Transformers in Time Series: A Survey

Qingsong Wen[1], **Tian Zhou**[2], **Chaoli Zhang**[2], **Weiqi Chen**[2], **Ziqing Ma**[2], **Junchi Yan**[3], **Liang Sun**[1]
[1]DAMO Academy, Alibaba Group, Bellevue, USA
[2]DAMO Academy, Alibaba Group, Hangzhou, China
[3]Department of CSE, MoE Key Lab of Artificial Intelligence, Shanghai Jiao Tong University
{qingsong.wen, tian.zt, chaoli.zcl, jarvus.cwq, maziqing.mzq, liang.sun}@alibaba-inc.com,
yanjunchi@sjtu.edu.cn

Abstract

Transformers have achieved superior performances in many tasks in natural language processing and computer vision, which also triggered great interest in the time series community. Among multiple advantages of Transformers, the ability to capture long-range dependencies and interactions is especially attractive for time series modeling, leading to exciting progress in various time series applications. In this paper, we systematically review Transformer schemes for time series modeling by highlighting their strengths as well as limitations. In particular, we examine the development of time series Transformers in two perspectives. From the perspective of network structure, we summarize the adaptations and modifications that have been made to Transformers in order to accommodate the challenges in time series analysis. From the perspective of applications, we categorize time series Transformers based on common tasks including forecasting, anomaly detection, and classification. Empirically, we perform robust analysis, model size analysis, and seasonal-trend decomposition analysis to study how Transformers perform in time series. Finally, we discuss and suggest future directions to provide useful research guidance. A corresponding resource that has been continuously updated can be found in the GitHub repository[1].

1 Introduction

The innovation of Transformer in deep learning [Vaswani *et al.*, 2017] has brought great interests recently due to its excellent performances in natural language processing (NLP) [Kenton and others, 2019], computer vision (CV) [Dosovitskiy *et al.*, 2021], and speech processing [Dong *et al.*, 2018]. Over the past few years, numerous Transformers have been proposed to advance the state-of-the-art performances of various tasks significantly. There are quite a few literature reviews from different aspects, such as in NLP applications [Han *et al.*, 2021], CV applications [Han *et al.*, 2022], and efficient Transformers [Tay *et al.*, 2022].

[1] https://github.com/qingsongedu/time-series-transformers-review

Transformers have shown great modeling ability for long-range dependencies and interactions in sequential data and thus are appealing to time series modeling. Many variants of Transformer have been proposed to address special challenges in time series modeling and have been successfully applied to various time series tasks, such as forecasting [Li *et al.*, 2019; Zhou *et al.*, 2022], anomaly detection [Xu *et al.*, 2022; Tuli *et al.*, 2022], and classification [Zerveas *et al.*, 2021; Yang *et al.*, 2021]. Specifically, seasonality or periodicity is an important feature of time series [Wen *et al.*, 2021a]. How to effectively model long-range and short-range temporal dependency and capture seasonality simultaneously remains a challenge [Wu *et al.*, 2021; Wen *et al.*, 2022]. We note that there exist several surveys related to deep learning for time series, including forecasting [Lim and Zohren, 2021; Benidis *et al.*, 2022; Torres *et al.*, 2021], classification [Ismail Fawaz *et al.*, 2019], anomaly detection [Choi *et al.*, 2021; Blázquez-García *et al.*, 2021], and data augmentation [Wen *et al.*, 2021b], but there is no comprehensive survey for Transformers in time series. As Transformer for time series is an emerging subject in deep learning, a systematic and comprehensive survey on time series Transformers would greatly benefit the time series community.

In this paper, we aim to fill the gap by summarizing the main developments of time series Transformers. We first give a brief introduction about vanilla Transformer, and then propose a new taxonomy from perspectives of both network modifications and application domains for time series Transformers. For network modifications, we discuss the improvements made on both low-level (i.e. module) and high-level (i.e. architecture) of Transformers, with the aim to optimize the performance of time series modeling. For applications, we analyze and summarize Transformers for popular time series tasks including forecasting, anomaly detection, and classification. For each time series Transformer, we analyze its insights, strengths, and limitations. To provide practical guidelines on how to effectively use Transformers for time series modeling, we conduct extensive empirical studies that examine multiple aspects of time series modeling, including robustness analysis, model size analysis, and seasonal-trend decomposition analysis. We conclude this work by discussing possible future directions for time series Transformers, including inductive biases for time series Transformers, Transformers and GNN for time series, pre-trained Transformers

for time series, Transformers with architecture level variants, and Transformers with NAS for time series. To the best of our knowledge, this is the first work to comprehensively and systematically review the key developments of Transformers for modeling time series data. We hope this survey will ignite further research interests in time series Transformers.

2 Preliminaries of the Transformer

2.1 Vanilla Transformer

The vanilla Transformer [Vaswani *et al.*, 2017] follows most competitive neural sequence models with an encoder-decoder structure. Both encoder and decoder are composed of multiple identical blocks. Each encoder block consists of a multi-head self-attention module and a position-wise feed-forward network while each decoder block inserts cross-attention models between the multi-head self-attention module and the position-wise feed-forward network.

2.2 Input Encoding and Positional Encoding

Unlike LSTM or RNN, the vanilla Transformer has no recurrence. Instead, it utilizes the positional encoding added in the input embeddings, to model the sequence information. We summarize some positional encodings below.

Absolute Positional Encoding

In vanilla Transformer, for each position index t, encoding vector is given by

$$PE(t)_i = \begin{cases} \sin(\omega_i t) & i\%2 = 0 \\ \cos(\omega_i t) & i\%2 = 1 \end{cases} \quad (1)$$

where ω_i is the hand-crafted frequency for each dimension. Another way is to learn a set of positional embeddings for each position which is more flexible [Kenton and others, 2019; Gehring *et al.*, 2017].

Relative Positional Encoding

Following the intuition that pairwise positional relationships between input elements is more beneficial than positions of elements, relative positional encoding methods have been proposed. For example, one of such methods is to add a learnable relative positional embedding to keys of attention mechanism [Shaw *et al.*, 2018].

Besides the absolute and relative positional encodings, there are methods using hybrid positional encodings that combine them together [Ke *et al.*, 2021]. Generally, the positional encoding is added to the token embedding and fed to Transformer.

