Leveraging 2D Information for Long-term Time Series Forecasting with Vanilla Transformers

Xin Cheng[1*] **Xiuying Chen**[2*] **Shuqi Li**[3] **Di Luo**[3]
Xun Wang[4] **Dongyan Zhao**[1] **Rui Yan**[3]

[1] Peking University [2] KAUST [3] Renmin University of China [3] Microsoft
chengxin1998@stu.pku.edu.cn xiuying.chen@kaust.edu.sa

Abstract

Time series prediction is crucial for understanding and forecasting complex dynamics in various domains, ranging from finance and economics to climate and healthcare. Based on Transformer architecture, one approach involves encoding multiple variables from the same timestamp into a single temporal token to model global dependencies. In contrast, another approach embeds the time points of individual series into separate variate tokens. The former method faces challenges in learning variate-centric representations, while the latter risks missing essential temporal information critical for accurate forecasting. In our work, we introduce GridTST, a model that combines the benefits of two approaches using innovative multi-directional attentions based on a vanilla Transformer. We regard the input time series data as a grid, where the x-axis represents the time steps and the y-axis represents the variates. A vertical slicing of this grid combines the variates at each time step into a *time token*, while a horizontal slicing embeds the individual series across all time steps into a *variate token*. Correspondingly, a *horizontal attention mechanism* focuses on time tokens to comprehend the correlations between data at various time steps, while a *vertical*, variate-aware *attention* is employed to grasp multivariate correlations. This combination enables efficient processing of information across both time and variate dimensions, thereby enhancing the model's analytical strength. The GridTST model consistently delivers state-of-the-art performance across various real-world datasets. We further conduct a comprehensive analysis to showcase the effectiveness of different components of our model, its generalization capability across diverse variates and time series, and its enhanced proficiency in utilizing arbitrary lookback windows. We will release our code and checkpoints to facilitate further research[1].

1 Introduction

Time series prediction involves analyzing and interpreting data points collected or recorded at successive time intervals to forecast future values [1, 2]. This task is fundamental in various fields, from finance and economics to weather forecasting and inventory management, where understanding trends, patterns, and potential future occurrences based on historical data is crucial. Among all the deep models, there is particular interest in applying Transformer architecture in time series prediction due to its strong performance, computational efficiency, and scalability [3–7].

The Transformer architecture, introduced in [8], is a groundbreaking neural network design that has transformed how we handle sequential data in natural language processing [9–11] and computer

[1] `https://github.com/Hannibal046/GridTST`

Preprint. Under review.

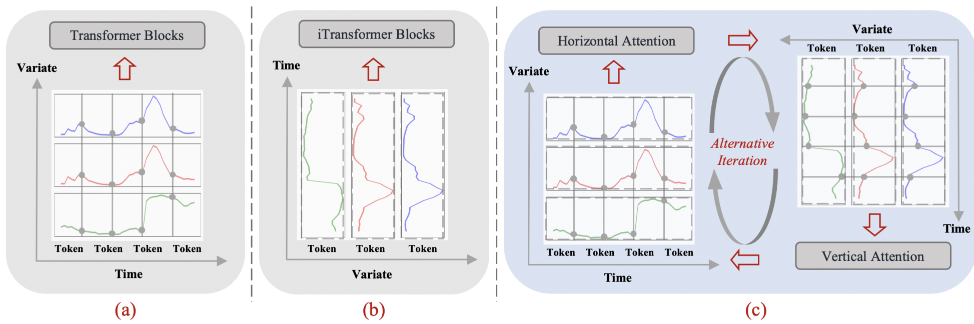

Figure 1: Comparison of the vanilla Transformer (a), inverse Transformer (b), and our proposed GridTST (c). Unlike baseline transformers that embed time steps into temporal tokens or the entire series into variate tokens separately, GridTST models both simultaneously. This approach captures multivariate and multi-time-step correlations using a bi-directional attention mechanism.

vision [12, 13]. When adapted for time series forecasting, the Transformer utilizes its ability to handle sequential dependencies and complex patterns, making it highly effective in predicting future trends and values in time-dependent data [4, 14–16].

Classic transformer-based forecasters embed multi-variate data points from the same timestamp into a single variable and use temporal tokens to capture temporal dependencies, as illustrated in Figure 1(a). [7] noted that single-time-step tokens may not effectively convey information due to their limited receptive field. Furthermore, [17] pointed out the inappropriate use of permutation-invariant attention in the temporal dimension. To address these issues, [7] developed the iTransformer, which inverts the roles of the attention mechanism and feed-forward network, representing time points as variate tokens (Figure 1(b)). However, this model still lacks adequate timestamp modeling. Effective time modeling is vital in time series prediction to capture unique patterns like trends and seasonality, as emphasized by [15, 18, 19]. These models employ the vanilla Transformer architecture, which ensures generalizability and compatibility with existing hardware accelerators, aiding in deployment and scalability for complex datasets. This raises a critical question: Can the vanilla Transformer architecture effectively capture both temporal and covariate information?

To address the above issues, in this work, we propose a novel method named GridTST, which captures the cross-time and cross-variate dependency by adapting the traditional attention mechanism and architecture from different views. Specifically, we adapt the Patching technique from [18] into our mechanism. This involves segmenting time series data into subseries-level patches that function as input tokens. The motivation behind this is that a single time step, unlike a word in a sentence, lacks inherent semantic meaning. Therefore, extracting local semantic information becomes crucial for analyzing their interconnections. Then, we visualize the input time series data as a grid, with the x-axis representing time steps and the y-axis representing the variates. By slicing this grid vertically, we combine variates at each time step into a 'time token', and through horizontal slicing, we embed the data of individual series across all time steps into a 'variate token'. Correspondingly, we can apply a horizontal attention mechanism on time tokens to analyze correlations between data at different time steps, and a vertical, variate-aware attention to capture the multivariate correlations. In this way, multivariate and multi-time-step correlations can be depicted by the attention mechanism. This method is illustrated in Figure 1(c). Correspondingly, the feed-forward network is able to learn generalizable representations that encapsulate both series- and time-aware features. The comparison with the existing method frameworks is shown in Table 5 in Appendix.

Our contributions lie in three aspects: Firstly, our observations confirm that both temporal and covariate information are crucial for the task of time series prediction. Secondly, we introduce GridTST, a model that effectively leverages the foundational Transformer architecture. It employs an innovative data structuring technique to concurrently and efficiently capture both temporal dynamics and covariate information. Finally, GridSTS achieves consistent state-of-the-art on real-world forecasting benchmarks.

2 Related Work

In recent years, a multitude of deep models have been introduced for temporal modeling. Recurrent Neural Networks (RNNs) are commonly utilized to capture temporal dependencies, as noted by [20, 21]. [22] introduced LSTNet, which integrates Convolutional Neural Networks (CNNs) with recurrent-skip connections. Moreover, a body of research has focused on leveraging Temporal Convolutional Networks (TCNs) to understand temporal causality through causal convolution, as exemplified in seminal works by [23, 24, 3, 25]. Additionally, recent studies such as those by [26] and [27] have explored the use of simpler linear layers for the modeling of complex temporal data, resulting in the development of models like DLinear and TSMixer, respectively.

