# How Much Can Time-related Features Enhance Time Series Forecasting?

Chaolv Zeng
Shanghai Jiao Tong University
Shanghai, China
zclzcl@sjtu.edu.cn

Yuan Tian
China Yangtze Power Co
Yichang, China
tian_yuan4@ctg.com.cn

Guanjie Zheng*
Shanghai Jiao Tong University
Shanghai, China
gjzheng@sjtu.edu.cn

Yunjun Gao
Zhejiang University
Hangzhou, China
gaoyj@zju.edu.cn

## ABSTRACT

Recent advancements in long-term time series forecasting (LTSF) have primarily focused on capturing cross-time and cross-variate (channel) dependencies within historical data. However, a critical aspect often overlooked by many existing methods is the explicit incorporation of **time-related features** (e.g., season, month, day of the week, hour, minute), which are essential components of time series data. The absence of this explicit time-related encoding limits the ability of current models to capture cyclical or seasonal trends and long-term dependencies, especially with limited historical input. To address this gap, we introduce a simple yet highly efficient module designed to encode time-related features, Time Stamp Forecaster (TimeSter), thereby enhancing the backbone's forecasting performance. By integrating TimeSter with a linear backbone, our model, TimeLinear, significantly improves the performance of a single linear projector, reducing MSE by an average of 23% on benchmark datasets such as Electricity and Traffic. Notably, TimeLinear achieves these gains while maintaining exceptional computational efficiency, delivering results that are on par with or exceed state-of-the-art models, despite using a fraction of the parameters.

**PVLDB Reference Format:**
Chaolv Zeng, Yuan Tian, Guanjie Zheng, and Yunjun Gao. How Much Can Time-related Features Enhance Time Series Forecasting?. PVLDB, 14(1): XXX-XXX, 2020.
doi:XX.XX/XXX.XX

**PVLDB Artifact Availability:**
The source code, data, and/or other artifacts have been made available at https://github.com/zclzcl0223/TimeLinear.

## 1 INTRODUCTION

Time series data [2, 13, 27, 30, 31, 38, 46] is a sequence of data collected at successive points in time, typically at uniform intervals. Unlike other types of sequence data, such as text, time series data is characterized by each data point being associated with a time-related feature or time stamp[1] (e.g., season, month, week, hour, minute, etc), along with the corresponding value of the observed variable. In cases where multiple variables are recorded, the time series transitions from univariate to multivariate series. Given the ubiquity of multivariate time series data [3], accurate forecasting is essential across a broad range of fields, including transportation [8, 23, 33], finance [32, 41], and climate [5, 44]. In response to this demand, recent years have seen the development of numerous neural network architectures that have achieved substantial advancements in time series forecasting [6, 20, 22, 26, 36, 39, 40, 43, 45, 47].

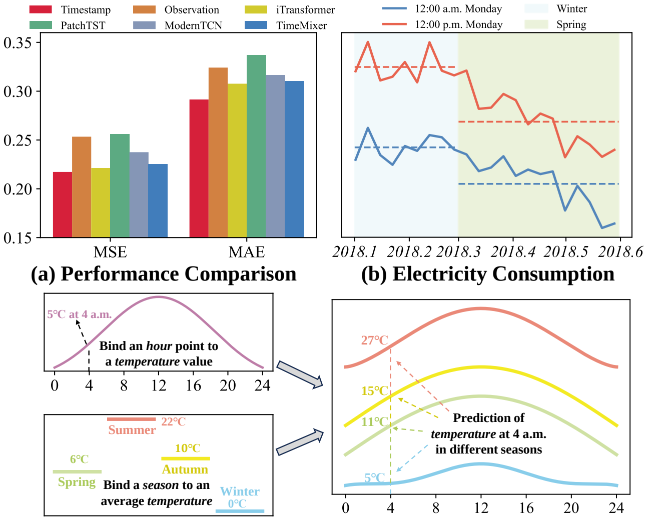
As mentioned above, each time series data point consists of a time stamp and the observed values of the measured variables. However, most existing forecasting models primarily focus on modeling cross-time and cross-variate dependencies based on the historical observations of the variables, often overlooking the rich semantic information embedded in the associated time stamps. Time stamps inherently provide crucial contextual information, as they constrain or dictate the values of the observed variables. For instance, in weather forecasting, summer months typically correlate with higher temperatures, while winter is associated with lower temperatures. Similarly, in transportation, weekdays usually see heavier traffic, whereas weekends tend to have lower traffic volumes. In other words, the combination of different time stamps corresponds to certain variable observations in a multidimensional vector space, within which variables fluctuate due to random noise.

To validate this claim, we conduct a case study using the Electricity dataset [40], where we encode the historical time stamps (hours, days, and seasons) to hidden space and make predictions based solely on this encoding using a single linear layer. As illustrated in Figure 1 (a), where *Timestamp* refers to models that use time-related features as input and *Observation* refers to models that use historical observations of the variables as input, both approaches leverage a single linear layer for predictions. Notably, under the setting of predicting the next 720 steps based on 96 historical data, forecasting based solely on time-related features outperforms all other baselines. This can be attributed to the cyclical stability and seasonal patterns present in residential electricity consumption. As depicted in Figure 1 (b), electricity consumption at noon and

*Corresponding Author

Proceedings of the VLDB Endowment, Vol. 14, No. 1 ISSN 2150-8097.
doi:XX.XX/XXX.XX

[1] We use the terms "time-related feature" and "time stamp" interchangeably.



**(a) Performance Comparison** **(b) Electricity Consumption**

**(c) Pipeline of Predicting with Time Stamps**

**Figure 1: (a) Comparison of forecasting using only time-related features, historical observations, and other baselines on Electricity with a look-back window of 96 and a future horizon of 720. (b) 100 people's (indexed as *190-290* on Electricity) average electricity consumption at noon and midnight every Monday during the first half of *2018*. (c) Temperature prediction using time-related features, where each hour and season is associated with some temperature values.**

midnight on Mondays exhibits relatively stable behavior within the same season, especially in winter months, fluctuating around an average value, represented by the dotted line. In contrast, values at the same time point differ significantly between spring and winter. The association of these consumption patterns with specific time-related features highlights the potential of utilizing time stamps to enhance time series prediction. Predicting with time stamps is analogous to binding a time-related feature to a specific observation, as shown in Figure 1 (c). The combination of different time-related features produces different predictions accordingly.

Based on the above discussion and the widespread existence of the relationship between observations and time stamps across various types of time series (e.g., weather, traffic, energy), we propose to predict with time-related features. However, given that we can only encode a limited number of time stamps (seasons, months, weeks, hours, minutes, etc.), while the future is unknown (such as years), relying solely on them would result in predictions that are invariant for all time points with the same time-related features. Therefore, while time stamps provide a robust foundation, historical observations are still needed to introduce variability and cope with noise in the predictions. Our proposed method consists of two key components to address multivariate time series forecasting with both time-related features and historical observations of variables. The first part, named **Time Stamp Forecaster** (TimeSter), encodes historical time-related features into a hidden space and makes predictions with a single linear layer. The second part, named **Backbone Forecaster** (BonSter), employs any other backbones (e.g., PatchTST [26], iTransformer [20]) to predict the noise components based on historical observations of variables. The outputs of these two components are combined to yield the final prediction. A simple combination of TimeSter and a linear layer backbone, named **TimeLinear**, can reduce the average MSE of a single linear projector on the Electricity and Traffic datasets by 23%. Technically, our contributions are summarized as:

- Realizing the relationship between time stamps and time series observations, we comprehensively study the potential of time stamps in enhancing time series forecasting.
- We introduce a lightweight and effective module to leverage time-related features, Time Stamp Forecaster (**TimeSter**), which can be easily integrated into various time series prediction models to boost their performance.
- We present a simple yet powerful model, **TimeLinear**, the combination of TimeSter and a linear layer, which achieves consistent state-of-the-art performance across seven real-world datasets with only *100k∼1M* parameters.

The rest of the paper will be organized as follows: We first introduce the preliminary knowledge in Section 2 and related works in Section 3. Then we introduce our method in Section 4. In Section 5, we present our experimental results and some visualization results in detail. Finally, we conclude in Section 6.

## 2 PRELIMINARY

**Multivariate Time Series Forecasting.** Given historical observations of $V$ variables $\mathbf{X} = \{\mathbf{x}_1, ..., \mathbf{x}_L\} \in \mathbb{R}^{L\times V}$, where $L$ denotes the look-back window, multivariate time series forecasting aims to predict the future values $\mathbf{Y} = \{\mathbf{x}_{L+1}, ..., \mathbf{x}_{L+T}\} \in \mathbb{R}^{T\times V}$, where $T$ is the prediction horizon. Specifically, this work focuses on long-term time series forecasting, where the prediction length $T$ is greater than or equal to 96. In the following sections, we denote multivariate series at a time point $i$ as $\mathbf{x}_i$ and univariate series as $x_i$.

**Time-related Features.** Time-related features are auxiliary information that captures temporal information at the point when measurements are recorded. These features typically include components such as season, month, day of the week, hour, and minute. For instance, a time stamp like *2024-07-15-14:10:00* can be represented by the categorical features $(Summer, July, Monday, 14h, 10min)$. In practice, these categorical features are numerically encoded (e.g., $Summer = 1$, $July = 6$) and then normalized to a common range, such as $[-0.5, 0.5]$, to facilitate training in machine learning models. In the following sections, we let $\mathbf{U} \in \mathbb{R}^{L\times r}$ denote the time-related features corresponding to the historical data, and $\mathbf{P} \in \mathbb{R}^{T\times r}$ denote the time-related features corresponding to the future data, where $r$ is the number of selected time-related features.

## 3 RELATED WORKS

### 3.1 Predicting with Historical Observations

The application of neural networks in time series forecasting has led to the development of numerous deep learning models tailored for this domain. These models span a variety of architectures, including CNN-based models like MICN [35], TimesNet [39], and ModernTCN [22]; Transformer-based models such as Crossformer [45], PatchTST [26], and iTransformer [20]; and models based on linear layers or MLPs, such as DLinear [43], RLinear [18], FITS [42], TimeMixer [36], MSD-Mixer [46], and SOFTS [10]. Despite their architectural differences, these models share a common goal of enhancing the modeling of multivariate historical observations to improve forecasting accuracy. Some models, such as DLinear, RLinear, FITS, PatchTST, and TimeMixer, focus on capturing long-range temporal dependencies, while others, like Crossformer, ModernTCN, TimesNet, iTransformer, and SOFTS, emphasize modeling the interactions among variables. However, a common limitation across these models is the lack of explicit consideration for the relationship between time-related features and the corresponding observations, which reduces their sensitivity to temporal dynamics, such as seasonal patterns and cyclical trends.

### 3.2 Predicting with Time-related Features

Incorporating time-related features, such as seasonality, day of the week, and time of day, is essential to improve predictive accuracy in time series forecasting. Models like Autoformer [40] and TimesNet [39] simply add time stamps to historical series through position encoding. iTransformer [20] treats different time-related features as different tokens and feeds them to the attention layer together with other time series variables. However, these models often treat time-related features as secondary, leading to suboptimal modeling of the interactions between these features and the target variables. This is further evidenced by empirical results [34], which demonstrate that removing time-related features from these models has minimal impact on performance. Many other models simply ignore time-related information [22, 26, 42, 43]. With the advent of LLMs and time series foundation models [1, 9], recent work [12, 21, 25, 37] has explored time-related features as prompts or metadata, yielding promising results. In smaller models, efforts have emerged to explicitly model time-related features and capture periodicity. GLAFF [34] applies a transformer to model dependencies between time-related features across different time points and then generates adaptive weights to balance global and local information. However, GLAFF lacks precise alignment between time-related features and historical observations, and its computational demands are high. More importantly, these methods that utilize time-related features all ignore the selection of time stamps, which is proven to be crucial in our experiments. CycleNet [19] learns a periodic sequence for each time series to model repetition-based periodicity. Nonetheless, it focuses solely on static periodic patterns and overlooks dynamic, timestamp-driven variations such as seasonality.

