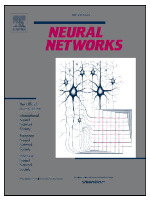
Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet



Full Length Article

Temporal structure-preserving transformer for industrial load forecasting

Senzhen Wu[a], Zhijin Wang[a,*], Xiufeng Liu[b,*], Yuan Zhao[c], Yue Hu[d], Yaohui Huang[e]

[a] College of Computer Engineering, Jimei University, Yinjiang Road 185, Xiamen, 361021, China
[b] Department of Technology, Management and Economics, Technical University of Denmark, 2800 Kgs., Lyngby, Denmark
[c] School of Economics and Management, Lanzhou University of Technology, Langongping Road 287, Lanzhou, 730050, China
[d] Chengyi College, Jimei University, Jimei Road 199, Xiamen, 361021, China
[e] School of Automation, Central South University, 410083 city, China

ARTICLE INFO

Keywords:
Industrial load forecasting
Transformer networks
Multiscale modeling
Temporal structure-preserving
Domain knowledge integration

ABSTRACT

Accurate power load forecasting in industrial parks is crucial for optimizing energy management and operational efficiency. Existing models struggle with industrial load series' complex, multi-target nature and the need to integrate diverse exogenous variables. This paper introduces the Temporal Structure-Preserving Transformer (TSPT), a novel architecture that addresses these challenges by decomposing multi-target series into univariate series, enabling parallel processing and integrating exogenous data. The TSPT model incorporates the Gated Feature Fusion (GFF), which learns to capture multiscale temporal patterns from each target sequence and exogenous factors by preserving the temporal structure of the series. This parallel processing and the structure-preserving transformations allow TSPT to effectively integrate domain-specific knowledge, such as weather, production, and efficiency data, enhancing its forecasting performance. Comprehensive experiments on a real-world industrial park dataset demonstrate TSPT's superiority over state-of-the-art methods in handling complex, multi-target forecasting tasks with integrated exogenous variables. The proposed approach offers a pathway for scalable and accurate load forecasting in industrial settings, improving energy management and operational decision-making.

1. Introduction

Accurate and reliable forecasting of electricity consumption within industrial parks is paramount for the optimal planning, operation, and control of energy systems, leading to significant cost savings and improved operational efficiency (Slowik & Urban, 2022; Zhang & Chiang, 2019). However, industrial load forecasting poses unique and complex challenges due to the multi-faceted nature of industrial load series and the inherent influence of diverse, often dynamic, exogenous factors (Lu et al., 2021). Industrial loads are characterized by intricate temporal dependencies that span across different time scales, non-trivial cross-series correlations among various building types. They are significantly impacted by external variables such as highly variable weather patterns, production schedules that can change quickly, and the diverse efficiency measures that vary throughout operations (Ge et al., 2020; Li et al., 2018; Som, 2023). These complexities necessitate forecasting methods that go beyond simple pattern recognition and can instead adapt to varying behaviors and changing operational conditions.

Traditional time series methods, such as ARIMA and exponential smoothing, which assume stationarity and linearity, struggle to capture these complex, non-linear dynamics and often fail to leverage the rich and heterogeneous information contained in exogenous variables, which are essential to more accurate forecasting (Ahmad et al., 2016; Porteiro et al., 2019). Similarly, while advanced deep learning techniques, such as recurrent neural networks (RNNs) and convolutional neural networks (CNNs), have shown promise in handling temporal data, they still exhibit limitations in effectively modeling long-range dependencies and integrating heterogeneous data sources from disparate sources in a unified way (Burg et al., 2021; Chen et al., 2018). Furthermore, these models often treat all data as homogeneous, while real-world industrial load data is far more complicated, and this lack of distinction can result in suboptimal performance. Although Ko et al. (2022) propose a deep concatenated residual network with bidirectional LSTM for wind power forecasting, highlighting the potential of hybrid architectures in capturing complex temporal patterns, their approach does not fully address the crucial multi-target nature of industrial load forecasting.

Recent advancements in transformer-based models, which use self-attention mechanisms, have demonstrated impressive capabilities in capturing long-range dependencies and have been successfully applied to various time series domains (Vaswani et al., 2017; Wang et al., 2023b;

* Corresponding authors.
E-mail addresses: szwbyte@gmail.com (S. Wu), zhijin@jmu.edu.cn (Z. Wang), xiuli@dtu.dk (X. Liu), zhaoyuan@lut.edu.cn (Y. Zhao), yuehu.xm@gmail.com (Y. Hu), yhhuang5212@gmail.com (Y. Huang).

https://doi.org/10.1016/j.neunet.2025.107887
Received 17 December 2024; Received in revised form 13 May 2025; Accepted 18 July 2025
Available online 23 July 2025

Wu et al., 2021a; Zhou et al., 2021). However, existing transformer models, such as Informer (Zhou et al., 2021) and Autoformer (Wu et al., 2021a), are primarily designed for univariate or homogeneous multivariate series. They typically lack mechanisms to effectively decompose multi-target series or efficiently integrate diverse exogenous variables, which are essential for industrial load forecasting. While some hybrid methods, such as Tatinati et al. (2022), combine decomposition with deep learning, they often do not fully exploit the parallel processing capabilities of modern hardware, nor do they preserve the temporal structure as it evolves across different scales.

Previous attempts to incorporate domain knowledge in industrial load forecasting, such as hybrid models combining Empirical Mode Decomposition (EMD) (Hossain et al., 2023) and LSTMs (Meng et al., 2022) or integrating features like production plans and energy efficiency indicators (Li et al., 2018), often lack the advanced multiscale temporal modeling capabilities offered by transformer architectures. The ELITE-Load model proposed in Zhang and Chiang (2019) addresses some of these challenges by combining optimal input-pruned neural networks with the TRUST-TECH methodology. However, it does not explicitly model multiscale temporal patterns, nor does it fully leverage the parallel processing capabilities of modern hardware for multi-target forecasting. Therefore, despite the progress, there is a pressing need for a model capable of addressing all these challenges simultaneously.

To bridge this critical gap, we introduce the Temporal Structure-Preserving Transformer (TSPT), a novel architecture specifically designed for the intricacies of industrial load forecasting. The key innovation of TSPT lies in decomposing longer time series into several lattice-structured time series that encapsulate local periodicities, enabling the parallel processing and effective integration of both long-term and short-term temporal features. This decomposition allows each local period to be modeled independently while incorporating relevant external data. Furthermore, TSPT utilizes a Gated Feature Fusion (GFF) mechanism, a novel component that learns to extract multiscale temporal patterns from each target sequence and exogenous factors by preserving the temporal structure of the series. Unlike methods that focus solely on individual target data, GFF enables the model to adapt to different patterns observed across various exogenous factors while capturing both short-term and long-term dependencies, thereby enhancing the model's capability to model the dynamic characteristics of industrial loads.

The main contributions of this work are as follows:

- We propose the Temporal Structure-Preserving Transformer (TSPT), a novel architecture that enables parallel processing and the integration of exogenous variables by decomposing multi-target load series into univariate series. This decomposition allows for more efficient and effective modeling of complex industrial load patterns.
- We introduce the Gated Feature Fusion (GFF), a novel component that learns to extract multiscale temporal patterns from each target sequence and exogenous factors by preserving the temporal series structure, enabling the model to adapt to different patterns observed across various exogenous factors. GFF sets itself apart from existing approaches by simultaneously modeling short-term and long-term dependencies, enhancing the extraction of rich temporal information.
- We demonstrate the superior forecasting performance of TSPT over state-of-the-art methods using a real-world industrial park dataset, showcasing its capability to handle complex, multi-target forecasting tasks with integrated exogenous variables, ultimately improving energy management, operational decision-making, and resource allocation.

The paper continues with the following structure: Section 2 provides a review of related work. Section 3 details the proposed TSPT architecture and components. Section 4 presents the experimental setup, and Section 5 presents the results and analyses. Section 6 discusses the findings, including limitations and implications. Section 7 concludes the paper by summarizing contributions and suggesting future directions.

2. Related work

2.1. Transformer models for time series forecasting

The Transformer architecture, introduced initially for natural language processing, has shown exceptional potential in modeling time series data due to its capability to capture long-range dependencies and complex temporal dynamics (Vaswani et al., 2017). Early adaptations of Transformers to time series tasks aimed to overcome limitations of traditional models like ARIMA and RNNs, particularly in handling long sequences and non-linear dependencies (Wen et al., 2023; Zhou et al., 2021).

Numerous variants of the Transformer have been proposed to address specific challenges in time series forecasting. Informer, for instance, introduces the ProbSparse self-attention mechanism, reducing the quadratic complexity of canonical attention to $\mathcal{O}(L \log L)$, and employs a generative decoder for efficient long-sequence forecasting (Zhou et al., 2021). Similarly, Autoformer integrates seasonal-trend decomposition directly into the attention mechanism, enhancing its interpretability and efficiency for multiscale patterns (Wu et al., 2021b). Preformer takes a novel approach with multiscale segment-wise correlations, providing segment-level attention to better model long-term dependencies (Du et al., 2023).

Despite these advancements, limitations persist. The permutation-invariant nature of attention mechanisms in Transformers can lead to temporal information loss, as highlighted by Zeng et al., who argue for models like LTSF-Linear that focus on preserving temporal ordering (Wen et al., 2023). Addressing non-stationarity has also been a significant focus. Non-stationary Transformers mitigate over-stationarization by reintroducing intrinsic non-stationary patterns into learned attention weights (Liu et al., 2022b).

Recently, models such as iTransformer and TimeXer have extended the application of Transformers by rethinking the embedding and attention mechanisms. iTransformer employs variate-level tokens to enhance the modeling of multivariate dependencies, particularly in scenarios with disparate scales and dynamics among variables (Liu et al., 2023b). TimeXer further leverages exogenous variables through a dual attention mechanism, reconciling temporal and cross-variate relationships for improved interpretability and accuracy in complex forecasting tasks (Wang et al., 2024).

Our proposed Temporal Structure-Preserving Transformer (TSPT) addresses these gaps by introducing the Gated Feature Fusion (GFF) mechanism. Unlike existing approaches, TSPT decomposes longer time series into several lattice-structured time series and leverages transformations that explicitly preserve the underlying temporal structure, effectively integrating domain-specific exogenous variables. By incorporating this structure-preserving approach with a multiscale representation and attention framework, TSPT can better capture both short- and long-term dependencies while maintaining scalability and interpretability for industrial load forecasting.

2.2. Industrial domain knowledge representations

Incorporating domain-specific knowledge into forecasting models is critical for improving predictive accuracy, particularly in complex industrial settings (Kerkkänen & Huiskonen, 2014; Seifert et al., 2015). Industrial systems often exhibit unique characteristics such as strong periodicity, non-linear relationships, and the influence of diverse exogenous variables, which data-driven models may struggle to capture without appropriate domain adaptations (Yanchenko et al., 2021). Effective integration of contextual information enhances interpretability, robustness, and utility of forecasting models in real-world applications (Kerkkänen & Huiskonen, 2014; Kreye et al., 2012).

Contextual information spans various forms, including product-specific data, manufacturing details, competitor activities, and macroeconomic forecasts (Seifert et al., 2015). Integrating these factors into forecasting models can assist decision-makers in making informed judgments by providing additional insights into the forecasting event (Alexander & Block, 2022). For instance, auxiliary contextual information, such as categorical data, can supply fine-grained details that improve the accuracy of coarse-grained event forecasts (Ma et al., 2023).

Several methodologies have been proposed to effectively incorporate domain knowledge into forecasting models. Bayesian forecasting approaches are widely used for integrating contextual constraints and decision-making processes within an organizational framework (Wang et al., 2023a). Similarly, the TEI@I methodology, combining qualitative and quantitative analyses, has effectively leveraged domain knowledge across diverse contexts (Huang et al., 2011).

Machine learning and graph-based modeling advances have further expanded the capacity to represent and utilize domain knowledge. Graph neural networks (GNNs) have been particularly effective in modeling spatial-temporal dependencies and non-Euclidean relationships, such as those found in energy grids, manufacturing systems, and transportation networks (Geng et al., 2019; Song et al., 2020; Zhang et al., 2020). By capturing pairwise correlations among regions and temporal structures, these approaches enhance forecasting accuracy and adaptability in industrial systems (Geng et al., 2019).

Transformer-based architectures have also shown promise in embedding domain-specific knowledge directly into forecasting models. For instance, integrating positional encodings and exogenous variable embeddings into attention mechanisms allows these models to leverage operational thresholds, energy constraints, and other domain-specific data effectively (Filipiak et al., 2023; Song et al., 2020).

Our proposed model, the Temporal Structure-Preserving Transformer (TSPT), builds upon these advances by introducing a multiscale framework that explicitly integrates domain knowledge through transformations that preserve the underlying temporal structure. By allowing the parallel decomposition of multi-target time series and integrating exogenous variables, TSPT captures intricate relationships unique to industrial systems. This approach enables the model to leverage domain knowledge without relying solely on manual feature engineering, ensuring scalability and adaptability across diverse industrial applications.

3. Proposed method

This section details the architecture and components of the proposed Temporal Structure-Preserving Transformer (TSPT) model, specifically designed for accurate multi-target industrial power load forecasting, incorporating diverse and dynamic exogenous variables. We first formulate the intricate forecasting problem inherent in industrial settings and then provide a comprehensive description of the TSPT model's components and their functionalities, emphasizing the theoretical underpinnings of our approach as a novel neural learning system.