There are also a large body of literature introducing Transformer [8] into time series analysis with specific modules designed for the unique aspects of time series, such as decomposition and periodicity detection. Decomposition stands as a foundational technique that breaks down complex time series into simpler, more predictable components [28, 15, 29]. Inspired by stochastic process theory, the aspect of periodicity within time series is increasingly being factored into models to better handle complex temporal variations [28, 15, 29, 30]. As one of these, TimesNet [30] explores the multi-periodicity of time series and captures the intra-period variations and inter-period variations of a single variate simultaneously which is also the first task-general foundation model achieving significant results across all five time series analysis tasks. Besides, Crossformer [31] models the cross-time dependency and the cross-dimension dependency to bridge the gap in focus on modeling the relationships among multivariate data.

The research landscape for adapting the Transformer architecture to time series analysis includes a promising line of inquiry that prioritizes zero modification of the original Transformer design. This approach ensures generalizability, compatibility with existing hardware accelerators, and scalability—all vital for efficient deployment. Two significant contributions in this domain are PatchTST [18] and iTransformer [7]. PatchTST introduces channel-independence and patching strategies for effective time-dependency modeling, which have demonstrated consistent performance enhancements and have quickly gained traction in subsequent studies [32, 33]. Contrasting with earlier methods that focus on global dependencies across temporal tokens in time series, iTransformer presents a novel perspective by treating individual series as variate tokens. This approach enables the model to capture intricate multivariate correlations. In the same vein of leveraging the unaltered Transformer architecture, GridTST emerges as a novel model by viewing the time series as a grid and modeling with alternated self-attention, thereby becoming the first to concurrently harness both temporal and covariate information using the vanilla Transformer framework.

3 GridTST

3.1 Problem Formulation

In multivariate time series forecasting, we take the historical observations as input, denoted by $X = \{x_1, \ldots, x_T\} \in \mathbb{R}^{T \times N}$, where T represents the number of time steps and N the number of variates. Our objective is to forecast F future time steps, which we denote as $Y = \{x_{T+1}, \ldots, x_{T+F}\} \in \mathbb{R}^{F \times N}$. We denote the prediction results as $\hat{Y} = \{\hat{x}_{T+1}, \ldots, \hat{x}_{T+F}\} \in \mathbb{R}^{F \times N}$. For brevity, $X_{t,:}$ refers to the multivariate data recorded at a single time step t, whereas $X_{:,n}$ represents the entire time series for a specific variate n.

3.2 Model Structure

As illustrated in Figure 2, GridTST utilizes the vanilla encoder-only architecture of Transformer, including the patched embedding, horizontal and vertical attentions. It's important to note that the horizontal and vertical attentions are standard Transformer attention modules, with the innovation lying in the dual-view of time series data.

Patched Time Tokens. A straightforward approach to generate input tokens involves considering each variate at every time step as a distinct token. Nevertheless, in contrast to words within a sentence, an isolated time step does not convey semantic value, underscoring the importance of extracting local semantic features to effectively analyze their interconnections. Consequently, we enhance the detection of subtle semantic nuances, which may elude analysis at the individual point level, by aggregating several consecutive time steps into segments, thus creating tokens at the subseries level, following PatchTST [18].

Specifically, we take each univariate input time series $X_{:,n} \in \mathbb{R}^{T\times 1}$ and segment it into patches, which may overlap or be distinctly separate. The length of each patch is defined as P, while the stride S is the distance between the start of one patch and the start of the next. This segmentation results in a series of patches $X^p_{:,n} \in \mathbb{R}^{M\times P}$, with the total number of patches given by $M = \lceil \frac{T-P}{S} \rceil + 2$. To maintain continuity at the boundary, we append S copies of the final value $X_{T,n} \in \mathbb{R}$ to the sequence before initiating the patching process.

With the use of patches, the number of input tokens can be reduced from T to approximately T/S. This implies the memory usage and computational complexity of the attention map are quadratically decreased by a factor of S. Thus constrained on the training time and GPU memory, patch design can allow the model to see the longer historical sequence.

Following, the patches are projected into the Transformer's latent space of dimension D using a trainable linear projection $W_p \in \mathbb{R}^{P\times D}$, supplemented by a learnable additive position encoding $W_{pos} \in \mathbb{R}^{M\times D}$, to maintain the temporal order of patches: $X^d_{:,n} = X^p_{:,n}W_p + W_{pos}$, where $X^d_{:,n} \in \mathbb{R}^{M\times D}$ represents the inputs for the Transformer encoder.

Henceforth, we define our input grid as $X^d = \{X^d_1, ..., X^d_M\} \in \mathbb{R}^{M\times N\times D}$. Here, $X^d_{t,:} \in \mathbb{R}^{N\times D}$ denotes the multivariate data encapsulated within the patch at step t, while $X^d_{:,n} \in \mathbb{R}^{M\times D}$ captures the complete patched time series corresponding to the specific variate.

Horizontal Attention. Horizontal attention in the GridTST model plays a crucial role in understanding the time-dependent aspects of time series data. It helps the model to analyze how each data point relates to its predecessors and successors over time. This mechanism is key for identifying and learning from patterns, trends, and changes that occur throughout the time series, thereby enabling more accurate and informed predictions about future data points. Essentially, horizontal attention ensures the model captures the sequential nature and temporal dynamics in time series data.

Concretely, take the n-th variate series as an example, we utilize horizontal attention on the patched time tokens $X^d_{:,n} \in \mathbb{R}^{M\times D}$ to model time dependencies. In the multi-head attention mechanism, each head $h = \{1, \ldots, H\}$ transforms these inputs into query matrices $Q^h_{:,n} = X^d_{:,n}W^Q_h$, key matrices $K^h_{:,n} = X^d_{:,n}W^K_h$, and value matrices $V^h_{:,n} = X^d_{:,n}W^V_h$, where $W^Q_h, W^K_h \in \mathbb{R}^{D\times d_k}$ and $W^V_h \in \mathbb{R}^{D\times D}$. The attention output $O^d_{:,n} \in \mathbb{R}^{M\times D}$ is then obtained using a scaled product:

$$O^d_{:,n} = \text{Attention}(Q^h_{:,n}, K^h_{:,n}, V^h_{:,n}) = \text{Softmax}\left(\frac{Q^h_{:,n}(K^h_{:,n})^T}{\sqrt{d_k}}\right) V^h_{:,n}. \tag{1}$$

Next, $O^d_{:,n}$ will be processed through BatchNorm layers and a feed-forward network with residual connections, as depicted in Figure 2. The overall process is summarized as:

$$O^{d,l}_{:,n} = \text{Attn}_{\text{horizontal}}(O^{d,l-1}_{:,n}), \tag{2}$$

where l denotes the layer index.

Vertical Attention. While horizontal attention focuses on understanding patterns over time, vertical attention is designed to capture the relationships between different variates at a given time step. This is crucial for accurately modeling and forecasting in scenarios where the interplay between different variables significantly influences the outcomes. The use of vertical attention also allows the model to overcome the limitations of methods that struggle with learning representations centered around variates. It ensures that the GridTST model doesn't overlook the critical inter-variable dynamics that are essential for precise forecasting in multivariate scenarios.