## 4 METHOD

In this section, we present TimeSter, our proposed module that makes predictions based on time-related features, and describe how it integrates with various backbone models. An overview of the architecture is shown in Figure 2. To evaluate the effectiveness of our approach, we consider the linear layer backbone and the resulting model, **TimeLinear**, as our main model.

### 4.1 Time Stamp Forecaster

**Time Stamp Encoder.** Given time-related features $\mathbf{U}_i \in \mathbb{R}^r$ at a specific time point $i$, where $r$ denotes the number of time-related features, the observation of a variable at this time $x_i$ is assumed to follow the conditional distribution $p_\phi(x_i|\mathbf{U}_i)$. As depicted in Figure 3, we visualize the probability density distribution of a selected training variable across different datasets at noon every Monday. The results indicate a time-dependent distribution pattern for these variables, yet directly modeling this distribution remains challenging. Inspired by [16], we approximate this distribution with a learnable model $q_\theta$. For generality, we consider the distribution of $V$ variables over an $L$-step historical window:

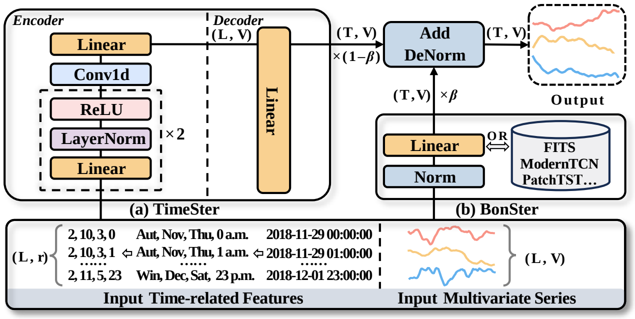


**Figure 2: Overview of the proposed method. (a) The TimeSter module encodes historical time-related features and predicts future values. (b) The BonSter, i.e., any backbone model, utilizes historical observations of multivariate time series to generate predictions. Their outputs are weighted and summed to yield the final prediction. The model after *Norm* could be replaced with any backbone. Here, we use a linear layer and name the whole model TimeLinear.**

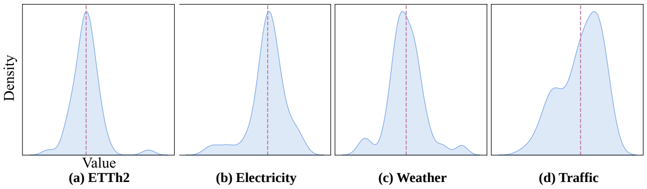


**Figure 3: Distribution of training data across datasets at noon on Mondays. From left to right, ETTh2: Variate *HUFL*, Electricity: Variate *1*, Weather: Variate *p (mbar)*, and Traffic: Variate *1*. From this, we can see a significant timestamp-related distribution.**

$$\mathbf{X}_\mathbf{U} \sim q_\theta(\mathbf{X}_\mathbf{U}|\mathbf{U}), \tag{1}$$

where $\mathbf{U} \in \mathbb{R}^{L\times r}$ represents historical time-related features and $\mathbf{X}_\mathbf{U} \in \mathbb{R}^{L\times V}$ represents the generated time-related observations. As shown in Figure 2 (a) left, we treat $q_\theta$ as an encoder consisting of two nonlinear hidden layers, a one-dimensional convolution layer, and a linear projection layer. Each linear layer projects along time-related feature and multivariate observation dimensions. The convolutional layer fuses features in the same dimension as the above linear layers and mixes channels in the time dimension to generate diverse outputs.

**Time-related Observation Decoder.** After generating the time-related multivariate observations, a single linear layer predicts the future observations, as depicted in Figure 2 (a) right:

$$\mathbf{Y_U} = \mathbf{W}\mathbf{X_U} + \mathbf{b}\,, \tag{2}$$

where $\mathbf{W} \in \mathbb{R}^{T \times L}$ and $\mathbf{b} \in \mathbb{R}^{T}$ are learnable parameters of the linear layer. The output $\mathbf{Y_U} \in \mathbb{R}^{T \times V}$ represents future predictions based on historical time-related observations.

## 4.2 Backbone Forecaster

Historical observations of time series are fundamental in time series modeling, as they directly capture the changing patterns of time series and provide a basis for accurate forecasting. To leverage both historical observations and time-related features, we incorporate other backbone models to process historical observations, complementing the TimeSter module. This integration allows TimeSter to be easily incorporated with various backbones. Technically, for the historical multivariate observations $\mathbf{X} \in \mathbb{R}^{L \times V}$, predictions are made with a specified backbone model, such as PatchTST [26]:

$$\mathbf{Y_B} = \mathrm{Backbone}(\mathbf{X})\,, \tag{3}$$

where $\mathbf{Y_B} \in \mathbb{R}^{T \times V}$ represents predictions based on historical observations. To isolate the effects of the backbone model and highlight the effectiveness of our time-related feature modeling, we use a single linear layer as our backbone in the following sections:

$$\mathbf{Y_B} = \mathbf{W_B}\mathbf{X} + \mathbf{b_B}\,. \tag{4}$$

Combining TimeSter, this model variant is referred to as **TimeLinear**. Finally, both predictions $\mathbf{Y_U}$ and $\mathbf{Y_B}$ are weighted and combined to produce the final result:

$$\mathbf{Y'} = \beta \mathbf{Y_B} + (1 - \beta)\mathbf{Y_U}\,, \tag{5}$$

where $\mathbf{Y'} \in \mathbb{R}^{T \times V}$ is the prediction, $\beta \in (0, 1)$ is a fixed coefficient that balances the contributions from each module. The full process is depicted in Figure 2 (b).

## 4.3 Simplified Reversible Instance Normalization

The Reversible Instance Normalization (RevIN) [14] is a widely used technique to mitigate distribution shifts in time series data. Since it was proposed, it has been widely used in almost all models [10, 18–20, 22, 26, 39, 42] in recent years and has played a significant role in improving their performance. To maintain consistent normalization across backbone models, we implement RevIN in all cases. For TimeLinear, we use a simplified RevIN variant without learnable affine parameters, as recent studies [19, 20] indicate it offers comparable performance with reduced parameters. Technically, we normalize historical observations with their respective means and standard deviations, and then denormalize the final forecasts using these same statistics. Note that we denormalize the final forecasts, not the output of the backbone network. Similar denormalization strategies also apply to other backbone networks. The process can be formulated as:

$$\begin{aligned} \hat{\mathbf{X}} &= \frac{\mathbf{X} - \mu}{\sqrt{\sigma^2 + \epsilon}}\,, \\ \hat{\mathbf{Y}}' &= \mathbf{Y'} \times \sqrt{\sigma^2 + \epsilon} + \mu\,, \end{aligned} \tag{6}$$

where $\mu$, $\sigma \in \mathbb{R}^{V}$ are the means and standard deviations of the multivariate historical observations, and $\epsilon$ is a small constant to avoid division by zero. The entire prediction process of TimeLinear is shown in Algorithm 1, where the *Linear* layer could be replaced with any other backbone models.

```
Algorithm 1: Pipeline of TimeLinear
Input: Historical time series $\mathbf{X} \in \mathbb{R}^{L \times V}$; historical time stamp
    feature $\mathbf{U} \in \mathbb{R}^{L \times r}$; future horizon $T$; trade-off coefficient $\beta$
  $\mu, \sigma \leftarrow \mathrm{Mean}(\mathbf{X}), \mathrm{STD}(\mathbf{X})$    // $\mu, \sigma \in \mathbb{R}^{V}$
  $\hat{\mathbf{X}} \leftarrow \frac{\mathbf{X}-\mu}{\sqrt{\sigma^2+\epsilon}}$    // $\hat{\mathbf{X}} \in \mathbb{R}^{L \times V}$
  $\mathbf{Y_U} \leftarrow \mathrm{TimeSter}(\mathbf{U})$    // $\mathbf{Y_U} \in \mathbb{R}^{T \times V}$
  $\mathbf{Y_B} \leftarrow \mathrm{Linear}(\hat{\mathbf{X}})$    // $\mathbf{Y_B} \in \mathbb{R}^{T \times V}$
  $\mathbf{Y'} \leftarrow \beta \mathbf{Y_B} + (1-\beta)\mathbf{Y_U}$    // $\mathbf{Y'} \in \mathbb{R}^{T \times V}$
  $\hat{\mathbf{Y}}' \leftarrow \mathbf{Y'} \times \sqrt{\sigma^2+\epsilon} + \mu$    // $\hat{\mathbf{Y}}' \in \mathbb{R}^{T \times V}$
Output: $\hat{\mathbf{Y}}'$    // The final prediction
```

# 5 EXPERIMENTS

In this section, we conduct a series of experiments to comprehensively evaluate the validity and effectiveness of our proposed method. We begin by detailing the experimental settings. In Section 5.1, we present the performance improvements achieved by the explicit incorporation of time-related features. Subsequently, in Section 5.2, we perform ablation studies to systematically verify the soundness of our modeling approach for time-related features. Finally, in Sections 5.3 and 5.4, we provide an in-depth analysis of why time-related features enhance performance, supported by additional insights and visualizations.

**Table 1: Overview of dataset.** *Variable* **denotes the number of variables.** *Timespan* **indicates the duration.** *Granularity* **indicates the interval between two time steps.** *Domain* **denotes the physical meaning of the observed value.**

| Dataset | Variable | Timespan | Granularity | Domain |
|---|---|---|---|---|
| ETTm1 & ETTm2 | 7 | 2016.7 - 2018.6 | 15 minutes | Electricity |
| ETTh1 & ETTh2 | 7 | 2016.7 - 2018.6 | 1 hour | Electricity |
| Electricity | 321 | 2016.7 - 2019.7 | 1 hour | Electricity |
| Weather | 21 | 2020.1 - 2021.1 | 10 minutes | Weather |
| Traffic | 862 | 2016.7 - 2018.7 | 1 hour | Transportation |

**Datasets.** We evaluate TimeLinear on seven well-built real-world datasets: four ETT series (ETTm1, ETTm2, ETTh1, ETTh2), Electricity, Weather, and Traffic, as detailed in Table 1. We adopt the dataset split and normalization approach used in previous studies [39, 40].

**Baselines.** We compare TimeLinear with models spanning various architectures: (i) Linear-based: DLinear [43], FITS [42], RLinear [18], CycleNet [19]; (ii) MLP-based: TimeMixer [36], TiDE [7], MSD-Mixer [46], SOFTS [10]; (iii) Convolution-based: TimesNet [39], ModernTCN [22]; and (iv) Transformer-based: Crossformer [45], PatchTST [26], iTransformer [20].

Linear-based models, including our TimeLinear backbone RLinear (consisting of a linear layer and the simplified RevIN), use a single linear layer for predictions, hence, they are simpler than

MLP-based models that employ multi-layer perceptrons for encoding. Moreover, GLAFF [34], which generates adaptive weights from time-related features to balance global and local dependencies, is included as a linear baseline with the same backbone as TimeLinear, denoted as GLAFFLinear.

**Implementation.** All models use a historical window length of $L = 96$ and are trained using Adam [15] with MSE loss. We evaluate models with Mean Squared Error (MSE) and Mean Absolute Error (MAE) over four prediction horizons $T \in \{96, 192, 336, 720\}$. Each experiment is repeated three times, and we report the mean values. All codes are implemented in Pytorch [28]. We conduct experiments on a single NVIDIA GeForce RTX 3090 24G GPU.

## 5.1 Main Results

In the main experiment, we start by comparing TimeLinear with state-of-the-art baseline models to validate the effectiveness of incorporating time-related features. Next, we integrate the TimeSter with various backbones to demonstrate the general applicability of our proposed method.