3.1. Problem formulation

Accurate electricity consumption forecasting within industrial parks is critical for energy management and operational efficiency. However, industrial load forecasting presents unique challenges due to the multifaceted nature of industrial load series and the influence of diverse exogenous factors. We consider a discrete time index set $\mathcal{T} = \{t_1, t_2, \ldots, t_L\}$, where $L \in \mathbb{Z}^+$ represents the total number of observed time steps. For each time step, we observe power loads of $N_T \in \mathbb{Z}^+$ different building types. We utilize a historical observation window of length $T \in \mathbb{Z}^+$, meaning that at any reference time $t_i \in \mathcal{T}$, we consider the T most recent observations as input.

Definition 1 (Multi-Target Power Load Matrix)**.** For a batch of B instances, the multi-target power load observation matrix at time t is denoted as $\mathbf{X}_t \in \mathbb{R}^{B\times T\times N_T}$. Each element $x_{b,t_j,k}$ represents the observed power load of target series k (e.g., a specific building type) for instance b at past time step t_j, where $t - T + 1 \leq t_j \leq t$. Each column of $\mathbf{X}_t$ corresponds to a distinct building type, and each row represents a time step within the historical window.

Table 1
Symbols and their semantics.

Symbol	Semantic
T	Length of the historical observation window (look-back)
H	Forecast horizon length
N_T	Number of target time series (building types)
N_E	Number of exogenous factor time series
$\mathbf{X}_t$	Historical power load observations batch, size $B \times T \times N_T$
$\mathbf{E}_t$	Historical exogenous factor observations batch, size $B \times T \times N_E$
$\mathbf{Y}_t$	True future load values batch, size $B \times H \times N_T$
$\hat{\mathbf{Y}}_t$	Predicted future load values batch, size $B \times H \times N_T$
B	Batch size
D	Latent feature dimension
P_l	Patch length
P_s	Patch stride
P_n	Number of patches per instance
d_{model}	Transformer embedding dimension
h	Number of attention heads
d_k, d_v	Key and Value dimension per head ($d_k = d_v = d_{model}/h$)
d_{ffn}	Dimension of Feed-Forward Network hidden layer
N_{enc}	Number of encoder layers
$\boldsymbol{\theta}$	Fusion gating tensor
$\boldsymbol{\mu}$	Positional encoding tensor

Definition 2 (Exogenous Factor Matrix)**.** In addition to power load observations, we incorporate $N_E \in \mathbb{Z}^+$ exogenous factors, such as weather conditions or production schedules, that significantly influence industrial power consumption. The exogenous factor matrix is defined as $\mathbf{E}_t \in \mathbb{R}^{B\times T\times N_E}$, where $e_{b,t_j,l}$ represents the observed value of the lth exogenous factor for instance b at past time step t_j.

Problem Statement: Given the historical multi-target power load matrix $\mathbf{X}_t$ and the historical exogenous factor matrix $\mathbf{E}_t$, the objective is to accurately predict the future energy demand for all N_T building types over a forecast horizon $H \in \mathbb{Z}^+$. Formally, we aim to estimate the future load matrix:

$$\mathbf{Y}_t = [\mathbf{y}_{t+1}, \mathbf{y}_{t+2}, \ldots, \mathbf{y}_{t+H}]^\top \in \mathbb{R}^{B\times H\times N_T},$$

where $\mathbf{y}_{t+h} \in \mathbb{R}^{B\times N_T}$ is the vector of true future loads across all targets and instances at future time step $t + h$. The TSPT model is designed to learn a complex non-linear mapping function f_p:

$$f_p : \mathbb{R}^{B\times T\times N_T} \times \mathbb{R}^{B\times T\times N_E} \to \mathbb{R}^{B\times H\times N_T},$$

such that the predicted future load matrix is $\hat{\mathbf{Y}}_t = f_p(\mathbf{X}_t, \mathbf{E}_t)$. To quantitatively evaluate forecasting accuracy, we utilize and minimize the Mean Squared Error (MSE) loss function, defined as:

$$\min_{f_p} \mathcal{L} = \frac{1}{B \cdot H \cdot N_T} \sum_{b=1}^{B} \sum_{h=1}^{H} \sum_{k=1}^{N_T} (y_{b,t+h,k} - \hat{y}_{b,t+h,k})^2, \tag{1}$$

where $y_{b,t+h,k}$ and $\hat{y}_{b,t+h,k}$ are the true and predicted power loads for target k, instance b, and at forecast horizon step h, respectively. Table 1 summarizes the key mathematical symbols used throughout the method description.

3.2. Overview of the TSPT architecture

The TSPT, illustrated in Fig. 1(a), presents a novel architecture for accurate and robust multi-target industrial load forecasting with diverse exogenous variables. As a specifically designed neural learning system, TSPT is structured to learn complex temporal dependencies and effectively fuse multimodal data streams, all while preserving the inherent temporal structure of the input series. The processing workflow is initiated with **Instance Normalization**, which is crucial for stabilizing training and standardizing the scales of both power load and exogenous data, addressing potential non-stationarity. Subsequently, **Feature Extraction Perceptrons** (TSFP and EFP) are employed to

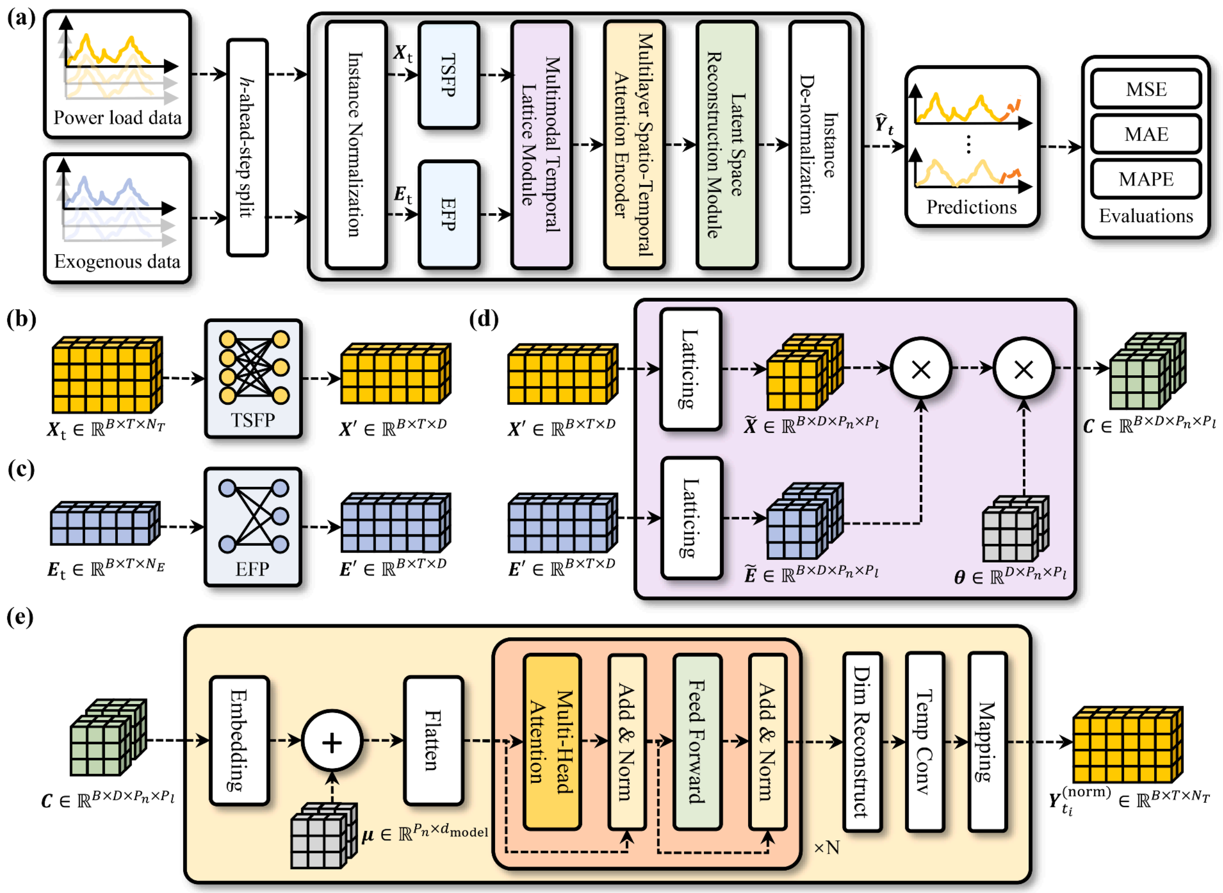

(a)
Power load data
Exogenous data
h-ahead-step split
Instance Normalization
X_t
E_t
TSFP
EFP
Multimodal Temporal Lattice Module
Multilayer Spatio-Temporal Attention Encoder
Latent Space Reconstruction Module
Instance De-normalization
$\hat{Y}_t$
Predictions
MSE
MAE
MAPE
Evaluations
(b)
$X_t \in \mathbb{R}^{B\times T\times N_T}$
TSFP
$X' \in \mathbb{R}^{B\times T\times D}$
(c)
$E_t \in \mathbb{R}^{B\times T\times N_E}$
EFP
$E' \in \mathbb{R}^{B\times T\times D}$
(d)
Latticing
$\tilde{X} \in \mathbb{R}^{B\times D\times P_n\times P_l}$
$\tilde{E} \in \mathbb{R}^{B\times D\times P_n\times P_l}$
$\theta \in \mathbb{R}^{D\times P_n\times P_l}$
$C \in \mathbb{R}^{B\times D\times P_n\times P_l}$
(e)
Embedding
$\mu \in \mathbb{R}^{P_n\times d_{\text{model}}}$
Flatten
Multi-Head Attention
Add & Norm
Feed Forward
Add & Norm
×N
Dim Reconstruct
Temp Conv
Mapping
$Y^{(\text{norm})}_{t_i} \in \mathbb{R}^{B\times T\times N_T}$

Fig. 1. The schematic illustration of the proposed TSPT model: (a) Overview of the TSPT Model Workflow; (b) Time-Series Feature Perceptron; (c) Exogenous Feature Perceptron; (d) Multimodal Temporal Lattice Module; (e) Multilayer Spatio-Temporal Attention Encoder and Latent Space Reconstruction Module.

project these normalized inputs into a shared, lower-dimensional latent space, extracting task-relevant features. To efficiently handle long input sequences and capture localized temporal patterns, **Temporal Patching** is applied to decompose the latent representations into smaller, overlapping patches. A key innovation of TSPT is the **Gated Feature Fusion (GFF)** mechanism, which adaptively combines these patched target and exogenous features, creating a unified multimodal representation that emphasizes relevant external influences. This richly fused representation is then fed into a **Multilayer Temporal Attention Encoder**, a Transformer-based module designed to capture intricate, long-range temporal patterns within the patch sequences. Finally, the **Prediction Head** is responsible for generating accurate multi-step forecasts from the encoded representations, with predictions then de-normalized to obtain the final output in the original scale. This carefully orchestrated and modular design empowers TSPT to effectively model the complexities of industrial load dynamics and seamlessly integrate diverse exogenous factors, leading to enhanced forecasting performance.

3.3. Instance normalization with RevIN

To address the challenges of varying scales and potential non-stationarity inherent in industrial power load time series, which can hinder model training and generalization, the TSPT model begins with Instance Normalization, specifically employing Reversible Instance Normalization (RevIN), also known as Instance Standard Scale, technique (Kim et al., 2021). RevIN is applied independently to each target time series within a batch $\mathbf{X}_t \in \mathbb{R}^{B\times T\times N_T}$. For each instance b and target variable k, RevIN first computes instance-specific mean $\mu_{b,k}$ and standard deviation $\sigma_{b,k}$ across the look-back window T:

$$\mu_{b,k} = \frac{1}{T}\sum_{j=1}^{T} x_{b,t-T+j,k}, \quad \sigma_{b,k} = \sqrt{\frac{1}{T}\sum_{j=1}^{T}(x_{b,t-T+j,k} - \mu_{b,k})^2}. \tag{2}$$

These statistics are then used to normalize the input time series segment:

$$x^{(\text{norm})}_{b,t_j,k} = \frac{x_{b,t_j,k} - \mu_{b,k}}{\sigma_{b,k} + \epsilon}, \tag{3}$$

where a small constant ϵ (e.g., 10^{-5}) is added to the denominator for numerical stability. Instance Normalization, in theory, reduces instance-specific distribution shifts, making the model less sensitive to scale variations and potential non-stationarities in the input data, thereby stabilizing the training process and improving convergence. Crucially, RevIN is fully reversible. The computed instance-specific statistics $\mu_{b,k}$ and $\sigma_{b,k}$ are stored and subsequently used to de-normalize the model's predictions. After the model produces normalized forecasts $\hat{\mathbf{Y}}^{(\text{norm})}_t$, the inverse RevIN transformation is applied to revert the predictions to the original scale:

$$\hat{y}_{b,t+h,k} = \hat{y}^{(\text{norm})}_{b,t+h,k} \cdot \sigma_{b,k} + \mu_{b,k}. \tag{4}$$

This reversible instance normalization strategy not only stabilizes the training process and improves convergence, but also preserves the original scale and interpretability of the forecasted values, facilitating practical deployment in real-world industrial applications. While instance normalization is applied to the target variables in the TSPT model, it can also be extended to the exogenous variables if beneficial for specific datasets or applications.

