IdealTSF: Can Non-Ideal Data Contribute to Enhancing the Performance of Time Series Forecasting Models?

Hua Wang[1], **Jinghao Lu**[1], **Fan Zhang**[2*]

[1]School of Computer and Artificial Intelligence, Ludong University, Yantai, 264025, China
[2]School of Computer Science and Technology, Shandong Technology and Business University, Yantai, 264005, China
hwa229@163.com, ljh@m.ldu.edu.cn, zhangfan@sdtbu.edu.cn

Abstract

Deep learning has shown strong performance in time series forecasting tasks. However, issues such as missing values and anomalies in sequential data hinder its further development in prediction tasks. Previous research has primarily focused on extracting feature information from sequence data or addressing these suboptimal data as positive samples for knowledge transfer. A more effective approach would be to leverage these non-ideal negative samples to enhance event prediction. In response, this study highlights the advantages of non-ideal negative samples and proposes the IdealTSF framework, which integrates both ideal positive and negative samples for time series forecasting. IdealTSF consists of three progressive steps: pretraining, training, and optimization. It first pretrains the model by extracting knowledge from negative sample data, then transforms the sequence data into ideal positive samples during training. Additionally, a negative optimization mechanism with adversarial disturbances is applied. Extensive experiments demonstrate that negative sample data unlocks significant potential within the basic attention architecture for time series forecasting. Therefore, IdealTSF is particularly well-suited for applications with noisy samples or low-quality data.

Code — https://github.com/LuckyLJH/IdealTSF

Introduction

Time series forecasting tasks are ubiquitous across various domains, including economics (Granger and Newbold 2014), energy (Martín et al. 2010; Qian et al. 2019), transportation planning (Chen et al. 2001; Yin et al. 2021), weather forecasting (Wu et al. 2023), healthcare, and natural sciences. Over the past decade, deep learning methods (LeCun, Bengio, and Hinton 2015) have gained popularity in forecasting, often outperforming statistical methods such as ARIMA (Shumway and Stoffer 2017). However, until recently, deep learning methods for forecasting have primarily focused on selecting superior feature extraction techniques, utilizing specific training schemes, and employing foundational architectures such as CNNs (Wang et al. 2023; Wu et al. 2022; Hewage et al. 2020), RNNs (Lai et al. 2018; Qin et al. 2017), Transformers (Vaswani et al. 2017; Zhou et al. 2022; Nie et al. 2022), and MLP variants (Challu et al. 2023; Murad, Aktukmak, and Yilmaz 2025), with the aim of capturing the trend characteristics of time series data as effectively as possible.

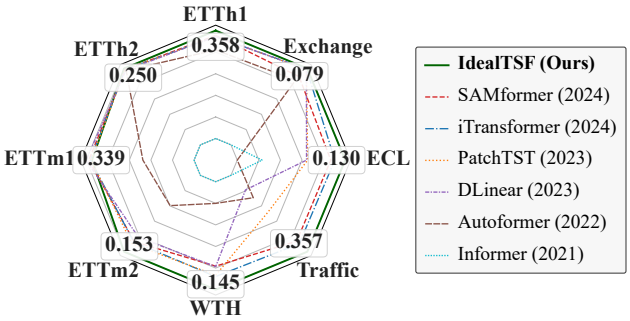

Figure 1: The performance of the model with a prediction length of 96.

Conventional time series analysis methods typically operate under the assumption that data are complete and devoid of anomalies. However, this assumption frequently fails to hold true in real-world applications (Schmidl, Wenig, and Papenbrock 2022; Qu et al. 2024). Common approaches to handling missing values, such as interpolation, typically presume a linear or smooth relationship between missing entries and observed data (Chen et al. 2023). Such methods tend to oversimplify the inherent complexity of time series data, neglecting potential nonlinear dependencies, abrupt fluctuations, and various non-stationary dynamics (Xu et al. 2022; Chen et al. 2024). For instance, in meteorological datasets, certain weather patterns may shift as a consequence of climate change, rendering simplistic interpolation techniques inadequate for capturing such evolving trends. In the presence of large-scale missing data, the efficacy of interpolation techniques becomes markedly limited, often failing to reconstruct the underlying data distribution accurately (Zeng et al. 2025a,b). Conversely, conventional outlier detection techniques frequently depend on statistical assumptions, such as normality of the data distribution (Zhang et al. 2025b,a; Yao, Li, and Xiao 2024). Such assumptions may lead to the misclassification of meaningful fluctuations

*Corresponding author
Copyright © 2026, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

as anomalies, thereby compromising the predictive performance of the model. For example, in power load forecasting, seasonal patterns or exceptional events—such as national holidays—can induce substantial fluctuations in load data (Xiao et al. 2024; Wang et al. 2025). Traditional anomaly detection methods may erroneously label these fluctuations as outliers, thereby overlooking critical trend information and introducing bias into forecasts (Xiao et al. 2025; Yao et al. 2023; Lu et al. 2025).

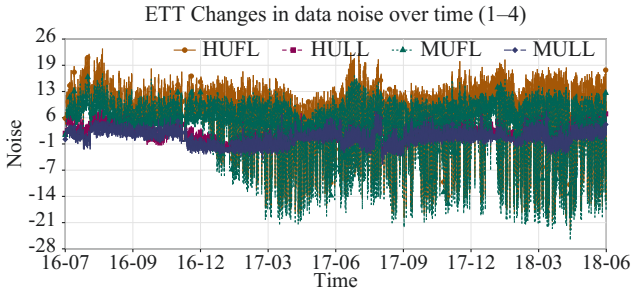

Figure 2: Irregular fluctuations in data.

