Communications in Transportation Research 5 (2025) 100218

Contents lists available at ScienceDirect

Communications in Transportation Research

journal homepage: www.journals.elsevier.com/communications-in-transportation-research

Full Length Article

HUTFormer: Hierarchical U-Net transformer for long-term traffic forecasting

Zezhi Shao[a], Fei Wang[a,b,*], Tao Sun[a], Chengqing Yu[a,b], Yuchen Fang[c], Guangyin Jin[d], Zhulin An[a], Yang Liu[e], Xiaobo Qu[e], Yongjun Xu[a,b]

[a] Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 100190, China
[b] School of Computer Science and Technology, University of Chinese Academy of Sciences, Beijing, 100049, China
[c] School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, 610054, China
[d] Department of Planning, Design, and Technology of Architecture, Sapienza University of Rome, Rome, 00196, Italy
[e] School of Vehicle and Mobility, Tsinghua University, Beijing, 100084, China

ARTICLE INFO

Keywords:
Traffic condition forecasting
Long-term time series forecasting
Multivariate time series forecasting

ABSTRACT

Traffic forecasting, which aims to predict traffic conditions based on historical observations, has been an enduring research topic and is widely recognized as an essential component of intelligent transportation. Recent proposals on Spatial-Temporal Graph Neural Networks (STGNNs) have made significant progress by combining sequential models with graph convolution networks. However, due to high complexity issues, STGNNs only focus on short-term traffic forecasting (e.g., 1-h ahead), while ignoring more practical long-term forecasting. In this paper, we make the first attempt to explore long-term traffic forecasting (e.g., 1-day ahead). To this end, we first reveal its unique challenges in exploiting multi-scale representations. Then, we propose a novel Hierarchical U-net TransFormer (HUTFormer) to address the issues of long-term traffic forecasting. HUTFormer consists of a hierarchical encoder and decoder to jointly generate and utilize multi-scale representations of traffic data. Specifically, for the encoder, we propose window self-attention and segment merging to extract multi-scale representations from long-term traffic data. For the decoder, we design a cross-scale attention mechanism to effectively incorporate multi-scale representations. In addition, HUTFormer employs an efficient input embedding strategy to address the complexity issues. Extensive experiments on four traffic datasets show that the proposed HUTFormer significantly outperforms state-of-the-art traffic forecasting and long time series forecasting baselines.

1. Introduction

Traffic forecasting aims at predicting future traffic conditions (e.g., traffic speed or flow) based on historical traffic conditions observed by sensors. With the development of Intelligent Transportation Systems (ITS), traffic forecasting fuels a wide range of services related to traffic scheduling, public safety, etc. (Chu et al., 2019, 2024; Guo et al., 2024; Jin et al., 2023; Li et al., 2025; Liu et al., 2023; Lv et al., 2023; Xu et al., 2023). For example, predicting long-term traffic changes (e.g., 1-day) is valuable for people to plan their route in advance to avoid possible traffic congestion.

In general, traffic data[1] is presented in the form of multiple time series, where each time series records traffic conditions observed by sensors deployed on a road network. A critical property of traffic data is that there exist strong correlations between time series owing to the connection of road networks. To make accurate traffic forecasting, state-of-the-art proposals (Jin et al., 2022; Li et al., 2018; Wu et al., 2019) usually adopt Spatial-Temporal Graph Neural Networks (STGNNs), which model the correlations between time series based on Graph Convolution Networks (GCNs) (Defferrard et al., 2016; Kipf and Welling, 2017; Li et al., 2018). However, graph convolution brings significant improvements in performance and 50 complexity at the same time. Computational complexity usually increases linearly or quadratically with the length and number of time series (Shao et al., 2022c). Therefore, it is difficult for STGNNs to scale to long-term historical traffic data, let alone predict long-term future traffic conditions. In fact, most existing works focus on short-term traffic prediction, e.g., predicting

* Corresponding author. Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 100190, China.
E-mail address: wangfei@ict.ac.cn (F. Wang).

[1] For the sake of brevity, in this paper, we use 'traffic condition data' and 'traffic data' interchangeably.

https://doi.org/10.1016/j.commtr.2025.100218
Received 10 February 2025; Received in revised form 29 April 2025; Accepted 15 May 2025
Available online xxxx

Nomenclature

T	The length of history traffic data
T_{f}	The length of future traffic data
N	The number of time series
C	The number of feature channels in a traffic sensor
$\mathcal{X}$	History data of shape $\mathbb{R}^{T\times N\times C}$
$\mathcal{Y}$	Future data of shape $\mathbb{R}^{T_f\times N\times C}$
X^i	History data of sensor i
Y^i	Future data of sensor i
$\widehat{\mathbf{Y}}^i$	Prediction data of sensor i
L	The segment size
P	The number of segments, $T = P \times L$
d	The hidden dimension
W	Parameter matrix of the fully connected layer
b	Parameter of the bias of the fully connected layer
E	Spatial embeddings of shape $\mathbb{R}^{N\times d_1}$
T	Temporal embeddings
N_D	The number of time slots of a day
N_W	The number of days in a week
S	Embeddings of each segment after segment embedding
U	Embeddings of each segment after spatial-temporal positional encoding
H	Hidden states

future 12 time steps (1 h in commonly used datasets). Such an inability to make long-term traffic forecasting limits the practicality of these models.