3.4. Feature extraction perceptrons: TSFP and EFP

To extract initial, task-relevant features from the raw input time series data, the TSPT architecture employs two parallel pathways of Multi-Layer Perceptrons (MLPs): the Time-Series Feature Perceptron (TSFP) and the Exogenous Factor Perceptron (EFP). These perceptrons act as dedicated feature extractors, mapping the input time series into a shared, lower-dimensional latent feature space of dimension D, which facilitates subsequent multimodal fusion and temporal encoding. This dimensionality reduction is theoretically beneficial as it focuses the model on the most salient features and reduces computational complexity in later stages. Furthermore, projecting both target and exogenous data into a shared latent space promotes feature alignment and enables effective cross-modal interaction.

The Time-Series Feature Perceptron (TSFP) is designed to process the instance-normalized target load data $\mathbf{X}_t^{(\text{norm})} \in \mathbb{R}^{B\times T\times N_T}$. The TSFP comprises a sequence of linear transformations. Specifically, in our implementation, TSFP consists of two linear layers. The first linear layer maps the input dimension N_T to an intermediate dimension D_{int}, followed by a second linear layer that projects to the latent dimension D. These layers are applied point-wise across the time dimension T, meaning that each time step is processed independently through the MLP. This operation effectively transforms the high-dimensional input space of target variables into a more compact and informative latent representation $\mathbf{X}' \in \mathbb{R}^{B\times T\times D}$:

$$\mathbf{X}' = \text{Linear}_2(\text{Linear}_1(\mathbf{X}_t^{(\text{norm})})) \in \mathbb{R}^{B\times T\times D}. \tag{5}$$

Here, $\text{Linear}_1 : \mathbb{R}^{N_T} \to \mathbb{R}^{D_{int}}$ and $\text{Linear}_2 : \mathbb{R}^{D_{int}} \to \mathbb{R}^{D}$ represent the linear transformations, where D_{int} is the intermediate dimension of the perceptrons. In our experiments, we set $D_{int} = 32$. While non-linear activations could be incorporated, the current design utilizes linear projections for initial feature extraction, focusing non-linearity in subsequent Transformer layers.

Similarly, the Exogenous Factor Perceptron (EFP) is tasked with extracting temporal correlations and relevant features from the exogenous data $\mathbf{E}_t \in \mathbb{R}^{B\times T\times N_E}$. Analogous to TSFP, the EFP utilizes its own independent MLP architecture, also consisting of two linear layers, to project the exogenous input into the same shared latent dimension D:

$$\mathbf{E}' = \text{Linear}'_2(\text{Linear}'_1(\mathbf{E}_t)) \in \mathbb{R}^{B\times T\times D}. \tag{6}$$

where $\text{Linear}'_1 : \mathbb{R}^{N_E} \to \mathbb{R}^{D_{int}}$ and $\text{Linear}'_2 : \mathbb{R}^{D_{int}} \to \mathbb{R}^{D}$ are linear transformations, mirroring the structure of TSFP. By employing separate, yet parallel, MLPs for target and exogenous data, the TSPT model allows for specialized feature extraction pathways tailored to the distinct characteristics of each data source, while simultaneously projecting them into a common latent space that enables effective cross-modal interaction in subsequent layers.

3.5. Temporal patching and gated feature fusion: Multimodal integration

To address the computational challenges associated with long input sequences and to facilitate effective integration of exogenous information, the TSPT model incorporates a Temporal Patching and Gated Feature Fusion (GFF) module. This module first segments the latent representations into smaller, manageable units, reducing sequence length and enabling localized pattern learning, and then adaptively fuses the target and exogenous features, creating a unified multimodal input for the Transformer encoder.

3.5.1. Temporal patching for efficient long-sequence processing

To enhance computational efficiency and enable the model to capture local temporal patterns, we adopt a temporal patching strategy, inspired by recent advancements in applying Transformers to long time series forecasting (Nie et al., 2023). The ‘PatchMaker‘ module segments the time dimension T of both latent representations, $\boldsymbol{X}'$ and $\boldsymbol{E}'$, into P_n overlapping patches. Each patch is of length P_l and is extracted with a stride P_s, allowing for overlap between consecutive patches. The patching operation, denoted as $\mathcal{P}(\cdot)$, transforms a tensor $\boldsymbol{Z}' \in \mathbb{R}^{B\times T\times D}$ into a patched representation $\tilde{\boldsymbol{Z}} = \mathcal{P}(\boldsymbol{Z}') \in \mathbb{R}^{B\times D\times P_n\times P_l}$. Specifically, for both target and exogenous latent representations, the patching operation $\mathcal{P} : \mathbb{R}^{B\times T\times D} \to \mathbb{R}^{B\times D\times P_n\times P_l}$ can be formally defined for each element as:

For an input tensor $\boldsymbol{Z}'$ with dimensions (B, T, D), the patched tensor $\tilde{\boldsymbol{Z}}$ with dimensions (B, D, P_n, P_l) is constructed such that for each batch $b \in \{1, \dots, B\}$, feature dimension $d \in \{1, \dots, D\}$, patch index $i \in \{1, \dots, P_n\}$, and element within the patch $j \in \{1, \dots, P_l\}$:

$$\tilde{z}_{b,d,i,j} = z'_{b,(i-1)\cdot P_s+j,d}, \tag{7}$$

where $z'_{b,t,d}$ is the element of $\boldsymbol{Z}'$ at batch b, time step t, and feature dimension d. The number of patches P_n is determined by $P_n = \lfloor (T - P_l)/P_s \rfloor + 1$. Padding is applied as configured in the ‘PatchMaker‘ module to handle cases where T is not perfectly divisible by the patch parameters. Applying temporal patching reduces the sequence length from T to P_n for the subsequent self-attention mechanism. The computational complexity of standard self-attention is quadratic with respect to sequence length, $\mathcal{O}(T^2)$. By reducing the sequence length to P_n, where typically $P_n \ll T$, temporal patching reduces the complexity to $\mathcal{O}(P_n^2)$, significantly improving computational efficiency and enabling the model to process longer historical contexts more effectively. Specifically, for both target and exogenous latent representations:

$$\tilde{\boldsymbol{X}} = \mathcal{P}(\boldsymbol{X}') \in \mathbb{R}^{B\times D\times P_n\times P_l}, \quad \tilde{\boldsymbol{E}} = \mathcal{P}(\boldsymbol{E}') \in \mathbb{R}^{B\times D\times P_n\times P_l}. \tag{8}$$

3.5.2. Gated feature fusion (GFF) for adaptive multimodal integration

To effectively integrate the information from the patched target series ($\tilde{\boldsymbol{X}}$) and the patched exogenous factors ($\tilde{\boldsymbol{E}}$), we propose a Gated Feature Fusion (GFF) mechanism, as conceptually illustrated in Fig. 1(d). GFF adaptively combines these multimodal representations through an element-wise multiplication, modulated by a learnable gating tensor $\boldsymbol{\theta} \in \mathbb{R}^{D\times P_n\times P_l}$. This gating tensor, shared across instances within a batch but learned independently for each feature dimension d, patch index i, and intra-patch position j, allows the model to learn spatially and featurally specific interaction weights between the target and exogenous features. The gating tensor $\boldsymbol{\theta}$ is implemented as a trainable parameter: $\boldsymbol{\theta} \in \mathbb{R}^{D\times P_n\times P_l} = \texttt{nn.Parameter(torch.ones(}D, P_n, P_l\texttt{))}$. The fused representation $\boldsymbol{C} \in \mathbb{R}^{B\times D\times P_n\times P_l}$ is then computed as:

$$\boldsymbol{C}_{b,d,i,j} = \tilde{\boldsymbol{X}}_{b,d,i,j} \cdot \tilde{\boldsymbol{E}}_{b,d,i,j} \cdot \boldsymbol{\theta}_{d,i,j}. \tag{9}$$

Here, $\cdot$ denotes element-wise multiplication. The gating tensor $\boldsymbol{\theta}$ acts as a dynamic filter, modulating the contribution of the exogenous factors to the fused representation based on the learned weights. This multiplicative gating mechanism allows for more nuanced interactions compared to simple additive or concatenative fusion, enabling the model to selectively emphasize or suppress the influence of exogenous variables depending on the specific temporal context and feature dimensions. The resulting fused tensor $\boldsymbol{C} \in \mathbb{R}^{B\times D\times P_n\times P_l}$ serves as a rich, multimodal representation that effectively integrates both target and exogenous information, providing a strong foundation for the subsequent temporal attention encoder. The adaptive nature of GFF, driven by the learned gating tensor, is theoretically advantageous as it allows the model to dynamically adjust its reliance on exogenous information based on the input context, potentially leading to more robust and accurate forecasting, particularly in complex industrial settings where the influence of exogenous factors can be highly variable.

3.6. Multilayer temporal attention encoder: capturing temporal dynamics

The core of the TSPT model’s ability to capture complex temporal dependencies lies in its Multilayer Temporal Attention Encoder. This encoder module is implemented as a stack of N_{enc} standard Transformer encoder layers (Vaswani et al., 2017), designed to process the sequence of P_n fused patches and extract intricate temporal relationships crucial for accurate forecasting.

3.6.1. *Input embedding and positional encoding for temporal awareness*

To prepare the fused patch representation $\boldsymbol{C} \in \mathbb{R}^{B\times D\times P_n\times P_l}$ for the Transformer encoder, two essential steps are performed: linear embedding and positional encoding. First, a linear embedding layer, $\text{Linear}_{\text{embed}} : \mathbb{R}^{P_l} \to \mathbb{R}^{d_{model}}$, projects each patch element from its original length P_l into the Transformer's model dimension d_{model}:

$$\boldsymbol{H}^{(0)}_{\text{emb}} = \text{Linear}_{\text{embed}}(\boldsymbol{C}) \in \mathbb{R}^{B\times D\times P_n\times d_{model}}. \tag{10}$$

Next, to align with the expected input format of standard Transformer encoder layers, which typically require input tensors of shape '(BatchSize, SequenceLength, FeatureDim)', the batch dimension B and the latent variable dimension D are flattened together, resulting in a reshaped tensor: $\text{Reshape} : \mathbb{R}^{B\times D\times P_n\times d_{model}} \to \mathbb{R}^{(B\cdot D)\times P_n\times d_{model}}$

$$\hat{\boldsymbol{H}}^{(0)}_{\text{emb}} = \text{Reshape}(\boldsymbol{H}^{(0)}_{\text{emb}}) \in \mathbb{R}^{(B\cdot D)\times P_n\times d_{model}}. \tag{11}$$

This reshaping strategy effectively treats each latent variable d within each instance b as an independent sequence of P_n patches to be processed by the encoder. Consequently, the self-attention mechanism within the encoder primarily focuses on capturing temporal dependencies **within** each latent variable's patch sequence. Inter-variable dependencies are learned mainly through the initial Feature Perceptrons and the final projection layers in the Prediction Head. To explicitly encode the temporal order of the patches, which is crucial for sequence modeling, learnable positional encodings $\boldsymbol{\mu} \in \mathbb{R}^{P_n\times d_{model}}$, are added to the reshaped embedded representation:

$$\boldsymbol{H}^{(0)} = \hat{\boldsymbol{H}}^{(0)}_{\text{emb}} + \boldsymbol{\mu}. \tag{12}$$

The positional encoding tensor $\boldsymbol{\mu}$ is learned during training and captures the position-specific information within the patch sequence, enabling the Transformer to be sensitive to the temporal order of the input. Dropout regularization is typically applied after the positional encodings are added to prevent overfitting. The resulting tensor $\boldsymbol{H}^{(0)}$ serves as the input to the first layer of the Transformer encoder stack, effectively initiating the temporal attention-based sequence learning process.

3.6.2. *Standard transformer encoder layer for temporal dependency extraction*

The Multilayer Temporal Attention Encoder, a core component of TSPT for capturing intricate temporal dynamics, is constructed from a stack of N_{enc} identical Transformer encoder layers. These layers are sequentially arranged to enable progressive feature refinement and hierarchical extraction of temporal dependencies from the input patch sequences. As conceptually illustrated in Fig. 1(e), each encoder layer is composed of two primary sub-layers: a Multi-Head Self-Attention (MHA) mechanism and a position-wise Feed-Forward Network (FFN). To ensure efficient training of this deep architecture and to facilitate unimpeded information propagation, residual connections and Layer Normalization (LayerNorm) Xu et al. (2019) are integrally incorporated around each sub-layer, promoting gradient stability and accelerating convergence. Let $\boldsymbol{H}^{(i-1)} \in \mathbb{R}^{(B\cdot D)\times P_n\times d_{\text{model}}}$ represent the input tensor to the ith encoder layer ($i = 1,\dots,N_{enc}$), representing a batch of $B\cdot D$ sequences of P_n patches, each with a feature dimension of d_{model}. The computational steps within each layer are detailed as follows:

(1) **Multi-Head Self-Attention (MHA) Sub-layer:** This sub-layer is the linchpin of the Transformer encoder's ability to model long-range temporal dependencies. By employing the self-attention mechanism, the model can dynamically weigh the importance of different temporal positions within the input sequence when processing each position, effectively capturing dependencies regardless of their temporal distance. The MHA sub-layer operates by projecting the input $\boldsymbol{H}^{(i-1)}$ into three sets of representations: queries ($\boldsymbol{Q}$), keys ($\boldsymbol{K}$), and values ($\boldsymbol{V}$), through linear transformations $\text{Linear}_Q, \text{Linear}_K, \text{Linear}_V : \mathbb{R}^{d_{model}} \to \mathbb{R}^{d_{model}}$. To enhance the attention mechanism's capacity and stability, multi-head attention is employed, where the attention computation is performed in parallel across h independent attention heads. For each head $j \in \{1,\dots,h\}$, we derive queries $\boldsymbol{Q}_j$, keys $\boldsymbol{K}_j$, and values $\boldsymbol{V}_j$ by projecting $\boldsymbol{H}^{(i-1)}$. The scaled dot-product attention is then computed for each head as:

$$\text{Attention}_j(\boldsymbol{Q}_j,\boldsymbol{K}_j,\boldsymbol{V}_j) = \text{softmax}\left(\frac{\mathrm{Q_j K_j^\top}}{\sqrt{\mathrm{d_k}}}\right)\mathrm{V_j}, \tag{13}$$

where $d_k = d_{model}/h$ is the dimension of the key and query vectors for each head, and the scaling factor $\sqrt{d_k}$ is applied to mitigate the effect of potentially large dot product values, preventing the softmax function from saturating and ensuring stable gradient behavior. The outputs from all h attention heads are then concatenated and linearly projected back to the model dimension d_{model} using a linear transformation $\text{Linear}_{\text{concat}} : \mathbb{R}^{h\cdot d_v} \to \mathbb{R}^{d_{model}}$, yielding the final output of the MHA sub-layer, denoted as $\text{MHA}(\boldsymbol{H}^{(i-1)})$. This multi-head self-attention mechanism empowers the model to simultaneously attend to information from different representation subspaces at different positions, enriching the learned temporal representations. Unlike recurrent neural networks, which process sequences sequentially, self-attention enables parallel processing of all positions and effectively captures long-range dependencies without the vanishing gradient problem inherent in deep RNNs.