In practical applications, time series data often exhibit unpredictable and non-ideal variations (Zhang et al. 2025c, 2024). Simply compensating for data imperfections may impair the model's generalization ability. Therefore, enhancing the model's robustness to imperfect conditions is critical. To this end, we propose IdealTSF, a time series forecasting framework designed to jointly strengthen the model's resilience across the pre-training, training, and optimization phases. As illustrated in Figure 2, the data exhibit pronounced jumps and heavy-tailed characteristics. IdealTSF explicitly addresses these non-ideal properties by introducing a negative sample pre-training module, which utilizes stable distributions to generate probabilistic data with targeted statistical features—mimicking jump processes and heavy-tailed stochastic behaviors. Additionally, it incorporates multi-scale noise injection and structured deletion to simulate anomalies and missingness commonly observed in real-world events. During training, we generate positive samples using hybrid smoothed interpolation, and integrate them with original inputs through a pre-trained attention mechanism to extract predictive features. Finally, in the optimization stage, negative perturbations are introduced to guide gradient descent toward flatter minima. To further enhance generalization, adversarial training is employed using FGSM (Fast Gradient Sign Method) or PGD (Projected Gradient Descent) attacks. This enables the model to converge more rapidly to flat optimal solutions and remain resilient to imperfect data. As demonstrated in Figure 1, IdealTSF outperforms state-of-the-art deep models, achieving approximately a 10% improvement in optimization metrics.

In summary, key contributions of this work are as follows:

- We demonstrate that non-ideal negative samples can also provide valuable information and can be leveraged in conjunction with positive samples to collaboratively extract informative features.
- To fully exploit the utility of negative data, the proposed IdealTSF framework adopts a three-stage progressive design encompassing pre-training, training, and optimization phases.
- Extensive experiments across multiple datasets and adversarial scenarios demonstrate that the proposed model can effectively leverage negative information to resist interference from non-ideal data, achieving performance improvements exceeding 10% over baseline methods.

Methods

IdealTSF receives input time series data $X \in \mathbb{R}^{B\times C\times L}$, where B denotes the batch size, C the number of input features, and L the number of time steps. The model uses historical data $X = \{X_1, X_2, ..., X_L\}$ to forecast future values Y. Forecasting grows harder with more variables and longer horizons, and real-world gaps/anomalies worsen it. We propose a lightweight attention model that learns from both positive and negative samples for efficient, robust prediction. Pretrain the attention module on synthetic "noisy/incomplete" data built from representative distributions, multi-scale noise, and structured deletion. Generate "clean" samples via hybrid smoothing + interpolation, then use the pre-trained attention to extract features for prediction. Inject adversarial perturbations (e.g., FGSM/PGD) during training to speed convergence and improve generalization, boosting robustness to real-world imperfections. The architecture of IdealTSF is illustrated in Figure 3.

Negative Sample Pre-training

To improve the model's adaptability to imperfect data, we design a negative sample pre-training module. In each training batch, artificially perturbed negative samples are constructed from the input data $X \in \mathbb{R}^{B\times C\times L}$ to train the model and enhance its robustness.

Stable Distributions To simulate jump processes and heavy-tailed stochastic behaviors, we employ stable distributions as defined in Equation 1, which generate probability distributions with specific statistical properties. The probability density function is determined by four parameters: the location parameter μ, the scale parameter γ, the stability index α, and the skewness parameter β. Here, μ controls the central location of the distribution, while γ determines the distribution's scale or volatility. The stability index α governs the tail behavior of the distribution, where $\alpha \in (0, 2]$ and $\alpha = 2$ corresponds to the normal distribution. The parameter β controls the asymmetry of the distribution, with $\beta = 0$ yielding a symmetric form.

$$X \sim \exp\left(i\mu t - \gamma|t|^{\alpha}\left(1 - i\beta\,\mathrm{sgn}(t)\,\tan\left(\frac{\pi\alpha}{2}\right)\right)\right) \quad (1)$$

To ensure that the generated increments are uniformly distributed across all possible directions—so that each increment has a random, rather than fixed, direction—we employ the polar coordinate method, as defined in Equation 2, to generate increments from a stable distribution. This involves sampling a random angle θ from a uniform distribution over the interval $[0, 2\pi]$. Additionally, a uniformly