In this study, we focus on long-term traffic forecasting, e.g., predicting a future day. Except for the correlations between time series, the long-term traffic forecasting task has its own uniqueness. In the following, we discuss them in detail to motivate model design. Examples of traffic data is shown in Fig. 1[2]. On the one hand, when observing from a global perspective, traffic data usually exhibit regular changes, e.g., daily periodicity. On the other hand, local details are crucial for traffic forecasting. For example, we must capture the rapidly decreasing traffic changes when daily traffic congestion occurs. To capture different patterns, exploiting multi-scale representations of traffic data is the key challenge of accurate long-term traffic forecasting. Specifically, smaller-scale and larger-scale representations are extracted based on smaller and larger receptive fields, respectively. The former is usually semantically weak but fine-grained, which facilitates the prediction of local details, e.g., rapid changes during traffic congestion. In contrast, the latter is coarse-grained but semantically strong, which is helpful in predicting global changes, e.g., daily periodicity. An illustration is shown in Fig. 1b. The prediction based on large-scale features captures daily periodicity but misses local details, which can be fixed by further incorporating small-scale features.

However, it is a challenging task to exploit multi-scale representations of traffic data. We discuss it from two aspects: Generating and utilizing multi-scale representations. On the one hand, most existing models cannot generate multi-scale representations of traffic data. State-of-the-art models for long time series forecasting (Zhou et al., 2021) mainly adopt Transformers to capture the long-term dependencies based on self-attention mechanisms (Vaswani et al., 2017). However, standard self-attention naturally has a global receptive field and thus can only generate representations on a fixed scale. On the other hand, utilizing multi-scale representations for traffic forecasting is also a challenging task, as it usually requires a specific decoder. For example, in computer vision tasks like object detection and semantic segmentation, researchers designed decoders such as FPN (Lin et al., 2017) and U-Net (Ronneberger et al., 2015) to utilize the multi-scale representations extracted by the pre-trained encoder (He et al., 2016). These architectures usually require pixel alignment of input and output images. However, the historical and future sequences in traffic forecasting problems are not the same sequences, i.e., not aligned, making existing approaches (Lin et al., 2017; Ronneberger et al., 2015) inapplicable.

Based on the above discussion, we summarize three challenges that the desired long-term traffic forecasting model should address. First, it must efficiently model the correlations between multiple long-term time series. Second, it should generate multi-scale representations of traffic data by an encoder. Third, it should include a decoder for traffic forecasting tasks to effectively utilize the multi-scale representations generated by the encoder.

To address the above challenges, we propose a novel Hierarchica U-Net TransFormer (named HUTFormer). As shown in Fig. 2, HUTFormer is a two-stage model consisting of a hierarchical encoder and a hierarchical decoder, forming an inverted U-shaped structure. To address the efficiency problem, HUTFormer designs an efficient input embedding strategy, which employs segment embedding and spatial-temporal positional encoding to significantly reduce the complexity of modeling multiple long-term time series in both temporal and spatial dimensions. To generate multi-scale representations, the HUTFormer encoder proposes a window Transformer layer to limit the receptive field, and then designs segment merging as a pooling layer to extract larger-scale features. Thus, lower layers of the encoder focus on smaller-scale features, while higher layers generate larger-scale features. Then, HUTFormer makes an intermediate prediction based on the top-level representations. To utilize multi-scale representations, the HUTFormer decoder proposes a cross-scale attention mechanism to address the misalignment issue, which retrieves information for each segment of the intermediate prediction from multi-scale representations, thus enabling the fine-tuning of the intermediate prediction. By exploiting the multi-scale representations of traffic data, HUTFormer is capable of making accurate long-term traffic forecasting. The main contributions of this paper are summarized as follows:

- To our best knowledge, this is the first attempt to study long-term traffic forecasting. We reveal its unique challenges in exploiting multi-scale representations of traffic data, and propose a novel Hierarchical U-Net TransFormer (HUTFormer) to address them.
- We propose window self-attention and cross-scale attention mechanisms to generate and utilize multi-scale representations effectively. In addition, to address complexity issues, we design an input embedding strategy that includes segment embedding and spatial-temporal positional encoding.
- Extensive experiments on four traffic datasets show that the proposed HUTFormer significantly outperforms state-of-the-art traffic forecasting and long-sequence time series forecasting baselines, and effectively exploits the multi-scale representations of traffic data.

2. Related work

2.1. Traffic forecasting

Previous traffic forecasting studies usually fall into two categories, i.e., knowledge-driven (e.g., queuing theory) and early data-driven models (Belhadi et al., 2020; Cho et al., 2014; Kumar and Vanajakshi, 2015; Liu et al., 2019, 2021a; Sutskever et al., 2014; Wang et al., 2020b; Yu and Koltun, 2016). However, these methods usually ignore the

[2] Fig. 1b is the future part of Fig. 1a

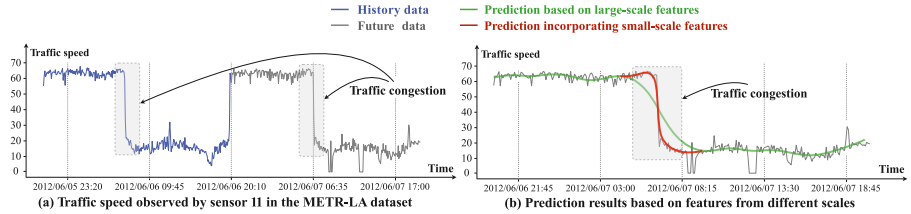


Fig. 1. Examples of long-term traffic forecasting.