(2) **Add & Norm Layer 1: Residual Connection and Layer Normalization for Attention Output:** To facilitate efficient training and stable gradient propagation, a residual connection is established by directly adding the input $\boldsymbol{H}^{(i-1)}$ to the output of the MHA sub-layer. Dropout regularization is applied to the MHA output before the residual connection to prevent overfitting. Subsequently, Layer Normalization is applied to the summed output to normalize the activations:

$$\boldsymbol{A}^{(i)} = \text{LayerNorm}(\boldsymbol{H}^{(i-1)} + \text{Dropout}(\text{MHA}(\boldsymbol{H}^{(i-1)}))) \tag{14}$$

Layer Normalization, applied independently to each sample, normalizes the features across all dimensions within each layer, thereby reducing internal covariate shift and promoting faster and more stable training dynamics. The residual connection, in conjunction with Layer Normalization, ensures that the information from earlier layers can be effectively propagated through the deeper network, mitigating the vanishing gradient problem and enabling the training of deeper and more complex models.

(3) **Feed-Forward Network (FFN) Sub-layer: Position-wise Non-linear Transformation:** The FFN sub-layer further processes the attention-aggregated representations $\boldsymbol{A}^{(i)}$ in a position-wise manner, independently applying the same non-linear transformation to each position in the sequence. This position-wise processing enhances the model's capacity to learn complex, non-linear feature transformations at each temporal location. The FFN typically consists of a two-layer feed-forward network with a non-linear activation function, such as the Gaussian Error Linear Unit (GELU), in between. Specifically, it comprises a linear transformation $\text{Linear}'_1 : \mathbb{R}^{d_{model}} \to \mathbb{R}^{d_{ffn}}$ that projects the input to a higher-dimensional intermediate space of dimension d_{ffn}, followed by the GELU non-linearity, and then a second linear transformation $\text{Linear}'_2 : \mathbb{R}^{d_{ffn}} \to \mathbb{R}^{d_{model}}$ that projects back to the model dimension:

$$\text{FFN}(\boldsymbol{A}^{(i)}) = \text{Linear}'_2(\text{GELU}(\text{Linear}'_1(\boldsymbol{A}^{(i)}))). \tag{15}$$

The GELU activation function introduces non-linearity, enabling the model to learn more intricate and non-linear relationships within the temporal data, and the expansion to a higher dimension d_{ffn} in the intermediate layer increases the model's representation capacity.

(4) **Add & Norm Layer 2: Residual Connection and Layer Normalization for FFN Output:** Analogous to the first Add & Norm layer, a second residual connection and Layer Normalization are applied to the output of the FFN sub-layer. The original output of the first Add & Norm" layer, $\boldsymbol{A}^{(i)}$, is added to the output of the FFN sub-layer, with Dropout regularization applied to the FFN output prior to the residual connection. The summed output is then normalized using

Layer Normalization, resulting in the final output of the ith encoder layer:

$$\boldsymbol{H}^{(i)} = \text{LayerNorm}(\boldsymbol{A}^{(i)} + \text{Dropout}(\text{FFN}(\boldsymbol{A}^{(i)}))). \tag{16}$$

The resulting output $\boldsymbol{H}^{(i)}$ from the ith encoder layer then serves as the input to the subsequent $(i+1)$th layer, creating a deep stack of encoder layers that progressively refine and abstract the temporal representations. This layered architecture, combined with the power of self-attention and non-linear transformations, allows the TSPT encoder to effectively capture and model complex, multi-scale temporal patterns inherent in industrial load time series data. Algorithm 1 provides a concise algorithmic summary of the standard Transformer encoder layer operations.

Algorithm 1: Standard Transformer Encoder Layer.

```
Input: Input representation $\boldsymbol{H}^{(i-1)} \in \mathbb{R}^{(B\cdot D)\times P_n\times d_{\text{model}}}$
Input: Layer parameters (MHA weights, FFN weights,
       LayerNorm parameters, Dropout rate)
Output: Output representation $\boldsymbol{H}^{(i)} \in \mathbb{R}^{(B\cdot D)\times P_n\times d_{\text{model}}}$
1  $\boldsymbol{A}_{\text{attn}} \leftarrow \text{MHA}(\boldsymbol{H}^{(i-1)})$ ;   // Compute multi-head
     self-attention using Eq. (13)
2  $\boldsymbol{A}^{(i)} \leftarrow \text{LayerNorm}(\boldsymbol{H}^{(i-1)} + \text{Dropout}(\boldsymbol{A}_{\text{attn}}))$ ;   // Residual
     connection 1 + Dropout + LayerNorm
3  $\boldsymbol{H}_{\text{ffn}} \leftarrow \text{FFN}(\boldsymbol{A}^{(i)})$ ; // Compute position-wise feed-forward
     network using Eq. (15)
4  $\boldsymbol{H}^{(i)} \leftarrow \text{LayerNorm}(\boldsymbol{A}^{(i)} + \text{Dropout}(\boldsymbol{H}_{\text{ffn}}))$ ;   // Residual
     connection 2 + Dropout + LayerNorm
5  return $\boldsymbol{H}^{(i)}$;
```

3.7. *Prediction head: Generating multi-horizon forecasts*

The final component of the TSPT architecture is the Prediction Head, responsible for mapping the high-level temporal representations extracted by the encoder to the desired multi-horizon forecasts $\hat{\mathbf{Y}}_t \in \mathbb{R}^{B\times H\times N_T}$. This module comprises a sequence of reshaping, flattening, and linear projection layers.

3.7.1. *Reshaping and feature consolidation*

To prepare the encoder output for the final projection steps, the tensor $\boldsymbol{H}^{(N_{enc})} \in \mathbb{R}^{(B\cdot D)\times P_n\times d_{model}}$ from the last encoder layer undergoes reshaping and flattening operations. First, it is reshaped to separate the batch and latent variable dimensions: Reshape$'$: $\mathbb{R}^{(B\cdot D)\times P_n\times d_{model}} \rightarrow \mathbb{R}^{B\times D\times P_n\times d_{model}}$

$$\boldsymbol{G}_{\text{enc}} = \text{Reshape}'(\boldsymbol{H}^{(N_{enc})}) \in \mathbb{R}^{B\times D\times P_n\times d_{model}}. \tag{17}$$

Then, to consolidate the temporal information encoded across the patches and the model dimension, the patch dimension P_n and the model dimension d_{model} are flattened together, resulting in a representation $\boldsymbol{G}_{\text{flat}}$ of shape $\mathbb{R}^{B\times D\times(P_n\cdot d_{model})}$: Flatten : $\mathbb{R}^{\mathrm{B\times D\times P_n\times d_{model}}} \rightarrow \mathbb{R}^{\mathrm{B\times D\times (P_n\cdot d_{model})}}$

$$\boldsymbol{G}_{\text{flat}} = \text{Flatten}(\mathrm{G_{enc}}, \text{dims} = (2,3)) \in \mathbb{R}^{\mathrm{B\times D\times (P_n\cdot d_{model})}}. \tag{18}$$

This flattened tensor $\boldsymbol{G}_{\text{flat}}$ effectively aggregates the temporal information learned by the encoder across all patches and feature dimensions, preparing it for the subsequent projection layers.

3.7.2. *Horizon projection layer for multi-step forecasting*

To generate multi-step forecasts spanning the horizon H, a linear projection layer, denoted as Linear$_{\text{horizon}}$: $\mathbb{R}^{(P_n\cdot d_{model})} \rightarrow \mathbb{R}^H$, is applied to the flattened representation $\boldsymbol{G}_{\text{flat}}$. This layer maps the consolidated temporal features to the desired forecast horizon, effectively summarizing the encoded sequence information into H future time steps for each latent dimension D:

$$\boldsymbol{G}_H = \text{Linear}_{\text{horizon}}(\boldsymbol{G}_{\text{flat}}) \in \mathbb{R}^{B\times D\times H}. \tag{19}$$

Table 2
Proportion of missing values for building types.

Building type	Missing rate per year (%)					
	2016	2017	2018	2019	2020	2021
Office	2.892	1.478	0.171	1.22	17.349	3.378
Residential	3.097	1.05	5.171	3.824	32.479	33.311
Commercial	11.452	17.551	11.13	7.021	5.316	5.719
Public	57.399	0.913	1.256	0.765	62.261	84.087

The Linear$_{\text{horizon}}$ layer utilizes weights $\boldsymbol{W}_t \in \mathbb{R}^{(P_n\cdot d_{model})\times H}$ and biases $\boldsymbol{b}_t \in \mathbb{R}^H$. In our implementation, Linear$_{\text{horizon}}$ is realized using a Graph-Augmented Regression (GAR) module (Liu et al., 2022a), which, in its simplest form, reduces to a linear layer but can be extended for more complex temporal projections if needed.

3.7.3. *Variable projection layer for target variable output*

Finally, to map the latent dimension D to the number of target output variables N_T, a variable projection layer, Linear$_{\text{variable}}$: $\mathbb{R}^D \rightarrow \mathbb{R}^{N_T}$, is employed. Before applying this layer, the dimensions of $\boldsymbol{G}_H$ are permuted to shape $\mathbb{R}^{B\times H\times D}$: Permute : $\mathbb{R}^{B\times D\times H} \rightarrow \mathbb{R}^{B\times H\times D}$. The variable projection layer then performs the final linear transformation:

$$\hat{\mathbf{Y}}_t^{(\text{norm})} = \text{Linear}_{\text{variable}}(\text{Permute}(\boldsymbol{G}_H)) \in \mathbb{R}^{B\times H\times N_T}. \tag{20}$$

The Linear$_{\text{variable}}$ layer uses weights $\boldsymbol{W}_m \in \mathbb{R}^{D\times N_T}$ and biases $\boldsymbol{b}_m \in \mathbb{R}^{N_T}$. The output of this layer, $\hat{\mathbf{Y}}_t^{(\text{norm})}$, represents the model's normalized multi-horizon forecast. This normalized forecast is subsequently denormalized using the inverse RevIN transformation Eq. (4) to produce the final, physically scaled output prediction $\hat{\mathbf{Y}}_t \in \mathbb{R}^{B\times H\times N_T}$, ready for evaluation and practical application.

4. Experimental setup

4.1. *Dataset and preprocessing*

This study employs a 6-year (from January 1, 2016 to December 31, 2021) power load dataset (Zhou et al., 2023) from various types of buildings in an industrial park located in Suzhou, China. The dataset consists of historical power load observations and historical exogenous factor observations. The historical power load observations include power load data for four building types: office buildings, residential buildings, commercial buildings, and public buildings. The data is available at three sampling intervals: 15 min, 30 min, and 1 h. The historical exogenous factor observations contain data on two environmental factors: temperature and humidity, sampled at a 1-h interval. For this study, we utilize the power load data and exogenous factor data with a 1-h sampling interval to ensure consistency and alignment between the two data types.

The original dataset contains missing values, as shown in Table 2. To address this issue, we employ the Multiple Imputation by Chained Equations (MICE) method during data preprocessing to impute the missing values. MICE is advantageous due to its ability to handle relationships between multiple variables, estimate missing values more accurately, and apply to different types of data, including continuous and categorical variables. This method preserves the underlying structure of the data and reduces information loss.

For model training and evaluation, we split the preprocessed dataset into training and test sets, with 80 % of the data used for training and the remaining 20 % for testing. The training set is used during the model training process and participates in the backpropagation algorithm for parameter optimization. The test set is used to evaluate the final performance of the trained model and compare it with other models.