*5.1.1 Long-term Forecasting Performance.* Table 2 presents the multivariate long-term forecasting results across TimeLinear and other baselines. Most baseline results follow the original papers, except when experimental settings differ, in which case we rerun them (e.g., ModernTCN, PatchTST) under the official hyperparameters. The results show that TimeLinear achieves leading performance. On the one hand, TimeLinear is the best-performing Linear-based model, ranking top 1 in **12** out of 14 settings among all Linear-based models. On the other hand, TimeLinear is comparable to those models with complex encoders, ranking top 1 in **7** out of 14 settings among all methods. These outstanding performances are not only seen in datasets with a small number of variables, such as the ETT dataset (7 variables) but also in datasets with a large number of variables, such as the Electricity dataset (321 variables), where the complex correlations between variables have a great impact on the prediction results. More remarkably, TimeLinear achieves these with significantly fewer parameters and faster training speed than all non-linear models. Figure 4 shows that TimeLinear, with only *100k* parameters, outperforms other models and demonstrates superior efficiency. The results fully demonstrate the importance of time-related features and the effectiveness of our modeling.

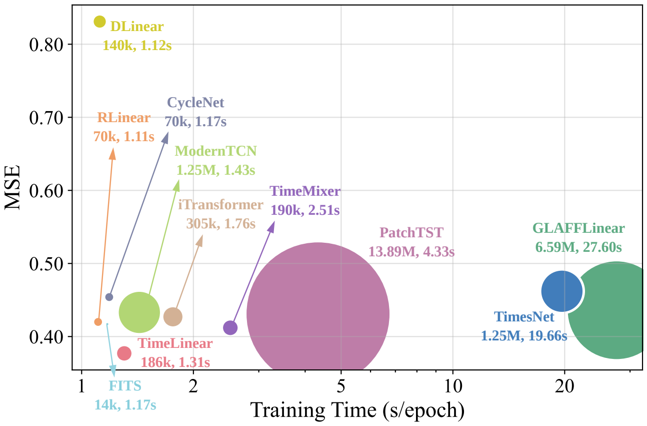


**Figure 4: Accuracy, parameter count, and training time of TimeLinear and other models on ETTh2, with historical window $L = 96$ and future horizon $T = 720$.**

*5.1.2 Generalization Performance.* To further verify the versatility of TimeSter, we evaluate its integration with various backbones. In addition to RLinear (the backbone of TimeLinear), we analyze TimeSter on five more models across different architectures and domains: FITS (Linear-based model in the frequency domain), ModernTCN and TimesNet (Convolution-based), iTransformer and PatchTST (Transformer-based). For a fair comparison, all results are reproduced on the same machine under the optimal hyperparameters, so some results might be different from Table 2.

As shown in Table 3 and 4, we can see that TimeSter generally enhances performance across different architectures, datasets, and domains. Especially, for datasets with strong periodicity and seasonality, such as Electricity, Weather, and Traffic, TimeSter can bring more significant improvements, which is consistent with the characteristics of these datasets. For these datasets, whether periodicity or seasonality, they are generally based on days, weeks, months, or seasons, which can be captured by time-related features.

However, the performance improvement of iTransformer (Table 4) on the Traffic dataset is minor after combining TimeSter with it. This may be caused by iTransformer's channel-dependent strategy [11], where the correlations between different time series variables are modeled. In contrast, another strategy is the channel-independent strategy, in which models do not model the relationship between different variables, but only treat different variables as independent training samples. While the cross-variate dependencies between historical observations are modeled by iTransformer, those between historical time-related features are not modeled by TimeSter, which might result in discrepancies between predictions of historical observations and time-related features. This can also explain the difference in improvement rates between different models: channel-independent models, shown in Table 3 (RLinear, FITS, and PatchTST, whose improvement rates are 10.47%, 10.02%, and 4.02% in MSE, respectively), have higher improvement rates while channel-dependent models, shown in Table 4 (ModernTCN, TimesNet, and iTransformer, whose improvement rates are 4.13%, 1.62%, and 3.81% in MSE, respectively), have lower improvement rates. Although differences in improvement rates among models exist, the results still show the strong applicability of TimeSter. Nonetheless, considering the relationship between different variables in time-related feature modeling is still an explorable topic, and we regard it as a future research direction.

## 5.2 Ablation Studies

In ablation studies, we first evaluate the selection of time-related features. Then, we verify the rationality of the design of the TimeSter module, including the encoder and decoder.

*5.2.1 Ablation of Time-related Features.* The selection of time-related features influences the correct time series pattern recognition, which is particularly important for data with different periodicity and seasonality. In TimeSter, we generally consider four primary time-related features: (i) Hour-of-day (H, 0, ..., 23); (ii) Day-of-week

**Table 2: Results of the multivariate long-term forecasting task under prediction lengths $T \in \{96, 192, 336, 720\}$. The historical window $L$ is fixed at 96. We report the average performance of four prediction lengths. For Linear-based models, we highlight the best in red and the runner-up in blue. For all architectures, we highlight the best performers with $^*$.**

| | Dataset | ETTm1 | | ETTm2 | | ETTh1 | | ETTh2 | | Electricity | | Weather | | Traffic | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Metric | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| Others | TimesNet [39] | 0.400 | 0.406 | 0.291 | 0.333 | 0.458 | 0.450 | 0.414 | 0.427 | 0.192 | 0.295 | 0.259 | 0.287 | 0.620 | 0.336 |
| | TiDE [7] | 0.419 | 0.419 | 0.358 | 0.404 | 0.541 | 0.507 | 0.611 | 0.550 | 0.251 | 0.344 | 0.271 | 0.320 | 0.760 | 0.473 |
| | Crossformer [45] | 0.513 | 0.496 | 0.757 | 0.610 | 0.529 | 0.522 | 0.942 | 0.684 | 0.244 | 0.334 | 0.259 | 0.315 | 0.550 | 0.304 |
| | PatchTST [26] | 0.387 | 0.400 | 0.283 | 0.327 | 0.445 | 0.441 | 0.378 | 0.403 | 0.191 | 0.280 | 0.256 | 0.278 | 0.461 | 0.291 |
| | MSD-Mixer [46] | 0.361* | 0.387* | 0.271* | 0.324 | 0.440 | 0.433 | 0.368 | 0.406 | 0.173 | 0.272 | 0.233* | 0.287 | 0.524 | 0.340 |
| | iTransformer [20] | 0.407 | 0.410 | 0.288 | 0.332 | 0.454 | 0.447 | 0.383 | 0.407 | 0.178 | 0.270 | 0.258 | 0.278 | 0.428 | 0.282 |
| | TimeMixer [36] | 0.381 | 0.395 | 0.275 | 0.323 | 0.447 | 0.440 | 0.364 | 0.395 | 0.182 | 0.272 | 0.240 | 0.271* | 0.484 | 0.297 |
| | ModernTCN [22] | 0.386 | 0.401 | 0.278 | 0.322 | 0.445 | 0.432 | 0.381 | 0.404 | 0.197 | 0.282 | 0.240 | 0.271* | 0.550 | 0.369 |
| | SOFTS [10] | 0.393 | 0.403 | 0.287 | 0.330 | 0.449 | 0.442 | 0.373 | 0.400 | 0.174 | 0.264 | 0.255 | 0.278 | 0.409* | 0.267* |
| Linear | DLinear [43] | 0.403 | 0.407 | 0.350 | 0.401 | 0.456 | 0.452 | 0.559 | 0.515 | 0.212 | 0.300 | 0.265 | 0.317 | 0.625 | 0.383 |
| | RLinear [18] | 0.412 | 0.406 | 0.286 | 0.327 | 0.446 | 0.433 | 0.377 | 0.399 | 0.215 | 0.291 | 0.273 | 0.291 | 0.623 | 0.371 |
| | FITS [42] | 0.415 | 0.408 | 0.286 | 0.328 | 0.444 | 0.432 | 0.374 | 0.397 | 0.217 | 0.295 | 0.273 | 0.292 | 0.627 | 0.375 |
| | CycleNet [19] | 0.386 | 0.395 | 0.272 | 0.315* | 0.433 | 0.428 | 0.384 | 0.405 | 0.170 | 0.261 | 0.255 | 0.279 | 0.486 | 0.313 |
| | GLAFFLinear [34] | 0.422 | 0.416 | 0.289 | 0.327 | 0.459 | 0.444 | 0.394 | 0.411 | 0.204 | 0.297 | 0.273 | 0.292 | 0.562 | 0.301 |
| | TimeLinear [Ours] | 0.385 | 0.395 | 0.273 | 0.315* | 0.432* | 0.426* | 0.358* | 0.389* | 0.165* | 0.259* | 0.251 | 0.276 | 0.480 | 0.304 |

(D, 0, ..., 6); (iii) Month-of-year (M, 0, ..., 11); and (iv) Season-of-year (S, 0, ..., 3).

Results, which are consistent with the analysis of dataset characteristics [17, 19, 47], are illustrated in Table 5. In general, ETTm1, ETTm2, and ETTh1 have strong daily cycles, while Traffic shows significant daily and weekly cycles. ETTh2, Electricity, and Weather exhibit more significant seasonality than other datasets. Notably, the Weather dataset, which is supposed to have strong seasonality, does not achieve significant gains in seasonal modeling (features M and S). This is because the time span of the Weather dataset is only one year, and the data used as the training set is only eight months, as shown in Table 1. As a result, the model can not fully capture all seasonal characteristics. However, TimeSter still achieves performance improvement under this limitation, which proves that TimeSter has strong generalization ability. The above results also illustrate the importance of time-related feature selection, which is ignored by previous methods [20, 21, 34].

#### 5.2.2 *Ablation of TimeSter Encoder.*

To further validate the design of TimeSter, we conduct extensive experiments on the effectiveness of each module in the TimeSter encoder. Results are shown in Table 6: (i) Removing ReLU significantly impacts datasets with more complex time-related feature dependencies, such as Electricity and ETTh2, where seasonality is considered. This highlights the importance of nonlinearity in learning intricate mappings. (ii) The Conv1d layer captures local feature correlations via convolution and global temporal dependencies via channel mixing. This not only enriches the semantic information of each feature but also enhances the temporal dependence of encoding. Therefore, compared to other modules, its removal will result in significant performance degradation. (iii) LayerNorm's stabilizing effect could provide a subtle advantage by ensuring consistent scaling, which aids the model's learning dynamics across varying time series patterns. (iv) Varying the hidden layer depth reveals that a deeper architecture achieves better performance, especially when compared to the *Zero Hidden Layer* model, which shows a substantial performance drop. This emphasizes the importance of model depth for capturing complex temporal patterns. The *One Hidden Layer* variant, while the performance is closer to the TimeSter, is still slightly inferior, indicating that additional depth enhances the model's representational power. However, the benefits of increasing the number of layers gradually decrease. Hence, for efficiency and accuracy considerations, the final model only considers two hidden layers. Overall, the full architecture of the TimeSter encoder is rational, as each component contributes to performance improvement.