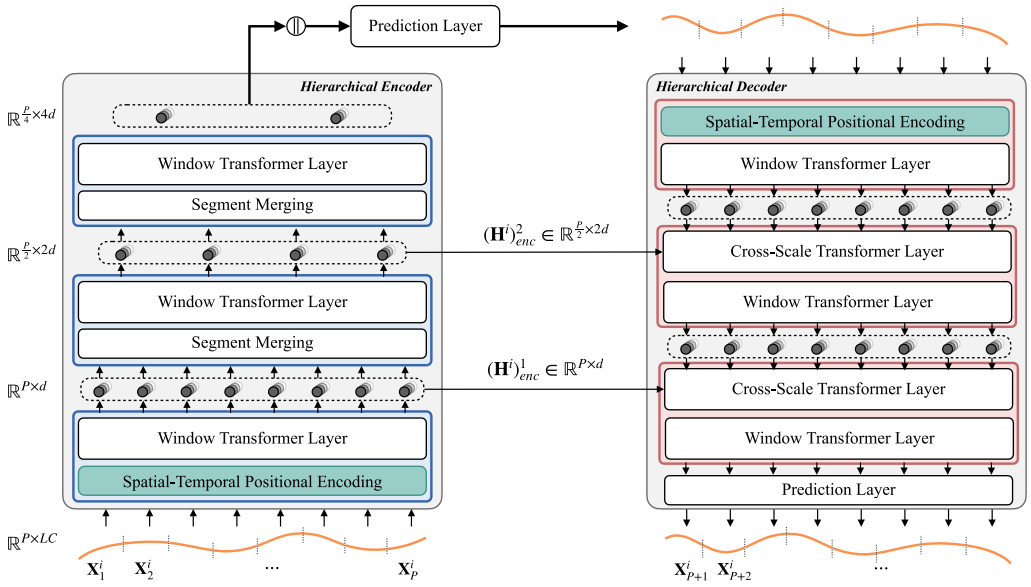


Fig. 2. Overview of the proposed HUTFormer. Left: The hierarchical encoder. It generates multi-scale features for traffic data based on window Transformer layer and segment merging, and makes an intermediate prediction. Right: The hierarchical decoder. It fine-tunes the intermediate prediction by incorporating multi-scale features based on cross-scale Transformer layer. In addition, segment embedding and spatial-temporal positional encoding are proposed to address complexity issues.

correlation between time series and the high non-linearity of time series (Shao et al., 2022d), which severely limits the effectiveness of these methods. With the development of deep learning (Huang et al., 2025; Shao et al., 2025b; Wang et al., 2023a), Spatial-temporal Graph Neural Networks (STGNNs) were proposed recently (Li et al., 2018) to model the complex spatial-temporal correlations in traffic data. Specifically, STGNNs combine Graph Neural Networks (GNNs) (Defferrard et al., 2016; Kipf and Welling, 2017; Shao et al., 2022a) and sequential models (e.g., CNN (Yu and Koltun, 2016) or RNN (Cho et al., 2014)), to model the complex spatial-temporal correlation in traffic data. For example, DCRNN (Li et al., 2018), ST-MetaNet (Pan et al., 2019), AGCRN (Bai et al., 2020), etc. (Li et al., 2023; Shang et al., 2021; Yu et al., 2018), are RNN-based methods, which combine GNN with recurrent neural networks. Graph WaveNet (Wu et al., 2019), MTGNN (Wu et al., 2020), STGCN (Yu et al., 2018), and StemGNN (Cao et al., 2020) are CNN-based methods (Han et al., 2021), which usually combines GNN with the Temporal Convolution Network (TCN (Yu and Koltun, 2016)). Moreover, techniques such as attention mechanisms and spectral theories are also widely employed in STGNNs (Guo et al., 2019, 2022; Li et al., 2024; Park et al., 2020; Shao et al., 2025c; Wang et al., 2020a; Yu et al., 2024; Zheng et al., 2020).

Although STGNNs have achieved significant progress, their complexity is high. Specifically, their complexity usually increases linearly or quadratically with the length and the number of time series (Shao et al., 2022c), since they need to deal with both temporal and spatial dependency at every step. Thus, most of them focus on short-term traffic forecasting based on short-term history data, e.g., predicting future 1-h traffic conditions based on 1-h historical data (Bogaerts et al., 2020; Li et al., 2018; Shao et al., 2022c, 2022d; Wu et al., 2019, 2020). A recent work STEP (Shao et al., 2022c) attempts to address this issue based on the time series pre-training model. It significantly enhances STGNN's ability to exploit long-term historical data. However, STEP requires a downstream STGNN as the backbone, which still focuses on short-term traffic forecasting.

Although STGNN-based traffic forecasting has made significant progress, these models only focus on short-term traffic forecasting, and cannot handle long-term traffic forecasting. On the one hand, due to the high complexity, most of them can not handle long-term history data, let alone predict long-term future traffic conditions. On the other hand, apart from efficiency issues, long-term traffic forecasting also has its unique challenges, which require exploiting multi-scale representations of traffic data to capture the complex long-term traffic dynamics.