4.2. *Baseline methods*

We compare the performance of the proposed TSPT model with 19 classical time series prediction models to assess its efficacy. These methods include linear models, classic recurrent neural networks, transformer models, derivative transformer-based series models, graph neural networks, and derivative graph neural networks. The baselines are described as follows:

(1) **AR** (Wang & Cai, 2022): AutoRegression is a linear model that uses independent weights for each variable.
(2) **LSTM** (Józefowicz et al., 2015): Long Short-Term Memory (LSTM), a recurrent neural network variant with memory cells and three distinct gates, adept at capturing long-term dependencies in sequential data.
(3) **GRU** (Chung et al., 2014): Gated Recurrent Unit (GRU), a simplified variant of LSTM with fewer parameters, combining the forget and input gates into a single update gate.
(4) **ED** (Cho et al., 2014): Encoder-Decoder (ED) architecture, consisting of an encoder RNN that processes the input sequence and a decoder RNN that generates the output sequence.
(5) **CNNRNNRes** (Wu et al., 2018): Convolutional Neural Network (CNN) based models for sequence modeling, combining the advantages of recurrent neural networks and residual networks.
(6) **LSTNet** (Lai et al., 2018): Long- and Short-term Time-series Network (LSTNet), a deep learning framework that captures both long-term and short-term temporal patterns using CNN and RNN components.
(7) **DLinear** (Zeng et al., 2023): Decomposition-Linear (DLinear), a linear model that decomposes the time series into trend, seasonal, and residual components before applying linear regression.
(8) **Informer** (Zhou et al., 2021): An efficient transformer-based model designed for long sequence time-series forecasting, utilizing a ProbSparse self-attention mechanism.
(9) **Autoformer** (Wu et al., 2021a): A transformer-based model with a novel decomposition architecture and an Auto-Correlation mechanism to capture both trend and seasonal patterns.
(10) **FEDformer** (Zhou et al., 2022): Frequency Enhanced Decomposed Transformer (FEDformer), a method that combines Transformer with seasonal-trend decomposition for improved forecasting performance.
(11) **STAEformer** (Liu et al., 2023a): Spatio-Temporal Adaptive Embedding Transformer (STAEformer), a model that leverages spatial and temporal attention mechanisms for multivariate time series forecasting.
(12) **Triformer** (Cirstea et al., 2022): A Transformer-based model that integrates triple attention mechanisms (temporal, feature, and instance attention) for improved time series forecasting.
(13) **PatchTST** (Nie et al., 2023): Patch-based Time Series Transformer (PatchTST), a transformer-based model that applies patching to capture local dependencies in time series data.
(14) **STID** (Shao et al., 2022): Spatial-Temporal Identity (STID), a model that learns spatial and temporal representations separately and fuses them for multivariate time series forecasting.
(15) **NHiTS** (Challu et al., 2023): Neural Hierarchical Interpolation for Time Series (NHiTS), a deep learning model that hierarchically interpolates time series at different scales for multi-horizon forecasting.
(16) **GAIN** (Wang et al., 2023a): Graph Ambient Intelligent Network (GAIN) combines graph attention networks with meteorological factors for multivariate time series forecasting.
(17) **MAGNet** (Chen et al., 2023): Multi-scale Attention Gated Network (MAGNet), a model that integrates multi-scale feature extraction, attention mechanisms, and gating for time series forecasting.
(18) **GraphWaveNet** (Wu et al., 2019): A model that combines graph convolutions with dilated causal convolutions for spatio-temporal forecasting on graph-structured time series.
(19) **TCOAT** (Hu et al., 2024): Novel wind power forecasting method employing temporal and spatial dependencies, attention mechanisms, and fusion layers.

The baseline methods for time series prediction encompass a wide range of approaches, including simple linear models (AR), classic recurrent neural networks (LSTM, GRU, ED), convolutional neural networks (CNNRNNRes), and decomposition-based methods (DLinear). State-of-the-art Transformer models and their derivatives (Informer, Autoformer, FEDformer, STAEformer, Triformer, PatchTST) leverage self-attention mechanisms to capture long-range dependencies and complex patterns, while graph-based models (GraphWaveNet) incorporate spatial and temporal dependencies. Other notable approaches include identity-based modeling (STID), hierarchical interpolation (NHiTS), multi-scale learning (MAGNet), and novel architectures that combine attention, gating, and fusion layers (GAIN, TCOAT).

4.3. Evaluation metrics

To evaluate the performance of the proposed TSPT model, we employ three widely used metrics in power load forecasting: Mean Squared Error (MSE), Mean Absolute Error (MAE), and Pearson Correlation Coefficient (PCC). These metrics are defined as follows:

$$\mathrm{MSE}(\boldsymbol{Y}, \hat{\boldsymbol{Y}}) = \frac{1}{N}\sum_{i=1}^{N}(Y_i - \hat{Y}_i)^2, \tag{21}$$

$$\mathrm{MAE}(\boldsymbol{Y}, \hat{\boldsymbol{Y}}) = \frac{1}{N}\sum_{i=1}^{N}\left|Y_i - \hat{Y}_i\right|, \tag{22}$$

$$\mathrm{PCC}(\boldsymbol{Y}, \hat{\boldsymbol{Y}}) = \frac{\sum_{i=1}^{N}(Y_i - \bar{Y})(\hat{Y}_i - \bar{\hat{Y}})}{\sqrt{\sum_{i=1}^{N}(Y_i - \bar{Y})^2} \cdot \sqrt{\sum_{i=1}^{N}(\hat{Y}_i - \bar{\hat{Y}})^2}}, \tag{23}$$

where $\boldsymbol{Y}$ represents the actual power load values, $\hat{\boldsymbol{Y}}$ denotes the predicted values, N is the number of time steps in the test set, and $\bar{Y}$ and $\bar{\hat{Y}}$ are the mean values of $\boldsymbol{Y}$ and $\hat{\boldsymbol{Y}}$, respectively.

4.4. Implementation and settings

All models are implemented using the PyTorch v2.0.1 framework. The experiments are conducted on a server equipped with an Intel(R) Xeon(R) Gold 6226R CPU (2.90GHz), 128GB of memory, and an NVIDIA RTX A6000 48GB GPU.

Each model is trained using the Adam optimizer with MSE as the loss function. The training process is conducted for a maximum of 2000 epochs, and the model checkpoint with the best validation performance in terms of MSE is selected as the final model. Each experiment is repeated three times with random seed 40, 41, and 42, respectively, and the reported results represent the average performance and variance across these three runs. Hyperparameter tuning is performed using a grid search approach to identify the optimal configuration for each model. The specific ranges explored and the final selected values are detailed in Table 3.

5. Results and analyses

5.1. Comparison with baseline methods

Table 4 compares the proposed TSPT model with 19 baseline methods for multi-step time series prediction. We evaluated models using MSE, MAE, and PCC for 24-h ($H = 24$) and 48-h ($H = 48$) horizons, both with and without exogenous factors.

- For $H = 24$ using only power load data, TSPT achieves the best MSE (7.568×10^3), MAE (52.096), and PCC (84.215 %), outperforming all baselines. PatchTST is second-best (MSE: 7.881×10^3, MAE: 53.288, PCC: 83.908 %).
- At $H = 48$, TSPT also maintains the best performance of all metrics with the lowest MSE (10.193×10^3), lowest MAE (61.676), and highest PCC (80.501 %). While PatchTST is close with a slightly worse

Table 3
Hyper-parameter settings.

Model	Parameter	Option range
LSTM	Hidden size	$\{2^4, 2^5, 2^6\}$
GRU	Layer number	1-3 (1 per step)
ED	Bidirectional	True, False
	CNN kernel size	3-9 (2 per step)
	CNN out channels	$\{2^2, 2^3, 2^4, 2^5, 2^6\}$
CNNRNNRes	RNN hidden size	$\{2^4, 2^5, 2^6\}$
	RNN layers	1-3 (1 per step)
	Residual window size	1-7 (1 per step)
	Residual ratio	0.1-0.5 (0.1 per step)
	Skip window size	1-7 (1 per step)
LSTNet	Skip GRU hidden size	$\{2^4, 2^5, 2^6\}$
	Skip GRU layers	1-3 (1 per step)
DLinear	Decomposition kernel size	3-9 (2 per step)
	Encoder layers	1-3 (1 per step)
Informer	Decoder layers	1-3 (1 per step)
Autoformer	The label length	1-10 (1 per step)
FEDformer	The numbers of heads	$\{2^2, 2^3, 2^4\}$
STAEformer	The dimension of the model	$\{2^4, 2^5, 2^6\}$
	The dimension of feed-forward	$\{2^4, 2^5, 2^6\}$
	The channel size	$\{2^3, 2^4, 2^5\}$
Triformer	The patch size	{2, 5}
	The dimension of memory state	{2, 5}
	Encoder layers	1-3 (1 per step)
PatchTST	The numbers of heads	$\{2^2, 2^3, 2^4\}$
	The dimension of the model	$\{2^4, 2^5, 2^6\}$
	The patch length	{2, 5}
	The patch stride	{1, 2, 5}
	The Number of Blocks	1-3 (1 per step)
NHiTS	The layer number	1-3 (1 per step)
	The hidden size	$\{2^4, 2^5, 2^6\}$
	The pooling size	2-8 (2 per step)
STID	Node dimension	$\{2^4, 2^5, 2^6\}$
	The numbers of heads	$\{2^2, 2^3, 2^4\}$
GAIN	GAT hidden size	$\{2^4, 2^5, 2^6\}$
	The number of heads of GAT	$\{2^0, 2^1, 2^2, 2^3, 2^4\}$
	The label length	1-10 (1 per step)
	The input channel size	$\{2^2, 2^3, 2^4\}$
MAGNet	The convolution channel size	$\{2^2, 2^3, 2^4\}$
	The residual channel size	$\{2^2, 2^3, 2^4\}$
	The skip channel size	$\{2^2, 2^3, 2^4\}$
	The end channel size	$\{2^2, 2^3, 2^4\}$
GraphWaveNet	The residual channels	$\{2^4, 2^5, 2^6\}$
	The dilation channels	$\{2^4, 2^5, 2^6\}$
	The skip channels	$\{2^6, 2^7, 2^8, 2^9\}$
	The end channels	$\{2^6, 2^7, 2^8, 2^9\}$
TCOAT	RNN hidden size	$\{2^4, 2^5, 2^6\}$
	RNN layers	1-3 (1 per step)
	Bidirectional	True, False
	Residual window size	1-7 (1 per step)
	Residual ratio	0.1-0.5 (0.1 per step)

MSE of 10.743×10^3, MAE of 63.827, and PCC of 78.928 %, other models like Triformer and STID achieve comparable MSE, MAE, and PCC scores.

- With exogenous factors, TSPT achieves the best results for $H = 24$ across all metrics. At $H = 48$, TSPT's performance is among the best as well. Compared with the version without integration of exogenous factors, the TSPT that considers external influences has improved in most metrics. The ability of TSPT to maintain performance under exogenous data highlights its adaptability to the variability of exogenous factors. Meanwhile, the slight performance improvement also demonstrates the GFF mechanism's effective capture of the patterns of exogenous factor variations, thereby aiding in enhancing the prediction performance of the target data.
- Some baselines (e.g., GRU, Informer, Autoformer) show poor performance with high MSE/MAE and low PCC scores. This suggests a limitation in these models in capturing temporal information effectively, particularly in a longer horizon.
- Linear models (e.g., AR, DLinear) demonstrate excellent performance across experiments with $H = 24$ and $H = 48$, both with and without the integration of exogenous data. This indicates that the time series of the industrial park smart meter dataset exhibits certain linear correlations over time. However, some non-linear models (e.g., Triformer, NHiTS) outperform linear models in prediction performance, suggesting the presence of non-linear relationships in the dataset at local or fine-grained temporal scales. In comparison with non-linear models that fail to accurately capture these non-linear relationships (e.g., LSTM, Informer), our findings provide reliable guidance for the model design of smart meter forecasting solutions.
- Most baseline methods do not exhibit an improvement in predictive performance in the fusion experiments with the inclusion of meteorological factors; in fact, their prediction accuracy even declines. Notably, the PatchTST model's predictive performance in the fusion experiments deteriorates significantly compared to the experiments without external factors. Specifically, the MSE degrades by an average of 102.75 %, the MAE by 65.66 %, and the PCC by 22.19 %. This indicates that most baselines are unable to adapt to the variability introduced by the inclusion of exogenous factors, nor can they extract effective information from the patterns of exogenous factor variations to enhance prediction accuracy.

In summary, the TSPT model consistently outperforms or remains competitive with state-of-the-art baselines. The use of exogenous factors enhances TSPT's performance, highlighting its ability to capture complex temporal dependencies and leverage external information effectively for accurate time series forecasting.

5.2. Cross-correlation analyses

To examine the robustness of the TSPT model against variations in cross-correlations among time series from different building categories within the dataset, specifically the changes in cross-correlations, we extracted data from the commercial and office building areas within the four building regions of the original dataset. These data were then combined with external data for experimentation. The experimental results are presented in Table 5. To better illustrate the impact of cross-correlation variations within the dataset on the model's predictive performance and to further substantiate the robustness of the TSPT model as demonstrated, we selected several representative or high-performing baseline models for comparative experiments conducted over the same period.

- In the prediction of the commercial building category, TSPT achieved the overall best predictive performance. At $H = 24$, compared to DLinear, which ranked second in comprehensive predictive performance, TSPT led by 4.40 % in MSE, 0.35 % in MAE, and 0.22 % in PCC. At $H = 48$, TSPT's MSE led by 1.40 %, and PCC remained in the leading position. Although the MAE performance was slightly worse than DLinear's 84.431, the gap was only 0.35 %. In the office building category, TSPT's predictive performance remained the best overall. At $H = 24$, compared to the suboptimal model NHiTS, TSPT led by 1.85 % in MSE and 0.56 % in MAE. At $H = 48$, TSPT's MSE, MAE, and PCC led by 8.61 %, 5.72 %, and 5.53 %, respectively. Although TSPT's PCC was slightly behind NHiTS's 77.438 at $H = 24$ and behind Triformer's 73.452 at $H = 48$, overall, TSPT's predictive performance remained the best. The fact that the TSPT model maintained the leading prediction accuracy in both commercial and office building categories demonstrates that the TSPT model possesses strong robustness against variations in the cross-correlations of the dataset.
- The PatchTST model shows the worst performance among all evaluated models in the cross-correlations experiment. Its high prediction errors in all metrics, inability to capture complex cross-correlations

Table 4
Performance comparison in terms of three metrics on two data collections. Bold data indicates the best performance, underlined second, bottom wavy line is the worst. The model marked with the " + " symbol indicates that the model also takes meteorological factors into account.