2.3 Multi-head Attention

With Query-Key-Value (QKV) model, the scaled dot-product attention used by Transformer is given by

$$Attention(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = softmax(\frac{\mathbf{Q}\mathbf{K}^\mathbf{T}}{\sqrt{D_k}})\mathbf{V} \quad (2)$$

where queries $\mathbf{Q} \in \mathcal{R}^{N \times D_k}$, keys $\mathbf{K} \in \mathcal{R}^{M \times D_k}$, values $\mathbf{V} \in \mathcal{R}^{M \times D_v}$, N, M denote the lengths of queries and keys (or values), and D_k, D_v denote the dimensions of keys (or queries) and values. Transformer uses multi-head attention

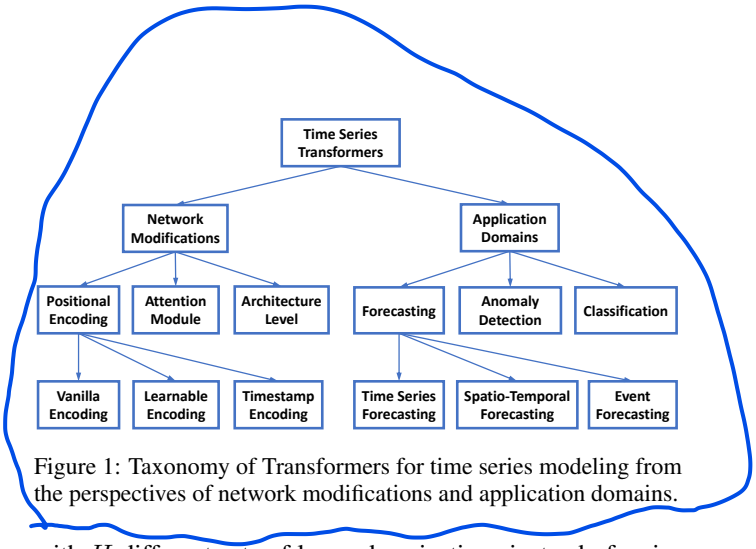

Figure 1: Taxonomy of Transformers for time series modeling from the perspectives of network modifications and application domains.

with H different sets of learned projections instead of a single attention function as

$$\text{MultiHeadAttn}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Concat}\,(head_1, \cdots, head_H)\mathbf{W}^O,$$

where $head_i = Attention(\mathbf{Q}\mathbf{W}_i^Q, \mathbf{K}\mathbf{W}_i^K, \mathbf{V}\mathbf{W}_i^V)$.

2.4 Feed-forward and Residual Network

The feed-forward network is a fully connected module as

$$FFN(\mathbf{H}') = ReLU(\mathbf{H}'\mathbf{W}^1 + \mathbf{b}^1)\mathbf{W}^2 + \mathbf{b}^2, \quad (3)$$

where $\mathbf{H}'$ is outputs of previous layer, $\mathbf{W}^1 \in \mathcal{R}^{D_m \times D_f}$, $\mathbf{W}^2 \in \mathcal{R}^{D_f \times D_m}$, $\mathbf{b}^1 \in \mathcal{R}^{D_f}$, $\mathbf{b}^2 \in \mathcal{R}^{D_m}$ are trainable parameters. In a deeper module, a residual connection module followed by a layer normalization module is inserted around each module. That is,

$$\mathbf{H}' = \text{LayerNorm}(SelfAttn(\mathbf{X}) + \mathbf{X}), \quad (4)$$

$$\mathbf{H} = \text{LayerNorm}(FFN(\mathbf{H}') + \mathbf{H}'), \quad (5)$$

where $SelfAttn(.)$ denotes self-attention module and $LayerNorm(.)$ denotes the layer normalization operation.

3 Taxonomy of Transformers in Time Series

To summarize the existing time series Transformers, we propose a taxonomy from perspectives of network modifications and application domains as illustrated in Fig. 1. Based on the taxonomy, we review the existing time series Transformers systematically. From the perspective of network modifications, we summarize the changes made on both module level and architecture level of Transformer in order to accommodate special challenges in time series modeling. From the perspective of applications, we classify time series Transformers based on their application tasks including forecasting, anomaly detection, and classification. In the following two sections, we would delve into the existing time series Transformers from these two perspectives.

4 Network Modifications for Time Series

4.1 Positional Encoding

As the ordering of time series matters, it is of great importance to encode the positions of input time series into Transformers. A common design is to first encode positional information as vectors and then inject them into the model as an

additional input together with the input time series. How to obtain these vectors when modeling time series with Transformers can be divided into three main categories.

Vanilla Positional Encoding. A few works [Li *et al.*, 2019] simply introduce vanilla positional encoding (Section 2.2) used in [Vaswani *et al.*, 2017], which is then added to the input time series embeddings and fed to Transformer. Although this approach can extract some positional information from time series, they were unable to fully exploit the important features of time series data.

Learnable Positional Encoding. As the vallina positional encoding is hand-crafted and less expressive and adaptive, several studies found that learning appropriate positional embeddings from time series data can be much more effective. Compared to fixed vanilla positional encoding, learned embeddings are more flexible and can adapt to specific tasks. [Zerveas *et al.*, 2021] introduces an embedding layer in Transformer that learns embedding vectors for each position index jointly with other model parameters. [Lim *et al.*, 2021] uses an LSTM network to encode positional embeddings, which can better exploit sequential ordering information in time series.

Timestamp Encoding. When modeling time series in real-world scenarios, the timestamp information is commonly accessible, including calendar timestamps (e.g., second, minute, hour, week, month, and year) and special timestamps (e.g., holidays and events). These timestamps are quite informative in real applications but hardly leveraged in vanilla Transformers. To mitigate the issue, Informer [Zhou *et al.*, 2021] proposed to encode timestamps as additional positional encoding by using learnable embedding layers. A similar timestamp encoding scheme was used in Autoformer [Wu *et al.*, 2021] and FEDformer [Zhou *et al.*, 2022].

4.2 Attention Module

Central to Transformer is the self-attention module. It can be viewed as a fully connected layer with weights that are dynamically generated based on the pairwise similarity of input patterns. As a result, it shares the same maximum path length as fully connected layers, but with a much less number of parameters, making it suitable for modeling long-term dependencies.

As we show in the previous section the self-attention module in the vanilla Transformer has a time and memory complexity of $\mathcal{O}(N^2)$ (N is the input time series length), which becomes the computational bottleneck when dealing with long sequences. Many efficient Transformers were proposed to reduce the quadratic complexity that can be classified into two main categories: (1) explicitly introducing a sparsity bias into the attention mechanism like LogTrans [Li *et al.*, 2019] and Pyraformer [Liu *et al.*, 2022a]; (2) exploring the low-rank property of the self-attention matrix to speed up the computation, e.g. Informer [Zhou *et al.*, 2021] and FEDformer [Zhou *et al.*, 2022]. Table 1 shows both the time and memory complexity of popular Transformers applied to time series modeling, and more details about these models will be discussed in Section 5.