#### 5.2.3 *Ablation of TimeSter Decoder.*

Our TimeSter module leverages time-related features by projecting the time-related observations generated by the encoder to future observations and adding them to the results generated by the backbone model. However, there are other embedding modes, e.g., adding the historical time-related observations with historical observations or adding the future time-related observations with the results generated by the backbone model. To demonstrate the effectiveness of our mode, we make a comprehensive comparison with all these variants in this section, including:

- Variant 1: $f(\mathbf{X})$ (i.e., RLinear), where we predict with historical observations using a single linear layer and simplified RevIN;
- Variant 2: $q_\theta(\mathbf{P})$, where we predict with the future time-related observations generated by TimeSter encoder;
- Variant 3: $f(q_\theta(\mathbf{U}))$, where we predict with a linear projector that takes the historical time-related observations as the input;
- Variant 4: $f(\mathbf{X}) + q_\theta(\mathbf{P})$, where we add the output of the linear projector and the future time-related observations;
- Variant 5: $f(\mathbf{X} + q_\theta(\mathbf{U}))$, where we add the input historical observations and time-related observations;

Table 3: Performance gain after combining TimeSter with channel-independent backbones, i.e., RLinear (Linear-based), FITS (Linear-based), and PatchTST (Transformer-based). The look-back window is $L = 96$. The performance of four prediction lengths $T \in \{96, 192, 336, 720\}$ is reported. *Avg* indicates the average performance of four prediction lengths. Red denotes improved performance and Green indicates decreasing performance. *Dataset Avg* denotes the average improvement of a model on all datasets.

| Models | | RLinear [18] | | w / TimeSter | | Improvement | | FITS [42] | | w / TimeSter | | Improvement | | PatchTST [26] | | w / TimeSter | | Improvement | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Metric | | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| ETTm1 | 96 | 0.350 | 0.370 | 0.325 | 0.364 | 7.06% | 1.78% | 0.353 | 0.374 | 0.329 | 0.368 | 6.77% | 1.73% | 0.325 | 0.364 | 0.316 | 0.363 | 3.00% | 0.36% |
| | 192 | 0.389 | 0.390 | 0.365 | 0.381 | 6.13% | 2.26% | 0.392 | 0.393 | 0.366 | 0.383 | 6.55% | 2.49% | 0.368 | 0.388 | 0.362 | 0.390 | 1.67% | -0.67% |
| | 336 | 0.422 | 0.413 | 0.395 | 0.401 | 6.43% | 2.88% | 0.425 | 0.415 | 0.397 | 0.402 | 6.67% | 2.98% | 0.396 | 0.405 | 0.387 | 0.405 | 2.42% | -0.03% |
| | 720 | 0.486 | 0.448 | 0.456 | 0.433 | 6.23% | 3.47% | 0.488 | 0.450 | 0.457 | 0.434 | 6.42% | 3.57% | 0.459 | 0.444 | 0.451 | 0.446 | 1.72% | -0.34% |
| | Avg | 0.412 | 0.406 | 0.385 | 0.395 | 6.43% | 2.64% | 0.415 | 0.408 | 0.387 | 0.397 | 6.59% | 2.74% | 0.387 | 0.400 | 0.379 | 0.401 | 2.15% | -0.18% |
| ETTm2 | 96 | 0.183 | 0.264 | 0.167 | 0.249 | 8.48% | 5.97% | 0.183 | 0.266 | 0.168 | 0.249 | 8.51% | 6.25% | 0.179 | 0.260 | 0.170 | 0.254 | 4.96% | 2.28% |
| | 192 | 0.246 | 0.304 | 0.233 | 0.291 | 5.41% | 4.28% | 0.247 | 0.305 | 0.234 | 0.292 | 5.45% | 4.44% | 0.242 | 0.302 | 0.241 | 0.302 | 0.45% | 0.00% |
| | 336 | 0.307 | 0.342 | 0.295 | 0.331 | 4.11% | 3.24% | 0.308 | 0.343 | 0.295 | 0.332 | 4.14% | 3.34% | 0.306 | 0.345 | 0.298 | 0.340 | 2.50% | 1.43% |
| | 720 | 0.407 | 0.398 | 0.395 | 0.390 | 2.96% | 2.03% | 0.408 | 0.398 | 0.396 | 0.390 | 2.92% | 2.25% | 0.406 | 0.402 | 0.401 | 0.400 | 1.30% | 0.39% |
| | Avg | 0.286 | 0.327 | 0.273 | 0.315 | 4.68% | 3.67% | 0.286 | 0.328 | 0.273 | 0.316 | 4.69% | 3.85% | 0.283 | 0.327 | 0.278 | 0.324 | 2.02% | 0.95% |
| ETTh1 | 96 | 0.389 | 0.395 | 0.378 | 0.391 | 2.75% | 1.00% | 0.386 | 0.394 | 0.378 | 0.392 | 2.00% | 0.42% | 0.377 | 0.396 | 0.377 | 0.396 | 0.03% | 0.10% |
| | 192 | 0.437 | 0.424 | 0.424 | 0.418 | 2.95% | 1.24% | 0.438 | 0.424 | 0.424 | 0.418 | 3.34% | 1.48% | 0.425 | 0.427 | 0.422 | 0.425 | 0.74% | 0.38% |
| | 336 | 0.479 | 0.445 | 0.463 | 0.438 | 3.29% | 1.41% | 0.479 | 0.444 | 0.460 | 0.436 | 3.89% | 1.85% | 0.461 | 0.447 | 0.459 | 0.446 | 0.52% | 0.31% |
| | 720 | 0.480 | 0.469 | 0.464 | 0.456 | 3.35% | 2.75% | 0.474 | 0.464 | 0.464 | 0.456 | 1.92% | 1.80% | 0.518 | 0.493 | 0.496 | 0.489 | 4.29% | 0.89% |
| | Avg | 0.446 | 0.433 | 0.432 | 0.426 | 3.11% | 1.64% | 0.444 | 0.432 | 0.432 | 0.425 | 2.82% | 1.42% | 0.445 | 0.441 | 0.438 | 0.439 | 1.57% | 0.44% |
| ETTh2 | 96 | 0.291 | 0.339 | 0.285 | 0.335 | 1.93% | 1.17% | 0.290 | 0.339 | 0.287 | 0.337 | 0.97% | 0.53% | 0.294 | 0.343 | 0.290 | 0.340 | 1.51% | 0.73% |
| | 192 | 0.376 | 0.390 | 0.373 | 0.390 | 0.80% | 0.07% | 0.375 | 0.390 | 0.373 | 0.392 | 0.42% | -0.49% | 0.377 | 0.398 | 0.364 | 0.389 | 3.41% | 2.14% |
| | 336 | 0.420 | 0.427 | 0.398 | 0.418 | 5.16% | 2.26% | 0.414 | 0.424 | 0.405 | 0.423 | 2.10% | 0.26% | 0.414 | 0.426 | 0.408 | 0.426 | 1.39% | 0.01% |
| | 720 | 0.422 | 0.440 | 0.377 | 0.412 | 10.78% | 6.41% | 0.417 | 0.436 | 0.386 | 0.418 | 7.43% | 4.18% | 0.426 | 0.445 | 0.421 | 0.441 | 1.15% | 0.89% |
| | Avg | 0.377 | 0.399 | 0.358 | 0.389 | 5.02% | 2.64% | 0.374 | 0.397 | 0.363 | 0.393 | 2.94% | 1.21% | 0.378 | 0.403 | 0.371 | 0.399 | 1.85% | 0.93% |
| Electricity | 96 | 0.197 | 0.274 | 0.140 | 0.234 | 29.10% | 14.48% | 0.200 | 0.278 | 0.141 | 0.237 | 29.28% | 14.83% | 0.165 | 0.256 | 0.142 | 0.240 | 13.92% | 5.95% |
| | 192 | 0.197 | 0.276 | 0.155 | 0.247 | 21.57% | 10.46% | 0.199 | 0.280 | 0.156 | 0.250 | 21.78% | 10.82% | 0.174 | 0.265 | 0.160 | 0.258 | 8.33% | 2.69% |
| | 336 | 0.212 | 0.292 | 0.169 | 0.265 | 20.10% | 9.11% | 0.214 | 0.295 | 0.171 | 0.268 | 20.18% | 9.34% | 0.191 | 0.282 | 0.173 | 0.273 | 9.50% | 3.13% |
| | 720 | 0.253 | 0.324 | 0.198 | 0.290 | 21.91% | 10.43% | 0.255 | 0.327 | 0.200 | 0.293 | 21.69% | 10.37% | 0.232 | 0.316 | 0.202 | 0.300 | 12.84% | 5.17% |
| | Avg | 0.215 | 0.291 | 0.165 | 0.259 | 23.04% | 11.06% | 0.217 | 0.295 | 0.167 | 0.262 | 23.09% | 11.27% | 0.191 | 0.280 | 0.169 | 0.268 | 11.20% | 4.25% |
| Weather | 96 | 0.195 | 0.234 | 0.166 | 0.212 | 14.91% | 9.29% | 0.194 | 0.235 | 0.167 | 0.213 | 14.38% | 9.16% | 0.173 | 0.214 | 0.157 | 0.205 | 9.49% | 4.27% |
| | 192 | 0.240 | 0.270 | 0.218 | 0.256 | 9.03% | 5.37% | 0.240 | 0.271 | 0.219 | 0.256 | 8.92% | 5.50% | 0.219 | 0.255 | 0.208 | 0.250 | 5.12% | 2.31% |
| | 336 | 0.291 | 0.306 | 0.272 | 0.294 | 6.60% | 4.02% | 0.292 | 0.307 | 0.273 | 0.294 | 6.59% | 4.21% | 0.277 | 0.297 | 0.265 | 0.290 | 4.58% | 2.22% |
| | 720 | 0.364 | 0.353 | 0.347 | 0.342 | 4.81% | 3.05% | 0.365 | 0.354 | 0.347 | 0.342 | 4.85% | 3.19% | 0.354 | 0.347 | 0.342 | 0.341 | 3.30% | 1.54% |
| | Avg | 0.273 | 0.291 | 0.251 | 0.276 | 8.02% | 5.10% | 0.273 | 0.292 | 0.251 | 0.277 | 7.91% | 5.20% | 0.256 | 0.278 | 0.243 | 0.272 | 5.08% | 2.42% |
| Traffic | 96 | 0.645 | 0.383 | 0.459 | 0.293 | 28.83% | 23.45% | 0.650 | 0.387 | 0.468 | 0.307 | 27.96% | 20.59% | 0.437 | 0.280 | 0.416 | 0.279 | 4.79% | 0.08% |
| | 192 | 0.598 | 0.359 | 0.467 | 0.298 | 21.83% | 16.92% | 0.602 | 0.363 | 0.480 | 0.315 | 20.17% | 13.28% | 0.448 | 0.284 | 0.429 | 0.284 | 4.24% | -0.23% |
| | 336 | 0.605 | 0.362 | 0.481 | 0.305 | 20.50% | 15.60% | 0.609 | 0.365 | 0.488 | 0.313 | 19.82% | 14.27% | 0.463 | 0.292 | 0.444 | 0.291 | 4.15% | 0.05% |
| | 720 | 0.643 | 0.381 | 0.512 | 0.320 | 20.43% | 16.11% | 0.647 | 0.385 | 0.516 | 0.328 | 20.23% | 14.90% | 0.497 | 0.310 | 0.477 | 0.309 | 3.91% | 0.17% |
| | Avg | 0.623 | 0.371 | 0.480 | 0.304 | 22.96% | 18.07% | 0.627 | 0.375 | 0.488 | 0.316 | 22.12% | 15.82% | 0.461 | 0.291 | 0.442 | 0.291 | 4.26% | 0.02% |
| Dataset Avg | | | | | | 10.47% | 6.40% | | | | | 10.02% | 5.93% | | | | | 4.02% | 1.26% |

- Variant 6: $f(\mathbf{X}) + g(q_\theta(\mathbf{U}))$, our TimeLinear, where we predict with two linear layers ($f$ and $g$) that take historical observations and historical time-related observations as input respectively and add the results up.

The results are shown in Table 7, where we can draw some interesting conclusions: (i) Predicting with time-related features surpasses predicting with historical observations in some datasets (Electricity and Weather), deriving from the comparison of Variant 1, 2, and 3. (ii) Historical time-related features and future time-related features exhibit similar effectiveness, deriving from Variant 2, 3 and Variant 4, 5. (iii) TimeLinear achieves the best overall performance. These findings not only prove the effectiveness of our time-related feature modeling but also verify the rationality of our dynamic projection strategy. Moreover, our experiments in the next section will show that as the historical window increases, our dynamic projection strategy is particularly more advantageous than others.

## 5.3 More Analysis

In this section, we begin with further experiments from the last section, evaluating the effectiveness of TimeSter under longer historical windows. Then, we explore the mechanism of time-related features visually. Furthermore, we use the quantitative indicator ACF to explore the relationship between time-related features and the characteristics of the corresponding time series. Finally, we conduct the robustness and hyperparameter analysis.