2.2. *Long-sequence time series forecasting*

Recently, long-sequence time series forecasting has received much attention (Liu et al., 2022; Shao et al., 2025a; Wu et al., 2021; Zeng et al., 2023; Zhou et al., 2021, 2022). They aim to make long-term future predictions by modeling long-term historical sequences. For example, Informer (Zhou et al., 2021) proposes a ProbSparse self-attention mechanism to replace the standard self-attention, which enhances the predictive ability of the standard Transformer in the long-sequence forecasting problem. Autoformer (Wu et al., 2021) designs an efficient Auto-Correlation mechanism to conduct dependencies discovery and information aggregation at the series level. DLinear (Zeng et al., 2023) rethinks Transformer-based techniques and proposes a simple linear

model based on decomposition to achieve better accuracy. Recently, many advanced Transformer-based models have been proposed, such as PatchTST (Nie et al., 2023), Crossformer (Zhang and Yan, 2023), Scaleformer (Shabani et al., 2023), and DSformer (Yu et al., 2023).

Although these models have made considerable progress in long-term time series forecasting, they are not designed for traffic data, which significantly affects their effectiveness in traffic forecasting problems. We discuss it from two aspects. First of all, there are strong correlations between multiple time series in traffic data, which is an important bottleneck in traffic forecasting (Li et al., 2018). However, long-sequence time series forecasting models usually do not pay attention to such spatial correlations (Cirstea et al., 2022; Liu et al., 2022; Shabani et al., 2023; Wu et al., 2021; Zhou et al., 2021, 2022), or are not efficient enough (Zhang and Yan, 2023). Second, as discussed in Section 1, long-term traffic forecasting requires exploiting multi-scale representations of traffic data to capture the complex long-term traffic dynamics. However, long-sequence forecasting models usually rely on the self-attention mechanism and its variants, which can not explicitly generate multi-scale features (Nie et al., 2023; Wu et al., 2021; Zhou et al., 2021).

3. Preliminaries

In this section, we define the notions of traffic data and traffic forecasting task.

Definition 1. Traffic data $\mathcal{X} \in \mathbb{R}^{T\times N\times C}$ denotes the observation from all sensors on the traffic network, where T is the number of time steps, N is the number of traffic sensors, and C is the number of features collected by sensors. We additionally denote the data from the sensor i as $\mathbf{X}^i \in \mathbb{R}^{T\times C}$.

Definition 2. Traffic forecasting aims to predict the traffic values $\mathcal{Y} \in \mathbb{R}^{T_f\times N\times C}$ of the T_f nearest future time steps based on the given historical traffic data $\mathcal{X} \in \mathbb{R}^{T_h\times N\times C}$ from the past T_h time steps. In this study, we focus on long-term traffic forecasting, e.g., forecasting for a day in the future.

4. Model architecture

4.1. Overview

As illustrated in Fig. 2, HUTFormer is based on a hierarchical U-Net structure to generate and utilize multi-scale representations of traffic data. In this subsection, we intuitively discuss each component of HUTFormer and its two-stage training strategy.

First, we discuss the hierarchical encoder. The window Transformer layer is the basis for generating multi-scale representations, which calculates self-attention within a small window to limit the receptive field. Then, segment merging acts as a pooling layer, reducing the sequence length to produce larger-scale representations. By combining them, lower layers can focus on smaller-scale features while higher layers focus on larger-scale features, thus successfully generating multi-scale features. Then, an intermediate prediction is made based on the top-layer representations. However, considering that the top-layer features are semantically strong but coarse-grained, the intermediate prediction may fail to capture rapidly changing local details, e.g., the red line in Fig. 1b.

To address the above problem, the hierarchical decoder aims to fine-tune the intermediate prediction by incorporating multi-scale representations. U-Net (Cao et al., 2022; Ronneberger et al., 2015) is a popular structure for utilizing multi-scale representations, especially in computer vision tasks (e.g., semantic segmentation). In these tasks, the pixels of the input and target images are aligned, i.e., models operate on the same image. However, for traffic forecasting tasks, the input and output sequences are not the same sequence, i.e., not aligned. Thus, the representations generated by the encoder and the decoder cannot be directly superimposed as regular U-Net structures (Cao et al., 2022; Ronneberger et al., 2015) do for computer vision tasks. To this end, we design a cross-scale Transformer layer, which uses the representations from the decoder as queries and the multi-scale features from the encoder as keys and values to retrieve information. Such a top–down pathway and lateral connects help to combine the multi-scale representations, thus enhancing the prediction accuracy.

In addition, HUTFormer addresses complexity issues based on an efficient input embedding strategy, which consists of segment embedding and spatial-temporal positional encoding. On the one hand, segment embedding reduces complexity from the temporal dimension by using time series segments as basic input tokens. This simple operation has significant benefits in both reducing the length of the input sequence and providing more robust semantics (Shao et al., 2022c). On the other hand, spatial-temporal positional encoding is designed to replace the standard positional encoding (Dosovitskiy et al., 2021; Vaswani et al., 2017) in Transformer. More importantly, it efficiently models the correlations among time series from the perspective of solving the indistinguishability of samples (Deng et al., 2021; Shao et al., 2022b; Wang et al., 2023b), avoiding the high complexity of conducting graph convolution (Li et al., 2018; Wu et al., 2019) in the spatial dimension.

