methodology

3.1 Motivation

The learning objective is a fundamental component in training effective forecasting models, yet its importance remains underexplored. Existing approaches predominantly employ the TMSE in (1) as the objective [23, 25]. 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 [41], 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 $\|Y - \hat{Y}\|_{\Sigma^{-1}}^2 = (Y - \hat{Y})^\top \Sigma^{-1}(Y - \hat{Y})$. The bias vanishes if different steps in Y are decorrelated.[1]

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 [33]) and transportation (enabling proactive traffic management [42]). 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 [43, 21], as gradients from different tasks often conflict [43, 21], impeding convergence and leading to suboptimal model performance.

Designing effective learning objectives to handle the two limitations is challenging. The recent pioneering work **FreDF** [34] 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 [34]). 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 fails to address the autocorrelation bias and the optimization difficulty.*

Given the critical role of objective in training forecast models and the limitations of existing approaches, 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?*

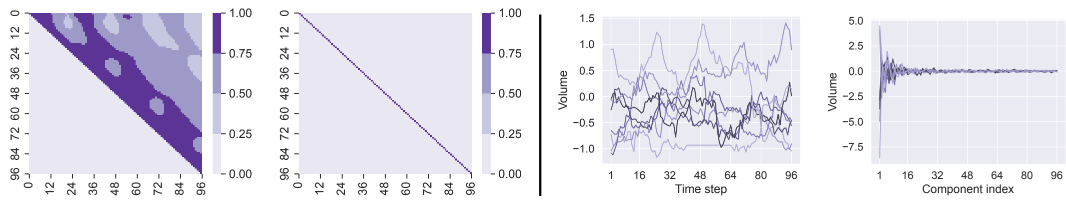
3.2 Transforming label sequence with optimized projection matrix

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[2].

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

[1] The pioneering work [34] identifies the bias under the first-order Markov assumption on the label sequence. This study generalizes this bias without the first-order Markov assumption.

[2] In the subsequent paragraphs, we use the univariate case with $\mathrm{D} = 1$ for clarity. In the multivariate case, label sequences are concatenated along the forecast horizon to produce decorrelated components.



(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).

calculated by solving the following problem:

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

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 escalates 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 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 [26, 11]. However, one key distinction warrants emphasis. Existing works dominantly employ principal component analysis on *input features* for obtaining informative representations [9, 7], in contrast, we apply 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 TransDF. The approach centers on extracting the latent components from the label sequence, then optimizing the forecasting model using the most significant components.

Algorithm 1 The workflow of TransDF.

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

Given an input historical sequence, the forecast model predicts a sequence $\hat{\mathbf{Y}}$. In line with prevailing preprocessing practices [23, 41, 27], 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 [34], 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 objectives calculated with the transformed components and the label sequence are fused following [34], with $0 \le \alpha \le 1$ controlling the relative contribution:

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

By projecting both forecasts and labels into a decorrelated latent space, TransDF effectively reduces autocorrelation bias. By focusing exclusively on the most significant components, TransDF reduces optimization difficulty with minimal information sacrifice. Key advantages of the canonical DF framework [23, 41], like efficient inference and multi-task learning capabilities, are preserved. Crucially, TransDF is model-agnostic, offering practitioners the flexibility to employ the most suitable forecasting model for each specific scenario.

4 Experiments

To demonstrate the efficacy of TransDF, there are six aspects empirically investigated:

1. **Performance:** *Does TransDF work?* We compare TransDF 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.2. Moreover, we compare it 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 TransDF, elucidating their roles in enhancing forecast accuracy.
3. **Generality:** *Does it support other forecasting models?* Section 4.5 verifies the adaptability of TransDF across different forecasting models, with additional results documented in Appendix E.3.

Table 1: Long-term forecasting performance.

Models	**PDF (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**	<u>0.387</u>	<u>0.398</u>	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**	<u>0.280</u>	<u>0.324</u>	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**	<u>0.447</u>	<u>0.434</u>	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**	<u>0.377</u>	0.402	0.386	0.407	0.524	0.496	0.409	0.420	0.620	0.546	0.378	<u>0.401</u>	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	<u>0.179</u>	<u>0.270</u>	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	<u>0.426</u>	<u>0.285</u>	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	<u>0.282</u>	0.269	0.289	<u>0.249</u>	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	<u>0.106</u>	<u>0.223</u>	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	<u>0.149</u>	<u>0.247</u>	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 [23]. **Bold** and <u>underlined</u> denote best and second-best results, respectively. *Avg* indicates average results over forecast horizons: T=96, 192, 336 and 720. TransDF employs the top-performing baseline on each dataset as its underlying forecasting model.

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 TransDF maintains efficacy across a broad range of parameter values.