**Table 4: Performance gain after combining TimeSter with channel-dependent backbones, i.e., ModernTCN (Convolution-based), TimesNet (Convolution-based), and iTransformer (Transformer-based). The look-back window is $L = 96$. The performance of four prediction lengths $T \in \{96, 192, 336, 720\}$ is reported. *Avg* indicates the average performance of four prediction lengths. Red denotes improved performance and Green indicates decreasing performance. *Dataset Avg* denotes the average improvement of a model on all datasets.**

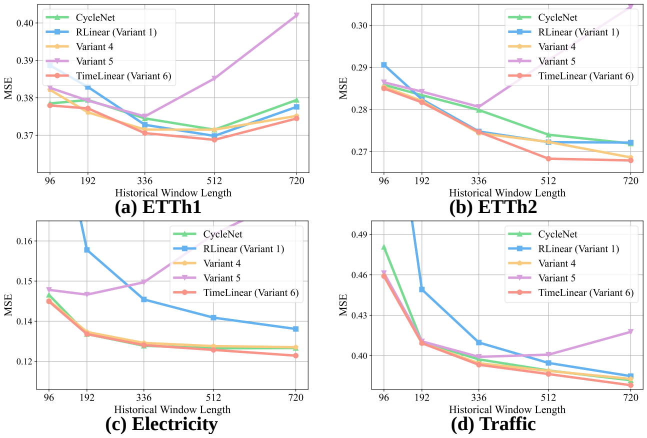
| Models | | ModernTCN [22] | | w / TimeSter | | Improvement | | TimesNet [39] | | w / TimeSter | | Improvement | | iTransformer [20] | | w / TimeSter | | Improvement | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Metric | | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| ETTm1 | 96 | 0.317 | 0.362 | 0.315 | 0.359 | 0.59% | 0.64% | 0.331 | 0.371 | 0.334 | 0.371 | -1.01% | -0.05% | 0.341 | 0.376 | 0.325 | 0.369 | 4.60% | 1.85% |
| | 192 | 0.363 | 0.389 | 0.358 | 0.387 | 1.62% | 0.55% | 0.392 | 0.404 | 0.379 | 0.396 | 3.40% | 1.75% | 0.382 | 0.395 | 0.367 | 0.390 | 3.90% | 1.47% |
| | 336 | 0.403 | 0.412 | 0.399 | 0.411 | 1.03% | 0.06% | 0.418 | 0.424 | 0.413 | 0.422 | 1.05% | 0.51% | 0.419 | 0.419 | 0.402 | 0.411 | 4.07% | 1.78% |
| | 720 | 0.461 | 0.443 | 0.457 | 0.439 | 0.94% | 0.96% | 0.490 | 0.460 | 0.480 | 0.456 | 1.90% | 1.05% | 0.490 | 0.458 | 0.469 | 0.448 | 4.39% | 2.31% |
| | Avg | 0.386 | 0.401 | 0.382 | 0.399 | 1.05% | 0.56% | 0.408 | 0.415 | 0.402 | 0.411 | 1.45% | 0.84% | 0.408 | 0.412 | 0.391 | 0.404 | 4.24% | 1.87% |
| ETTm2 | 96 | 0.173 | 0.255 | 0.169 | 0.252 | 1.85% | 1.14% | 0.188 | 0.266 | 0.186 | 0.264 | 0.82% | 0.66% | 0.185 | 0.270 | 0.174 | 0.258 | 5.80% | 4.64% |
| | 192 | 0.235 | 0.296 | 0.237 | 0.297 | -0.88% | -0.20% | 0.255 | 0.309 | 0.256 | 0.308 | -0.65% | 0.43% | 0.252 | 0.312 | 0.243 | 0.303 | 3.55% | 3.06% |
| | 336 | 0.308 | 0.344 | 0.299 | 0.336 | 2.80% | 2.20% | 0.319 | 0.347 | 0.312 | 0.343 | 2.23% | 1.12% | 0.314 | 0.351 | 0.302 | 0.340 | 3.79% | 3.07% |
| | 720 | 0.398 | 0.394 | 0.395 | 0.392 | 0.61% | 0.34% | 0.419 | 0.406 | 0.413 | 0.401 | 1.39% | 1.05% | 0.412 | 0.406 | 0.403 | 0.398 | 2.23% | 2.00% |
| | Avg | 0.278 | 0.322 | 0.275 | 0.319 | 1.09% | 0.87% | 0.295 | 0.332 | 0.292 | 0.329 | 1.08% | 0.85% | 0.291 | 0.335 | 0.280 | 0.325 | 3.50% | 3.06% |
| ETTh1 | 96 | 0.386 | 0.394 | 0.385 | 0.394 | 0.09% | 0.05% | 0.399 | 0.419 | 0.401 | 0.420 | -0.48% | -0.16% | 0.385 | 0.404 | 0.383 | 0.402 | 0.67% | 0.49% |
| | 192 | 0.436 | 0.423 | 0.436 | 0.423 | 0.01% | 0.01% | 0.453 | 0.449 | 0.451 | 0.450 | 0.51% | -0.27% | 0.443 | 0.437 | 0.431 | 0.429 | 2.65% | 1.83% |
| | 336 | 0.479 | 0.445 | 0.477 | 0.444 | 0.37% | 0.22% | 0.507 | 0.480 | 0.502 | 0.477 | 0.88% | 0.57% | 0.487 | 0.458 | 0.473 | 0.449 | 2.94% | 1.98% |
| | 720 | 0.481 | 0.469 | 0.479 | 0.466 | 0.40% | 0.54% | 0.532 | 0.502 | 0.505 | 0.490 | 4.99% | 2.42% | 0.517 | 0.497 | 0.470 | 0.470 | 9.07% | 5.61% |
| | Avg | 0.445 | 0.432 | 0.444 | 0.432 | 0.23% | 0.22% | 0.473 | 0.463 | 0.465 | 0.459 | 1.66% | 0.70% | 0.458 | 0.449 | 0.439 | 0.437 | 4.12% | 2.61% |
| ETTh2 | 96 | 0.292 | 0.340 | 0.290 | 0.339 | 0.35% | 0.40% | 0.322 | 0.363 | 0.324 | 0.367 | -0.57% | -0.91% | 0.299 | 0.350 | 0.295 | 0.346 | 1.44% | 1.04% |
| | 192 | 0.377 | 0.395 | 0.376 | 0.392 | 0.39% | 0.64% | 0.404 | 0.413 | 0.399 | 0.411 | 1.18% | 0.53% | 0.380 | 0.399 | 0.377 | 0.398 | 0.77% | 0.17% |
| | 336 | 0.424 | 0.434 | 0.419 | 0.431 | 1.05% | 0.72% | 0.448 | 0.449 | 0.438 | 0.443 | 2.26% | 1.42% | 0.423 | 0.432 | 0.419 | 0.431 | 0.93% | 0.37% |
| | 720 | 0.433 | 0.448 | 0.422 | 0.443 | 2.56% | 1.14% | 0.453 | 0.460 | 0.434 | 0.450 | 4.27% | 2.22% | 0.429 | 0.446 | 0.419 | 0.441 | 2.22% | 1.18% |
| | Avg | 0.381 | 0.404 | 0.377 | 0.401 | 1.18% | 0.75% | 0.407 | 0.421 | 0.399 | 0.417 | 1.99% | 0.92% | 0.383 | 0.407 | 0.378 | 0.404 | 1.35% | 0.69% |
| Electricity | 96 | 0.173 | 0.260 | 0.142 | 0.238 | 17.50% | 8.62% | 0.167 | 0.271 | 0.166 | 0.269 | 0.80% | 0.56% | 0.148 | 0.240 | 0.134 | 0.232 | 9.49% | 3.56% |
| | 192 | 0.181 | 0.267 | 0.160 | 0.253 | 11.74% | 5.26% | 0.186 | 0.288 | 0.184 | 0.286 | 1.15% | 0.69% | 0.165 | 0.256 | 0.152 | 0.248 | 7.68% | 3.19% |
| | 336 | 0.196 | 0.283 | 0.175 | 0.271 | 11.02% | 4.21% | 0.203 | 0.303 | 0.194 | 0.295 | 4.58% | 2.63% | 0.178 | 0.270 | 0.167 | 0.267 | 6.34% | 1.45% |
| | 720 | 0.238 | 0.316 | 0.206 | 0.298 | 13.43% | 5.70% | 0.225 | 0.320 | 0.228 | 0.322 | -1.57% | -0.61% | 0.221 | 0.308 | 0.195 | 0.292 | 11.67% | 5.04% |
| | Avg | 0.197 | 0.282 | 0.171 | 0.265 | 13.33% | 5.90% | 0.195 | 0.295 | 0.193 | 0.293 | 1.19% | 0.80% | 0.178 | 0.269 | 0.162 | 0.259 | 8.96% | 3.36% |
| Weather | 96 | 0.155 | 0.203 | 0.151 | 0.197 | 2.49% | 3.34% | 0.172 | 0.222 | 0.166 | 0.216 | 3.50% | 2.88% | 0.174 | 0.213 | 0.160 | 0.205 | 8.18% | 4.09% |
| | 192 | 0.202 | 0.247 | 0.200 | 0.244 | 1.33% | 1.37% | 0.229 | 0.268 | 0.216 | 0.259 | 5.48% | 3.66% | 0.224 | 0.257 | 0.214 | 0.252 | 4.66% | 1.99% |
| | 336 | 0.263 | 0.293 | 0.260 | 0.289 | 1.23% | 1.13% | 0.284 | 0.305 | 0.274 | 0.299 | 3.65% | 1.87% | 0.281 | 0.299 | 0.271 | 0.294 | 3.69% | 1.69% |
| | 720 | 0.341 | 0.343 | 0.338 | 0.338 | 0.99% | 1.34% | 0.357 | 0.354 | 0.351 | 0.349 | 1.67% | 1.29% | 0.359 | 0.351 | 0.349 | 0.346 | 2.77% | 1.47% |
| | Avg | 0.240 | 0.271 | 0.237 | 0.267 | 1.37% | 1.67% | 0.261 | 0.287 | 0.252 | 0.281 | 3.35% | 2.30% | 0.260 | 0.280 | 0.249 | 0.274 | 4.33% | 2.15% |
| Traffic | 96 | 0.523 | 0.350 | 0.465 | 0.304 | 11.07% | 13.15% | 0.590 | 0.315 | 0.588 | 0.314 | 0.21% | 0.25% | 0.393 | 0.269 | 0.392 | 0.267 | 0.43% | 0.40% |
| | 192 | 0.543 | 0.366 | 0.475 | 0.312 | 12.50% | 14.75% | 0.617 | 0.329 | 0.612 | 0.326 | 0.78% | 0.85% | 0.413 | 0.277 | 0.414 | 0.277 | -0.34% | 0.13% |
| | 336 | 0.544 | 0.366 | 0.494 | 0.323 | 9.17% | 11.75% | 0.636 | 0.337 | 0.630 | 0.334 | 0.98% | 0.73% | 0.424 | 0.283 | 0.427 | 0.283 | -0.66% | 0.03% |
| | 720 | 0.590 | 0.395 | 0.532 | 0.344 | 9.85% | 12.83% | 0.668 | 0.353 | 0.665 | 0.349 | 0.43% | 1.06% | 0.459 | 0.301 | 0.455 | 0.297 | 1.03% | 1.30% |
| | Avg | 0.550 | 0.369 | 0.492 | 0.321 | 10.63% | 13.12% | 0.628 | 0.333 | 0.624 | 0.331 | 0.60% | 0.73% | 0.422 | 0.282 | 0.422 | 0.281 | 0.13% | 0.48% |
| Dataset Avg | | | | | | 4.13% | 3.30% | | | | | 1.62% | 1.02% | | | | | 3.81% | 2.03% |

*5.3.1 Longer Historical Windows.* In long-term time series forecasting, the historical window length is an important hyperparameter, which determines the richness of temporal information within input data. Both theory [4] and practice [19, 20, 26] have proven that a longer historical window often leads to better forecasting performance. For time-related features, a longer historical window indicates richer time-related information, which is supposed to be useful for more accurate forecasting. Figure 5 illustrates the performance of CycleNet [19] and our variants in Table 7 under different historical window lengths, where some interesting findings could be drawn: (i) TimeLinear outperforms baselines and other variants at almost all historical window lengths. (ii) Variant 5 struggles to provide better prediction over a longer historical window. This is because longer historical windows also mean greater noise, and simply adding time-related observations with historical observations will lead to mutual interference of noise [29, 34], which will eventually lead to worse prediction results. (iii) While Variant 4 is close to TimeLinear under shorter historical windows, it gradually loses ground as the historical window increases, for it only relies on future time stamps and cannot utilize richer input time stamp information. (iv) The growth of the historical window also leads to a weakening of the gain brought by the time stamp, as richer historical observation information contains more cycle and seasonal information, which dilutes the benefits of time-related information modeling. Such a phenomenon is worthy of further exploration.