Specifically, vertical attention focuses on the variate tokens $X^d_{t,:} \in \mathbb{R}^{N\times D}$. Similar to horizontal attention, in this vertical multi-head attention mechanism, each head processes these inputs into query matrices $\hat{Q}^h_{t,:} = X^d_{t,:}W^{\hat{Q}}_h$, key matrices $\hat{K}^h_{t,:} = X^d_{t,:}W^{\hat{K}}_h$, and value matrices $\hat{V}^h_{t,:} = X^d_{t,:}W^{\hat{V}}_h$. Here, $W^{\hat{Q}}_h, W^{\hat{K}}_h \in \mathbb{R}^{D\times d_k}$ and $W^{\hat{V}}_h \in \mathbb{R}^{D\times D}$. The attention output $\hat{O}^d_{t,:} \in \mathbb{R}^{N\times D}$ is subsequently derived

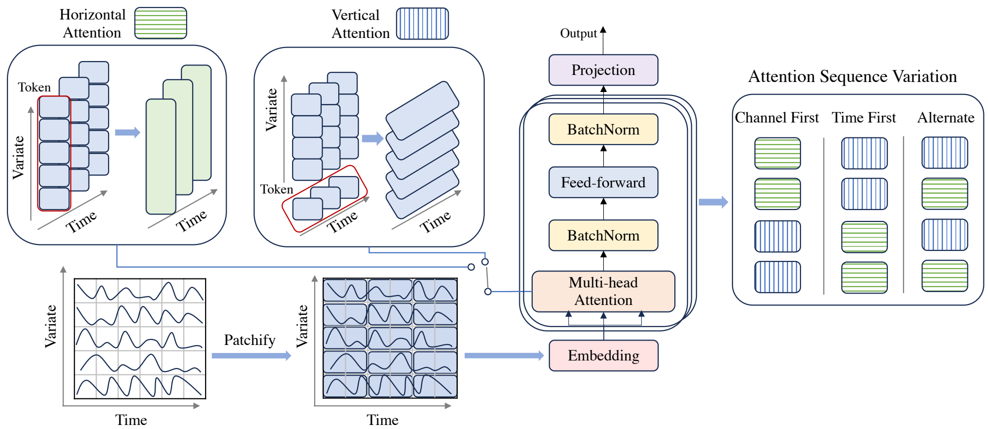

Figure 2: Overview of our proposed GridTST. Firstly, we transform the inputs by breaking them down into grids. These grids undergo processing via vanilla transformer attention, incorporating distinct horizontal and vertical directions. Finally, our model generates projected prediction results.

through a scaled product:

$$\hat{O}^{d}_{t,:} = \text{Attention}(\hat{Q}^{h}_{t,:}, \hat{K}^{h}_{t,:}, \hat{V}^{h}_{t,:}) = \text{Softmax}\left(\frac{\hat{Q}^{h}_{t,:}(\hat{K}^{h}_{t,:})^{T}}{\sqrt{d_k}}\right)\hat{V}^{h}_{t,:}. \tag{3}$$

This multi-head attention block also incorporates BatchNorm layers and a feed-forward network with residual connections. Similarly, the process is denoted as:

$$\hat{O}^{d,l}_{:,n} = \text{Attn}_{\text{vertical}}(\hat{O}^{d,l-1}_{t:}). \tag{4}$$

The core code is in Figure 1 in Appendix.

Attention Sequencing. In our GridSTS model, the order of applying horizontal and vertical attentions can be flexibly configured to optimize performance. We explored three distinct configurations: 1. Time-first: Applying horizontal attention first, followed by vertical attention. 2. Channel-first: Applying vertical attention first, followed by horizontal attention. 3. Alternate: An iterative approach where vertical and horizontal attentions are applied in an alternating manner.

Our experimental results in § 4.3 demonstrate the effectiveness of these different configurations. Notably, we discovered that the sequence starting with vertical attention and then transitioning to horizontal yields the best performance. This is likely because vertical attention first captures complex variate relationships, laying the groundwork for horizontal attention to then effectively grasp temporal patterns, resulting in more accurate forecasts.

Complexity To offer a clear perspective, consider a time series with m covariates and n patches, and a Transformer model with a hidden size of d. The computational complexity of the attention layer for PatchTST is $\mathcal{O}(n^2 d)$, whereas for GridTST, it is $\mathcal{O}\left(\frac{m^2 d}{2} + \frac{n^2 d}{2}\right)$. This implies that for datasets with a relatively small number of covariates—such as ETTh1 and ETTm1, which have seven covariates—GridTST can operate more efficiently than PatchTST. For dataset with large covariate number, we also design an efficient training algorithm by variate sampleing, as detailed in § 4.5.

Normalization. We denote the representation after the attention sequence process as $Z \in \mathbb{R}^{M \times N \times D}$. This representation is subsequently processed through a flatten layer with a linear head to generate the prediction results. Notably, we apply instance normalization before patching and instance denormalization after this linear head. This approach, recently introduced, effectively mitigates the distribution shift between training and testing data [34, 35]. Specifically, it normalizes each variate series instance $X_{t,:}$ to have zero mean and unit standard deviation. Essentially, each $X_{t,:}$ is normalized before patching, and then its mean and deviation are reintegrated into the output prediction. The final output, denoted as $\hat{Y} = \{\hat{x}_{T+1}, \ldots, \hat{x}_{T+F}\} \in \mathbb{R}^{F \times N}$, represents the forecast over the horizon F.

Loss Function. We employ the Mean Squared Error (MSE) loss to quantify the difference between our predictions and the actual ground truth. The loss for each channel is accumulated and then averaged across F time series to calculate the overall objective loss, expressed as $\mathcal{L} = \left\| \hat{X}_{L+1:L+F} - X_{L+1:L+T} \right\|_2^2$.

4 EXPERIMENTS

Datasets. We conduct experiments on seven real-world datasets to evaluate the performance of the proposed GridTST. The first four datasets are collected by [28]: *Weather* includes 21 meteorological factors collected every 10 minutes from the Weather Station of the Max Planck Biogeochemistry Institute in 2020; *Traffic* collects hourly road occupancy rates measured by 862 sensors of the San Francisco Bay area freeways from January 2015 to December 2016; *Electricity* records the hourly electricity consumption data of 321 clients; *Illness* dataset collects the number of patients and influenza-like illness ratio on a weekly frequency. Additionally, *ETTh1* and *Ettm1* by [27] contain 7 factors of electricity transformer from July 2016 to July 2018, recorded hourly. Finally, *Solar-Energy* by [22] records the solar power production of 137 PV plants in 2006, sampled every 10 minutes. Details are in Appendix A.

To ensure consistency in our evaluation, we adhere to the data processing protocol and forecasting length settings as established in TimesNet [30] and SCINet [36]. Specifically, for the Weather, Traffic, Electricity, Ett, and Solar-Energy datasets, we maintain a fixed length of 96 for the lookback series. The prediction length, however, varies and is set within the range of {96, 192, 336, 720}, allowing us to assess the model's performance across different forecast horizons.

4.1 Prediction Performance

Baselines and Experimental Settings. Our study incorporates state-of-the-art Transformer-based models as baseline comparisons. These models include iTransformer [7], PatchTST [18], Autoformer [28], CrossFormer [31], Informer [37], Pyraformer [38], along with a recent non-Transformer-based model, DLinear [17]. Each of these models adheres to a uniform experimental setup. Specifically, for the Illness dataset, the prediction length T is set within the range {24, 36, 48, 60}, while for other datasets, T spans {96, 192, 336, 720}, consistent with the methodologies detailed in their respective original papers. Notably, all models utilize a default look-back window $L = 336$, in alignment with the DLinear model's parameters. Our evaluation criteria focus on the Mean Squared Error (MSE) and Mean Absolute Error (MAE) in the context of multivariate time series forecasting.