Model	$H = 24$			$H = 48$		
	MSE ($\times10^3$)	MAE	PCC (%)	MSE ($\times10^3$)	MAE	PCC (%)
AR	9.305±0.048	57.311±0.261	82.802±0.017	12.434±0.07	67.374±0.244	77.489±0.058
LSTM	18.873±1.513	87.65±2.74	66.006±3.005	28.845±0.494	112.488±0.97	55.388±2.079
GRU	21.571±1.552	91.628±3.251	68.555±1.578	24.527±3.717	100.728±5.119	62.668±2.93
ED	20.803±2.29	89.471±4.268	69.726±1.567	33.339±2.718	117.778±5.283	51.073±3.013
CNNRNNRes	16.384±1.226	82.859±2.431	74.343±1.213	26.314±2.022	109.441±5.549	62.343±3.614
LSTNet	10.572±0.557	64.099±1.941	81.296±0.76	14.368±0.163	77.79±0.758	74.667±0.536
DLinear	10.334±0.019	60.841±0.13	81.542±0.068	13.998±0.062	72.812±0.238	76.421±0.028
Informer	39.453 ± 22.107	132.401 ± 43.914	49.469 ± 9.168	37.092 ± 4.47	127.668 ± 8.365	39.586 ± 5.262
Autoformer	26.993±3.297	110.136±8.959	63.492±1.161	29.795±0.053	116.614±0.661	55.006±0.693
FEDformer	15.153±0.584	81.52±1.097	72.265±0.497	18.951±0.664	90.225±1.522	67.521±0.605
STAEformer	12.423±1.658	68.979±4.113	78.995±2.552	16.931±0.373	79.285±0.653	71.63±0.518
Triformer	9.062±0.241	59.023±0.789	83.082±0.385	11.587±0.434	66.57±1.579	78.031±0.338
PatchTST	7.881 ± 0.067	53.288 ± 0.563	83.908 ± 0.194	10.743 ± 0.106	63.827 ± 0.603	78.928 ± 0.332
STID	9.961±0.702	57.549±0.647	80.427±0.903	11.719±0.225	66.489±0.568	77.573±0.767
NHiTS	8.909±0.25	57.064±0.498	82.284±0.876	11.734±0.207	67.235±1.104	78.334±0.307
GAIN	31.154±5.146	103.684±8.392	68.911±4.052	37.161±0.84	120.764±1.239	58.982±1.472
MAGNet	13.164±1.022	73.969±2.055	78.115±2.73	16.528±1.011	84.323±1.497	71.44±2.698
GraphWaveNet	10.597±0.076	62.328±0.176	80.869±0.104	14.354±0.217	74.722±0.958	75.507±0.274
TCOAT	12.243±0.951	69.085±2.206	78.785±1.284	16.375±0.185	82.198±0.696	74.469±0.312
TSPT	**7.568±0.049**	**52.096±0.706**	**84.215±0.212**	**10.193±0.068**	**61.676±0.452**	**80.501±0.603**
AR(+)	9.164±0.024	56.892±0.4	82.608±0.02	12.356±0.004	67.455±0.149	77.511±0.026
LSTM(+)	20.168±1.169	90.071±1.639	70.412±1.401	29.606±0.919	113.723±1.894	54.999±3.23
GRU(+)	24.466 ± 0.708	99.776±1.919	65.556±0.307	34.636 ± 1.921	123.829 ± 2.163	50.225 ± 3.038
ED(+)	21.83±2.222	93.491±6.958	68.137±2.061	31.372±1.844	118.341±5.896	56.34±1.839
CNNRNNRes(+)	19.824±0.946	92.725±2.724	69.9±2.208	23.705±1.753	104.122±3.826	62.279±1.075
LSTNet(+)	10.31±0.819	64.729±3.6	81.979±0.422	12.919±0.861	73.41±3.692	77.032±1.577
DLinear(+)	10.16±0.019	60.543±0.027	81.514±0.006	13.752±0.021	72.177±0.143	76.329±0.021
Informer(+)	18.874±1.047	87.717±3.028	71.071±4.057	27.583±0.328	107.543±1.404	61.047±1.095
Autoformer(+)	22.562±2.628	100.194 ± 5.317	65.493 ± 1.989	24.299±5.204	102.643±10.281	62.141±4.144
FEDformer(+)	11.691±0.79	70.772±2.115	78.583±1.239	15.703±0.659	82.995±1.479	72.054±0.581
STAEformer(+)	11.031±1.091	65.755±3.629	79.975±0.985	17.782±1.011	81.562±2.615	71.327±0.342
Triformer(+)	9.083±0.484	58.744±2.187	83.09±0.222	12.286±1.038	70.432±4.246	77.62±1.225
PatchTST(+)	17.451±7.79	92.564±30.666	64.574±15.161	20.309±6.689	101.453±26.062	62.127±12.072
STID(+)	8.41 ± 0.316	55.583 ± 1.73	83.018 ± 0.716	11.364 ± 0.267	65.861 ± 0.69	78.572 ± 0.033
NHiTS(+)	8.92±0.494	57.851±2.061	82.03±1.193	11.595±0.29	66.48±1.179	78.103±0.682
GAIN(+)	33.62±1.182	108.33±3.149	66.877±1.919	36.548±2.714	118.589±3.607	59.081±3.038
MAGNet(+)	11.267±0.396	68.727±1.62	79.915±0.396	15.842±0.763	83.56±3.579	73.329±0.273
GraphWaveNet(+)	11.344±1.146	66.157±5.678	81.017±0.36	14.337±0.114	74.241±0.262	75.669±0.593
TCOAT(+)	12.285±0.39	69.088±1.08	80.485±0.697	16.149±0.821	80.624±1.927	75.056±0.906
TSPT(+)	**7.456±0.117**	**51.792±0.209**	**84.505±0.102**	**10.252±0.137**	**61.33±0.367**	**80.386±0.352**

Table 5
Performance comparison in terms of three metrics on "commercial" and "office" sub data collections. Bold data indicates the best performance, underlined second, bottom wavy line is the worst. These two datasets all take meteorological factors into account.

Model		$H = 24$			$H = 48$		
		MSE ($\times10^3$)	MAE	PCC (%)	MSE ($\times10^3$)	MAE	PCC (%)
commercial	AR(+)	12.34±0.013	71.553±0.058	92.098±0.024	17.084±0.007	85.138±0.05	89.147±0.026
	LSTNet(+)	13.759±0.272	79.082±1.32	91.324±0.064	24.73±2.484	107.402±4.734	88.793±0.222
	DLinear(+)	12.33±0.135	71.102 ± 0.401	92.154 ± 0.008	16.975 ± 0.021	**84.431±0.198**	89.289±0.028
	Triformer(+)	14.459±1.111	82.515±3.574	91.868±0.385	19.607±1.03	96.61±3.644	88.465±0.345
	PatchTST(+)	24.565 ± 9.485	111.878 ± 28.818	82.836 ± 5.996	31.635 ± 8.922	130.579 ± 22.019	78.311 ± 5.232
	STID(+)	12.141 ± 0.09	73.207±0.362	92.034±0.218	17.116±0.389	87.501±1.088	89.298 ± 0.502
	NHiTS(+)	14.324±0.875	81.767±2.549	91.671±0.261	20.084±0.515	96.067±1.32	88.335±0.228
	TSPT(+)	**11.787±0.252**	**70.85±0.504**	**92.358±0.412**	**16.737±0.462**	84.727 ± 0.966	**89.313±0.456**
office	AR(+)	15.778±0.026	72.631±0.297	71.999±0.143	20.557 ± 0.019	85.556±0.415	65.931±0.086
	LSTNet(+)	13.011±0.582	69.42±3.279	74.614±0.334	18.098±0.124	85.204±3.736	69.309±1.186
	DLinear(+)	15.618±0.054	70.924±0.292	72.994±0.08	20.184±0.019	83.393±0.309	67.122±0.062
	Triformer(+)	12.475±1.573	64.068±6.388	75.536±1.792	14.842±1.636	72.912±5.96	**73.452±1.379**
	PatchTST(+)	15.801 ± 4.601	83.112 ± 18.736	62.181 ± 12.58	19.014±3.308	92.895 ± 10.453	59.252 ± 11.143
	STID(+)	10.506±0.003	57.852±1.033	76.606±0.816	13.996±0.261	70.646±1.2	70.243±1.176
	NHiTS(+)	9.911 ± 0.365	55.199 ± 1.464	**77.438±0.561**	13.767 ± 0.926	68.281 ± 1.89	68.489±1.384
	TSPT(+)	**9.728±0.239**	**54.892±1.757**	77.234 ± 0.374	**12.581±0.321**	**64.374±1.108**	72.275 ± 1.148

Table 6
Model ablation study.

Model	$H = 24$			$H = 48$		
	MSE ($\times 10^3$)	MAE	PCC (%)	MSE ($\times 10^3$)	MAE	PCC (%)
w/o Instance Scale	13.956±4.871	75.744±12.437	78.845±2.792	15.762±1.1	82.96±4.91	73.111±3.977
w/o TSFP	7.947±0.135	53.358±0.709	84.205±0.29	10.75±0.406	63.907±1.346	79.389±0.655
w/o Encoder	9.63±0.017	59.364±0.139	82.127±0.116	12.789±0.061	69.669±0.06	77.18±0.193
TSPT(+) (all)	7.456±0.117	51.792±0.209	84.505±0.102	10.252±0.137	61.33±0.367	80.386±0.352

due to rigid structure, and lack of flexibility in handling different data characteristics across building categories contribute to its poor performance. Compared with other models, its deviations from actual values and lower accuracy are significant. In conclusion, these limitations highlight the importance of effectively capturing cross-correlations and adapting to data characteristics for accurate predictions, emphasizing the need for improvement in such models.

- In the context of the commercial building category, linear models such as AR and DLinear demonstrate a higher level of prediction accuracy when compared to non-linear models like LSTNet and Transformer. On the other hand, within the office building category, the predictive performance of linear models falls short when contrasted with that of non-linear models. This disparity in performance suggests that, in the prediction of smart meter data within industrial park settings, the meter data from specific building type areas display linear correlations, whereas the meter data from other building categories exhibit substantial non-linear relationships. It becomes evident that for a predictive model to attain a high level of prediction accuracy, it is essential to simultaneously handle both the linear and non-linear relationships inherent within time series data.

In summary, the TSPT model demonstrated remarkable robustness against variations in cross-correlations, achieving the best overall predictive performance in both commercial and office building categories. Moreover, the performance disparity between linear and non-linear models in different building categories revealed the presence of both linear and non-linear relationships within the smart meter data. Overall, these findings underscore the significance of model robustness and the ability to handle diverse data relationships for accurate prediction in the context of smart meter time series analysis.

5.3. Model ablation study

To evaluate the contribution of each component within the proposed TSPT model, we performed an ablation study by systematically removing key modules and assessing the resulting impact on prediction performance. The following variants were tested against the full TSPT model:

(1) **w/o Instance Scale**: The Instance Scale Normalization is removed, and the raw input features are directly passed to the model.
(2) **w/o Temporal Patching**: The Temporal Patching operation is removed, and unpatched data is passed to the Transformer encoder module.
(3) **w/o Encoder**: The Transformer encoder part is removed.
(4) **TSPT (all)**: The complete TSPT model serves as a control, including all proposed modules.

Table 6 summarizes the results of this ablation study. The most significant performance degradation occurs when the Instance Scale is removed. Specifically, the “w/o Instance Scale" variant demonstrates a substantial increase in MSE to 13.956 (an 87.18 % increase relative to the full TSPT model), an increase in MAE to 75.744 (a 46.25 % increase), and a decrease in PCC to 78.845 (a 6.70 % decrease) under the forecasting task with the prediction window length of 24. Similar conclusions can be drawn for the scenario where the prediction window length is set to 48. Compared to the complete TSPT model, the “w/o Instance Scale" variant exhibits a 53.75 % increase in MSE, a 35.27 % increase in MAE, and a 9.05 % decrease in PCC. These results strongly indicate that the Instance Scale is the most critical component for achieving high prediction accuracy. This severe degradation underscores the crucial role of the Instance Scale in constraining the input space of the data and optimizing the model’s learning efficiency.

The removal of the Temporal Patching (“w/o Temporal Patching") results in the performance decrease slightly, with a prediction length of 24, the MSE increasing to 7.947×10^3 (a 6.59 % increase), MAE increasing to 53.358 (a 3.02 % increase), and PCC decreasing to 84.205 % (a 0.36 % decrease), with a prediction length of 48, the MSE increasing to 10.75×10^3 (a 4.86 % increase), MAE increasing to 63.907 (a 4.20 % increase), and PCC decreasing to 79.389 % (a 1.24 % decrease). These results indicate that Temporal Patching plays a significant role in effectively capturing local periodic dependencies across different temporal scales, although it is not as critical as the Instance Scale. By effectively decomposing and reconstructing time-series information, the Temporal Patching operation enhances the performance of the model. The fact that the performance impact is relatively small yet still statistically significant indicates that directly extracting periodic dependencies from raw long time-series data is indeed beneficial but insufficient to achieve the same level of performance as the complete TSPT.