Finally, we propose the training strategy: a two-stage strategy. The first stage aims to train the hierarchical encoder based on the Mean Absolute Error (MAE) between the intermediate prediction and ground truth. In the second stage, we only train the decoder, while the parameters of the encoder are fixed to act as the multi-scale feature extractor. The reason for adopting the two-stage strategy is that traffic forecasting tasks are different from tasks that employ an end-to-end strategy (e.g., semantic segmentation (Cao et al., 2022; Ronneberger et al., 2015) and object detection (Lin et al., 2017) in computer vision). Specifically, in computer vision tasks, pre-trained vision models (e.g., pre-trained ResNet (He et al., 2016)) usually serve as the backbone to extract multi-scale features (Lin et al., 2017). However, there is no pre-trained model for time series that can extract multi-scale features. Therefore, optimizing the feature extractor (i.e., the encoder) and downstream networks (i.e., the decoder) in an end-to-end fashion may be insufficient. The experimental results in Section 5.5 also verify this hypothesis. Next, we introduce each component in detail.

4.2. Input embedding

Segment embedding. Most previous works usually use single data points as the basic input units. However, isolated points of time series usually give less semantics (Shao et al., 2022c) and are more easily affected by noise. Therefore, HUTFormer adopts segment embedding, i.e., dividing the input sequence into several segments to get the input tokens. Specifically, given the time series $\mathbf{X}^i \in \mathbb{R}^{T\times C}$ from sensor i, HUTFormer divides it into P non-overlapping segments of length L, i.e., $T = P^*L$. We denote the jth segment as $\mathbf{X}^i_j \in \mathbb{R}^{LC}$. Then, we conduct the input embedding layer based on these segments:

$$\mathbf{S}^i_j = \mathbf{W}\cdot\mathbf{X}^i_j + \mathbf{b} \tag{1}$$

where $\mathbf{S}^i_j \in \mathbb{R}^d$ is the embedding of segments j of the time series from sensor i, and d is the hidden dimension. $\mathbf{W} \in \mathbb{R}^{d\times(LC)}$ and $\mathbf{b} \in \mathbb{R}^d$ are learnable parameters shared by all segments.

In summary, applying segment embedding brings two benefits. First, it provides more robust semantics. Second, it significantly reduces the sequence length to reduce computational complexity.

Spatial-temporal positional encoding. In this study, we propose to replace the standard positional encoding in Transformer-based networks (Dosovitskiy et al., 2021; Vaswani et al., 2017) with Spatial-temporal Positional Encoding (ST-PE). Specifically, given the segment

embedding $\mathbf{S}_j^i \in \mathbb{R}^d$ of segments j from time series i, ST-PE conduct positional encoding on the spatial and temporal dimensions simultaneously:

$$\mathbf{U}_j^i = \text{Linear}(\mathbf{S}_j^i \Big\| \mathbf{E}^i \Big\| \mathbf{T}_j^{TiD} \Big\| \mathbf{T}_j^{DiW}) \tag{2}$$

On the spatial dimension, we define the spatial positional embeddings $\mathbf{E} \in \mathbb{R}^{N \times d_1}$, where N is the number of time series (i.e., sensors), and d_1 is the hidden dimension. On the temporal dimension, we define two semantic positional embeddings, $\mathbf{T}^{TiD} \in \mathbb{R}^{N_D \times d_2}$ and $\mathbf{T}^{DiW} \in \mathbb{R}^{N_W \times d_3}$, where N_D is the number of time slots of a day (determined by the sensor's sampling frequency) and $N_W = 7$ is the number of days in a week. The temporal embeddings are thus shared among slots for the same time of the day and the same day of the week. Semantic temporal positional embeddings are helpful since traffic systems usually reflect the periodicity of human society. In addition, kindly note that all other baseline models (Li et al., 2018; Liu et al., 2022; Shao et al., 2022d; Wu et al., 2019, 2021; Zhou et al., 2022) also use such temporal features, so there is no unfairness. Linear(·) is a linear layer to reduce the hidden dimension. $\mathbf{E}$, $\mathbf{T}^{TiD}$, and $\mathbf{T}^{DiW}$ are trainable parameters.

Embedding $\mathbf{E}$ is vital for reducing the complexity of modeling the spatial correlations between time series. This is because attaching spatial embeddings plays a similar role to GCN in terms of solving the indistinguishability of samples (Shao et al., 2022b), but with two primary advantages. On the one hand, it is more efficient than GCNs, which usually have $\mathcal{O}(N^2)$ complexity. On the other hand, it does not generate many additional network parameters than approaches based on variable-specific modeling (Bai et al., 2020; Cirstea et al., 2022).

4.3. Hierarchical encoder

Window Transformer layer. Standard Transformer layers (Vaswani et al., 2017) are designed based on the multi-head self-attention mechanism. As shown in Fig. 3a, it computes the attention among all input tokens. Therefore, each layer of the Transformer layer has an infinite receptive field, and many works (Liu et al., 2022; Wu et al., 2021; Zhou et al., 2021, 2022) try to capture long-term dependencies based on such a feature.