*5.3.2 Time-related Features Visualization.* To reveal the relationship between time stamps and variable observations, we visualize the learned time-related embedding and the output of different

**Table 5: Ablation of time-related features. *H* denotes Hour-of-day. *D* indicates Day-of-week. *M* is Month-of-year. *S* indicates Season-of-year. × means forecasting without TimeSter (i.e., only RLinear). The historical window $L$ is 96 and the future horizon $T$ is 720. The best is marked in red.**

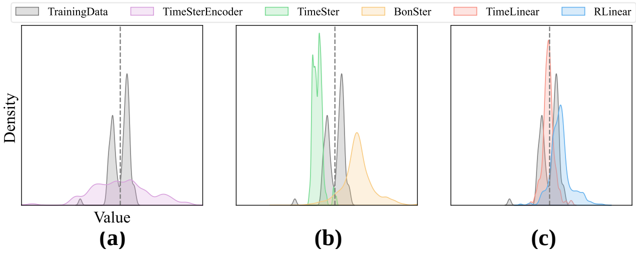
| Features | ETTm1 | ETTm2 | ETTh1 | ETTh2 | Electricity | Weather | Traffic |
|---|---|---|---|---|---|---|---|
| | MSE | MSE | MSE | MSE | MSE | MSE | MSE |
| × | 0.486 | 0.407 | 0.480 | 0.422 | 0.253 | 0.364 | 0.643 |
| H | 0.457 | 0.395 | 0.464 | 0.447 | 0.249 | 0.348 | 0.646 |
| H_D | 0.462 | 0.410 | 0.477 | 0.449 | 0.210 | 0.349 | 0.512 |
| H_M | 0.534 | 0.398 | 0.521 | 0.398 | 0.241 | 0.351 | 0.656 |
| H_S | 0.479 | 0.398 | 0.508 | 0.417 | 0.236 | 0.347 | 0.656 |
| H_D_M | 0.547 | 0.412 | 0.515 | 0.382 | 0.202 | 0.355 | 0.535 |
| H_D_S | 0.473 | 0.404 | 0.502 | 0.418 | 0.198 | 0.350 | 0.553 |
| H_M_S | 0.559 | 0.399 | 0.519 | 0.377 | 0.241 | 0.351 | 0.665 |
| H_D_M_S | 0.544 | 0.402 | 0.581 | 0.397 | 0.204 | 0.353 | 0.536 |



(a) ETTh1 (b) ETTh2 (c) Electricity (d) Traffic

**Figure 5: Performance promotion with longer historical windows. The forecasting length is 96.**

modules, shown in Figure 6, where the distribution is depicted. On the left figure, we find that the embedding of the TimeSter encoder tends to be more uniform, indicating that it pays more attention to learning diverse encoding. The next figure shows that TimeSter learns the shape of the data distribution well and corrects BonSter's high prediction values. The right figure also proves this point. After combining TimeSter, the model can make predictions that are more in line with the characteristics of that time point, that is, they are distributed along the mean (shown by the dotted line).



(a) (b) (c)

**Figure 6: Data distribution of the Traffic dataset at noon on Mondays. The variable index is 26. Equipping TimeSter enables the model to make predictions that are closer to the true distribution.**

### 5.3.3 *Correlations between Datasets and Time-related Features.*

Different datasets have different periodicities and seasonalities, which in turn affect the selection of time-related features. For example, hour-of-day (H) often corresponds to daily periodicity, while season-of-year (S) corresponds to seasonality. To analyze the periodicity and seasonality of different datasets, and explain the relationship between these properties and time-related feature selection, we introduce the Autocorrelation Function (ACF) [19, 24].

The Autocorrelation Function (ACF) is a statistical tool used to measure the correlation of a time series with its lagged versions. It provides insight into the degree of similarity between values separated by lags, which helps identify patterns, seasonality, or periodicity in time series. The autocorrelation at lag $k$ is defined as the correlation between observations separated by $k$ time steps. For a univariate time series $X$ with mean $\mu$, the ACF $\rho_k$ at lag $k$ is computed as:

$$\rho_k = \frac{\sum_{t=1}^{N-k}(x_t - \mu)(x_{t+k} - \mu)}{\sum_{t=1}^{N}(x_t - \mu)^2},$$

where $N$ is the number of observations, $\mu$ indicates the mean of the time series $X$, $x_t$ denotes the value of the time series at time step $t$, and $x_{t+k}$ is the value at time step $t + k$, representing a lag of $k$. The values of $\rho_k$ range from $-1$ to 1, where 1 denotes perfect positive correlation at lag $k$, 0 indicates no correlation at lag $k$, and $-1$ means perfect negative correlation at lag $k$.

For an ACF curve, with $k$ as the horizontal axis, its periodic peaks indicate the period of the time series. For instance, a peak that occurs every 12 time steps means that the original sequence has a component with a period of 12. If we downsample the original time series, ACF can also be used to explore the seasonality of the original series. Figure 7 illustrates the ACF curve of the Electricity and Traffic dataset respectively, where the granularity is hour for the left figure and day for the right one. From the left figure, we can see that both datasets show strong daily periodicity, which is reflected in the peaks of the ACF curve that appear every 24 hours. In addition, both datasets also show a certain weekly periodicity, which is reflected in the larger peaks that appear every 168 hours (7 days). From the right figure, where the granularity is day, we can find that the Electricity dataset shows a strong partition distribution phenomenon, that is, there is a peak approximately every 90 days (3 months), and there are long-term similar distributions on both sides of the peak. In contrast, such a phenomenon does not occur in the Traffic dataset. This indicates that the Electricity dataset has more significant seasonality than the Traffic dataset. These characteristics correspond one-to-one with the selection of time-related features, which further proves the importance and effectiveness of time-related features for extracting the periodicity and seasonality of time series.

### 5.3.4 *Minute-of-Hour for Datasets with Minute Granularity.*

In ablation studies, we evaluate the influence of time-related features under the hour granularity. However, the minute-of-hour (Min)

**Table 6: Ablation of TimeSter encoder. The historical window $L$ is fixed at 96 and the average performance of four prediction lengths is reported. The best performers for each dataset are highlighted in red.**

| Dataset | ETTm1 | | ETTm2 | | ETTh1 | | ETTh2 | | Electricity | | Weather | | Traffic | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Metric | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| TimeLinear | 0.385 | 0.395 | 0.273 | 0.315 | 0.432 | 0.426 | 0.358 | 0.389 | 0.165 | 0.259 | 0.251 | 0.276 | 0.480 | 0.304 |
| w / o ReLU | 0.386 | 0.395 | 0.273 | 0.315 | 0.433 | 0.426 | 0.369 | 0.395 | 0.172 | 0.265 | 0.251 | 0.276 | 0.477 | 0.312 |
| w / o Conv1d | 0.387 | 0.396 | 0.275 | 0.318 | 0.435 | 0.427 | 0.367 | 0.394 | 0.170 | 0.263 | 0.252 | 0.278 | 0.484 | 0.310 |
| w / o LayerNorm | 0.386 | 0.396 | 0.274 | 0.317 | 0.434 | 0.426 | 0.369 | 0.395 | 0.168 | 0.262 | 0.251 | 0.276 | 0.480 | 0.306 |
| One Hidden Layer | 0.386 | 0.395 | 0.273 | 0.316 | 0.434 | 0.427 | 0.365 | 0.394 | 0.167 | 0.261 | 0.251 | 0.276 | 0.482 | 0.308 |
| Zero Hidden Layer | 0.388 | 0.396 | 0.278 | 0.322 | 0.440 | 0.429 | 0.370 | 0.397 | 0.184 | 0.274 | 0.253 | 0.278 | 0.525 | 0.350 |

**Table 7: Ablation of TimeSter decoder. Both $f$ and $g$ are linear projectors. $\mathbf{X} \in \mathbb{R}^{L \times V}$ denotes historical observations of $V$ variables. $\mathbf{U} \in \mathbb{R}^{L \times r}$ denotes the time-related features corresponding to the historical data, and $\mathbf{P} \in \mathbb{R}^{T \times r}$ denotes the time-related features corresponding to the future data. $q_\theta(\cdot)$ is the TimeSter encoder. The best performers for each dataset are highlighted in red.**

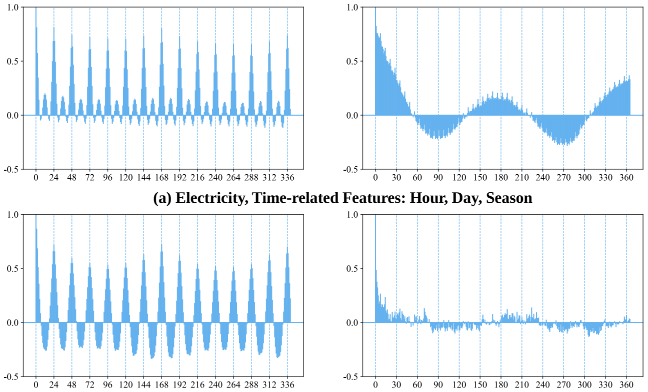
| Variant | Mode | ETTm1 | | ETTm2 | | ETTh1 | | ETTh2 | | Electricity | | Weather | | Traffic | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| 1 | $f(\mathbf{X})$ | 0.412 | 0.406 | 0.286 | 0.327 | 0.446 | 0.433 | 0.377 | 0.399 | 0.215 | 0.291 | 0.273 | 0.291 | 0.623 | 0.371 |
| 2 | $q_\theta(\mathbf{P})$ | 0.419 | 0.426 | 0.292 | 0.328 | 0.451 | 0.453 | 0.421 | 0.429 | 0.196 | 0.295 | 0.265 | 0.289 | 0.653 | 0.356 |
| 3 | $f(q_\theta(\mathbf{U}))$ | 0.419 | 0.426 | 0.293 | 0.328 | 0.451 | 0.454 | 0.390 | 0.411 | 0.195 | 0.294 | 0.266 | 0.290 | 0.657 | 0.356 |
| 4 | $f(\mathbf{X}) + q_\theta(\mathbf{P})$ | 0.386 | 0.396 | 0.272 | 0.315 | 0.436 | 0.428 | 0.379 | 0.402 | 0.167 | 0.261 | 0.251 | 0.276 | 0.479 | 0.303 |
| 5 | $f(\mathbf{X} + q_\theta(\mathbf{U}))$ | 0.386 | 0.395 | 0.275 | 0.318 | 0.434 | 0.427 | 0.367 | 0.394 | 0.168 | 0.264 | 0.251 | 0.276 | 0.481 | 0.307 |
| 6 | $f(\mathbf{X}) + g(q_\theta(\mathbf{U}))$ | 0.385 | 0.395 | 0.273 | 0.315 | 0.432 | 0.426 | 0.358 | 0.389 | 0.165 | 0.259 | 0.251 | 0.276 | 0.480 | 0.304 |