Table 1: Complexity comparisons of popular time series Transformers with different attention modules.

Methods	Training		Testing
	Time	Memory	Steps
Transformer [Vaswani *et al.*, 2017]	$\mathcal{O}\left(N^2\right)$	$\mathcal{O}\left(N^2\right)$	N
LogTrans [Li *et al.*, 2019]	$\mathcal{O}(N \log N)$	$\mathcal{O}\left(N \log N\right)$	1
Informer [Zhou *et al.*, 2021]	$\mathcal{O}(N \log N)$	$\mathcal{O}(N \log N)$	1
Autoformer [Wu *et al.*, 2021]	$\mathcal{O}(N \log N)$	$\mathcal{O}(N \log N)$	1
Pyraformer [Liu *et al.*, 2022a]	$\mathcal{O}(N)$	$\mathcal{O}(N)$	1
Quatformer [Chen *et al.*, 2022]	$\mathcal{O}(2cN)$	$\mathcal{O}(2cN)$	1
FEDformer [Zhou *et al.*, 2022]	$\mathcal{O}(N)$	$\mathcal{O}(N)$	1
Crossformer [Zhang and Yan, 2023]	$\mathcal{O}(\frac{D}{L_{seg}^2}N^2)$	$\mathcal{O}(N)$	1

4.3 Architecture-based Attention Innovation

To accommodate individual modules in Transformers for modeling time series, a number of works [Zhou *et al.*, 2021; Liu *et al.*, 2022a] seek to renovate Transformers on the architecture level. Recent works introduce hierarchical architecture into Transformer to take into account the multi-resolution aspect of time series. Informer [Zhou *et al.*, 2021] inserts max-pooling layers with stride 2 between attention blocks, which down-sample series into its half slice. Pyraformer [Liu *et al.*, 2022a] designs a C-ary tree-based attention mechanism, in which nodes at the finest scale correspond to the original time series, while nodes in the coarser scales represent series at lower resolutions. Pyraformer developed both intra-scale and inter-scale attentions in order to better capture temporal dependencies across different resolutions. Besides the ability to integrate information at different multi-resolutions, a hierarchical architecture also enjoys the benefits of efficient computation, particularly for long-time series.

5 Applications of Time Series Transformers

In this section, we review the applications of Transformer to important time series tasks, including forecasting, anomaly detection, and classification.

5.1 Transformers in Forecasting

Here we examine three common types of forecasting tasks here, i.e. time series forecasting, spatial-temporal forecasting, and event forecasting.

Time Series Forecasting

A lot of work has been done to design new Transformer variants for time series forecasting tasks in the latest years. Module-level and architecture-level variants are two large categories and the former consists of the majority of the up-to-date works.

Module-level variants In the module-level variants for time series forecasting, their main architectures are similar to the vanilla Transformer with minor changes. Researchers introduce various time series inductive biases to design new modules. The following summarized work consists of three different types: designing new attention modules, exploring the innovative way to normalize time series data, and utilizing the bias for token inputs, as shown in Figure 2.

The first type of variant for module-level Transformers is to design new attention modules, which is the category with the largest proportion. Here we first describe six typical works: LogTrans [Li *et al.*, 2019], Informer [Zhou *et al.*, 2021],

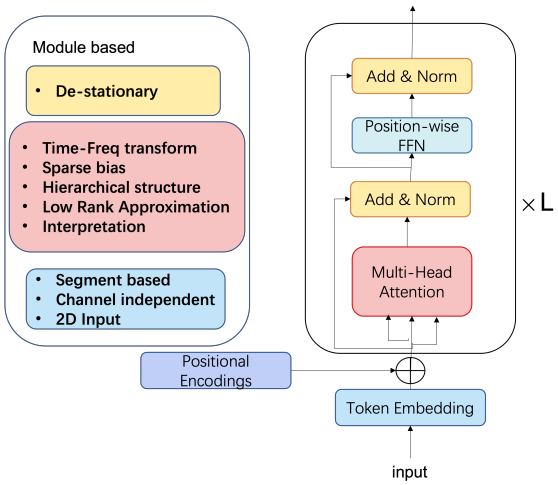

Figure 2: Categorization of module-level Transformer variants for time series forecasting.

AST [Wu *et al.*, 2020a], Pyraformer [Liu *et al.*, 2022a], Quatformer [Chen *et al.*, 2022], and FEDformer [Zhou *et al.*, 2022], all of which exploit sparsity inductive bias or low-rank approximation to remove noise and achieve a low-order calculation complexity. LogTrans [Li *et al.*, 2019] proposes convolutional self-attention by employing causal convolutions to generate queries and keys in the self-attention layer. It introduces sparse bias, a Logsparse mask, in self-attention model that reduces computational complexity from $\mathcal{O}(N^2)$ to $\mathcal{O}(N \log N)$. Instead of using explicit sparse bias, Informer [Zhou *et al.*, 2021] selects dominant queries based on queries and key similarities, thus achieving similar improvements as LogTrans in computational complexity. It also designs a generative style decoder to produce long-term forecasting directly and thus avoids accumulative error in using one forward-step prediction for long-term forecasting. AST [Wu *et al.*, 2020a] uses a generative adversarial encoder-decoder framework to train a sparse Transformer model for time series forecasting. It shows that adversarial training can improve time series forecasting by directly shaping the output distribution of the network to avoid error accumulation through one-step ahead inference. Pyraformer [Liu *et al.*, 2022a] designs a hierarchical pyramidal attention module with a binary tree following the path, to capture temporal dependencies of different ranges with linear time and memory complexity. FEDformer [Zhou *et al.*, 2022] applies attention operation in the frequency domain with Fourier transform and wavelet transform. It achieves a linear complexity by randomly selecting a fixed-size subset of frequency. Note that due to the success of Autoformer and FEDformer, it has attracted more attention in the community to explore self-attention mechanisms in the frequency domain for time series modeling. Quatformer [Chen *et al.*, 2022] proposes learning-to-rotate attention (LRA) based on quaternions that introduce learnable period and phase information to depict intricate periodical patterns. Moreover, it decouples LRA using a global memory to achieve linear complexity.