However, the infinite receptive field makes the standard transformer layers unable to generate multi-scale features (Liu et al., 2021b). Inspired by recent development in computer vision (Liu et al., 2021b), we apply the window self-attention in HUTFormer to extract the hierarchical multi-scale features. An example of window self-attention with windows size 2 is shown in Fig. 3b. Window self-attention forces calculating attention inside non-overlapping windows, thereby limiting the size of the receptive field. By replacing multi-head self-attention in standard Transformer layers (Vaswani et al., 2017) with the Window Multi-head Self-Attention (W-MSA), we present the window Transformer layer:

$$\begin{aligned} \mathbf{H}^{\text{in}'} &= \text{W} - \text{MSA}(\text{LN}(\mathbf{H}^{\text{in}})) + \mathbf{H}^{\text{in}} \\ \mathbf{H}^{\text{out}} &= \text{MLP}(\text{LN}(\mathbf{H}^{\text{in}'})) + \mathbf{H}^{\text{in}'} \end{aligned} \tag{3}$$

where LN(·) is the layer normalization, and MLP(·) is the multi-layer perceptron. $\mathbf{H}^{\text{in}} \in \mathbb{R}^{P \times d}$ and $\mathbf{H}^{\text{out}} \in \mathbb{R}^{P \times d}$ are the input and output sequences. P is the sequence length, and d is the hidden dimension. By limiting the receptive field size, the window transformer layer is the basis for extracting multi-scale features.

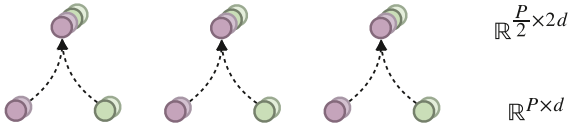


Fig. 4. An illustration of segment merging.

Segment merging. To generate hierarchical multi-scale representations, we adopt segment merging, which reduces the number of tokens and increases the number of hidden dimensions as the network gets deeper. As illustrated in Fig. 4, segment merging divides the token series into non-overlapping groups of size 2, and concatenates the features within each group.

By combining the segment merging and window transformer layer, we get the basic block of the hierarchical encoder (i.e., the blue block in Fig. 2). Assuming $(\mathbf{H}^i)_{\text{enc}}^l \in \mathbb{R}^{P^l \times d^l}$ is the representation of series i after block l ($l \geq 1$) of the encoder, the $(l+1)$th block is computed as

$$\begin{aligned} (\mathbf{H}^i)_{\text{enc}}^{l'} &= \text{SegmentMerging}((\mathbf{H}^i)_{\text{enc}}^{l}) \\ (\mathbf{H}^i)_{\text{enc}}^{l+1} &= \text{WindowTransformer}((\mathbf{H}^i)_{\text{enc}}^{(l')}) \end{aligned} \tag{4}$$

where $(\mathbf{H}^i)_{\text{enc}}^{l+1} \in \mathbb{R}^{P^{l+1} \times d^{l+1}}$ is the representation of time series i after block $l+1$ of the encoder. $P^{l+1} = \frac{P^l}{2}$ is the number of tokens after $(l+1)$th layer, and $d^{l+1} = 2d^{l+1}$ is the hidden dimension.

Prediction layer. Assuming there are S blocks in the encoder, HUTFormer makes an intermediate prediction with a linear layer:

$$\widehat{\mathbf{Y}}_{\text{enc}}^i = \text{Linear}(\Big\|_{j=1}^{P^S} (\mathbf{H}_j^i)_{\text{enc}}^S) \tag{5}$$

where P^S is the number of tokens after the Sth block. $\widehat{\mathbf{Y}}^i \in \mathbb{R}^{T_f \times C}$ is the prediction of time series i. Considering the prediction from all N time series, $\widehat{\mathcal{Y}}^{\text{enc}} \in \mathbb{R}^{T_f \times N \times C}$, we compute the Mean Absolute Error (MAE) as regression loss to train the hierarchical encoder:

$$\mathcal{L}_{\text{enc}} = \frac{1}{T_f N C} \sum_{j=1}^{T_f} \sum_{i=1}^{N} \sum_{k=1}^{C} |\widehat{\mathcal{Y}}_{ijk}^{enc} - \mathcal{Y}_{ijk}| \tag{6}$$

4.4. Hierarchical decoder

Cross-scale Transformer layer. The hierarchical decoder aims to effectively utilize the multi-scale features, to fine-tune each segment of the intermediate prediction. However, as discussed in Section 4.1, the history and future sequence in traffic forecasting tasks are not aligned, making the feature sequences extracted by the encoder and the decoder cannot be directly superimposed. Therefore, we design a cross-scale attention mechanism to select and incorporate multi-scale features. Different from self-attention, cross-scale attention utilizes the

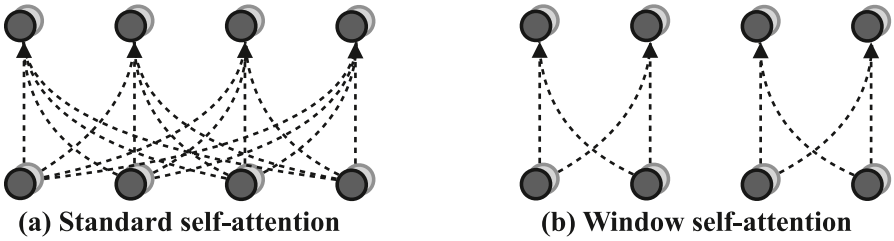


Fig. 3. Standard self-attention v. s. Window self-attention.