Similarly, the ablation of the Encoder, denoted “w/o Encoder", results in a notable performance drop, with MSE reaching 9.63×10^3 (a 29.16 % increase relative to the full TSPT model), MAE increasing to 59.364 (a 14.62 % increase), PCC decreasing to 82.127 % (a 2.81 % decrease) in $H = 24$, and MSE increasing to 12.789×10^3 (a 24.75 % increase), MAE increasing to 69.669 (a 13.60 % increase), PCC decreasing to 77.18 % (a 3.99 % decrease) in $H = 48$. This highlights that the multilayer spatial-temporal attention encoder’s ability to capture multi-scale temporal patterns is essential to the model’s performance. The model, therefore, benefits significantly from modeling information across multiple temporal scales rather than relying on a single temporal scale.

Comparing the performance of all ablated variants to the full TSPT model (which achieves an MSE of 7.456×10^3, MAE of 51.792, and PCC of 84.505 % in $H = 24$, an MSE of 10.252×10^3, MAE of 61.33, and PCC of 80.386 % in $H = 48$), we observe that the complete model consistently outperforms all of the ablated variants across all three evaluation metrics. These results demonstrate that each module contributes to the overall performance of the TSPT model and validates the rationale behind the proposed design of TSPT. Each module plays a unique and statistically significant role in handling the complexities inherent in multi-step industrial load forecasting.

In conclusion, this ablation study clearly demonstrates that the TSPT’s strong performance depends on the synergistic interaction of all of its components. By strategically incorporating the Instance Scale, Temporal Latticing, and Multilayer spatiotemporal attention Encoder, the TSPT model addresses the nuances of industrial power load forecasting.

5.4. Comparative analysis of prediction across building types

Fig. 2 provides a visual comparison of the power load forecasting performance for four different building types within the industrial park: Commercial, Office, Public, and Residential. We compare the TSPT model’s predictions against three baseline methods: LSTM (a recurrent neural network known for capturing temporal dependencies), DLinear

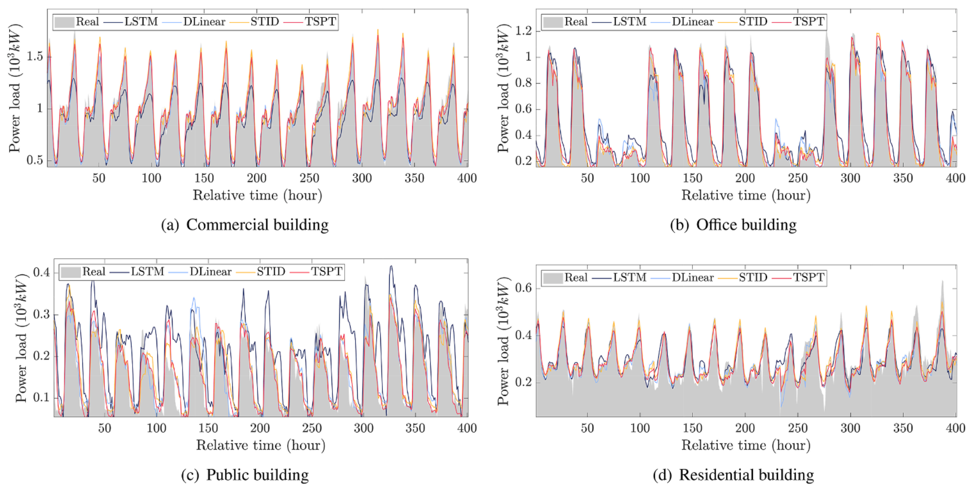


Fig. 2. Performance comparison of the TSPT model against LSTM, DLinear, and STID models.

(representing a traditional linear regression approach), and a standard Transformer model (known for its self-attention mechanism to handle long-range dependencies). These baselines were selected to represent a range of common approaches to time-series forecasting and to offer a robust benchmark for the proposed model. Each subfigure shows the real (ground truth) power load along with predictions from these models across a timespan of 400 h. The visualization results are selected segments from the validation set at the optimal epochs where each model achieves its best performance under the conditions of $H = 24$, with external factors considered, and the random seed set to 40.

Across all building types, the TSPT model's predictions closely follow the real power load, demonstrating both robustness and adaptability to diverse consumption patterns. In the Commercial building (Fig. 2(a)) and Office building (Fig. 2(b)) plots, we observe that the TSPT model accurately captures the daily cyclical patterns with minimal deviation, demonstrating its effectiveness in handling regular oscillations. Specifically, the TSPT model predictions closely match the peaks and troughs of the actual load, something that the baseline methods often fail to do. The LSTM model tends to over-predict during peaks, while DLinear is insensitive to the alternation of short-term and long-term variations in electricity meter readings. Although STID can closely align with the actual value curve, it fails to efficiently capture the downward trend in power load. This suggests that the TSPT's design, particularly the GFF, is effective at modeling periodic temporal dependencies.

For the Public building (Fig. 2(c)), where the load pattern is more irregular and characterized by sudden spikes and drops, the TSPT model demonstrates a strong capability to adapt to these less predictable fluctuations. While the other models tend to smooth out these rapid changes, the TSPT model adeptly predicts both sharp increases and decreases in power consumption, demonstrating its ability to respond to transient events. This highlights the effectiveness of the attention mechanisms in the CEE in handling irregular temporal dynamics.

In the Residential building (Fig. 2(d)) type, which typically exhibits more consistent daily patterns, the TSPT model's predictions closely mirror the real load values. While all methods tend to perform better on this load type, the TSPT model is more precise in mirroring the daily pattern, indicating that it captures both the daily patterns and the magnitude of the load changes. Other models have similar daily behaviors but also contain more variability and errors.

The baseline methods generally follow the overall trends of the power load but exhibit more significant deviations, particularly during periods of rapid change or at peak times. This indicates that these models, while competent at capturing long-term trends, often lack the precision of the TSPT model. These results are consistent with our quantitative results. The TSPT model's superior performance, particularly in accurately capturing both periodic oscillations and sudden changes in power load, demonstrates its effectiveness across different building types and confirms its potential for more reliable and precise forecasting of industrial power loads, showcasing the potential of the proposed method.

6. Discussion

The experimental results presented in Section 5 demonstrate the efficacy of the proposed TSPT model for multi-target industrial load forecasting, particularly in comparison to a comprehensive set of state-of-the-art baseline methods. TSPT consistently achieved superior or highly competitive performance across various evaluation metrics (MSE, MAE, PCC) and forecast horizons (24-h, 48-h) on a real-world industrial park dataset.

The superior performance of TSPT can be attributed to its key architectural innovations and design choices. The Temporal Patching mechanism effectively reduced the computational burden associated with long input sequences, enabling the model to capture long-range temporal dependencies relevant to industrial load patterns while maintaining computational efficiency. The Gated Feature Fusion (GFF) module proved crucial for effectively integrating diverse exogenous variables, allowing the model to adaptively weigh and incorporate external factors like weather conditions, which inherently influence energy consumption within industrial parks. This adaptive fusion, facilitated by the learnable gating tensor, enabled TSPT to outperform baselines, especially those less adept at handling heterogeneous input data. Furthermore, leveraging the power of self-attention, the Multilayer Temporal Attention Encoder successfully captured complex, non-linear temporal dynamics inherent in industrial load profiles, outperforming traditional linear models and many deep learning baselines.

The ablation study (Section 5.3) further highlighted the contribution of individual components within the TSPT architecture. The significant performance degradation observed upon removing Instance Normalization underscores the importance of stabilizing input distributions and handling potential non-stationarity in industrial load data. Removing Temporal Patching and the Transformer Encoder also resulted in noticeable performance drops, validating their roles in efficient long-range dependency modeling and feature extraction. These ablation results collectively support the synergistic effect of the TSPT model's modular design.

While TSPT demonstrates promising results, it is important to acknowledge certain limitations and potential avenues for future research. The empirical validation is primarily based on a single, albeit real-world, dataset from an industrial park in Suzhou, China. Further rigorous evaluation across more diverse datasets, including those from different geographical locations, industrial sectors, and with varying data characteristics (e.g., noise levels, missing data patterns, cross-series correlation structures), is necessary to comprehensively assess the generalizability and robustness of TSPT. Furthermore, while the Gated Feature Fusion offers an effective integration mechanism, exploring alternative fusion strategies, such as cross-attention mechanisms or more complex adaptive fusion networks, could yield further performance improvements. The impact of the non-standard encoder layer design, specifically the attention score accumulation and omission of Layer Normalization, warrants further investigation. While the current implementation accurately reflects the described architecture, a comparative ablation against a strictly standard Transformer encoder block would provide valuable insights into the specific contributions, if any, of these deviations. Finally, future work could explore the interpretability of the TSPT model, particularly examining the learned attention weights and the gating tensor, to gain deeper insights into the model's decision-making process and the relative importance of different temporal patterns and exogenous factors in industrial load forecasting.

Despite these limitations, the TSPT model offers a practically relevant and effective solution for industrial load forecasting. Its ability to handle multi-target series, integrate exogenous variables, and achieve high forecasting accuracy holds significant implications for optimizing energy management, enhancing operational efficiency, and supporting informed decision-making in industrial park settings. The improved forecasting accuracy translates directly to potential cost savings through optimized energy procurement and reduced operational risks associated with inaccurate load predictions.

7. Conclusions and future work

This paper introduces the Temporal Structure-Preserving Transformer (TSPT), a novel deep-learning architecture specifically designed for the complex task of multi-target industrial load forecasting with integrated exogenous variables. TSPT leverages a modular pipeline comprising Instance Normalization, Feature Extraction Perceptrons, Temporal Patching, Gated Feature Fusion (GFF), and a Multilayer Temporal Attention Encoder. Key innovations of TSPT include the GFF mechanism for adaptive multimodal data integration and the efficient temporal processing enabled by patch-based Transformer encoding. Comprehensive experimental evaluations on a real-world industrial park dataset demonstrate the superior forecasting performance of TSPT compared to a wide range of state-of-the-art time series forecasting models, including both traditional statistical methods and advanced deep learning architectures. TSPT's ability to effectively capture intricate temporal dependencies and seamlessly integrate diverse exogenous factors contributes to its enhanced accuracy and robustness in complex industrial load forecasting scenarios. The ablation study further validates the contribution of individual components within the TSPT architecture, highlighting the synergistic effect of its modular design.

The proposed TSPT model offers a significant advancement in industrial load forecasting, providing a practical tool for improving energy management, optimizing operational efficiency, and supporting data-driven decision-making in industrial settings. Future research directions include extending the empirical validation to more diverse datasets, exploring alternative fusion and encoder mechanisms, and investigating the interpretability of the model to further enhance its practical utility and theoretical understanding.

CRediT authorship contribution statement

Senzhen Wu: Writing – review & editing, Writing – original draft, Software, Methodology, Data curation; **Zhijin Wang:** Writing – review & editing, Writing – original draft, Validation, Software, Project administration, Methodology, Conceptualization; **Xiufeng Liu:** Writing – review & editing, Writing – original draft, Supervision, Project administration, Methodology, Funding acquisition, Formal analysis, Conceptualization; **Yuan Zhao:** Writing – review & editing, Writing – original draft, Validation, Software, Methodology, Conceptualization; **Yue Hu:** Writing – review & editing, Writing – original draft, Methodology, Data curation; **Yaohui Huang:** Writing – review & editing, Writing – original draft, Software, Methodology.

Data availability

Data will be made available on request.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was partly supported by the RE-INTEGRATE project (no. 101118217) funded by the European Union Horizon 2020 research and innovation program. We thank the editors and anonymous reviewers for their valuable comments and suggestions, which greatly benefited this paper.