The following three works focus on building an explicit interpretation ability of models, which follows the trend of Explainable Artificial Intelligence (XAI). TFT [Lim *et al.*, 2021] designs a multi-horizon forecasting model with static covariate encoders, gating feature selection, and temporal self-attention decoder. It encodes and selects useful information from various covariates to perform forecasting. It also preserves interpretability by incorporating global, temporal dependency, and events. ProTran [Tang and Matteson, 2021] and SSDNet [Lin *et al.*, 2021] combine Transformer with state space models to provide probabilistic forecasts. ProTran designs a generative modeling and inference procedure based on variational inference. SSDNet first uses Transformer to learn the temporal pattern and estimate the parameters of SSM, and then applies SSM to perform the seasonal-trend decomposition and maintain the interpretable ability.

The second type of variant for module-level Transformers is the way to normalize time series data. To the best of our knowledge, Non-stationary Transformer [Liu *et al.*, 2022b] is the only work that mainly focuses on modifying the normalization mechanism as shown in Figure 2. It explores the over-stationarization problem in time series forecasting tasks with a relatively simple plugin series stationary and De-stationary module to modify and boost the performance of various attention blocks.

The third type of variant for module-level Transformer is utilizing the bias for token input. Autoformer [Wu *et al.*, 2021] adopts a segmentation-based representation mechanism. It devises a simple seasonal-trend decomposition architecture with an auto-correlation mechanism working as an attention module. The auto-correlation block measures the time-delay similarity between inputs signal and aggregates the top-k similar sub-series to produce the output with reduced complexity. PatchTST [Nie *et al.*, 2023] utilizes channel-independent where each channel contains a single univariate time series that shares the same embedding within all the series, and subseries-level patch design which segmentation of time series into subseries-level patches that are served as input tokens to Transformer. Such ViT [Dosovitskiy *et al.*, 2021] alike design improves its numerical performance in long-time time-series forecasting tasks a lot. Crossformer [Zhang and Yan, 2023] proposes a Transformer-based model utilizing cross-dimension dependency for multivariate time series forecasting. The input is embedded into a 2D vector array through the novel dimension-segment-wise embedding to preserve time and dimension information. Then, a two-stage attention layer is used to efficiently capture the cross-time and cross-dimension dependency.

Architecture-level variants Some works start to design a new transformer architecture beyond the scope of the vanilla transformer. Triformer [Cirstea *et al.*, 2022] design a triangular,variable-specific patch attention. It has a triangular tree-type structure as the later input size shrinks exponentially and a set of variable-specific parameters making a multi-layer Triformer maintain a lightweight and linear complexity. Scaleformer [Shabani *et al.*, 2023] proposes a multi-scale framework that can be applied to the baseline transformer-based time series forecasting models (FEDformer[Zhou *et al.*, 2022], Autoformer[Wu *et al.*, 2021], etc.). It can improve the baseline model's performance by iteratively refining the forecasted time series at multiple scales with shared weights.

Remarks Note that DLinear [Zeng *et al.*, 2023] questions the necessity of using Transformers for long-term time series forecasting, and shows that a simpler MLP-based model can achieve better results compared to some Transformer baselines through empirical studies. However, we notice that a recent Transformer model PatchTST [Nie *et al.*, 2023] achieves a better numerical result compared to DLinear for long-term time series forecasting. Moreover, there is a thorough theoretical study [Yun *et al.*, 2020] showing that the Transformer models are universal approximators of sequence-to-sequence functions. It is a overclaim to question the potential of any type of method for time series forecasting based solely on experimental results from some variant instantiations of such method, especially for Transformer models which already demonstrate the performances in most machine learning-based tasks. Therefore, we conclude that summarizing the recent Transformer-based models for time series forecasting is necessary and would benefit the whole community.

Spatio-Temporal Forecasting

In spatio-temporal forecasting, both temporal and spatio-temporal dependencies are taken into account in time series Transformers for accurate forecasting.

Traffic Transformer [Cai *et al.*, 2020] designs an encoder-decoder structure using a self-attention module to capture temporal-temporal dependencies and a graph neural network module to capture spatial dependencies. Spatial-temporal Transformer [Xu *et al.*, 2020] for traffic flow forecasting takes a step further. Besides introducing a temporal Transformer block to capture temporal dependencies, it also designs a spatial Transformer block, together with a graph convolution network, to better capture spatial-spatial dependencies. Spatio-temporal graph Transformer [Yu *et al.*, 2020] designs an attention-based graph convolution mechanism that is able to learn a complicated temporal-spatial attention pattern to improve pedestrian trajectory prediction. Earthformer [Gao *et al.*, 2022] proposes a cuboid attention for efficient space-time modeling, which decomposes the data into cuboids and applies cuboid-level self-attention in parallel. It shows that Earthformer achieves superior performance in weather and climate forecasting. Recently, AirFormer [Liang *et al.*, 2023] devises a dartboard spatial self-attention module and a causal temporal self-attention module to efficiently capture spatial correlations and temporal dependencies, respectively. Furthermore, it enhances Transformers with latent variables to capture data uncertainty and improve air quality forecasting.

Event Forecasting

Event sequence data with irregular and asynchronous timestamps are naturally observed in many real-life applications, which is in contrast to regular time series data with equal sampling intervals. Event forecasting or prediction aims to predict the times and marks of future events given the history of past events, and it is often modeled by temporal point processes (TPP) [Yan *et al.*, 2019; Shchur *et al.*, 2021].

Recently, several neural TPP models incorporate Transformers in order to improve the performance of event prediction. Self-attentive Hawkes process (SAHP) [Zhang *et al.*, 2020] and Transformer Hawkes process (THP) [Zuo *et al.*, 2020] adopt Transformer encoder architecture to summarize the influence of historical events and compute the intensity function for event prediction. They modify the positional encoding by translating time intervals into sinusoidal functions such that the intervals between events can be utilized. Later, a more flexible named attentive neural datalog through time (A-NDTT) [Mei *et al.*, 2022] is proposed to extend SAHP/THP schemes by embedding all possible events and times with attention as well. Experiments show that it can better capture sophisticated event dependencies than existing methods.

5.2 Transformers in Anomaly Detection

Transformer based architecture also benefits the time series anomaly detection task with the ability to model temporal dependency, which brings high detection quality [Xu *et al.*, 2022]. Besides, in multiple studies, including TranAD [Tuli *et al.*, 2022], MT-RVAE [Wang *et al.*, 2022], and TransAnomaly [Zhang *et al.*, 2021], researchers proposed to combine Transformer with neural generative models, such as VAEs [Kingma and Welling, 2014] and GANs [Goodfellow *et al.*, 2014], for better performance in anomaly detection. We will elaborate on these models in the following part.