representations of the decoder as queries to retrieve the multi-scale features from the encoder. For brevity, we denote $\mathbf{H}_{\text{enc}} \in \mathbb{R}^{P_{\text{enc}} \times d_{\text{enc}}}$ as the representation from the encoder and $\mathbf{H}_{\text{dec}} \in \mathbb{R}^{P_{\text{dec}} \times d_{\text{dec}}}$ as the corresponding representation from the decoder. Then, the Cross-scale Attention (CA) is computed as

$$\begin{aligned} \text{CA}(\mathbf{H}_{\text{enc}}, \mathbf{H}_{\text{dec}}) &= \text{Softmax}\left(\frac{\mathbf{H}_{\text{dec}}(\mathbf{H}'_{\text{enc}})^T}{\sqrt{d_{\text{dec}}}}\right)\mathbf{H}'_{\text{enc}} \\ \mathbf{H}'_{\text{enc}} &= \text{Linear}(\mathbf{H}_{\text{enc}}) \end{aligned} \tag{7}$$

The Linear(·) layer is used to transform the hidden dimension from d_{enc} to d_{dec}. By replacing the multi-head self-attention with Multi-head Cross-scale Attention (MCA), we present the cross-scale Transformer layer as

$$\begin{aligned} \mathbf{H}^{\text{in}'}_{\text{dec}} &= \text{MCA}(\text{LN}(\mathbf{H}^{\text{in}}_{\text{enc}}, \mathbf{H}^{\text{in}}_{\text{dec}})) + \mathbf{H}^{\text{in}}_{\text{dec}} \\ \mathbf{H}^{\text{out}}_{\text{dec}} &= \text{MLP}(\text{LN}(\mathbf{H}^{\text{in}'}_{\text{dec}})) + \mathbf{H}^{\text{in}'}_{\text{dec}} \end{aligned} \tag{8}$$

where $\mathbf{H}^{\text{in}}_{\text{enc}}$ is the multi-scale feature from the encoder, and $\mathbf{H}^{\text{in}}_{\text{dec}}$ is the input feature from the decoder. $\mathbf{H}^{\text{out}}_{\text{dec}}$ is the output of the cross-scale Transformer layer.

Prediction layer. Assuming $(\mathbf{H}^i_j)^S_{\text{dec}} \in \mathbb{R}^{d_{\text{dec}}}$ is the representation of the decoder's last block (i.e., the Sth block) for jth segment of ith time series, HUTFormer makes the final prediction for each segment with a shared linear layer:

$$(\widehat{\mathbf{Y}}^i_j)_{\text{dec}} = \text{Linear}((\mathbf{H}^i_j)^S_{\text{dec}}) \tag{9}$$

where $(\widehat{\mathbf{Y}}^i_j)_{\text{dec}} \in \mathbb{R}^{LC}$ is the final prediction of segment j of time series i. Similar to the encoder, we consider the prediction from all P_{dec} segments ($P_{\text{dec}} \times L = T_f$) of all N time series, $\widehat{\mathcal{Y}}^{\text{dec}} \in \mathbb{R}^{T_f \times N \times C}$, and compute the MAE loss to train the hierarchical decoder:

$$\mathcal{L}_{\text{dec}} = \frac{1}{T_f NC}\sum_{j=1}^{T_f}\sum_{i=1}^{N}\sum_{k=1}^{C}|\widehat{\mathcal{Y}}^{\text{dec}}_{ijk} - \mathcal{Y}_{ijk}| \tag{10}$$

Kindly note that the parameters of the encoder are fixed during this stage to serve as a pre-training model for extracting robust hierarchical multi-scale representations of traffic data.

5. Experiments

In this section, we conduct extensive experiments on four real-world traffic datasets to validate the effectiveness of HUTFormer for long-term traffic forecasting. First, we introduce the experimental settings, including datasets, baselines, and implementation details. Then, we compare HUTFormer with other state-of-the-art traffic forecasting baselines and long-sequence time series forecasting baselines. Furthermore, we conduct more experiments to evaluate the impact of important components and strategies, including the effectiveness of the hierarchical U-Net structure, the input embedding strategy, and the two-stage training strategy.

5.1. Experimental setting

Datasets. We conduct experiments on four commonly used traffic datasets, including two traffic speed datasets (METR-LA and PEMS-BAY) and two traffic flow datasets (PEMS04 and PEMS08). The statistical information is summarized in Table 1.

Table 1
Statistics of datasets.

Type	Dataset	# Sample	# Sensor	Sample rate
Speed	METR-LA	34,272	207	5 min
	PEMS-BAY	52,116	325	5 min
Flow	PEMS04	16,992	307	5 min
	PEMS08	17,856	170	5 min

- METR-LA is a traffic speed dataset collected from loop-detectors located on the LA County road network (Jagadish et al., 2014). It contains data of 207 selected sensors over a period of 4 months from Mar to Jun in 2012 (Li et al., 2018). The traffic information is recorded at the rate of every 5 min, and the total number of time slices is 34,272.
- PEMS-BAY is a traffic speed dataset collected from California Transportation Agencies (CalTrans) Performance Measurement System (PeMS) (Chen et al., 2001). It contains data of 325 sensors in the Bay Area over a period of 6 months from Jan 1st, 2017 to May 31st, 2017 (Li et al., 2018). The traffic information is recorded at the rate of every 5 min, and the total number of time slices is 52,116.
- PEMS04 is a traffic flow dataset also collected from CalTrans PeMS. It contains data of 307 sensors over a period of 2 months from Jan 1st, 2018 to Feb 28th, 2018 (Guo et al., 2019). The traffic information is recorded at the rate of every 5 min, and the total number of time slices is 16,992.
- PEMS08 is a public traffic flow dataset collected from CalTrans PeMS. Specifically, PEMS08 contains data of 170 sensors in San Bernardino over a period of 2 months from July 1st, 2018 to Aug 31st, 2018 (Guo et al., 2019). The traffic information is recorded at the rate of every 5 min, and the total number of time slices is 17,856.
- ETTh1, ETTm1, and Weather are non-traffic time series datasets used to verify the generalization of the proposed HUTFormer. Due to space limitations, we omit their details. Interested readers can refer to (Zhou et al., 2021).