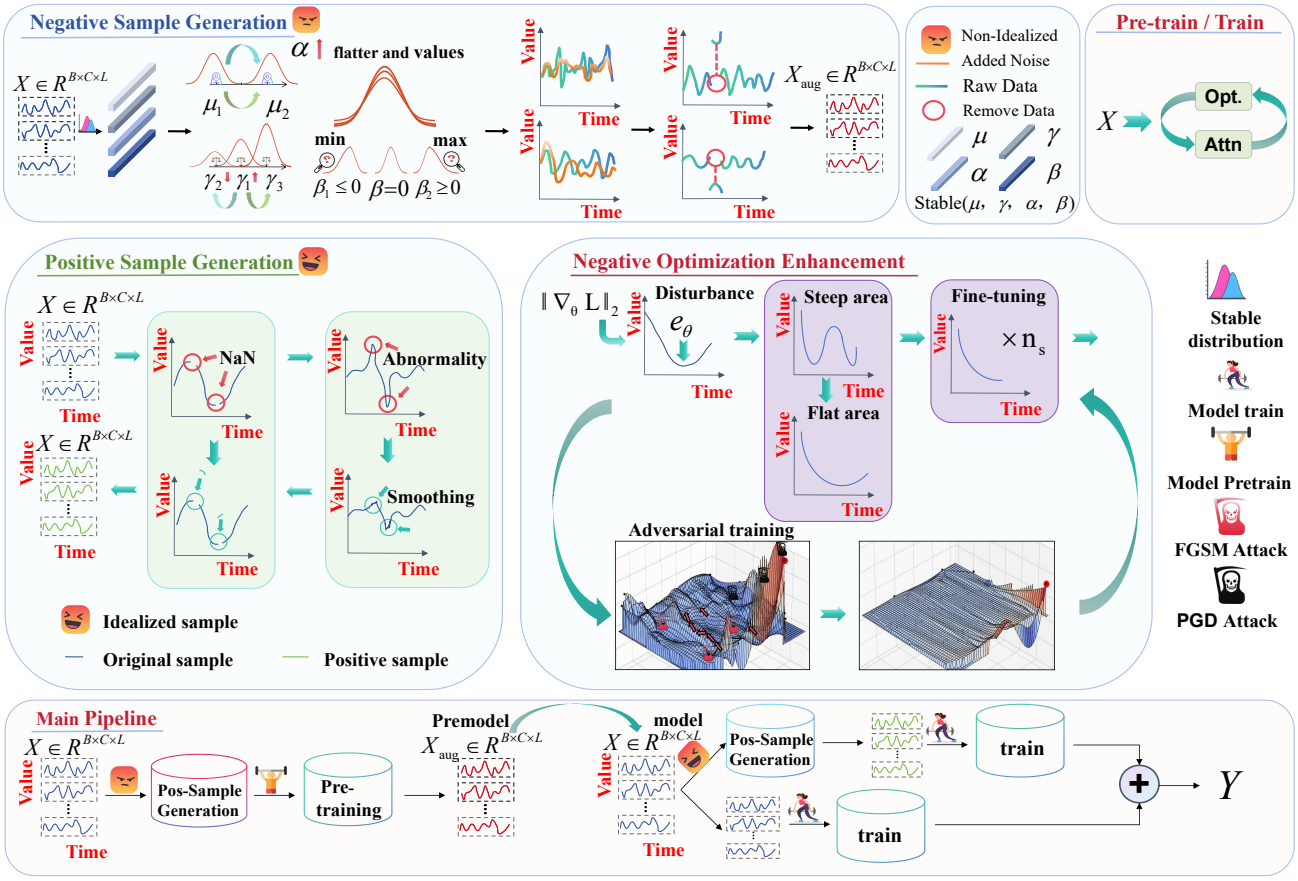

Figure 3: The architecture of IdealTSF.

distributed random variable independent of direction is sampled to maintain equal probability across all angles. The randomized angle θ is a critical component of the polar method, as it ensures diversity in the generated negative samples by avoiding directional bias.

$$\theta \sim \text{Uniform}(0, 2\pi) \tag{2}$$

Once the directional angle θ is determined, the magnitude R is computed using Equation 3 to control the size of the increment from the stable distribution. A smaller value of the stability index α leads to greater volatility and jump behavior in the negative samples, effectively simulating extreme time steps. The scale parameter γ controls the intensity of fluctuations, and the Gamma function $\Gamma(\alpha)$ is used in the computation of the distribution magnitude.

$$R = \left[\frac{\gamma}{2}\left(\frac{|\Gamma(\alpha)|}{1-\alpha}\right)\right]^{\frac{1}{\alpha}} \tag{3}$$

After obtaining the increment magnitude R, it is combined with the random angle θ using Equation 4 to generate the final increment Δx_i. This step ensures that the increment is random in both magnitude and direction, thereby capturing the jump characteristics inherent in the data. The resulting negative sample sequence is constructed using Equation 5, yielding $X'(T) = x'_1, x'_2, \ldots, x'_n$.

$$\Delta x_i = R \cdot \cos(\theta) \tag{4}$$

$$x'_i = x_i + \Delta x_i \tag{5}$$

Multi-scale Noise In real-world applications, time series data may experience various abrupt perturbations. To enhance the model's adaptability to such scenarios, we introduce multi-scale noise during the negative sample pretraining phase. Specifically, disturbances of varying frequencies are simulated by adding noise at different scales w_i. Noise $n(t)$ is added to the perturbed time series $X'(T)$ using Equation 6, thereby improving model robustness. For each scale w_i, the corresponding noise $n(t)$ is generated via Equation 7 with a distinct noise intensity σ_i, where $\mathcal{N}(0, \sigma_i^2)$ denotes a standard normal distribution representing the noise at each time step. A sliding window operation is applied to the noise across different scales to simulate the blurring effects of multi-scale perturbations. Lower-frequency (longer time-scale) noise is assigned higher intensity, while higher-frequency (shorter time-scale) noise is assigned lower intensity.

$$x_{\text{noise}}(t) = X'(T) + n(t), \tag{6}$$

$$n(t) = \frac{1}{w_i} \sum_{\tau=t-w_i+1}^{t} \mathcal{N}(0, \sigma_i^2) \tag{7}$$

Structured Deletion To further improve robustness to missing or irregular data, the model adopts a structured deletion strategy by randomly removing continuous segments from negative samples to simulate real-world data loss. For the input $x_{\text{noise}}(t)$, data is deleted over a randomly selected time interval using Equation 8, where L denotes the length of the deletion segment, and $L \in [L_{\min}, L_{\max}]$. Within the deleted interval, time series values are set to zero. Here, t_s represents the start time of deletion, and $L_d \sim \mathcal{U}(L_{\min}, L_{\max})$ denotes the deletion length sampled from a uniform distribution. The final augmented sequence x_{aug} is obtained using Equation 9.

$$x_{\text{aug}}(t) = \begin{cases} 0, & t \in [t_s, t_s + L_d] \\ x_{\text{noise}}(t), & \text{otherwise} \end{cases} \tag{8}$$

$$x_{\text{aug}} = [x_{\text{aug}}(1), ..., x_{\text{aug}}(L)] \tag{9}$$

Negative Sample Training We pre-train the attention module by minimizing the mean squared error (MSE) between the model output and the ground truth labels, as defined in Equation 10. Additionally, we incorporate the EcoSystem Optimizer (ECOS) to enhance the stability of parameter updates via negative enhancement strategies. Here, θ denotes the trainable parameters of the model, $\mathcal{L}_{\text{MSE}}$ is the mean squared error loss function, and $f_\theta(X)$ represents the model prediction based on input X and parameters θ.