Main results. The comprehensive forecasting results are detailed in Table 1, with the top-performing models highlighted in red and the second-best in blue. A lower MSE/MAE signifies more accurate predictions, and in this context, the proposed GridTST consistently delivers state-of-the-art performance. Notably, PatchTST, previously the leading model on the Electricity and Weather datasets, struggles in several Traffic dataset scenarios. This is likely due to the dataset's large number of variates, where PatchTST's patching mechanism might lose focus on specific localities, thus failing to handle rapid fluctuations effectively. In contrast, the iTransformer stands out in the traffic datasets, adeptly handling up to 800 variates. However, it falls short in other datasets where the number of variates is fewer and time information plays a more crucial role. This disparity highlights the iTransformer's limitations in contexts where temporal dynamics are more significant.

Our proposed method enhances this by modeling both temporal and variate aspects through bidirectional attentions. For example, it exhibits superior performance in 26 out of 28 tasks when evaluated using two metrics. By aggregating the entire series of time and variate variations for representation, it can more adeptly manage such forecasting challenges. Consequently, native Transformer components prove effective for temporal modeling and multivariate correlation, and our proposed bidirectional architecture is well-suited for tackling complex real-world time series forecasting scenarios.

Increasing lookback length. Numerous prior studies [18] have noted that increasing the lookback length does not invariably enhance forecasting performance in Transformers, a phenomenon often attributed to attention becoming dispersed over an expanding input sequence. In this context, we assessed the performance of our GridTST, along with time-centric PatchTST and variate-centric iTransformer, as illustrated in Figure 3, particularly focusing on scenarios with extended lookback lengths. The full data can be found in Table 6 to Table 9 in Appendix. It is apparent that the

Table 1: Multivariate long-term forecasting results with GridTST. We use prediction lengths $T \in \{24, 36, 48, 60\}$ for Illness dataset and $T \in \{96, 192, 336, 720\}$ for the others. The best results are in **red** and second best is in **blue**. We fix the lookback length T = 336.

Models		GridTST		iTransformer		PatchTST		DLinear		CrossFormer		Autoformer		Informer		Pyraformer	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Weather	96	**0.145**	**0.195**	0.163	0.212	0.151	0.200	0.176	0.237	0.226	0.301	0.249	0.329	0.354	0.405	0.896	0.556
	192	**0.191**	**0.241**	0.203	0.249	0.195	0.241	0.220	0.282	0.215	0.289	0.325	0.370	0.419	0.434	0.622	0.624
	336	**0.243**	**0.280**	0.255	0.289	0.247	0.281	0.265	0.319	0.319	0.317	0.351	0.391	0.583	0.543	0.739	0.753
	720	**0.316**	**0.333**	0.326	0.337	0.321	0.333	0.323	0.362	0.381	0.379	0.415	0.426	0.916	0.705	1.004	0.934
Traffic	96	**0.337**	**0.241**	0.358	0.258	0.366	0.251	0.410	0.282	0.556	0.347	0.597	0.371	0.733	0.410	2.085	0.468
	192	**0.373**	**0.259**	0.375	0.268	0.388	0.263	0.423	0.287	0.588	0.320	0.607	0.382	0.777	0.435	0.867	0.467
	336	**0.378**	**0.259**	0.389	0.274	0.398	0.265	0.436	0.296	0.601	0.347	0.623	0.387	0.776	0.434	0.869	0.469
	720	**0.403**	**0.276**	0.422	0.290	0.434	0.287	0.466	0.315	0.605	0.345	0.639	0.395	0.827	0.466	0.881	0.473
Electricity	96	**0.123**	**0.219**	0.131	0.228	0.129	0.222	0.140	0.237	0.166	0.293	0.196	0.313	0.304	0.393	0.386	0.449
	192	**0.142**	**0.237**	0.155	0.249	0.148	0.240	0.153	0.249	0.187	0.302	0.211	0.324	0.327	0.417	0.386	0.443
	336	**0.158**	**0.254**	0.170	0.266	0.166	0.259	0.169	0.267	0.205	0.324	0.214	0.327	0.333	0.422	0.378	0.443
	720	**0.186**	**0.280**	0.207	0.300	0.210	0.298	0.203	0.301	0.211	0.338	0.236	0.342	0.351	0.427	0.376	0.445
Illness	24	**1.638**	**0.833**	2.060	0.960	1.816	0.819	2.215	1.081	2.424	1.045	2.906	1.182	4.657	1.449	1.420	2.012
	36	**1.707**	**0.854**	2.151	1.020	2.098	0.978	1.963	0.963	2.411	1.011	2.585	1.038	4.650	1.463	7.394	2.031
	48	**1.699**	**0.877**	2.060	0.990	1.735	0.892	2.130	1.024	2.438	1.028	3.024	1.145	5.004	1.542	7.551	2.057
	60	**1.555**	**0.812**	2.220	1.030	1.578	0.818	2.368	1.096	2.442	1.022	2.761	1.114	5.071	1.543	7.662	2.100
ETTh1	96	**0.368**	**0.395**	0.399	0.417	0.378	0.400	0.375	0.399	0.386	0.425	0.435	0.446	0.941	0.769	0.664	0.612
	192	**0.409**	**0.418**	0.442	0.446	0.414	0.421	0.405	0.416	0.434	0.456	0.456	0.457	1.007	0.786	0.790	0.681
	336	**0.436**	**0.440**	0.463	0.462	0.440	0.440	0.439	0.443	0.481	0.472	0.486	0.487	1.038	0.784	0.891	0.738
	720	**0.451**	**0.464**	0.496	0.501	0.456	0.471	0.472	0.490	0.481	0.512	0.515	0.517	1.144	0.857	0.963	0.782
ETTm1	96	**0.279**	**0.339**	0.302	0.356	0.292	0.342	0.299	0.343	0.316	0.380	0.510	0.492	0.626	0.560	0.543	0.510
	192	**0.327**	**0.368**	0.344	0.382	0.330	0.368	0.335	0.365	0.361	0.409	0.514	0.495	0.725	0.619	0.557	0.537
	336	**0.360**	**0.388**	0.378	0.403	0.365	0.391	0.369	0.386	0.392	0.425	0.510	0.492	1.005	0.741	0.754	0.655
	720	**0.417**	**0.428**	0.438	0.437	0.417	0.424	0.425	0.423	0.446	0.458	0.527	0.493	1.133	0.845	0.908	0.724
Solar	96	**0.169**	**0.232**	0.196	0.249	0.194	0.249	0.221	0.289	0.241	0.299	0.266	0.311	0.208	0.237	0.228	0.249
	192	**0.184**	**0.241**	0.219	0.266	0.228	0.261	0.249	0.285	0.268	0.314	0.271	0.315	0.229	0.259	0.237	0.268
	336	**0.193**	**0.250**	0.218	0.266	0.221	0.254	0.263	0.291	0.288	0.311	0.281	0.317	0.235	0.272	0.247	0.278
	720	**0.203**	**0.261**	0.230	0.279	0.219	0.271	0.244	0.296	0.271	0.315	0.295	0.319	0.233	0.275	0.255	0.286