TranAD [Tuli *et al.*, 2022] proposes an adversarial training procedure to amplify reconstruction errors as a simple Transformer-based network tends to miss small deviation of anomaly. GAN style adversarial training procedure is designed by two Transformer encoders and two Transformer decoders to gain stability. Ablation study shows that, if Transformer-based encoder-decoder is replaced, F1 score drops nearly 11%, indicating the effect of Transformer architecture on time series anomaly detection.

MT-RVAE [Wang *et al.*, 2022] and TransAnomaly [Zhang *et al.*, 2021] combine VAE with Transformer, but they share different purposes. TransAnomaly combines VAE with Transformer to allow more parallelization and reduce training cost by nearly 80%. In MT-RVAE, a multiscale Transformer is designed to extract and integrate time-series information at different scales. It overcomes the shortcomings of traditional Transformers where only local information is extracted for sequential analysis.

GTA [Chen *et al.*, 2021c] combines Transformer with graph-based learning architecture for multivariate time series anomaly detection. Note that, MT-RVAE is also for multivariate time series but with few dimensions or insufficient close relationships among sequences where the graph neural network model does not work well. To deal with such challenge, MT-RVAE modifies the positional encoding module and introduces feature-learning module. Instead, GTA contains a graph convolution structure to model the influence propagation process. Similar to MT-RVAE, GTA also considers "global" information, yet by replacing vanilla multi-head attention with a multi-branch attention mechanism, that is, a combination of global-learned attention, vanilla multi-head attention, and neighborhood convolution.

AnomalyTrans [Xu *et al.*, 2022] combines Transformer and Gaussian prior-Association to make anomalies more distinguishable. Sharing similar motivation as TranAD, AnomalyTrans achieves the goal in a different way. The insight is that it is harder for anomalies to build strong associations with the whole series while easier with adjacent time points com-

pared with normality. In AnomalyTrans, prior-association and series-association are modeled simultaneously. Besides reconstruction loss, the anomaly model is optimized by the minimax strategy to constrain the prior- and series- associations for more distinguishable association discrepancy.

5.3 Transformers in Classification

Transformer is proved to be effective in various time series classification tasks due to its prominent capability in capturing long-term dependency. GTN [Liu *et al.*, 2021] uses a two-tower Transformer with each tower respectively working on time-step-wise attention and channel-wise attention. To merge the feature of the two towers, a learnable weighted concatenation (also known as 'gating') is used. The proposed extension of Transformer achieves state-of-the-art results on 13 multivariate time series classifications. [Rußwurm and Körner, 2020] studied the self-attention based Transformer for raw optical satellite time series classification and obtained the best results compared with recurrent and convolutional neural networks. Recently, TARNet [Chowdhury *et al.*, 2022] designs Transformers to learn task-aware data reconstruction that augments classification performance, which utilizes attention score for important timestamps masking and reconstruction and brings superior performance.

Pre-trained Transformers are also investigated in classification tasks. [Yuan and Lin, 2020] studies the Transformer for raw optical satellite image time series classification. The authors use self-supervised pre-trained schema because of limited labeled data. [Zerveas *et al.*, 2021] introduced an unsupervised pre-trained framework and the model is pre-trained with proportionally masked data. The pre-trained models are then fine-tuned in downstream tasks such as classification. [Yang *et al.*, 2021] proposes to use large-scale pre-trained speech processing model for downstream time series classification problems and generates 19 competitive results on 30 popular time series classification datasets.

6 Experimental Evaluation and Discussion

We conduct empirical studies on a typical challenging benchmark dataset ETTm2 [Zhou *et al.*, 2021] to analyze how Transformers work on time series data. Since classic statistical ARIMA/ETS [Hyndman and Khandakar, 2008] models and basic RNN/CNN models perform inferior to Transformers in this dataset as shown in [Zhou *et al.*, 2021; Wu *et al.*, 2021], we focus on popular time series Transformers with different configurations in the experiments.

Robustness Analysis

A lot of works we describe above carefully design attention modules to lower the quadratic calculation and memory complexity, though they practically use a short fixed-size input to achieve the best result in their reported experiments. It makes us question the actual usage of such an efficient design. We perform a robust experiment with prolonging input sequence length to verify their prediction power and robustness when dealing with long-term input sequences in Table 2.

As in Table 2, when we compare the prediction results with prolonging input length, various Transformer-based model deteriorates quickly. This phenomenon makes a lot of carefully designed Transformers impractical in long-term forecasting tasks since they cannot effectively utilize long input information. More works and designs need to be investigated to fully utilize long sequence input for better performance.

Table 2: The MSE comparisons in robustness experiment of forecasting 96 steps for ETTm2 dataset with prolonging input length.

Model		Transformer	Autoformer	Informer	Reformer	LogFormer
Input Len	96	**0.557**	**0.239**	0.428	**0.615**	**0.667**
	192	0.710	0.265	**0.385**	0.686	0.697
	336	1.078	0.375	1.078	1.359	0.937
	720	1.691	0.315	1.057	1.443	2.153
	1440	0.936	0.552	1.898	0.815	0.867

Table 3: The MSE comparisons in model size experiment of forecasting 96 steps for ETTm2 dataset with different number of layers.

Model		Transformer	Autoformer	Informer	Reformer	LogFormer
Layer Num	3	0.557	**0.234**	0.428	0.597	0.667
	6	**0.439**	0.282	0.489	**0.353**	**0.387**
	12	0.556	0.238	0.779	0.481	0.562
	24	0.580	0.266	0.815	1.109	0.690
	48	0.461	NaN	1.623	OOM	2.992

Model Size Analysis

Before being introduced into the field of time series prediction, Transformer has shown dominant performance in NLP and CV communities [Vaswani *et al.*, 2017; Kenton and others, 2019; Han *et al.*, 2021; Han *et al.*, 2022]. One of the key advantages Transformer holds in these fields is being able to increase prediction power through increasing model size. Usually, the model capacity is controlled by Transformer's layer number, which is commonly set between 12 to 128. Yet as shown in the experiments of Table 3, when we compare the prediction result with different Transformer models with various numbers of layers, the Transformer with 3 to 6 layers often achieves better results. It raises a question about how to design a proper Transformer architecture with deeper layers to increase the model's capacity and achieve better forecasting performance.

Seasonal-Trend Decomposition Analysis

In recent studies, researchers [Wu *et al.*, 2021; Zhou *et al.*, 2022; Lin *et al.*, 2021; Liu *et al.*, 2022a] begin to realize that the seasonal-trend decomposition [Cleveland *et al.*, 1990; Wen *et al.*, 2020] is a crucial part of Transformer's performance in time series forecasting. As an experiment shown in Table 4, we adopt a simple moving average seasonal-trend decomposition architecture proposed in [Wu *et al.*, 2021] to test various attention modules. It can be seen that the simple seasonal-trend decomposition model can significantly boost model's performance by 50 % to 80%. It is a unique block and such performance boosting through decomposition seems a consistent phenomenon in time series forecasting for Transformer's application, which is worth further investigating for more advanced and carefully designed time series decomposition schemes.