References

Ahmad, A., Javaid, N., Guizani, M., Alrajeh, N., & Khan, Z. A. (2016). An accurate and fast converging short-term load forecasting model for industrial applications in a smart grid. *IEEE Transactions on Industrial Informatics*, *13*(5), 2587–2596. https://doi.org/10.1109/TII.2016.2638322

Alexander, S., & Block, P. (2022). Integration of seasonal precipitation forecast information into local-level agricultural decision-making using an agent-based model to support community adaptation. *Climate Risk Management*, *36*, 100417. https://doi.org/10.1016/j.crm.2022.100417

Burg, L., Gürses-Tran, G., Madlener, R., & Monti, A. (2021). Comparative analysis of load forecasting models for varying time horizons and load aggregation levels. *Energies*, *14*(21), 7128. https://doi.org/10.3390/en14217128

Challu, C., Olivares, K. G., Oreshkin, B. N., Ramirez, F. G., Canseco, M. M., & Dubrawski, A. (2023). NHITS: Neural hierarchical interpolation for time series forecasting. In *Proceedings of the AAAI conference on artificial intelligence* (pp. 6989–6997). Vancouver, Canada: AAAI Press. https://doi.org/10.1609/AAAI.V37I6.25854

Chen, Y., Kloft, M., Yang, Y., Li, C., & Li, L. (2018). Mixed kernel based extreme learning machine for electric load forecasting. *Neurocomputing*, *312*, 90–106. https://doi.org/10.1016/J.NEUCOM.2018.05.068

Chen, Z., Zhang, F., Li, T., & Li, C. (2023). MAGNet: Muti-scale attention and evolutionary graph structure for long sequence time-series forecasting. In *Proceedings of international conference on artificial neural networks* (pp. 218–230). Xanthi, Greece: Springer. https://doi.org/10.1007/978-3-031-44223-0_18

Cho, K., van Merrienboer, B., Bahdanau, D., & Bengio, Y. (2014). On the properties of neural machine translation: Encoder-decoder approaches. In *Proceedings of the 8th workshop on syntax, semantics and structure in statistical translation* (pp. 103–111). Doha, Qatar: Association for Computational Linguistics. https://doi.org/10.3115/v1/W14-4012

Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555. https://doi.org/10.48550/arXiv.1412.3555

Cirstea, R.-G., Guo, C., Yang, B., Kieu, T., Dong, X., & Pan, S. (2022). Triformer: Triangular, variable-specific attentions for long sequence multivariate time series forecasting–full version. *CoRR*, abs/2204.13767. https://doi.org/10.48550/ARXIV.2204.13767

Du, D., Su, B., & Wei, Z. (2023). Preformer: predictive transformer with multi-scale segment-wise correlations for long-term time series forecasting. In *Proceedings of the acoustics, speech and signal processing (ICASSP)* (pp. 1–5). Rhodes Island, Greece: IEEE. https://doi.org/10.1109/ICASSP49357.2023.10096881

Filipiak, B. C., Bassill, N. P., Corbosiero, K. L., Lang, A. L., & Lazear, R. A. (2023). Probabilistic forecasting methods of winter mixed-precipitation events in new york state utilizing a random forest. *Artificial Intelligence for the Earth Systems*, *2*(3), e220080. https://doi.org/10.1175/aies-d-22-0080.1

Ge, Q., Guo, C., Jiang, H., Lu, Z., Yao, G., Zhang, J., & Hua, Q. (2020). Industrial power load forecasting method based on reinforcement learning and PSO-LSSVM. *IEEE Transactions on Cybernetics*, *52*(2), 1112–1124. https://doi.org/10.1109/TCYB.2020.2983871

Geng, X., Li, Y., Wang, L., Zhang, L., Yang, Q., Ye, J., & Liu, Y. (2019). Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting. In *Proceedings of the AAAI conference on artificial intelligence* (pp. 3656–3663). New York, New York, USA: AAAI Press. https://doi.org/10.1609/aaai.v33i01.33013656

Hossain, M. A., Gray, E., Lu, J., Islam, M. R., Alam, M. S., Chakrabortty, R., & Pota, H. R. (2023). Optimized forecasting model to improve the accuracy of very short-term wind power prediction. *IEEE Transactions on Industrial Informatics*, *19*(10), 10145–10159. https://doi.org/10.1109/TII.2022.3230726

Hu, Y., Liu, H., Wu, S., Zhao, Y., Wang, Z., & Liu, X. (2024). Temporal collaborative attention for wind power forecasting. *Applied Energy*, *357*, 122502. https://doi.org/10.1016/j.apenergy.2023.122502

Huang, A., Xiao, J., & Wang, S. (2011). A combined forecast method integrating contextual knowledge. *International Journal of Knowledge and Systems Science (IJKSS)*, *2*(4), 39–53. https://doi.org/10.4018/jkss.2011100104

Józefowicz, R., Zaremba, W., & Sutskever, I. (2015). An empirical exploration of recurrent network architectures. In *Proceedings of the 32nd international conference on machine learning* (pp. 2342–2350). Lille, France: PMLR. http://proceedings.mlr.press/v37/jozefowicz15.html.

Kerkkänen, A., & Huiskonen, J. (2014). The role of contextual information in demand forecasting. *International Journal of Information and Decision Sciences*, *6*(2), 109–126. https://doi.org/10.1504/IJIDS.2014.061765

Kim, T., Kim, J., Tae, Y., Park, C., Choi, J.-H., & Choo, J. (2021). Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International conference on learning representations*.

Ko, M.-S., Lee, K., & Hur, K. (2022). Feedforward error learning deep neural networks for multivariate deterministic power forecasting. *Transactions on Industrial Informatics*, *18*(9), 6214–6223. https://doi.org/10.1109/TII.2022.3160628

Kreye, M. E., Goh, Y. M., Newnes, L. B., & Goodwin, P. (2012). Approaches to displaying information to assist decisions under uncertainty. *Omega*, *40*(6), 682–692. https://doi.org/10.1016/j.omega.2011.05.010

Lai, G., Chang, W., Yang, Y., & Liu, H. (2018). Modeling long- and short-term temporal patterns with deep neural networks. In *Proceedings of the 41st international conference on research and development in information retrieval* (pp. 95–104). Ann Arbor, MI, USA: ACM. https://doi.org/10.1145/3209978.3210006

Li, C., Tao, Y., Ao, W., Yang, S., & Bai, Y. (2018). Improving forecasting accuracy of daily enterprise electricity consumption using a random forest based on ensemble empirical mode decomposition. *Energy*, *165*, 1220–1227. https://doi.org/10.1016/j.energy.2018.10.113

Liu, H., Dong, Z., Jiang, R., Deng, J., Deng, J., Chen, Q., & Song, X. (2023a). Spatio-temporal adaptive embedding makes vanilla transformer sota for traffic forecasting. In *Proceedings of the 32nd ACM international conference on information and knowledge management* (pp. 4125–4129). Birmingham, United Kingdom: ACM. https://doi.org/10.1145/3583780.3615160

Liu, X., Lyu, X., Zhang, X., Gao, J., & Chen, J. (2022a). Memory augmented graph learning networks for multivariate time series forecasting. In *Proceedings of the 31st ACM international conference on information & knowledge management* (pp. 4254–4258).

Liu, Y., Hu, T., Zhang, H., Wu, H., Wang, S., Ma, L., & Long, M. iTransformer: Inverted transformers are effective for time series forecasting. *International Conference on Learning Representations*, (2023b). https://arxiv.org/abs/2310.06625.

Liu, Y., Wu, H., Wang, J., & Long, M. Non-stationary transformers: Exploring the stationarity in time series forecasting. In *Proceedings of the advances in neural information processing systems* (pp. 9881–9893). Curran Associates, Inc. (*35*), (2022b). https://doi.org/10.48550/arXiv.2205.14415

Lu, R., Bai, R., Ding, Y., Wei, M., Jiang, J., Sun, M., Xiao, F., & Zhang, H.-T. (2021). A hybrid deep learning-based online energy management scheme for industrial microgrid. *Applied Energy*, *304*, 117857. https://doi.org/10.1016/j.apenergy.2021.117857

Ma, Y., Ye, C., Wu, Z., Wang, X., Cao, Y., & Chua, T.-S. (2023). Context-aware event forecasting via graph disentanglement. In *Proceedings of the 29th ACM SIGKDD conference on knowledge discovery and data mining* (pp. 1643–1652). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3580305.3599285

Meng, Z., Xie, Y., & Sun, J. (2022). Short-term load forecasting using neural attention model based on EMD. *Electrical Engineering*, (pp. 1–10). https://doi.org/10.1007/s00202-021-01420-4

Nie, Y., Nguyen, N. H., Sinthong, P., & Kalagnanam, J. (2023). A time series is worth 64 words: Long-term forecasting with transformers. In *Proceedings of the 11th international conference on learning representations ICLR* (pp. 1–5). OpenReview.net. http://openreview.net/pdf?id=Jbdc0vTOcol.

Porteiro, R., Nesmachnow, S., & Hernández-Callejo, L. (2019). Short term load forecasting of industrial electricity using machine learning. In *Smart cities: Second Ibero-American congress, ICSC-CITIES* (pp. 146–161). Moscow, Russia: Springer. https://doi.org/10.1007/978-3-030-38889-8_12

Seifert, M., Siemsen, E., Hadida, A. L., & Eisingerich, A. B. (2015). Effective judgmental forecasting in the context of fashion products. *Journal of Operations Management*, *36*, 33–45. https://doi.org/10.1016/j.jom.2015.02.001

Shao, Z., Zhang, Z., Wang, F., Wei, W., & Xu, Y. (2022). Spatial-temporal identity: A simple yet effective baseline for multivariate time series forecasting. In *Proceedings of the 31st ACM international conference on information & knowledge management* (pp. 4454–4458). Atlanta, GA, USA: ACM. https://doi.org/10.1145/3511808.3557702

Slowik, M., & Urban, W. (2022). Machine learning short-term energy consumption forecasting for microgrids in a manufacturing plant. *Energies*, *15*(9), 3382. https://doi.org/10.3390/en15093382

Som, T. (2023). Time load forecasting: A smarter expertise through modern methods. Springer.

Song, C., Lin, Y., Guo, S., & Wan, H. (2020). Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting. In *Proceedings of the AAAI conference on artificial intelligence* (pp. 914–921). Palo Alto, California, USA: AAAI Press. https://doi.org/10.1609/aaai.v34i01.5438

Tatinati, S., Wang, Y., & Khong, A. W. H. (2022). Hybrid method based on random convolution nodes for short-term wind speed forecasting. *Transactions on Industrial Informatics*, *18*(10), 7019–7029. https://doi.org/10.1109/TII.2020.3043451

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., & Polosukhin, I. (2017). Attention is all you need. In *Proceedings of the 31th international conference on neural information processing systems* (pp. 5998–6008). Long Beach, CA, USA: Curran Associates, Inc. https://doi.org/10.5555/3295222.3295349

Wang, Y., Wu, H., Dong, J., Qin, G., Zhang, H., Liu, Y., Qiu, Y., Wang, J., & Long, M. (2024). TimeXer: Empowering transformers for time series forecasting with exogenous variables. arXiv, https://arxiv.org/abs/2402.19072.

Wang, Z., & Cai, B. (2022). COVID-19 cases prediction in multiple areas via shapelet learning. *Applied Intelligence*, *52*(1), 595–606. https://doi.org/10.1007/s10489-021-02391-6

Wang, Z., Liu, X., Huang, Y., Zhang, P., & Fu, Y. (2023a). A multivariate time series graph neural network for district heat load forecasting. *Energy*, *278*, 127911. https://doi.org/10.1016/j.energy.2023.127911

Wang, Z., Zhang, P., Huang, Y., Chao, G., Xie, X., & Fu, Y. (2023b). Oriented transformer for infectious disease case prediction. *Applied Intelligence*, *53*, 30097-30112. https://doi.org/10.1007/s10489-023-05101-6

Wen, Q., Zhou, T., Zhang, C., Chen, W., Ma, Z., Yan, J., & Sun, L. (2023). Transformers in time series: A survey. In *Proceedings of the 32nd international joint conference on artificial intelligence, IJCAI* (pp. 6778–6786). Macao, SAR, China: International Joint Conferences on Artificial Intelligence Organization. https://doi.org/10.24963/ijcai.2023/759

Wu, H., Xu, J., Wang, J., & Long, M. (2021a). Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In *Proceedings of the 34th annual conference on neural information processing systems* (pp. 22419–22430). Virtual: NeurIPS. https://doi.org/10.48550/arXiv.2106.13008

Wu, H., Xu, J., Wang, J., & Long, M. (2021b). Autoformer: decomposition transformers with auto-correlation for long-term series forecasting. In *Proceedings of the 35th international conference on neural information processing system* (pp. 22419–22430). Red Hook, NY, USA: Curran Associates Inc. https://doi.org/10.5555/3540261.3541978

Wu, Y., Yang, Y., Nishiura, H., & Saitoh, M. (2018). Deep learning for epidemiological predictions. In *Proceedings of the 41st international conference on research and development in information retrieval* (pp. 1085–1088). Ann Arbor, MI, USA: ACM. https://doi.org/10.1145/3209978.3210077

Wu, Z., Pan, S., Long, G., Jiang, J., & Zhang, C. (2019). Graph wavenet for deep spatial-temporal graph modeling. In *Proceedings of the 28th international joint conference on artificial intelligence* (pp. 1907–1913). Macao, China: ijcai.org. https://doi.org/10.24963/IJCAI.2019/264

Xu, J., Sun, X., Zhang, Z., Zhao, G., & Lin, J. (2019). Understanding and improving layer normalization. *Advances in Neural Information Processing Systems*, *32*.

Yanchenko, A. K., Tierney, G., Lawson, J., Hellmayr, C., Cron, A., & West, M. (2021). Multivariate dynamic modeling for bayesian forecasting of business revenue. arXiv, https://doi.org/10.1002/asmb.2704

Zeng, A., Chen, M., Zhang, L., & Xu, Q. (2023). Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence* (pp. 11121–11128). AAAI. https://arxiv.org/abs/2205.13504.

Zhang, Q., Chang, J., Meng, G., Xiang, S., & Pan, C. (2020). Spatio-temporal graph structure learning for traffic forecasting. In *Proceedings of the AAAI conference on artificial intelligence* (pp. 1177–1185). https://doi.org/10.1609/aaai.v34i01.5470

Zhang, Y.-F., & Chiang, H.-D. (2019). Enhanced ELITE-load: A novel CMPSOATT methodology constructing short-term load forecasting model for industrial applications. *Transactions on Industrial Informatics*, *16*(4), 2325–2334. https://doi.org/10.1109/TII.2019.2930064

Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., & Zhang, W. (2021). Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence* (pp. 11106–11115). San Diego, CA, USA: AAAI. https://doi.org/10.1609/aaai.v35i12.17325

Zhou, K., Hu, D., Hu, R., & Zhou, J. (2023). High-resolution electric power load data of an industrial park with multiple types of buildings in china. *Scientific Data*, *10*(1), 870. https://doi.org/10.1038/s41597-023-02786-9

Zhou, T., Ma, Z., Wen, Q., Wang, X., Sun, L., & Jin, R. (2022). FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *Proceedings of the 39th international conference on machine learning* (pp. 27268–27286). Baltimore, Maryland, USA: PMLR. https://doi.org/10.48550/arXiv.2201.12740