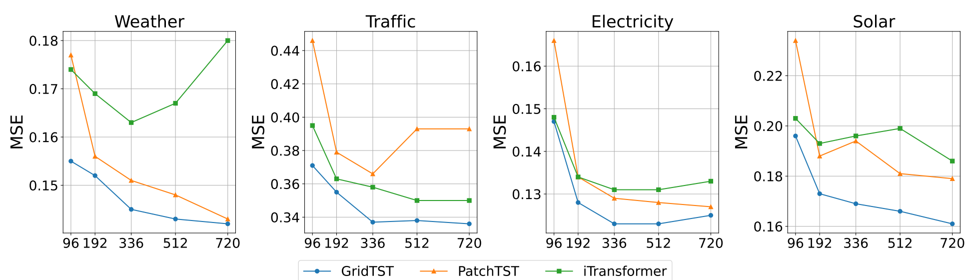

Figure 3: Forecasting Performance with Lookback Length $T \in \{96, 172, 336, 512, 720\}$ and Fixed Prediction Length $F = 96$. The performance of time-centric PatchTST or variate-centric iTransformer forecasters does not markedly improve with increased lookback length. In contrast, our GridTST framework enhances the vanilla Transformer, yielding improved performance when utilizing an enlarged lookback window.

variate-centric iTransformer exhibits subpar performance on the Weather dataset, which comprises a limited number of variates, while the time-centric PatchTST underperforms on the Traffic dataset, characterized by hundreds of variates. Conversely, our model demonstrates consistent enhancement in performance with the expansion of the lookback window size, irrespective of whether the number of variates is large or small. This improvement is particularly notable in the traffic and electricity datasets, where the number of variate channels is substantial. These findings validate the effectiveness of applying bidirectional attention across both temporal and variate dimensions, enabling Transformers to harness an extended lookback window. This approach effectively models complex relationships among hundreds of variables, thereby yielding more precise predictions. It's crucial to note that PatchTST and GridTST both utilize patching, whereas iTransformer relies on mapping. Our patch-based methods provide flexibility by allowing us to adjust patch lengths based on the lookback window size. This adaptability is a distinct advantage compared to mapping-based methods, which lack the capability to tailor their approach for different lookback windows.

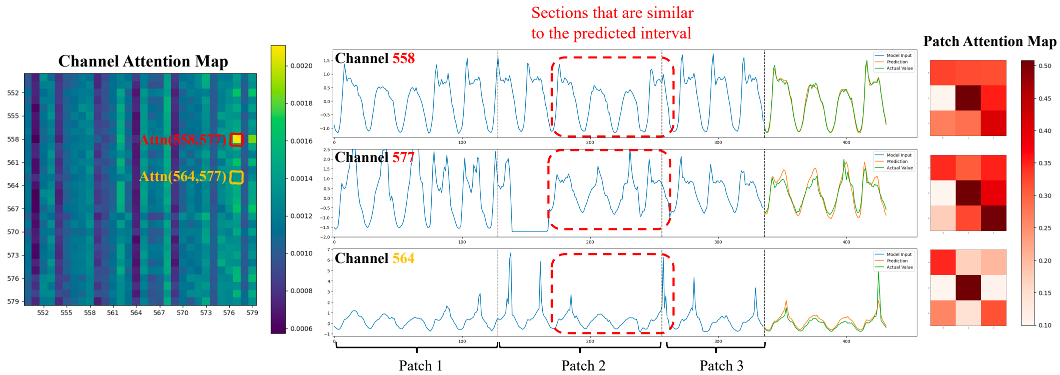

Figure 4: Visualization of attention maps and time series forecasts from the Traffic dataset. For each time series, the input data is represented in blue, the GridTST model's predictions in orange, and the actual observed values in green. The three black demarcation lines indicate the three segmented patches. The synergy of horizontal and vertical attention mechanisms enables the model to refine its forecasts by concentrating on the spatial and temporal information deemed most pertinent.

4.2 Visualization Case Study

As shown in Figure 4, we present an intuitive visualization of the multivariate and multi-time-step correlations via attention maps and the forecasting outcomes for three selected time series from the Traffic dataset, processed using GridTST in the channel-first arrangement. The model configuration includes a lookback length of 336, a fixed prediction length of 96, and a patch length also set to 96. The attention maps depicted are the result of averaging the attention matrices across all heads.

The channel attention map on the left highlights discernible correlations that mirror the inherent trends in the raw lookback series. Notably, time series 558 and 577 exhibit more closely aligned trends in contrast to series 564 and 577, which are not highlighted. This observation substantiates the model's proficiency in detecting inter-channel similarities by leveraging vertical attention mechanisms. Such mechanisms enable mutual reinforcement between analogous time series, thereby enhancing the prediction capabilities by leveraging the diversity of the channels.

Delving into the patch attention maps on the right, we observe that time series with similar trends share similar patch attention distributions. Furthermore, these maps frequently emphasize segments that mirror the predicted intervals, as delineated by the red dashed lines. This tendency underscores the significance of historical patterns within the data, and the horizontal attention's adeptness at capturing these temporal relationships.

The central temporal graphs demonstrate the model's adeptness at pinpointing critical time patches while simultaneously conducting inter-channel comparisons. The synergy of horizontal and vertical attention mechanisms enables the model to forecast by focusing on the spatial and temporal information deemed most pertinent.

4.3 Ablation study

Table 2 presents an ablation study on various attention sequence arrangements, offering valuable insights into the performance dynamics of different models. It is evident that all models achieve relatively good performance, underscoring the robustness and consistency of our proposed model. Specifically, the channel-first arrangement emerges as the most effective, outperforming other configurations in more metrics and settings. This approach aligns intuitively with human cognitive processes, where we typically analyze variables at a single time point before extending our analysis along the timeline. We select attention strategy for different dataset by the their performance on the validation set.

Table 2: Ablation study of GridTST

Models		Channel First		Time First		Alternate	
Metric		MSE	MAE	MSE	MAE	MSE	MAE
Weather	96	**0.145**	**0.195**	0.149	0.198	0.146	0.196
	192	0.191	0.240	0.193	**0.240**	**0.190**	**0.238**
	336	0.243	0.280	0.241	**0.277**	**0.240**	0.278
	720	0.316	0.333	**0.315**	**0.331**	0.335	0.342
Traffic	96	**0.337**	**0.241**	0.352	0.254	0.351	0.250
	192	**0.373**	**0.259**	0.382	0.270	0.381	0.267
	336	**0.378**	**0.260**	0.386	0.272	0.384	0.267
	720	**0.403**	**0.276**	0.414	0.285	0.414	0.285
Electricity	96	**0.123**	**0.219**	0.125	0.222	0.125	0.222
	192	**0.142**	**0.237**	0.146	0.242	0.145	0.241
	336	**0.158**	**0.254**	0.166	0.263	0.158	0.255
	720	0.206	0.295	0.191	0.286	**0.186**	**0.280**
ETTh1	96	**0.368**	**0.395**	0.374	0.397	0.370	0.396
	192	**0.409**	**0.418**	0.413	0.419	0.412	0.420
	336	**0.436**	0.440	**0.436**	**0.439**	0.439	0.440
	720	0.462	0.474	**0.450**	**0.464**	0.451	**0.464**

Table 3: Speedup ratio of our framework with BetterTransformer Library.