7 Future Research Opportunities

Here we highlight a few directions that are potentially promising for future research of Transformers in time series.

Table 4: The MSE comparisons in ablation experiments of seasonal-trend decomposition analysis. 'Ori' means the original version without the decomposition. 'Decomp' means with decomposition. The experiment is performed on ETTm2 dataset with prolonging output length.

Model		FEDformer		Autoformer		Informer		LogTrans		Reformer		Transformer		Promotion
MSE		Ori	Decomp	Ori	Decomp	Ori	Decomp	Ori	Decomp	Ori	Decomp	Ori	Decomp	Relative
Out Len	96	0.457	0.203	0.581	0.255	0.365	0.354	0.768	0.231	0.658	0.218	0.604	0.204	53%
	192	0.841	0.269	1.403	0.281	0.533	0.432	0.989	0.378	1.078	0.336	1.060	0.266	62%
	336	1.451	0.325	2.632	0.339	1.363	0.481	1.334	0.362	1.549	0.366	1.413	0.375	75%
	720	3.282	0.421	3.058	0.422	3.379	0.822	3.048	0.539	2.631	0.502	2.672	0.537	82%

7.1 Inductive Biases for Time Series Transformers

Vanilla Transformer does not make any assumptions about data patterns and characteristics. Although it is a general and universal network for modeling long-range dependencies, it also comes with a price, i.e. lots of data are needed to train Transformer to improve the generalization and avoid data overfitting. One of the key features of time series data is its seasonal/periodic and trend patterns [Wen *et al.*, 2019; Cleveland *et al.*, 1990]. Some recent studies have shown that incorporating series periodicity [Wu *et al.*, 2021] or frequency processing [Zhou *et al.*, 2022] into time series Transformer can enhance performance significantly. Moreover, it is interesting that some studies adopt a seemly opposite inductive bias, but both achieve good numerical improvement: [Nie *et al.*, 2023] removes the cross-channel dependency by utilizing a channel-independent attention module, while an interesting work [Zhang and Yan, 2023] improves its experimental performance by utilizing cross-dimension dependency with a two-stage attention mechanism. Clearly, we have noise and signals in such a cross-channel learning paradigm, but a clever way to utilize such inductive bias to suppress the noise and extract the signal is still desired. Thus, one future direction is to consider more effective ways to induce inductive biases into Transformers based on the understanding of time series data and characteristics of specific tasks.

7.2 Transformers and GNN for Time Series

Multivariate and spatio-temporal time series are becoming more and more common in applications, calling for additional techniques to handle high dimensionality, especially the ability to capture the underlying relationships among dimensions. Introducing graph neural networks (GNNs) is a natural way to model spatial dependency or relationships among dimensions. Recently, several studies have demonstrated that the combination of GNN and Transformers/attentions could bring not only significant performance improvement like in traffic forecasting [Cai *et al.*, 2020; Xu *et al.*, 2020] and multi-modal forecasting [Li *et al.*, 2021], but also better understanding of the spatio-temporal dynamics and latent casuality. It is an important future direction to combine Transformers and GNNs for effectively spatial-temporal modeling in time series.

7.3 Pre-trained Transformers for Time Series

Large-scale pre-trained Transformer models have significantly boosted the performance for various tasks in NLP [Kenton and others, 2019; Brown *et al.*, 2020] and CV [Chen *et al.*, 2021a]. However, there are limited works on pre-trained Transformers for time series, and existing studies mainly focus on time series classification [Zerveas *et al.*, 2021; Yang *et al.*, 2021]. Therefore, how to develop appropriate pre-trained Transformer models for different tasks in time series remains to be examined in the future.

7.4 Transformers with Architecture Level Variants

Most developed Transformer models for time series maintain the vanilla Transformer's architecture with modifications mainly in the attention module. We might borrow the idea from Transformer variants in NLP and CV which also have architecture-level model designs to fit different purposes, such as lightweight [Wu *et al.*, 2020b; Mehta *et al.*, 2021], cross-block connectivity [Bapna *et al.*, 2018], adaptive computation time [Dehghani *et al.*, 2019; Xin *et al.*, 2020], and recurrence [Dai *et al.*, 2019]. Therefore, one future direction is to consider more architecture-level designs for Transformers specifically optimized for time series data and tasks.

7.5 Transformers with NAS for Time Series

Hyper-parameters, such as embedding dimension, number of heads, and number of layers, can largely affect the performance of Transformers. Manual configuring these hyper-parameters is time-consuming and often results in suboptimal performance. Neural architecture search (NAS) [Elsken *et al.*, 2019; Wang *et al.*, 2020] has been a popular technique for discovering effective deep neural architectures, and automating Transformer design using NAS in NLP and CV can be found in recent studies [So *et al.*, 2019; Chen *et al.*, 2021b]. For industry-scale time series data which can be of both high dimension and long length, automatically discovering both memory- and computational-efficient Transformer architectures is of practical importance, making it an important future direction for time series Transformers.

8 Conclusion

We have provided a survey on time series Transformers. We organize the reviewed methods in a new taxonomy consisting of network design and application. We summarize representative methods in each category, discuss their strengths and limitations by experimental evaluation, and highlight future research directions.

References

[Bapna *et al.*, 2018] Ankur Bapna, Mia Xu Chen, Orhan Firat, Yuan Cao, and Yonghui Wu. Training deeper neural machine translation models with transparent attention. In *EMNLP*, 2018.

[Benidis *et al.*, 2022] Konstantinos Benidis, Syama Sundar Rangapuram, Valentin Flunkert, Yuyang Wang, Danielle Maddix, , et al. Deep learning for time series forecasting: Tutorial and literature survey. *ACM Computing Surveys*, 55(6):1–36, 2022.

[Blázquez-García *et al.*, 2021] Ane Blázquez-García, Angel Conde, Usue Mori, et al. A review on outlier/anomaly detection in time series data. *ACM Computing Surveys*, 54(3):1–33, 2021.

[Brown *et al.*, 2020] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, et al. Language models are few-shot learners. *NeurIPS*, 2020.

[Cai *et al.*, 2020] Ling Cai, Krzysztof Janowicz, Gengchen Mai, Bo Yan, and Rui Zhu. Traffic transformer: Capturing the continuity and periodicity of time series for traffic forecasting. *Transactions in GIS*, 24(3):736–755, 2020.