**Table 8: Performance changes after adding minute-of-hour (Min) to datasets with minute granularity. ↑ indicates improved performance, = indicates the same performance, and ↓ denotes decreasing performance. The best performer under different time-related feature combinations is bold. The historical window $L$ is 96 and the future horizon $T$ is 720.**

| Time Features | ETTm1 | | +Min | | ETTm2 | | +Min | | Weather | | +Min | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| H | **0.457** | **0.434** | **0.456**↑ | **0.433**↑ | **0.395** | **0.390** | **0.395**= | **0.390**= | 0.348 | 0.344 | 0.348= | 0.344= |
| H_D | 0.462 | 0.437 | 0.463↓ | 0.438↓ | 0.410 | 0.400 | 0.413↓ | 0.402↓ | 0.349 | 0.345 | 0.349= | 0.344↑ |
| H_M | 0.534 | 0.480 | 0.529↑ | 0.479↑ | 0.398 | 0.393 | 0.401↓ | 0.395↓ | 0.351 | 0.346 | 0.349↑ | 0.345↑ |
| H_S | 0.479 | 0.449 | 0.483↓ | 0.452↓ | 0.398 | 0.392 | 0.399↓ | 0.391↑ | **0.347** | **0.342** | **0.347**= | **0.343**↓ |
| H_D_M | 0.547 | 0.488 | 0.525↑ | 0.479↑ | 0.412 | 0.402 | 0.412= | 0.402= | 0.355 | 0.350 | 0.359↓ | 0.351↓ |
| H_D_S | 0.473 | 0.445 | 0.472↑ | 0.445= | 0.404 | 0.394 | 0.405↓ | 0.397↓ | 0.350 | 0.345 | 0.350= | 0.345= |
| H_M_S | 0.559 | 0.494 | 0.544↑ | 0.487↑ | 0.399 | 0.394 | 0.404↓ | 0.398↓ | 0.351 | 0.347 | 0.351= | 0.348↓ |
| H_D_M_S | 0.544 | 0.487 | 0.546↓ | 0.488↓ | 0.402 | 0.395 | 0.406↓ | 0.397↓ | 0.353 | 0.347 | 0.353= | 0.348↓ |
| ↑:=:↓ | 9:1:6 | | | | 1:4:11 | | | | 3:8:5 | | | |

**Table 9: Time-related features for different datasets and prediction lengths. Min: Minute-of-hour, H: Hour-of-day, D: Day-of-week, M: Month-of-year, S: Season-of-year.**

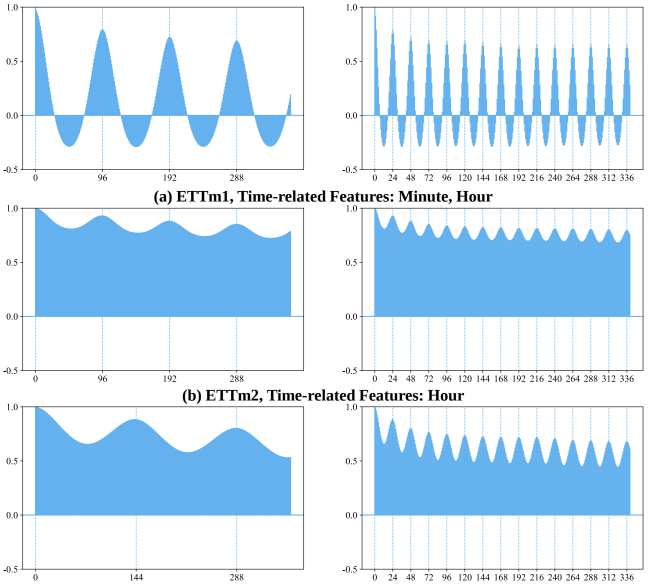
| Dataset | ETTm1 | ETTm2 | ETTh1 | ETTh2 | Electricity | Weather | Traffic |
|---|---|---|---|---|---|---|---|
| 96 | Min_H | H | H | H | H_D | H_S | H_D |
| 192 | Min_H | H | H | H | H_D | H_S | H_D |
| 336 | Min_H | H | H | H_D_M | H_D_S | H_S | H_D |
| 720 | Min_H | H | H | H_M_S | H_D_S | H_S | H_D |



(a) Electricity, Time-related Features: Hour, Day, Season

(b) Traffic, Time-related Features: Hour, Day

**Figure 7: ACF of Electricity and Traffic datasets. The variable index is 28 for both datasets. Granularity is hour for the left figure and day for the right figure. The selection of time-related features is highly consistent with the periodic and seasonal characteristics of the time series itself.**



(a) ETTm1, Time-related Features: Minute, Hour

(b) ETTm2, Time-related Features: Hour

(c) Weather, Time-related Features: Hour, Season

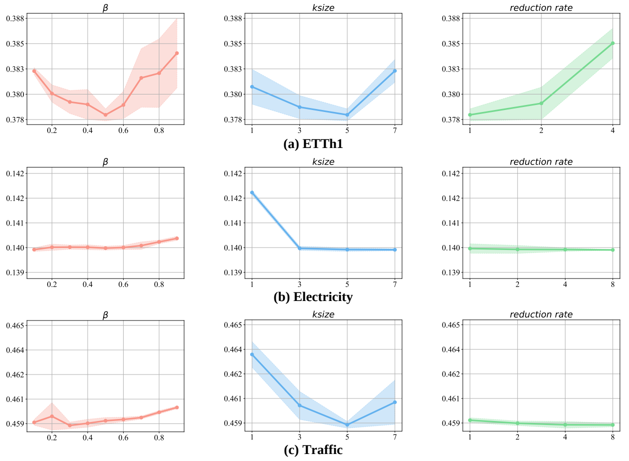
**Figure 8: ACF of ETTm1, ETTm2, and Weather. Granularity is minute for the left figure (15 minutes for ETTm1, ETTm2, and 10 minutes for Weather) and hour for the right figure. These datasets show stronger hourly periodicity.**

feature for datasets with minute granularity (ETTm1, ETTm2, and Weather) needs further analysis. In Table 8, we compare the results after adding the minute-of-hour (Min) feature with the original results in Table 5. The results show that although the minute feature improves the best performance of ETTm1, in most cases, it not only does not bring improvement but deteriorates model performance. We attribute the reason to the fact that these time series are not strongly correlated with the minute. To prove this, we illustrate the Autocorrelation Function (ACF) of ETTm1, ETTm2, and Weather under different granularity in Figure 8. We can see that these datasets clearly exhibit a stronger hourly periodicity, for the peak always occurs in a 24-hour cycle. ETTm1 exhibits a slightly different distribution, which explains why its performance improves after adding the minute feature. The above findings further demonstrate that our modeling of time stamps well reflects the intrinsic cyclical and seasonal characteristics of time series.

*5.3.5 Time-related Feature Choice for Different Datasets and Prediction Lengths.* Due to the differences in granularity and temporal characteristics of datasets and the prediction time span, the optimal time-related features are different for different datasets and prediction lengths. In Table 9, we illustrate the optimal time-related features of each dataset and prediction length under TimeLinear.

Generally, the choice of time-related features is consistent with the characteristics of the dataset itself, which are shown in both Figure 7 and Figure 8. As the prediction length increases, time-related features with longer time spans (e.g., month-of-year and season-of-year) become more effective for time series with stronger seasonality, as shown in the ETTh2 and Electricity with 336 or 720 prediction length. The same phenomenon also exists for longer historical windows. It should be noted that the above results are only obtained from experiments under our TimeLinear model. For other architectures (e.g., PatchTST [26], ModernTCN [22]), although the overall selection of time-related features is consistent with TimeLinear, due to the different abilities of models in capturing cross-time and cross-variate dependencies in historical observations, the optimal selection of time-related features might also vary slightly.

*5.3.6 Robustness Analysis.* Robustness is another important indicator to measure model performance. Table 10 shows the mean and standard deviation of TimeLinear under multiple random trainings. It can be seen that TimeLinear has strong stability as it shows a standard deviation close to 0 in almost all settings. This further indicates the practicality of our model.



(a) ETTh1

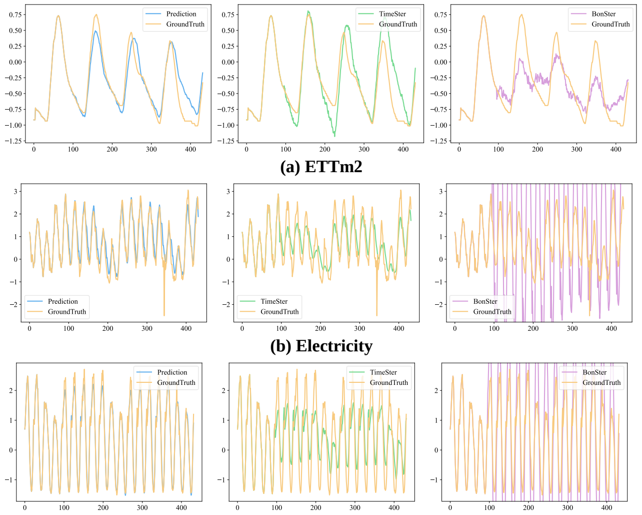
(b) Electricity

(c) Traffic

**Figure 9: Hyperparameter sensitivity of trade-off coefficient $\beta$, convolution kernel size *ksize*, and hidden layer size *reduction rate* for ETTh1, Electricity, and Traffic.**

Table 10: Robustness of TimeLinear performance. The average results from three random seeds {2020, 2021, 2022} are reported.

| Dataset<br>Horizon | | ETTm1 | ETTm2 | ETTh1 | ETTh2 | Electricity | Weather | Traffic |
|---|---|---|---|---|---|---|---|---|
| 96 | MSE | 0.325±0.000 | 0.167±0.000 | 0.378±0.001 | 0.285±0.001 | 0.140±0.000 | 0.166±0.000 | 0.459±0.000 |
| | MAE | 0.364±0.000 | 0.249±0.000 | 0.391±0.000 | 0.335±0.000 | 0.234±0.000 | 0.212±0.000 | 0.293±0.000 |
| 192 | MSE | 0.365±0.000 | 0.233±0.000 | 0.424±0.001 | 0.373±0.001 | 0.155±0.000 | 0.218±0.000 | 0.467±0.000 |
| | MAE | 0.381±0.001 | 0.291±0.000 | 0.418±0.001 | 0.390±0.001 | 0.247±0.000 | 0.256±0.001 | 0.298±0.000 |
| 336 | MSE | 0.395±0.000 | 0.295±0.000 | 0.463±0.001 | 0.398±0.001 | 0.169±0.000 | 0.272±0.000 | 0.481±0.000 |
| | MAE | 0.401±0.000 | 0.331±0.000 | 0.438±0.000 | 0.418±0.002 | 0.265±0.000 | 0.294±0.000 | 0.305±0.000 |
| 720 | MSE | 0.456±0.000 | 0.395±0.000 | 0.464±0.001 | 0.377±0.009 | 0.198±0.000 | 0.347±0.000 | 0.512±0.000 |
| | MAE | 0.433±0.000 | 0.390±0.000 | 0.456±0.000 | 0.412±0.003 | 0.290±0.000 | 0.342±0.000 | 0.320±0.000 |



(a) ETTm2

(b) Electricity

(c) Traffic

**Figure 10: Prediction results of three datasets. $L$ is 96 and $T$ is 336. From left to right, the result of TimeLinear, the result of TimeSter, and the result of BonSter, respectively.**

*5.3.7 Hyperparameter Sensitivity.* In addition to the basic learning rate, batch size, etc., various hyperparameters affect the performance of TimeLinear, mainly including the trade-off coefficient $\beta$, convolution kernel size *ksize*, and hidden layer size. In this section, we detail the sensitivity of TimeLinear to these hyperparameters on different datasets. Note that here we only consider the size of the first hidden layer, because the size of the second hidden layer is always set to the number of variables in the dataset. In addition, for the size of the first hidden layer, we express it as an *reduction rate* relative to the number of variables, e.g., *reduction rate* = 2 for Traffic with 862 variables means the hidden size is 431.

The results in Figure 9 show that $\beta$ and *ksize* have significant impacts on each dataset, for they directly affect the fusion of features and prediction results, which is crucial for effective time-related feature modeling. The size of the hidden layer determined by the *reduction rate* is related to the number of variables in the dataset, so its impact varies from dataset to dataset. Generally, a hidden layer that is too large not only reduces model efficiency but also leads to a certain degree of performance degradation, as shown in the results of the Traffic dataset. On the other hand, a hidden layer size that is too small will lead to insufficient modeling capabilities of the encoder. Therefore, this is a parameter that needs to be well controlled.