$$\theta^* = \begin{cases} \arg\min_\theta \mathbb{E}_{(X,Y)} \left[\mathcal{L}_{\text{MSE}}(f_\theta(X), Y)\right] \\ \text{s.t.} \quad \theta \leftarrow \text{ECOS}(\nabla_\theta \mathcal{L}_{\text{MSE}}) \end{cases} \tag{10}$$

EcoSystem Optimizer (ECOS)

The stability of deep learning models depends heavily on a robust optimization environment. Inspired by the resilience of ecosystems—where recovery is possible even after natural disasters—we propose the EcoSystem Optimizer (ECOS). This method simulates external disturbances using adversarial attacks such as FGSM or PGD, while internal perturbations are introduced by injecting noise into the gradient descent process. Together, these mechanisms enhance the robustness and generalization ability of time series forecasting models.

Adversarial Sample Generation Before optimization, adversarial samples $\mathbf{x}_{\text{adv}}$ are generated using adversarial training techniques such as FGSM or PGD. For each input $\mathbf{x}$ and its corresponding label $\mathbf{y}$, the perturbation is computed according to Equation 11, where α is the perturbation step size, and $\nabla \mathbf{x} L(\mathbf{x}, \mathbf{y})$ is the gradient of the loss function with respect to the input, guiding the direction of the adversarial modification. Furthermore, Equation 12 ensures that the generated adversarial sample remains within a bounded perturbation region ϵ around the original input $\mathbf{x}$.

$$\mathbf{x}_{adv} = \mathbf{x} + \alpha \cdot \text{sign}(\nabla_{\mathbf{x}} L(\mathbf{x}, \mathbf{y})) \tag{11}$$

$$\mathbf{x}_{adv}^{k+1} = \text{clip}_{\mathbf{x},\epsilon} \left(\mathbf{x}_{adv}^k + \alpha \cdot \text{sign}\left(\nabla_{\mathbf{x}} L(\mathbf{x}_{adv}^k, y)\right)\right) \tag{12}$$

Phase I: Internal Perturbation Resistance and Flat Region Exploration After completing adversarial preparation, the goal of the first phase is to enhance the model's robustness against internal perturbations by "ascending" the loss landscape—adjusting parameters toward a local maximum. First, the gradient of the loss with respect to the model parameters is computed using Equation 13, yielding $\nabla_\theta L$. Then, perturbation e_θ is generated via Equation 14, where ρ controls the magnitude of the perturbation. Gradient normalization is applied to ensure stable updates. Finally, the perturbation is applied to the parameters using Equation 15, improving the model's adaptability to previously unseen samples.

$$\|\nabla_\theta L\|_2 = \sqrt{\sum_i (\nabla_{\theta_i} L)^2} \tag{13}$$

$$e_\theta = \frac{\rho}{\|\nabla_\theta L\|_2} \cdot \nabla_\theta L \tag{14}$$

$$\theta_{\text{new}} = \theta + e_\theta \tag{15}$$

Phase II: Multi-step Fine-tuning After the parameter updates in the first phase, the model proceeds to a multi-step fine-tuning stage. A base optimizer (e.g., Adam) is used to perform n_s small-step updates on each parameter with a learning rate η. In each step, forward propagation is conducted, gradients ∇L are computed, and backpropagation is applied for parameter adjustment. This iterative process avoids large updates, progressively guides the model toward a better solution, and enhances both training stability and convergence performance.

$$\theta_i^s = \theta_i^{s-1} - \eta \cdot \nabla L(\theta_i^{s-1}) \tag{16}$$

Phase III: Parameter Restoration and Base Optimization In the second phase, the parameters θ are restored to their pre-perturbation state using Equations 17–18, followed by a standard optimization update. Although this step is referred to as a "restoration," the actual objective is not to return to the original point, but rather to guide the parameters toward a flatter region in the loss landscape, thereby reducing the risk of falling into sharp local minima.

$$\theta_{\text{recovered}} = \theta_{\text{new}} - e_\theta \tag{17}$$

$$\theta_{\text{final}} = \theta_{\text{recovered}} - \eta \nabla_\theta L \tag{18}$$

Adversarial Training In the final step of optimization, ECOS performs forward propagation using the adversarial samples $\mathbf{x}_{\text{adv}}$, and computes the mean squared error loss $L(\mathbf{x}_{\text{adv}}, y)$ as defined in Equation 19. Subsequently, gradients are computed via backpropagation using Equation 20, and the parameters are updated through the base optimizer to ensure that the model is effectively optimized even on adversarial examples.

$$L_{\text{adv}} = L(x_{adv}, y) \tag{19}$$

$$\nabla_\theta L_{\text{adv}} = \frac{\partial L_{\text{adv}}}{\partial \theta} \tag{20}$$

$$\theta = \theta - \eta \nabla_\theta L_{\text{adv}} \tag{21}$$

Pre-training → Positive Sample Training

In the preceding stage, negative sample pre-training enabled the model to adapt to imperfect data conditions. In the formal training phase, we addressed potential data anomalies to guide the model in learning the normal patterns of the time series.