[Chen *et al.*, 2021a] Hanting Chen, Yunhe Wang, Tianyu Guo, Chang Xu, Yiping Deng, Zhenhua Liu, Siwei Ma, Chunjing Xu, et al. Pre-trained image processing transformer. In *CVPR*, 2021.

[Chen *et al.*, 2021b] Minghao Chen, Houwen Peng, Jianlong Fu, and Haibin Ling. AutoFormer: Searching transformers for visual recognition. In *CVPR*, 2021.

[Chen *et al.*, 2021c] Zekai Chen, Dingshuo Chen, Xiao Zhang, Zixuan Yuan, and Xiuzhen Cheng. Learning graph structures with transformer for multivariate time series anomaly detection in IoT. *IEEE Internet of Things Journal*, 2021.

[Chen *et al.*, 2022] Weiqi Chen, Wenwei Wang, Bingqing Peng, Qingsong Wen, Tian Zhou, and Liang Sun. Learning to rotate: Quaternion transformer for complicated periodical time series forecasting. In *KDD*, 2022.

[Choi *et al.*, 2021] Kukjin Choi, Jihun Yi, Changhwa Park, and Sungroh Yoon. Deep learning for anomaly detection in time-series data: Review, analysis, and guidelines. *IEEE Access*, 2021.

[Chowdhury *et al.*, 2022] Ranak Roy Chowdhury, Xiyuan Zhang, Jingbo Shang, Rajesh K Gupta, and Dezhi Hong. TARNet: Task-aware reconstruction for time-series transformer. In *KDD*, 2022.

[Cirstea *et al.*, 2022] Razvan-Gabriel Cirstea, Chenjuan Guo, Bin Yang, Tung Kieu, Xuanyi Dong, and Shirui Pan. Triformer: Triangular, variable-specific attentions for long sequence multivariate time series forecasting. In *IJCAI*, 2022.

[Cleveland *et al.*, 1990] Robert Cleveland, William Cleveland, Jean McRae, et al. STL: A seasonal-trend decomposition procedure based on loess. *Journal of Official Statistics*, 6(1):3–73, 1990.

[Dai *et al.*, 2019] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G. Carbonell, Quoc V. Le, et al. Transformer-XL: Attentive language models beyond a fixed-length context. In *ACL*, 2019.

[Dehghani *et al.*, 2019] Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. Universal transformers. In *ICLR*, 2019.

[Dong *et al.*, 2018] Linhao Dong, Shuang Xu, and Bo Xu. Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition. In *ICASSP*, 2018.

[Dosovitskiy *et al.*, 2021] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

[Elsken *et al.*, 2019] Elsken, Thomas, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: A survey. *Journal of Machine Learning Research*, 2019.

[Gao *et al.*, 2022] Zhihan Gao, Xingjian Shi, Hao Wang, Yi Zhu, Bernie Wang, Mu Li, et al. Earthformer: Exploring space-time transformers for earth system forecasting. In *NeurIPS*, 2022.

[Gehring *et al.*, 2017] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. Convolutional sequence to sequence learning. In *ICML*, 2017.

[Goodfellow *et al.*, 2014] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, et al. Generative adversarial nets. *NeurIPS*, 2014.

[Han *et al.*, 2021] Xu Han, Zhengyan Zhang, Ning Ding, Yuxian Gu, Xiao Liu, Yuqi Huo, Jiezhong Qiu, Liang Zhang, et al. Pre-trained models: Past, present and future. *AI Open*, 2021.

[Han *et al.*, 2022] Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, et al. A survey on vision transformer. *IEEE TPAMI*, 45(1):87–110, 2022.

[Hyndman and Khandakar, 2008] Rob J Hyndman and Yeasmin Khandakar. Automatic time series forecasting: the forecast package for r. *Journal of statistical software*, 27:1–22, 2008.

[Ismail Fawaz *et al.*, 2019] 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*, 2019.

[Ke *et al.*, 2021] Guolin Ke, Di He, and Tie-Yan Liu. Rethinking positional encoding in language pre-training. In *ICLR*, 2021.

[Kenton and others, 2019] Jacob Devlin Ming-Wei Chang Kenton et al. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019.

[Kingma and Welling, 2014] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014.

[Li *et al.*, 2019] 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. In *NeurIPS*, 2019.

[Li *et al.*, 2021] Longyuan Li, Jian Yao, Li Wenliang, Tong He, Tianjun Xiao, Junchi Yan, David Wipf, and Zheng Zhang. Grin: Generative relation and intention network for multi-agent trajectory prediction. In *NeurIPS*, 2021.

[Liang *et al.*, 2023] Yuxuan Liang, Yutong Xia, Songyu Ke, Yiwei Wang, Qingsong Wen, Junbo Zhang, Yu Zheng, and Roger Zimmermann. AirFormer: Predicting nationwide air quality in china with transformers. In *AAAI*, 2023.

[Lim and Zohren, 2021] Bryan Lim and Stefan Zohren. Time-series forecasting with deep learning: a survey. *Philosophical Transactions of the Royal Society*, 2021.

[Lim *et al.*, 2021] Bryan Lim, Sercan Ö Arık, Nicolas Loeff, and Tomas Pfister. Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, 37(4):1748–1764, 2021.

[Lin *et al.*, 2021] Yang Lin, Irena Koprinska, and Mashud Rana. SSDNet: State space decomposition neural network for time series forecasting. In *ICDM*, 2021.

[Liu *et al.*, 2021] Minghao Liu, Shengqi Ren, Siyuan Ma, Jiahui Jiao, Yizhou Chen, Zhiguang Wang, and Wei Song. Gated transformer networks for multivariate time series classification. *arXiv preprint arXiv:2103.14438*, 2021.

[Liu *et al.*, 2022a] 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 *ICLR*, 2022.

[Liu *et al.*, 2022b] Yong Liu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Non-stationary transformers: Exploring the stationarity in time series forecasting. In *NeurIPS*, 2022.

[Mehta *et al.*, 2021] Sachin Mehta, Marjan Ghazvininejad, Srini Iyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. Delight: Deep and light-weight transformer. In *ICLR*, 2021.

[Mei *et al.*, 2022] Hongyuan Mei, Chenghao Yang, and Jason Eisner. Transformer embeddings of irregularly spaced events and their participants. In *ICLR*, 2022.

[Nie *et al.*, 2023] Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *ICLR*, 2023.

[Rußwurm and Körner, 2020] Marc Rußwurm and Marco Körner. Self-attention for raw optical satellite time series classification. *ISPRS J. Photogramm. Remote Sens.*, 169:421–435, 11 2020.