## 5.4 Prediction Results Visualization

In this section, we visualize the forecasting results of some datasets under TimeLinear, TimeSter, and BonSter. As shown in Figure 10, we can see that the TimeSter, which forecasts with time-related features, captures the periodicity and seasonality associated with time stamps well. In contrast, the BonSter, predicting with historical multivariate time series observations, tends to make predictions that are trend-based but have large deviations from the ground truth due to the influence of historical information. Although the results of individual predictions are not satisfactory, the combination of the TimeSter and BonSter has a complementary effect, resulting in better prediction accuracy. The above results not only reflect the importance of time-related features for more accurate long-term predictions but also verify the effectiveness and rationality of our model and pipeline.

# 6 DISCUSSION AND FUTURE WORK

This paper explores the potential of time-related features in enhancing multivariate long-term time series forecasting. Methodologically, we propose Time Stamp Forecaster (TimeSter), a plug-and-play module for time series prediction using time-related features. Extensive experiments demonstrate the versatility of TimeSter on different architectures. Moreover, its combination with a simple linear layer, named TimeLinear, achieves better performance than state-of-the-art models on multiple datasets while maintaining efficiency. In addition, we also study the existing shortcomings of our method, including the lack of modeling the relationship between different variables and the decrease in gain brought by time-related features after the historical window is increased. In the future, we hope to address these deficiencies and explore the possibilities of time-related features in various time series tasks.

## REFERENCES

[1] Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, Jasper Zschiegner, Danielle C. Maddix, Hao Wang, Michael W. Mahoney, Kari Torkkola, Andrew Gordon Wilson, Michael Bohlke-Schneider, and Yuyang Wang. 2024. Chronos: Learning the Language of Time Series. arXiv:2403.07815 [cs.LG]

[2] Parikshit Bansal, Prathamesh Deshpande, and Sunita Sarawagi. 2021. Missing Value Imputation on Multidimensional Time Series. *Proceedings of the VLDB Endowment* 14, 11 (2021), 2533–2545.

[3] Joos-Hendrik Böse, Valentin Flunkert, Jan Gasthaus, Tim Januschowski, Dustin Lange, David Salinas, Sebastian Schelter, Matthias Seeger, and Yuyang Wang. 2017. Probabilistic Demand Forecasting at Scale. *Proceedings of the VLDB Endowment* 10, 12 (2017), 1694–1705.

[4] George EP Box and Gwilym M Jenkins. 1968. Some Recent Advances in Forecasting and Control. *Journal of the Royal Statistical Society. Series C (Applied Statistics)* 17, 2 (1968), 91–109.

[5] Shengchao Chen, Guodong Long, Tao Shen, Jing Jiang, and Chengqi Zhang. 2024. Federated Prompt Learning for Weather Foundation Models on Devices. arXiv:2305.14244 [cs.LG]

[6] Yue Cui, Kai Zheng, Dingshan Cui, Jiandong Xie, Liwei Deng, Feiteng Huang, and Xiaofang Zhou. 2021. METRO: A Generic Graph Neural Network Framework for Multivariate Time Series Forecasting. *Proceedings of the VLDB Endowment* 15, 2 (2021), 224–236.

[7] Abhimanyu Das, Weihao Kong, Andrew Leach, Shaan Mathur, Rajat Sen, and Rose Yu. 2024. Long-term Forecasting with TiDE: Time-series Dense Encoder. arXiv:2304.08424 [stat.ML]

[8] Ziquan Fang, Lu Pan, Lu Chen, Yuntao Du, and Yunjun Gao. 2021. MDTP: A Multi-source Deep Trafic Prediction Framework over Spatio-Temporal Trajectory Data. *Proceedings of the VLDB Endowment* 14, 8 (2021), 1289–1297.

[9] Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. 2024. MOMENT: A Family of Open Time-series Foundation Models. In *International Conference on Machine Learning*.

[10] Lu Han, Xu-Yang Chen, Han-Jia Ye, and De-Chuan Zhan. 2024. SOFTS: Efficient Multivariate Time Series Forecasting with Series-Core Fusion. In *Advances in Neural Information Processing Systems*.

[11] Lu Han, Han-Jia Ye, and De-Chuan Zhan. 2024. The Capacity and Robustness Trade-off: Revisiting the Channel Independent Strategy for Multivariate Time Series Forecasting. *IEEE Transactions on Knowledge and Data Engineering* 36, 11 (2024), 7129–7142.

[12] Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, et al. 2024. Time-LLM: Time Series Forecasting by Reprogramming Large Language Models. In *International Conference on Learning Representations*.

[13] Mourad Khayati, Alberto Lerner, Zakhar Tymchenko, and Philippe Cudré-Mauroux. 2020. Mind the Gap: An Experimental Evaluation of Imputation of Missing Values Techniques in Time Series. *Proceedings of the VLDB Endowment* 13, 5 (2020), 768–782.

[14] Taesung Kim, Jinhee Kim, Yunwon Tae, Cheonbok Park, Jang-Ho Choi, and Jaegul Choo. 2021. Reversible Instance Normalization for Accurate Time-Series Forecasting against Distribution Shift. In *International Conference on Learning Representations*.

[15] Diederik P. Kingma and Jimmy Ba. 2017. Adam: A Method for Stochastic Optimization. arXiv:1412.6980 [cs.LG]

[16] Diederik P Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *International Conference on Learning Representations*.

[17] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. 2018. Modeling Long- and Short-Term Temporal Patterns with Deep Neural Networks. In *International ACM SIGIR Conference on Research and Development in Information Retrieval*.

[18] Zhe Li, Shiyi Qi, Yiduo Li, and Zenglin Xu. 2023. Revisiting Long-term Time Series Forecasting: An Investigation on Linear Mapping. arXiv:2305.10721 [cs.LG]

[19] Shengsheng Lin, Weiwei Lin, Xinyi Hu, Wentai Wu, Ruichao Mo, and Haocheng Zhong. 2024. CycleNet: Enhancing Time Series Forecasting through Modeling Periodic Patterns. In *Advances in Neural Information Processing Systems*.

[20] Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. 2024. iTransformer: Inverted Transformers Are Effective for Time Series Forecasting. In *International Conference on Learning Representations*.

[21] Yong Liu, Guo Qin, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. 2024. AutoTimes: Autoregressive Time Series Forecasters via Large Language Models. In *Advances in Neural Information Processing Systems*.

[22] Donghao Luo and Xue Wang. 2024. ModernTCN: A Modern Pure Convolution Structure for General Time Series Analysis. In *International Conference on Learning Representations*.

[23] Yisheng Lv, Yanjie Duan, Wenwen Kang, Zhengxi Li, and Fei-Yue Wang. 2015. Traffic Flow Prediction With Big Data: A Deep Learning Approach. *IEEE Transactions on Intelligent Transportation Systems* 16, 2 (2015), 865–873.

[24] Henrik Madsen. 2007. *Time Series Analysis*. Chapman and Hall/CRC.

[25] Sai Shankar Narasimhan, Shubhankar Agarwal, Oguzhan Akcin, Sujay Sanghavi, and Sandeep Chinchali. 2024. Time Weaver: A Conditional Time Series Generation Model. In *International Conference on Machine Learning*.

[26] Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. 2023. A Time Series is Worth 64 Words: Long-term Forecasting with Transformers. In *International Conference on Learning Representations*.

[27] John Paparrizos, Yuhao Kang, Paul Boniol, Ruey S. Tsay, Themis Palpanas, and Michael J. Franklin. 2022. TSB-UAD: An End-to-End Benchmark Suite for Univariate Time-Series Anomaly Detection. *Proceedings of the VLDB Endowment* 15, 8 (2022), 1697–1711.

[28] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems*.

[29] Luis Manuel Pereira, Addisson Salazar, and Luis Vergara. 2023. A Comparative Analysis of Early and Late Fusion for the Multimodal Two-Class Problem. *IEEE Access* 11 (2023), 84283–84300.

[30] Xiangfei Qiu, Jilin Hu, Lekui Zhou, Xingjian Wu, Junyang Du, Buang Zhang, Chenjuan Guo, Aoying Zhou, Christian S. Jensen, Zhenli Sheng, and Bin Yang. 2024. TFB: Towards Comprehensive and Fair Benchmarking of Time Series Forecasting Methods. *Proceedings of the VLDB Endowment* 17, 9 (2024), 2363–2377.

[31] Sebastian Schmidl, Phillip Wenig, and Thorsten Papenbrock. 2022. Anomaly Detection in Time Series: A Comprehensive Evaluation. *Proceedings of the VLDB Endowment* 15, 9 (2022), 1779–1797.

[32] Zhuangwei Shi. 2024. MambaStock: Selective state space model for stock prediction. arXiv:2402.18959 [cs.CE]

[33] Luan Tran, Min Y. Mun, Matthew Lim, Jonah Yamato, Nathan Huh, and Cyrus Shahabi. 2020. DeepTRANS: A Deep Learning System for Public Bus Travel Time Estimation using Traffic Forecasting. *Proceedings of the VLDB Endowment* 13, 12 (2020), 2957–2960.

[34] Chengsen Wang, Qi Qi, Jingyu Wang, Haifeng Sun, Zirui Zhuang, Jinming Wu, and Jianxin Liao. 2024. Rethinking the Power of Timestamps for Robust Time Series Forecasting: A Global-Local Fusion Perspective. In *Advances in Neural Information Processing Systems*.

[35] Huiqiang Wang, Jian Peng, Feihu Huang, Jince Wang, Junhui Chen, and Yifei Xiao. 2023. MICN: Multi-scale Local and Global Context Modeling for Long-term Series Forecasting. In *International Conference on Learning Representations*.

[36] Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Y Zhang, and JUN ZHOU. 2024. TimeMixer: Decomposable Multiscale Mixing for Time Series Forecasting. In *International Conference on Learning Representations*.

[37] Xinlei Wang, Maike Feng, Jing Qiu, Jinjin Gu, and Junhua Zhao. 2024. From News to Forecast: Integrating Event Analysis in LLM-Based Time Series Forecasting with Reflection. In *Advances in Neural Information Processing Systems*.

[38] Phillip Wenig, Sebastian Schmidl, and Thorsten Papenbrock. 2022. TimeEval: A Benchmarking Toolkit for Time Series Anomaly Detection Algorithms. *Proceedings of the VLDB Endowment* 15, 12 (2022), 3678–3681.

[39] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. 2023. TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis. In *International Conference on Learning Representations*.

[40] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. 2021. Autoformer: Decomposition Transformers with Auto-Correlation for Long-Term Series Forecasting. In *Advances in Neural Information Processing Systems*.

[41] Yumo Xu and Shay B Cohen. 2018. Stock Movement Prediction from Tweets and Historical Prices. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

[42] Zhijian Xu, Ailing Zeng, and Qiang Xu. 2024. FITS: Modeling Time Series with $10k$ Parameters. In *International Conference on Learning Representations*.

[43] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. 2023. Are Transformers Effective for Time Series Forecasting?. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

[44] Yuchen Zhang, Mingsheng Long, Kaiyuan Chen, Lanxiang Xing, Ronghua Jin, Michael I Jordan, and Jianmin Wang. 2023. Skilful Nowcasting of Extreme Precipitation with NowcastNet. *Nature* 619, 7970 (2023), 526–532.

[45] Yunhao Zhang and Junchi Yan. 2023. Crossformer: Transformer Utilizing Cross-Dimension Dependency for Multivariate Time Series Forecasting. In *International Conference on Learning Representations*.

[46] Shuhan Zhong, Sizhe Song, Weipeng Zhuo, Guanyao Li, Yang Liu, and S.-H. Gary Chan. 2024. A Multi-Scale Decomposition MLP-Mixer for Time Series Analysis. *Proceedings of the VLDB Endowment* 17, 7 (2024), 1723–1736.

[47] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. 2021. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*.