Missing Data Detection First, the missing data points T_{missing}, representing unavailable time steps, are identified. This set is defined by Equation 22, and the missing values are marked using NaN.

$$T_{\text{missing}} = \{t \mid x(t) = \text{NaN}\} \tag{22}$$

Anomaly Detection A hybrid anomaly detection approach is employed to identify outliers. First, the Z-score is calculated using Equation 23 to measure the deviation of data from the mean μ; if the absolute value exceeds a predefined threshold α, the point is flagged as a preliminary anomaly. Additionally, the interquartile range is computed using Equation 24 to detect further anomalies, where $Q1$ and $Q3$ represent the lower and upper quartiles, respectively.

$$Z(t) = \frac{x(t) - \mu}{\sigma} \tag{23}$$

$$\text{IQR} = Q3 - Q1 \tag{24}$$

According to Equation 25, anomalous data points are identified using a combined approach based on the Z-score and the Interquartile Range (IQR) methods. The threshold $\wp$ is typically set to 2 or 3. While the Z-score is suitable for normally distributed data, the IQR value I is more appropriate for non-normal distributions. This dual-criteria strategy enhances the model's ability to accurately detect outliers.

$$x(t) = \begin{cases} & \text{if } |Z(t)| > \wp \quad \text{or} \\ \text{Abnormal}, & \quad x(t) < Q1 - 1.5 \times \text{I} \quad \text{or} \\ & \quad x(t) > Q3 + 1.5 \times \text{I} \\ \text{Correct}, & \text{otherwise} \end{cases} \tag{25}$$

Positive Sample Generation To better leverage the robustness of the pre-trained attention mechanism, linear spline interpolation is employed to fill long-duration missing segments. For each missing time step $t \in T_{\text{missing}}$, interpolation is performed using Equation 26, which estimates the missing value based on the nearest known time steps t_1 and t_2. In addition, the interpolated results are smoothed using Equation 27 to eliminate short-term fluctuations, where W denotes the size of the sliding window. By applying the above hybrid interpolation methods to fill in missing values and correct anomalies, the complete time series is ultimately reconstructed using Equation 28. The resulting $\hat{x}$ serves as the positive sample for subsequent model training.

$$x(t) = x(t_1) + \frac{t - t_1}{t_2 - t_1} \cdot (x(t_2) - x(t_1)), \quad t \in [t_1, t_2] \tag{26}$$

$$\hat{x}(t) = \frac{1}{W} \sum_{t'=t-W+1}^{t} x(t') \tag{27}$$

$$x_{\text{aug}} = [\hat{x}(1), \hat{x}(2), ..., \hat{x}(L)] \tag{28}$$

Dual-Channel Feature Capture As shown in Equation 31, the original time series x_{orig} and the generated positive sample x_{aug} are combined as a dual-channel input and fed into the pre-trained attention mechanism. This design aims to demonstrate that, under complete data conditions, even a basic attention mechanism can achieve strong performance.

$$z_{\text{orig}} = [x_{\text{orig}}(1), x_{\text{orig}}(2), ..., x_{\text{orig}}(L)] \in \mathbb{R}^{B \times C \times L} \tag{29}$$

$$z_{\text{aug}} = [x_{\text{aug}}(1), x_{\text{aug}}(2), ..., x_{\text{aug}}(L)] \in \mathbb{R}^{B \times C \times L} \tag{30}$$

$$Attention_{train} \begin{cases} z_{\text{orig}} \\ z_{\text{aug}} \end{cases} \tag{31}$$

Experiments

In this section, we present the experimental results of IdealTSF on several mainstream long-term series forecasting (LTSF) benchmark datasets, covering both long- and short-term forecasting tasks for multivariate time series. To further validate the effectiveness of negative samples in IdealTSF, we conduct additional experiments that involve pre-training solely on negative samples without using any positive samples.

Benchmarks We conducted extensive experiments to evaluate the performance and efficiency of IdealTSF, covering both long-term and short-term forecasting tasks. For long-term forecasting, we conduct experiments on eight benchmark datasets: the ETT series—including four subsets (ETTh1, ETTh2, ETTm1, and ETTm2)—as well as Weather, Electricity, and Traffic datasets. For short-term forecasting, we evaluate on the PEMS dataset.

Long/Short-Term Time Series Forecasting

Table 1 presents a comparison between IdealTSF and other baseline models across eleven benchmark datasets. The results show that IdealTSF consistently ranks among the top two on all datasets, often achieving or closely approaching state-of-the-art performance. These datasets span various series with different sampling frequencies, numbers of variables, and real-world application scenarios. Notably, IdealTSF significantly outperforms models like Twinformer (Zhou et al. 2025), TimeMixer++ (Wang et al. 2024), TimeKAN (Huang et al. 2025), SAMformer (Ilbert et al. 2024), and PatchTST (Nie et al. 2022), and, relative to

		Ours		2025						2023-2024							
Models	H	IdealTSF		Twins*		Time**		TimeKAN		SAM*		iTrans*		PatchTST		DLinear	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	Avg	**0.402**	**0.419**	0.446	0.440	0.419	0.432	0.417	0.427	0.432	0.424	0.454	0.448	0.438	0.449	0.441	0.439
ETTh2	Avg	**0.338**	0.393	0.373	0.400	0.339	**0.380**	0.383	0.404	0.344	0.392	0.383	0.407	0.384	0.414	0.548	0.521
ETTm1	Avg	0.409	0.431	0.393	0.404	**0.369**	**0.378**	0.377	0.395	0.373	0.388	0.407	0.410	0.391	0.412	0.400	0.412
ETTm2	Avg	**0.248**	**0.302**	0.277	0.323	0.269	0.320	0.277	0.323	0.269	0.327	0.288	0.332	0.280	0.316	0.350	0.392
WTH	Avg	**0.201**	**0.249**	0.246	0.271	0.226	0.262	0.243	0.272	0.261	0.293	0.258	0.278	0.261	0.280	0.274	0.349
Traffic	Avg	**0.371**	**0.253**	0.407	0.274	0.416	0.264	0.422	0.269	0.425	0.297	0.428	0.282	0.555	0.395	0.632	0.397
ECL	Avg	**0.156**	**0.252**	0.167	0.261	0.165	0.253	0.182	0.274	0.181	0.275	0.176	0.270	0.209	0.306	0.211	0.303
PEMS03	Avg	**0.106**	**0.208**	0.109	0.219	0.116	0.226	—	—	0.180	0.304	0.169	0.272	0.376	0.329	0.159	0.262
PEMS04	Avg	**0.102**	**0.202**	0.111	0.219	0.121	0.232	—	—	0.195	0.307	0.209	0.317	0.353	0.420	0.130	0.241
PEMS07	Avg	0.118	0.220	**0.094**	**0.196**	0.100	0.204	—	—	0.211	0.303	0.235	0.315	0.380	0.440	0.125	0.226
PEMS08	Avg	0.182	0.250	**0.133**	**0.222**	0.151	0.234	—	—	0.280	0.321	0.268	0.306	0.440	0.363	0.192	0.271