[Shabani *et al.*, 2023] Amin Shabani, Amir Abdi, Lili Meng, and Tristan Sylvain. Scaleformer: iterative multi-scale refining transformers for time series forecasting. In *ICLR*, 2023.

[Shaw *et al.*, 2018] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. In *NAACL*, 2018.

[Shchur *et al.*, 2021] Oleksandr Shchur, Ali Caner Türkmen, Tim Januschowski, and Stephan Günnemann. Neural temporal point processes: A review. In *IJCAI*, 2021.

[So *et al.*, 2019] David So, Quoc Le, and Chen Liang. The evolved transformer. In *ICML*, 2019.

[Tang and Matteson, 2021] Binh Tang and David Matteson. Probabilistic transformer for time series analysis. In *NeurIPS*, 2021.

[Tay *et al.*, 2022] Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. Efficient transformers: A survey. *ACM Computing Surveys*, 55(6):1–28, 2022.

[Torres *et al.*, 2021] José F. Torres, Dalil Hadjout, Abderrazak Sebaa, Francisco Martínez-Álvarez, and Alicia Troncoso. Deep learning for time series forecasting: a survey. *Big Data*, 2021.

[Tuli *et al.*, 2022] Shreshth Tuli, Giuliano Casale, and Nicholas R Jennings. TranAD: Deep transformer networks for anomaly detection in multivariate time series data. In *VLDB*, 2022.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, et al. Attention is all you need. In *NeurIPS*, 2017.

[Wang *et al.*, 2020] Xiaoxing Wang, Chao Xue, Junchi Yan, Xiaokang Yang, Yonggang Hu, et al. MergeNAS: Merge operations into one for differentiable architecture search. In *IJCAI*, 2020.

[Wang *et al.*, 2022] Xixuan Wang, Dechang Pi, Xiangyan Zhang, et al. Variational transformer-based anomaly detection approach for multivariate time series. *Measurement*, page 110791, 2022.

[Wen *et al.*, 2019] Qingsong Wen, Jingkun Gao, Xiaomin Song, Liang Sun, Huan Xu, et al. RobustSTL: A robust seasonal-trend decomposition algorithm for long time series. In *AAAI*, 2019.

[Wen *et al.*, 2020] Qingsong Wen, Zhe Zhang, Yan Li, and Liang Sun. Fast RobustSTL: Efficient and robust seasonal-trend decomposition for time series with complex patterns. In *KDD*, 2020.

[Wen *et al.*, 2021a] Qingsong Wen, Kai He, Liang Sun, Yingying Zhang, Min Ke, et al. RobustPeriod: Time-frequency mining for robust multiple periodicities detection. In *SIGMOD*, 2021.

[Wen *et al.*, 2021b] Qingsong Wen, Liang Sun, Fan Yang, Xiaomin Song, Jingkun Gao, Xue Wang, and Huan Xu. Time series data augmentation for deep learning: A survey. In *IJCAI*, 2021.

[Wen *et al.*, 2022] Qingsong Wen, Linxiao Yang, Tian Zhou, and Liang Sun. Robust time series analysis and applications: An industrial perspective. In *KDD*, 2022.

[Wu *et al.*, 2020a] Sifan Wu, Xi Xiao, Qianggang Ding, Peilin Zhao, Ying Wei, and Junzhou Huang. Adversarial sparse transformer for time series forecasting. In *NeurIPS*, 2020.

[Wu *et al.*, 2020b] Zhanghao Wu, Zhijian Liu, Ji Lin, Yujun Lin, and Song Han. Lite transformer with long-short range attention. In *ICLR*, 2020.

[Wu *et al.*, 2021] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In *NeurIPS*, 2021.

[Xin *et al.*, 2020] Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy J. Lin. DeeBERT: Dynamic early exiting for accelerating bert inference. In *ACL*, 2020.

[Xu *et al.*, 2020] Mingxing Xu, Wenrui Dai, Chunmiao Liu, Xing Gao, Weiyao Lin, Guo-Jun Qi, and Hongkai Xiong. Spatial-temporal transformer networks for traffic flow forecasting. *arXiv preprint arXiv:2001.02908*, 2020.

[Xu *et al.*, 2022] Jiehui Xu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Anomaly Transformer: Time series anomaly detection with association discrepancy. In *ICLR*, 2022.

[Yan *et al.*, 2019] Junchi Yan, Hongteng Xu, and Liangda Li. Modeling and applications for temporal point processes. In *KDD*, 2019.

[Yang *et al.*, 2021] Chao-Han Huck Yang, Yun-Yun Tsai, and Pin-Yu Chen. Voice2series: Reprogramming acoustic models for time series classification. In *ICML*, 2021.

[Yu *et al.*, 2020] Cunjun Yu, Xiao Ma, Jiawei Ren, Haiyu Zhao, and Shuai Yi. Spatio-temporal graph transformer networks for pedestrian trajectory prediction. In *ECCV*, 2020.

[Yuan and Lin, 2020] Yuan Yuan and Lei Lin. Self-supervised pretraining of transformers for satellite image time series classification. *IEEE J-STARS*, 14:474–487, 2020.

[Yun *et al.*, 2020] Chulhee Yun, Srinadh Bhojanapalli, Ankit Singh Rawat, Sashank J. Reddi, et al. Are transformers universal approximators of sequence-to-sequence functions? In *ICLR*, 2020.

[Zeng *et al.*, 2023] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *AAAI*, 2023.

[Zerveas *et al.*, 2021] George Zerveas, Srideepika Jayaraman, Dhaval Patel, Anuradha Bhamidipaty, and Carsten Eickhoff. A transformer-based framework for multivariate time series representation learning. In *KDD*, 2021.

[Zhang and Yan, 2023] Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *ICLR*, 2023.

[Zhang *et al.*, 2020] Qiang Zhang, Aldo Lipani, Omer Kirnap, and Emine Yilmaz. Self-attentive Hawkes process. In *ICML*, 2020.

[Zhang *et al.*, 2021] Hongwei Zhang, Yuanqing Xia, et al. Unsupervised anomaly detection in multivariate time series through transformer-based variational autoencoder. In *CCDC*, 2021.

[Zhou *et al.*, 2021] 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 *AAAI*, 2021.

[Zhou *et al.*, 2022] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *ICML*, 2022.

[Zuo *et al.*, 2020] Simiao Zuo, Haoming Jiang, Zichong Li, Tuo Zhao, and Hongyuan Zha. Transformer Hawkes process. In *ICML*, 2020.