6. **Efficiency:** *Is TransDF computationally expensive?* Section E.1 investigates the running cost of TransDF in diverse settings.

4.1 Setup

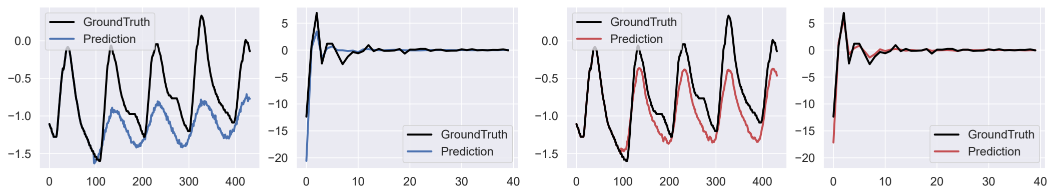
Datasets. In this work, we conduct experiments on public datasets following prior works [37]. For long-term forecasting, we use ETT (4 subsets), ECL, Traffic, Weather, and PEMS [23]. For short-term forecasting, we use M4. All datasets are split chronologically into training, validation, and testing sets. Detailed dataset statistics are provided in Appendix D.1.

Baselines. We compare TransDF against several established methods, grouped as: (1) Transformer-based methods: Transformer [30], Autoformer [38], FEDformer [44], iTransformer [23], and Fredformer [27]; (2) MLP-based methods: DLinear [41], TiDE [6], and FreTS [40]; and (3) other competitive models: TimesNet [37] and MICN [35].

Implementation. The baseline models are reproduced using the scripts provided by Fredformer [27]. Notably, we disable the drop-last trick to ensure fair comparison following Qiu et al. [28]. They are trained using the Adam [12] optimizer to minimize the TMSE loss. Datasets are split chronologically into training, validation, and test sets. Following the protocol outlined in the comprehensive benchmark [28], the dropping-last trick is disabled during the test phase. When integrating TransDF to enhance an established model, we adhere to the associated hyperparameter settings in the public benchmark [27, 23], only tuning α, γ and learning rate conservatively. Experiments are conducted on Intel(R) Xeon(R) Platinum 8383C CPUs and NVIDIA RTX H100 GPUs. More implementation details are provided in Appendix D.2.

4.2 Overall performance

Table 1 presents the long-term forecasting results. TransDF consistently improves base model performance. For example, on ETTh1, it reduces Fredformer's MSE by 0.016. Similar gains 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 TransDF: its decorrelation effect for debiased training, and its discrimination on significant components, which simplifies optimization.



(a) The generated forecast with DF. (b) The generated forecast with TransDF.

Figure 2: The visualization of forecast sequence generated by DF and TransDF. 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		**TransDF**		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.

Showcases. We visualize the forecast sequences and the generated components to showcase the improvements of TransDF 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, TransDF, 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 TransDF against other time-series learning objectives: FreDF [34], Koopman [14], Dilate [15], Soft-DTW [5], and DPTA [29]. 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 [15]. This phenomenon is rationalized by the fact that they do not mitigate the label correlation 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. TransDF directly addresses these two limitations of FreDF, leading to its superior overall performance.

4.4 Ablation studies

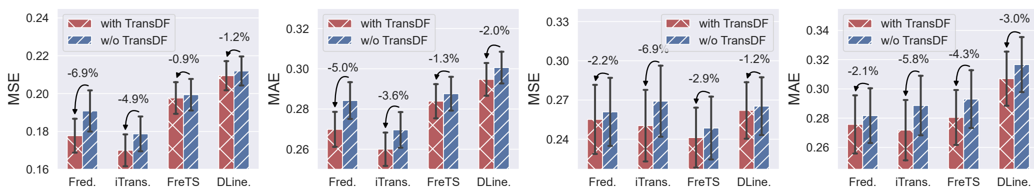
Table 3 presents an ablation study dissecting the contributions of critical factors in TransDF: the decorrelation effect and the task reduction effect. The main findings are summarized as follows.

- TransDF† 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.

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
TransDF†	✗	✓	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
TransDF‡	✓	✗	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
TransDF	✓	✓	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.



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

Figure 3: Improvement of TransDF 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.

- TransDF‡ 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.
- TransDF 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 TransDF with different transformation strategies and forecast models, to showcase the generality of TransDF. 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) [3], SVD [8], and factor analysis [13]. 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.

Table 4: Varying transformations results.

	ECL				Weather			
Trans	MSE	RI	MAE	RI	MSE	RI	MAE	RI
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: RI refers to the relative improvement (error reduction) over the baseline. **Bold** and underlined denote best and second-best results.