* indicates Former; ** indicates Mixer++.

Table 1: Average performance on both long-term and short-term time series forecasting tasks. Bold indicates the best result. The above results are the average values of 5 different random seeds.

TimeKAN, reduces MSE on ECL and ETTh1 by approximately 17% and 3.5%, respectively. Importantly, IdealTSF also demonstrates robust performance on datasets with inherently low predictability, such as Traffic and ECL, further validating its generalization ability. In addition to long-term forecasting, IdealTSF achieves strong results in short-term forecasting tasks as well. On the PEMS benchmark—which consists of multiple time series recorded across a city-wide traffic network—many advanced models (e.g., PatchTST (2023) and DLinear (2023)) exhibit performance degradation due to complex spatiotemporal dependencies among variables. In contrast, IdealTSF maintains competitive performance on this challenging task, highlighting its effectiveness in modeling complex multivariate time series.

Ablation Study

To evaluate the effectiveness of each component in IdealTSF, we conducted ablation studies by systematically removing individual modules (denoted as w/o). Table 2 presents detailed results and corresponding analysis. Specifically, "w/o Neg" refers to the removal of the negative sample pre-training module, "w/o Pos" denotes the removal of the positive sample generation process, and "w/o ECOS" indicates the exclusion of the Ecosystem Optimizer. "w/o Pos+ECOS" removes both the positive sample training and ECOS, while "w/o Neg+ECOS" removes both negative sample pre-training and ECOS. From the MSE and MAE metrics across multiple datasets (ETTh1, ETTh2, ETTm1, and ETTm2), it is evident that IdealTSF consistently outperforms the ablated versions, demonstrating superior forecasting accuracy and stability. These results confirm the contribution of each module to the overall performance and robustness of the model.

Module-Specific Experiments

Feasibility Heatmap of Negative Sample Pre-training

As shown in the attention heatmaps of the ETTh dataset in Figure 4–5, the left panel illustrates the attention distribution without negative sample pre-training, while the right panel displays the final attention weights after applying negative sample pre-training. It can be observed that, after pre-training, the attention distribution shifts from a uniform pattern to a focused concentration on specific time steps. This indicates that the model has learned meaningful temporal dependencies and is able to adaptively identify and focus on critical time points or intervals in the time series.

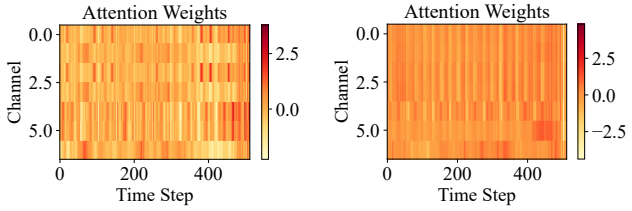

Figure 4: (a) The left figure shows the attention heatmap of ETTh1 without negative sample pre-training. (b) The right figure shows the attention heatmap of ETTh1 after applying negative sample pre-training.

ECOS Defense Performance Under Adversarial Attacks

Just as ecosystems exhibit resilience to external disruptions, the robustness of an optimizer can be evaluated by its per-

Models		IdealTSF		w/o Neg		w/o Pos		w/o ECOS		w/o Pos+ECOS		w/o Neg+ECOS	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	Avg	**0.402**	**0.423**	0.436	0.451	0.411	0.432	0.421	0.435	0.419	0.453	0.436	0.503
ETTh2	Avg	**0.338**	**0.393**	0.420	0.437	0.347	0.410	0.388	0.398	0.405	0.435	0.423	0.454
ETTm1	Avg	**0.409**	**0.431**	0.465	0.510	0.417	0.441	0.436	0.473	0.450	0.487	0.468	0.515
ETTm2	Avg	**0.247**	**0.301**	0.301	0.361	0.259	0.322	0.270	0.343	0.286	0.357	0.297	0.381

Table 2: IdealTSF of Ablation Studiese on ETT Datasets.

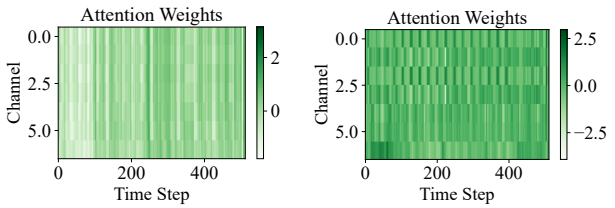

Figure 5: (a) The left figure shows the attention heatmap of ETTh2 without negative sample pre-training. (b) The right figure shows the attention heatmap of ETTh2 after applying negative sample pre-training.