Input_length	64	128	256	512	1024	2048	4096
GPU setting	x1.49	x1.62	x1.62	x1.80	x2.04	x2.46	x2.62
CPU setting	x1.19	x1.54	x1.34	x1.21	x1.17	x1.24	x1.20

4.4 Scalability of GridTST

One advantage of adopting the vanilla Transformer architecture is the ability to tap into the established ecosystem that has been purposefully developed to optimize Transformer performance on CPUs and GPUs. By leveraging specialized techniques such as sparsity and fused kernels, significant speedups can be achieved. For instance, using the BetterTransformer library, a single line of code can dramatically accelerate processing times of our GridTST by utilizing nested tensors for sparsity and integrating flash-attention [39]. This convenience is not available for various heavily modified Transformer structures [31, 38], and thus cannot benefit from the existing ecosystem. As demonstrated in Table 3, on GPUs, the speedup increases with the length of the input sequence, indicating that GridTST is particularly effective when scaling to extremely long time series. On CPUs, the acceleration remains consistent, making it a viable option for deployment scenarios.

4.5 Efficient training strategy

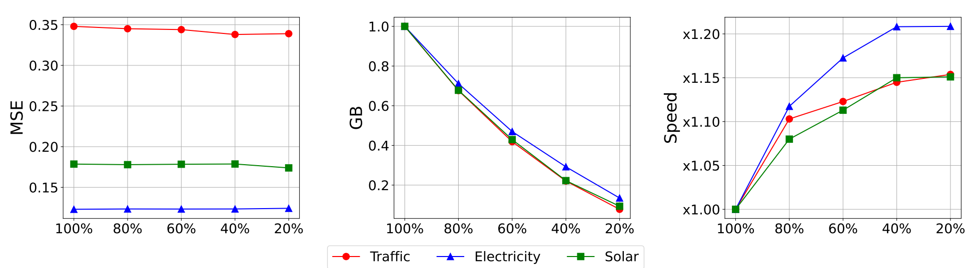

Figure 5: Analysis of the Proposed Training Strategy. The performance (Left) maintains stable across partially trained variants of each batch, with the sampled ratio ranging from 20% to 100%. Concurrently, there is a notable reduction in both the memory footprint (Middle) and the latency (Right) of the training process.

In the Transformer architecture, the self-attention mechanism's quadratic complexity can become burdensome during training as the number of variables increases. One approach to mitigate this complexity is to randomly select a subset of variables for each training batch, focusing the model's training on these chosen variables. In Figure 5, we demonstrate the effectiveness of this method by training the model with a randomly sampled set of variables and evaluating its performance. Full data can be found in Table 10 in Appendix. The results reveal that our proposed strategy achieves performance comparable to training with the full set of variables, while significantly reducing memory usage and increasing training speed. This underscores the effectiveness of our vertical attention approach in capturing variable dependencies from a limited set of training samples.

5 Conclusion and Broader Impacts

In our study, we introduced GridTST, an innovative model for time series prediction with broad applications in fields like finance, economics, climate, and healthcare. GridTST combines the strengths of two prevailing approaches by treating time series data as a grid, incorporating both time and variate dimensions. Our model employs horizontal and vertical attentions to efficiently capture temporal and multivariate correlations, enhancing analytical capabilities. We consistently achieved state-of-the-art performance on real-world datasets, showcasing GridTST's effectiveness. For future work, we aim to delve into another direction by extending GridTST's capabilities to address multi-modal time series data.

References

[1] Shiyang Li, Xiaoyong Jin, Yao Xuan, Xiyou Zhou, Wenhu Chen, Yu-Xiang Wang, and Xifeng Yan. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. *Advances in neural information processing systems*, 32, 2019.

[2] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. *Data mining and knowledge discovery*, 33(4):917–963, 2019.

[3] Anastasia Borovykh, Sander Bohte, and Cornelis W Oosterlee. Conditional time series forecasting with convolutional neural networks. *arXiv preprint arXiv:1703.04691*, 2017.

[4] Daizong Ding, Mi Zhang, Xudong Pan, Min Yang, and Xiangnan He. Modeling extreme events in time series prediction. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1114–1122, 2019.

[5] Qingsong Wen, Tian Zhou, Chaoli Zhang, Weiqi Chen, Ziqing Ma, Junchi Yan, and Liang Sun. Transformers in time series: A survey. *arXiv preprint arXiv:2202.07125*, 2022.

[6] Thomas Hartvigsen, Walter Gerych, and Marzyeh Ghassemi. On detecting covid-risky behavior from smartphones. In *epiDAMIK 5.0: The 5th International workshop on Epidemiology meets Data Mining and Knowledge discovery at KDD 2022*, 2022.

[7] Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. *ICLR*, 2024.

[8] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[9] Luciano Floridi and Massimo Chiriatti. Gpt-3: Its nature, scope, limits, and consequences. *Minds and Machines*, 30:681–694, 2020.

[10] Xin Cheng, Di Luo, Xiuying Chen, Lemao Liu, Dongyan Zhao, and Rui Yan. Lift yourself up: Retrieval-augmented text generation with self memory. *arXiv preprint arXiv:2305.02437*, 2023.

[11] Xiuying Chen, Mingzhe Li, Shen Gao, Zhangming Chan, Dongyan Zhao, Xin Gao, Xiangliang Zhang, and Rui Yan. Follow the timeline! generating an abstractive and extractive timeline summary in chronological order. *ACM Transactions on Information Systems*, 41(1):1–30, 2023.

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2021.

[13] Mingzhe Li, Xiuying Chen, Shen Gao, Zhangming Chan, Dongyan Zhao, and Rui Yan. Vmsmo: Learning to generate multimodal summary for video-based news articles. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9360–9369, 2020.

[14] Pulak Agarwal, Pranav Aluru, and B Aditya Prakash. Real-time anomaly detection in epidemic data streams. In *epiDAMIK 5.0: The 5th International workshop on Epidemiology meets Data Mining and Knowledge discovery at KDD 2022*, 2022.

[15] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International Conference on Machine Learning*, pages 27268–27286. PMLR, 2022.

[16] Laila Melkas, Rafael Savvides, Suyog H Chandramouli, Jarmo Mäkelä, Tuomo Nieminen, Ivan Mammarella, and Kai Puolamäki. Interactive causal structure discovery in earth system sciences. In *The KDD'21 Workshop on Causal Discovery*, pages 3–25. PMLR, 2021.

[17] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 11121–11128, 2023.

[18] Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *The Eleventh International Conference on Learning Representations*, 2022.

[19] Keisuke Kiritoshi, Tomonori Izumitani, Kazuki Koyama, Tomomi Okawachi, Keisuke Asahara, and Shohei Shimizu. Estimating individual-level optimal causal interventions combining causal models and machine learning models. In *The KDD'21 Workshop on Causal Discovery*, pages 55–77. PMLR, 2021.

[20] Rose Yu, Stephan Zheng, Anima Anandkumar, and Yisong Yue. Long-term forecasting using tensor-train rnns. *Arxiv*, 2017.

[21] David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 36(3):1181–1191, 2020.

[22] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, pages 95–104, 2018.