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**	<u>0.3959</u>	<u>0.3674</u>	**0.3943**	<u>0.2433</u>	<u>0.2849</u>
1	<u>0.3850</u>	**0.3933**	**0.3606**	<u>0.3890</u>	0.2753	0.3209

Note: **Bold** and <u>underlined</u> 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	<u>0.2437</u>	<u>0.2845</u>
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	<u>0.3814</u>	<u>0.3940</u>	<u>0.3624</u>	<u>0.3903</u>	0.2491	0.2924

Note: **Bold** and <u>underlined</u> denote best and second-best results.

Varying forecast models. We explore the versatility of TransDF in augmenting representative forecast models: Fredformer [27], iTransformer [23], FreTS [40], and DLinear [41]. As illustrated in Fig. 3, TransDF improves forecast performance in all cases. For instance, on the Weather dataset, iTransformer and FreTS with TransDF achieve substantial reductions in MSE—up to 6.9% and 2.9%, respectively. Further evidence of TransDF's versatility can be found in Appendix E.3. These results confirm TransDF'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 TransDF. The results are presented in Table 5 and Table 6. Additional trends across different datasets and forecast lengths are provided in Appendix E.4. 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, TransDF 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, TransDF 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 TransDF. 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 correlation due to the decorrelation between components and reduces task amount by discarding non-significant components. Experiments demonstrate that TransDF improves the performance of forecast models across diverse datasets.

Limitations & future works. In this work, we investigate the challenges of label correlation and exssive 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 TransDF 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.

References

[1] Kaifeng Bi, Lingxi Xie, Hengheng Zhang, Xin Chen, Xiaotao Gu, and Qi Tian. Accurate medium-range global weather forecasting with 3d neural networks. *Nature*, 619(7970):533–538, 2023.

[2] Michela Bia, Martin Huber, and Lukáš Lafférs. Double machine learning for sample selection models. *J. Bus. Econ. Stat.*, 42(3):958–969, 2024.

[3] 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.

[4] Victor Chernozhukov, Denis Chetverikov, Mert Demirer, Esther Duflo, Christian Hansen, Whitney Newey, and James Robins. Double/debiased machine learning for treatment and structural parameters: Double/debiased machine learning. *Econom. J.*, 21(1), 2018.

[5] Marco Cuturi and Mathieu Blondel. Soft-dtw: a differentiable loss function for time-series. In *International conference on machine learning*, pages 894–903. PMLR, 2017.

[6] Abhimanyu Das, Weihao Kong, Andrew Leach, Rajat Sen, and Rose Yu. Long-term forecasting with tide: Time-series dense encoder. *arXiv preprint arXiv:2304.08424*, 2023.

[7] 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.

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

[9] 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.

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

[11] Ian T Jolliffe. *Principal component analysis for special types of data*. Springer, 2002.

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

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

[14] 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.

[15] 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.

[16] Vincent Le Guen and Nicolas Thome. Probabilistic time series forecasting with shape and temporal diversity. In *Proc. Adv. Neural Inf. Process. Syst.*, volume 33, pages 4427–4440, 2020.

[17] Haoxuan Li, Kunhan Wu, Chunyuan Zheng, Yanghao Xiao, Hao Wang, Zhi Geng, Fuli Feng, Xiangnan He, and Peng Wu. Removing hidden confounding in recommendation: a unified multi-task learning approach. *Proc. Adv. Neural Inf. Process. Syst.*, 36:54614–54626, 2024.

[18] 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.

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

[20] 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.

[21] 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.

[22] 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.

[23] 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.

[24] 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.

[25] 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.

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

[27] 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.

[28] 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.

[29] Thanawin Rakthanmanon, Bilson Campana, Abdullah Mueen, Gustavo Batista, Brandon Westover, Qiang Zhu, Jesin Zakaria, and Eamonn Keogh. Searching and mining trillions of time series subsequences under dynamic time warping. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 262–270, 2012.

[30] 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.

[31] 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.*, 2023.

[32] Hao Wang, Zhichao Chen, Zhaoran Liu, Haozhe Li, Degui Yang, Xinggao Liu, and Haoxuan Li. Entire space counterfactual learning for reliable content recommendations. *IEEE Trans. Inf. Forensics Security*, pages 1–12, 2024.

[33] 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.*, 2024.

[34] Hao Wang, Lichen 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 *The Thirteenth International Conference on Learning Representations*, 2025.

[35] 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.

[36] Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Y Zhang, and Jun Zhou. Timemixer: Decomposable multiscale mixing for time series forecasting. In *Proc. Int. Conf. Learn. Represent.*, 2024.

[37] 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.

[38] 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.

[39] 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.

[40] 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.

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

[42] 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.

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

[44] 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.

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.