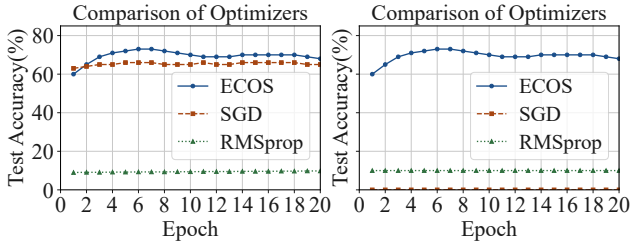

Figure 6: (a) Performance of IdealTSF under FGSM attack using different optimizers. (b) Performance of IdealTSF under PGD attack using different optimizers.

formance under adversarial inputs. The left plot of Figure 6 illustrates that under FGSM attacks, the ECOS optimizer maintains higher accuracy and stability compared to SGD and RMSprop. Notably, RMSprop shows a significant performance drop when exposed to adversarial perturbations. The right plot of Figure 6 further demonstrates that under the stronger PGD attack, only ECOS remains robust, while other optimizers suffer substantial performance degradation after the first few attack steps. These results highlight that ECOS is considerably more resilient to adversarial perturbations such as FGSM and PGD, making it particularly well-suited for scenarios involving risk, noise, or data anomalies.

Effectiveness of the ECOS Optimizer

To evaluate the effectiveness of the ECOS optimizer, we compare the performance of different optimization strategies—Adam vs. ECOS+Adam and SGD vs. ECOS+SGD—on the CIFAR-10 dataset, as shown in Figure 7. In the left panel, the accuracy of the standard Adam

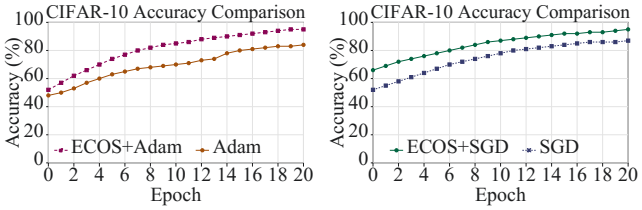

Figure 7: (a) The left figure shows the performance of the Adam and ECOS+Adam optimizers. (b) The right figure presents the performance of the SGD and ECOS+SGD optimizers.

optimizer steadily increases with training epochs, ultimately reaching around 80%. In contrast, ECOS+Adam exhibits a faster accuracy gain in the early stages and maintains higher stability in later epochs, achieving an accuracy of 90%. In the right panel, the standard SGD optimizer shows relatively slow improvement, stabilizing at approximately 80%, while ECOS+SGD significantly outperforms it, reaching nearly 90% accuracy. Overall, the ECOS optimizer substantially enhances both training speed and accuracy on CIFAR-10, improving optimization performance when applied to either Adam or SGD.

Conclusions

This paper proposes a time series forecasting model, IdealTSF, which leverages imperfect data (negative samples) during the pre-training, training, and optimization phases to enhance model robustness. By incorporating negative sample pre-training, hybrid smoothing interpolation, and adversarial training, IdealTSF effectively handles missing and anomalous data, thereby improving adaptability to complex real-world scenarios. Under conditions of low data quality or high uncertainty, the model outperforms traditional approaches in both accuracy and generalization. Experimental results demonstrate the effectiveness of the proposed method. Future innovations and improvements will focus more on multi-dimensional and complex time-series datasets with strong practical significance, so as to further enhance the usability and generalization ability of IdealTSF in real-world business scenarios. Meanwhile, we will investigate robust training strategies under harsher data-quality conditions, including negative-sample construction that better matches real noise distributions.

Acknowledgements

This work was supported in part by the following: the Joint Fund of the National Natural Science Foundation of China under Grant Nos. U24A20328, U24A20219, the Youth Innovation Technology Project of Higher School in Shandong Province under Grant No. 2023KJ212, the National Natural Science Foundation of China under Grant No. 62272281, the Special Funds for Taishan Scholars Project under Grant No. tsqn202306274, and the Natural Science Foundation of Shandong Province under Grant No. ZR20250C712.

References

Challu, C.; Olivares, K. G.; Oreshkin, B. N.; Ramirez, F. G.; Canseco, M. M.; and Dubrawski, A. 2023. Nhits: Neural hierarchical interpolation for time series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, 6989–6997.

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

Chen, Q.; Qu, D.; Tang, Y.; Song, H.; Zhang, Y.; Zhao, B.; Wang, D.; and Li, X. 2024. FreeGaussian: Guidance-free Controllable 3D Gaussian Splats with Flow Derivatives.

Chen, Y.; Zhang, C.; Ma, M.; Liu, Y.; Ding, R.; Li, B.; He, S.; Rajmohan, S.; Lin, Q.; and Zhang, D. 2023. Imdiffusion: Imputed diffusion models for multivariate time series anomaly detection. *arXiv preprint arXiv:2307.00754*.

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

Hewage, P.; Behera, A.; Trovati, M.; Pereira, E.; Ghahremani, M.; Palmieri, F.; and Liu, Y. 2020. Temporal convolutional neural (TCN) network for an effective weather forecasting using time-series data from the local weather station. *Soft Computing*, 24(21): 16453–16482.

Huang, S.; Zhao, Z.; Li, C.; and Bai, L. 2025. Timekan: Kan-based frequency decomposition learning architecture for long-term time series forecasting. *arXiv preprint arXiv:2502.06910*.

Ilbert, R.; Odonnat, A.; Feofanov, V.; Virmaux, A.; Paolo, G.; Palpanas, T.; and Redko, I. 2024. Samformer: Unlocking the potential of transformers in time series forecasting with sharpness-aware minimization and channel-wise attention. *arXiv preprint arXiv:2402.10198*.

Lai, G.; Chang, W.-C.; Yang, Y.; and Liu, H. 2018. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, 95–104.

LeCun, Y.; Bengio, Y.; and Hinton, G. 2015. Deep learning. *nature*, 521(7553): 436–444.

Lu, J.; Zhang, F.; Zhang, X.; Sun, Y.; and Wang, H. 2025. MCNR: Multiscale Feature-Based Latent Data Component Extraction Linear Regression Model. *Expert Systems with Applications*, 128634.