[23] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[24] Rajat Sen, Hsiang-Fu Yu, and Inderjit S Dhillon. Think globally, act locally: A deep neural network approach to high-dimensional time series forecasting. *Advances in neural information processing systems*, 32, 2019.

[25] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

[26] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? *arXiv preprint arXiv:2205.13504*, 2022.

[27] Si-An Chen, Chun-Liang Li, Nate Yoder, Sercan O Arik, and Tomas Pfister. Tsmixer: An all-mlp architecture for time series forecasting. *arXiv preprint arXiv:2303.06053*, 2023.

[28] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in Neural Information Processing Systems*, 34:22419–22430, 2021.

[29] Gerald Woo, Chenghao Liu, Doyen Sahoo, Akshat Kumar, and Steven Hoi. Etsformer: Exponential smoothing transformers for time-series forecasting. *arXiv preprint arXiv:2202.01381*, 2022.

[30] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. *arXiv preprint arXiv:2210.02186*, 2022.

[31] Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *The Eleventh International Conference on Learning Representations*, 2022.

[32] 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. Time-llm: Time series forecasting by reprogramming large language models. *arXiv preprint arXiv:2310.01728*, 2023.

[33] Tian Zhou, Peisong Niu, Xue Wang, Liang Sun, and Rong Jin. One fits all: Power general time series analysis by pretrained lm. *arXiv preprint arXiv:2302.11939*, 2023.

[34] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016.

[35] Taesung Kim, Jinhee Kim, Yunwon Tae, Cheonbok Park, Jang-Ho Choi, and Jaegul Choo. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International Conference on Learning Representations*, 2021.

[36] Minhao Liu, Ailing Zeng, Muxi Chen, Zhijian Xu, Qiuxia Lai, Lingna Ma, and Qiang Xu. Scinet: Time series modeling and forecasting with sample convolution and interaction. *Advances in Neural Information Processing Systems*, 35:5816–5828, 2022.

[37] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 11106–11115, 2021.

[38] Shizhan Liu, Hang Yu, Cong Liao, Jianguo Li, Weiyao Lin, Alex X Liu, and Schahram Dustdar. Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In *International conference on learning representations*, 2021.

[39] Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*, 35:16344–16359, 2022.

A Dataset

Table 4: Detailed dataset descriptions. Channels denotes the variate number of each dataset. Dataset Partition denotes the ratio number of time points in (Train, Validation, Test) split respectively. Prediction Length denotes the future time points to be predict and four prediction settings are included in each dataset. Frequency denotes the sampling interval of time points.

Dataset	# Channels	# TimeSteps	Prediction Length	Dataset Partition	Frequency	Category
Weather	21	52696	{96,192,336,720}	7:1:2	10min	Weather
Traffic	862	17544	{96,192,336,720}	7:1:2	Hourly	Transportation
Electricity	321	26304	{96,192,336,720}	7:1:2	Hourly	Electricity
Illness	7	966	{12,24,48,60}	7:1:2	Weekly	Illness
Etth1	7	17420	{96,192,336,720}	6:2:2	Hourly	Electricity
Ettm1	7	69680	{96,192,336,720}	6:2:2	15min	Electricity
Solar	137	52560	{96,192,336,720}	7:1:2	10min	Energy

We conduct experiments on seven real-world datasets to evaluate the performance of the proposed GridTST. The first four datasets are collected by [28]: *Weather* includes 21 meteorological factors collected every 10 minutes from the Weather Station of the Max Planck Biogeochemistry Institute in 2020; *Traffic* collects hourly road occupancy rates measured by 862 sensors of the San Francisco Bay area freeways from January 2015 to December 2016; *Electricity* records the hourly electricity consumption data of 321 clients; *Illness* dataset collects the number of patients and influenza-like illness ratio on a weekly frequency. Additionally, *ETTh1* and *Ettm1* by [27] contain 7 factors of electricity transformer from July 2016 to July 2018, recorded hourly. Finally, *Solar-Energy* by [22] records the solar power production of 137 PV plants in 2006, sampled every 10 minutes.

B Comparison with existing baselines

Table 5: Comparison of Transformer Models

	Vanilla Transformer	Multivariate Modeling	Sequential Modeling
DLinear	×	×	×
CrossFormer	×	✓	✓
PatchTST	✓	×	✓
iTransformer	✓	✓	×
GridTST	✓	✓	✓

Table 5 presents a clear comparison of various transformer models, with GridTST emerging as the most versatile. Unlike DLinear, CrossFormer, and PatchTST, GridTST supports both multivariate and sequential modeling, alongside leveraging the foundational vanilla Transformer architecture. This dual capability allows GridTST to effectively capture the intricate dynamics of time series data. While iTransformer also utilizes the vanilla Transformer framework and multivariate modeling, it falls short in sequential modeling, which GridTST accommodates. Therefore, GridTST stands out for its comprehensive approach, making it well-suited for complex time series analysis that demands capturing both simultaneous variable relationships and sequential dependencies.

C Increasing lookback length

The performance of data with different lookback lengths is available in Table 6 to Table 9.

Listing 1: GridTST forward pass with alternate attention.

```
def gridtst_forward_with_alternate_attention(time_series,
    GridTSTLayers):
    """
    The input time_series is a 4D tensor of shape: [batch_size,
         num_variates, num_patches,
    d_model]
    """
    batch_size, num_variates, num_patches, d_model =
        time_series.shape
    time_series = time_series.view(batch_size * num_variates,
        num_patches, d_model)

    for idx, layer in enumerate(GridTSTLayers):
        if idx % 2 == 0:
            time_series = layer(time_series)
        else:
            time_series = time_series.reshape(-1, num_variates,
                 num_patches, d_model)
            time_series = time_series.permute(0, 2, 1, 3)
            time_series = time_series.reshape(-1, num_variates,
                 d_model)

            time_series = layer(time_series)

            time_series = time_series.reshape(-1, num_patches,
                num_variates, d_model)
            time_series = time_series.permute(0, 2, 1, 3)
            time_series = time_series.reshape(-1, num_patches,
                d_model)
    return time_series
```

Table 6: Performance of GridTST, iTransformer and PatchTST when using lookback windows of different lengths, The prediction length is fixed at 96. The performance is measured by MSE.

Models		Channel First		Time First		Alternate	
Metric		MSE	MAE	MSE	MAE	MSE	MAE
Weather	96	0.155	0.2	0.177	0.218	0.174	0.214
	192	0.206	0.247	0.225	0.259	0.221	0.254
	336	0.264	0.289	0.278	0.297	0.278	0.296
	720	0.343	0.343	0.354	0.348	0.358	0.349
Traffic	96	0.371	0.247	0.446	0.283	0.395	0.268
	192	0.396	0.256	0.452	0.285	0.417	0.276
	336	0.406	0.261	0.467	0.291	0.433	0.283
	720	0.436	0.282	0.5	0.309	0.467	0.302
Electricity	96	0.147	0.239	0.166	0.252	0.148	0.24
	192	0.162	0.256	0.175	0.261	0.162	0.253
	336	0.178	0.272	0.19	0.277	0.178	0.269
	720	0.214	0.305	0.23	0.311	0.225	0.317
Solar	96	0.196	0.242	0.234	0.286	0.203	0.235
	192	0.225	0.265	0.267	0.31	0.233	0.261
	336	0.245	0.281	0.29	0.315	0.248	0.273
	720	0.251	0.291	0.289	0.317	0.249	0.275

Table 7: Performance of GridTST, iTransformer and PatchTST when using lookback windows of different lengths, The prediction length is fixed at 192. The performance is measured by MSE.