Martín, L.; Zarzalejo, L. F.; Polo, J.; Navarro, A.; Marchante, R.; and Cony, M. 2010. Prediction of global solar irradiance based on time series analysis: Application to solar thermal power plants energy production planning. *Solar energy*, 84(10): 1772–1781.

Murad, M. M. N.; Aktukmak, M.; and Yilmaz, Y. 2025. Wpmixer: Efficient multi-resolution mixing for long-term time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 19581–19588.

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

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

Qin, Y.; Song, D.; Chen, H.; Cheng, W.; Jiang, G.; and Cottrell, G. 2017. A dual-stage attention-based recurrent neural network for time series prediction. *arXiv preprint arXiv:1704.02971*.

Qu, D.; Chen, Q.; Zhang, P.; Gao, X.; Zhao, B.; Wang, Z.; Wang, D.; and Li, X. 2024. LiveScene: Language Embedding Interactive Radiance Fields for Physical Scene Control and Rendering. *Advances in Neural Information Processing Systems*, 37: 12271–12292.

Schmidl, S.; Wenig, P.; and Papenbrock, T. 2022. Anomaly detection in time series: a comprehensive evaluation. *Proceedings of the VLDB Endowment*, 15(9): 1779–1797.

Shumway, R. H.; and Stoffer, D. S. 2017. ARIMA models. In *Time series analysis and its applications: with R examples*, 75–163. Springer.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Wang, H.; Peng, J.; Huang, F.; Wang, J.; Chen, J.; and Xiao, Y. 2023. Micn: Multi-scale local and global context modeling for long-term series forecasting. In *The eleventh international conference on learning representations*.

Wang, S.; Li, J.; Shi, X.; Ye, Z.; Mo, B.; Lin, W.; Ju, S.; Chu, Z.; and Jin, M. 2024. Timemixer++: A general time series pattern machine for universal predictive analysis. *arXiv preprint arXiv:2410.16032*.

Wang, S.; Li, Z.; Li, Y.; Xiao, C.; Zhan, H.; Yao, Z.; Zhang, X.; Kang, J.; Li, L.; Liu, W.; et al. 2025. C3-OWD: A Curriculum Cross-modal Contrastive Learning Framework for Open-World Detection. *arXiv preprint arXiv:2509.23316*.

Wu, H.; Hu, T.; Liu, Y.; Zhou, H.; Wang, J.; and Long, M. 2022. Timesnet: Temporal 2d-variation modeling for general time series analysis. *arXiv preprint arXiv:2210.02186*.

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

Xiao, C.; Hou, L.; Fu, L.; and Chen, W. 2025. Diffusion-Based Self-Supervised Imitation Learning from Imperfect Visual Servoing Demonstrations for Robotic Glass Installation. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, 10401–10407. IEEE.

Xiao, C.; et al. 2024. Confusion-resistant federated learning via diffusion-based data harmonization on non-IID data. *Advances in Neural Information Processing Systems*, 37: 137495–137520.

Xu, L.; Xu, K.; Qin, Y.; Li, Y.; Huang, X.; Lin, Z.; Ye, N.; and Ji, X. 2022. TGAN-AD: Transformer-based GAN for anomaly detection of time series data. *Applied Sciences*, 12(16): 8085.

Yao, J.; Li, C.; Sun, K.; Cai, Y.; Li, H.; Ouyang, W.; and Li, H. 2023. Ndc-scene: Boost monocular 3d semantic scene completion in normalized device coordinates space. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, 9421–9431. IEEE Computer Society.

Yao, J.; Li, C.; and Xiao, C. 2024. Swift sampler: Efficient learning of sampler by 10 parameters. *Advances in Neural Information Processing Systems*, 37: 59030–59053.

Yin, X.; Wu, G.; Wei, J.; Shen, Y.; Qi, H.; and Yin, B. 2021. Deep learning on traffic prediction: Methods, analysis, and future directions. *IEEE Transactions on Intelligent Transportation Systems*, 23(6): 4927–4943.

Zeng, S.; Chang, X.; Xie, M.; Liu, X.; Bai, Y.; Pan, Z.; Xu, M.; and Wei, X. 2025a. FutureSightDrive: Thinking Visually with Spatio-Temporal CoT for Autonomous Driving. *arXiv preprint arXiv:2505.17685*.

Zeng, S.; Qi, D.; Chang, X.; Xiong, F.; Xie, S.; Wu, X.; Liang, S.; Xu, M.; and Wei, X. 2025b. JanusVLN: Decoupling Semantics and Spatiality with Dual Implicit Memory for Vision-Language Navigation. *arXiv preprint arXiv:2509.22548*.

Zhang, F.; Chen, G.; Wang, H.; and Zhang, C. 2024. CF-DAN: Facial-expression recognition based on cross-fusion dual-attention network. *Computational Visual Media*, 10(3): 593–608.

Zhang, F.; Wang, M.; Li, L.; Liu, Y.; and Wang, H. 2025a. Probabilistic intervals prediction based on adaptive regression with attention residual connections and covariance constraints. *Engineering Applications of Artificial Intelligence*, 156: 111013.

Zhang, F.; Wang, M.; Zhang, W.; and Wang, H. 2025b. THATSN: Temporal hierarchical aggregation tree structure network for long-term time-series forecasting. *Information Sciences*, 692: 121659.

Zhang, X.; Zeng, F.; Quan, Y.; Hui, Z.; and Yao, J. 2025c. Enhancing multimodal large language models complex reason via similarity computation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 10203–10211.

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

Zhou, Y.; Ye, Y.; Zhang, P.; Du, X.; and Chen, M. 2025. TwinsFormer: Revisiting Inherent Dependencies via Two Interactive Components for Time Series Forecasting.