Models		Channel First		Time First		Alternate	
Metric		MSE	MAE	MSE	MAE	MSE	MAE
Weather	96	0.152	0.199	0.156	0.201	0.169	0.216
	192	0.195	0.241	0.202	0.243	0.214	0.255
	336	0.252	0.285	0.252	0.285	0.266	0.293
	720	0.329	0.339	0.331	0.336	0.341	0.345
Traffic	96	0.355	0.249	0.379	0.253	0.363	0.258
	192	0.385	0.264	0.398	0.261	0.39	0.268
	336	0.397	0.274	0.411	0.268	0.399	0.282
	720	0.424	0.291	0.443	0.288	0.434	0.294
Electricity	96	0.128	0.223	0.134	0.225	0.134	0.228
	192	0.147	0.241	0.151	0.242	0.156	0.25
	336	0.16	0.256	0.169	0.262	0.171	0.266
	720	0.19	0.283	0.206	0.295	0.195	0.288
Solar	96	0.173	0.23	0.188	0.241	0.193	0.232
	192	0.196	0.262	0.221	0.256	0.223	0.26
	336	0.211	0.263	0.228	0.264	0.231	0.271
	720	0.214	0.266	0.231	0.274	0.227	0.27

Table 8: Performance of GridTST, iTransformer and PatchTST when using lookback windows of different lengths, The prediction length is fixed at 512. The performance is measured by MSE.

Models		Channel First		Time First		Alternate	
Metric		MSE	MAE	MSE	MAE	MSE	MAE
Weather	96	0.143	0.194	0.148	0.198	0.167	0.219
	192	0.189	0.238	0.192	0.24	0.208	0.254
	336	0.241	0.278	0.243	0.28	0.266	0.294
	720	0.313	0.33	0.313	0.332	0.341	0.345
Traffic	96	0.338	0.24	0.393	0.283	0.35	0.256
	192	0.358	0.251	0.408	0.29	0.376	0.268
	336	0.376	0.261	0.419	0.294	0.386	0.274
	720	0.399	0.274	0.452	0.313	0.417	0.289
Electricity	96	0.123	0.22	0.128	0.221	0.131	0.227
	192	0.144	0.24	0.149	0.244	0.153	0.25
	336	0.162	0.258	0.163	0.259	0.168	0.264
	720	0.194	0.285	0.199	0.291	0.191	0.284
Solar	96	0.166	0.232	0.181	0.239	0.199	0.242
	192	0.174	0.242	0.187	0.256	0.207	0.269
	336	0.176	0.244	0.202	0.267	0.223	0.266
	720	0.212	0.275	0.221	0.287	0.223	0.284

Table 9: Performance of GridTST, iTransformer and PatchTST when using lookback windows of different lengths, The prediction length is fixed at 720. The performance is measured by MSE.

Models		Channel First		Time First		Alternate	
Metric		MSE	MAE	MSE	MAE	MSE	MAE
Weather	96	0.142	0.194	0.143	0.193	0.18	0.231
	192	0.188	0.238	0.188	0.239	0.225	0.266
	336	0.239	0.277	0.242	0.283	0.285	0.311
	720	0.314	0.331	0.306	0.327	0.352	0.357
Traffic	96	0.336	0.244	0.393	0.288	0.35	0.256
	192	0.351	0.251	0.402	0.29	0.364	0.265
	336	0.376	0.263	0.418	0.297	0.383	0.272
	720	0.438	0.304	0.456	0.317	0.41	0.286
Electricity	96	0.125	0.221	0.13	0.225	0.133	0.229
	192	0.152	0.248	0.155	0.256	0.154	0.249
	336	0.17	0.267	0.188	0.269	0.168	0.265
	720	0.199	0.293	0.201	0.296	0.191	0.285
Solar	96	0.161	0.23	0.171	0.232	0.186	0.237
	192	0.189	0.267	0.199	0.269	0.202	0.27
	336	0.202	0.276	0.211	0.279	0.217	0.283
	720	0.229	0.306	0.231	0.298	0.221	0.283

D Efficiency

We show the MAE, MAE, peak memory, and training speed on different datasets with different variant sample ratios in Table 10.

Table 10: Ablation study on different variate sample ratios. The lookback window size is 336 and the prediction length is 96.

Variant sample ratio	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
Traffic										
MSE	0.339	0.339	0.338	0.338	0.341	0.344	0.3467	0.3451	0.3482	0.348
MAE	0.242	0.243	0.242	0.241	0.242	0.243	0.2464	0.2438	0.2471	0.246
Peak Memory	0.033	0.0782	0.1397	0.3084	0.4189	0.5355	0.6764	0.8251	0.9821	1
Training Speed	0.8424	0.8667	0.9322	0.956	0.9462	0.9732	0.9646	0.9922	0.9813	1
Electricity										
MSE	0.1254	0.1242	0.1237	0.1234	0.1233	0.1234	0.1234	0.1234	0.123	0.123
MAE	0.2197	0.2187	0.2184	0.2186	0.2188	0.2188	0.2191	0.219	0.219	0.219
Peak Memory	0.0575	0.1349	0.2077	0.2923	0.3765	0.47	0.5986	0.7115	0.8342	1
Training Speed	0.8492	0.8274	0.8587	0.9245	0.9697	0.9997	0.1003	0.9702	0.9893	1
Solar										
MSE	0.1775	0.1739	0.18	0.1787	0.1784	0.1777	0.1779	0.1785	0.1786	0.1786
MAE	0.2395	0.2368	0.2488	0.2494	0.25	0.2495	0.2503	0.2521	0.2511	0.2516
Peak Memory	0.0376	0.0936	0.152	0.2266	0.3248	0.4293	0.5454	0.6779	0.8317	1
Training Speed	0.8439	0.8688	0.9263	0.9388	0.9633	0.9671	0.9897	0.9995	1.0078	1

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: **[TODO]**

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: In the final section.

   Guidelines:

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

3. **Theory Assumptions and Proofs**

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

   Answer: [Yes]

   Justification: See model section.

   Guidelines:

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

- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

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

   Answer: [Yes]

   Justification: We will release code.

   Guidelines:

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

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

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

   Answer: [Yes]

   Justification: We will release code.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).

- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

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

   Answer: [Yes]

   Justification: See dataset section.

   Guidelines:

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

7. **Experiment Statistical Significance**

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

   Answer: [Yes]

   Justification: See experiment section.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

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

Answer: [Yes]

Justification: See efficiency section.

Guidelines:

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

9. **Code Of Ethics**

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

Answer: [Yes]

Guidelines:

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

10. **Broader Impacts**

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

Answer: [Yes]

Justification: See last section.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [Yes]

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [NA]

    Guidelines:

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

13. **New Assets**

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

    Answer: [NA]

    Guidelines:

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

14. **Crowdsourcing and Research with Human Subjects**

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

Answer: [NA]

Justification: Not applicable.

Guidelines:

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

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

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

Answer: [NA]

Justification: Not applicable.

Guidelines:

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