Spatial attributes—especially road-network topology—are indispensable to traffic forecasting tasks. The following describes in detail the spatial information used by each traffic dataset.[3]

- METR-LA and PEMS-BAY: These datasets comprise 207 and 325 sensor nodes, respectively. Their geographic distributions are illustrated in Fig. 5. Following prior work (Li et al., 2018), the spatial adjacency matrix A is computed with a thresholded Gaussian kernel (Shuman et al., 2013):

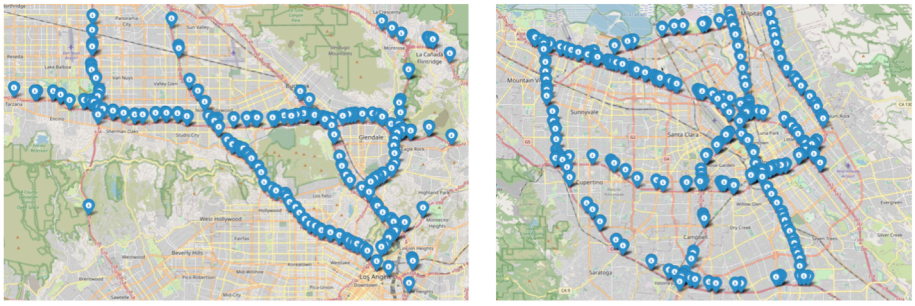
$$A_{ij} = \begin{cases} \exp\left(-\frac{dist(v_i, v_j)^2}{\sigma^2}\right), & dist(v_i, v_j) \le \kappa \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

  where dist(v_i, v_j) denotes the road-network distance between nodes v_i and v_j, σ is the standard deviation of all pairwise distances, and κ is the distance threshold.
- PEMS04 and PEMS08: These datasets contain 307 and 170 nodes, respectively. Because the original releases (Guo et al., 2019; Yu et al., 2018) only contain the distance between sensors without the raw latitude–longitude coordinates, we omit their spatial visualizations. Consistent with the previous studies (Guo et al., 2019; Yu et al., 2018), the spatial adjacency matrix A is defined as

$$A_{ij} = \begin{cases} 1, & dist(v_i, v_j) \le 3.5 \text{ mile} \\ 0, & \text{otherwise} \end{cases} \tag{12}$$

Baselines. On the one hand, we select six traffic forecasting baselines,

[3] ETTh1, ETTm1, and Weather are not traffic time-series datasets; they are non-spatiotemporal benchmarks used to evaluate the generalization capability of HUTFormer, and are therefore omitted here.



(a) METR-LA (b) PEMS-BAY

Fig. 5. Spatial typologies of METR-LA and PEMS-BAY datasets.

including:

- DCRNN (Li et al., 2018) is one of the earliest works for STGNN-based traffic forecasting, which replaces the fully connected layer in GRU (Cho et al., 2014) by diffusion convolutional layer to form a Diffusion Convolutional Gated Recurrent Unit.
- Graph WaveNet (Wu et al., 2019) is a traffic forecasting model, which stacks gated temporal convolutional layer and GCN layer by layer to jointly capture the spatial and temporal dependencies.
- MTGNN (Wu et al., 2020) is a traffic forecasting model, which extends Graph WaveNet through the mix-hop propagation layer in the spatial module, the dilated inception layer in the temporal module, and a delicate graph learning layer.
- STID (Shao et al., 2022b) is a simple but effective baseline for traffic forecasting, which identifies the indistinguishability of samples in both spatial and temporal dimensions as a key bottleneck, and addresses the indistinguishability by attaching spatial and temporal identities.
- STEP (Shao et al., 2022c) is a traffic forecasting model, which enhances existing STGNNs with the help of a time series pre-training model. It significantly extends the length of historical data.
- D^2STGNN (Shao et al., 2022d) is a state-of-the-art traffic forecasting model, which identifies the diffusion process and inherent process in traffic data, and further decouples them for better modeling.

On the other hand, we also select six long-sequence forecasting baselines, including:

- HI (Cui et al., 2021) is a basic baseline for long-sequence time series forecasting problems, which directly takes the most recent time steps in the input as output.
- DLinear (Zeng et al., 2023) is a simple but effective long-sequence time series forecasting model, which decomposes the time series into a trend and a remainder series and employs two one-layer linear networks to model these two series.
- Informer (Zhou et al., 2021) is a model for long-sequence time series forecasting, which designs a ProbSparse self-attention mechanism and distilling operation to handle the challenges of the quadratic complexity in the standard Transformer. Also, it carefully designs a generative decoder to alleviate the limitation of standard encoder-decoder architecture.
- Autoformer (Wu et al., 2021) is a model for long-sequence time series forecasting, which is proposed as a decomposition architecture by embedding the series decomposition block as an inner operator. Besides, it designs an efficient Auto-Correlation mechanism to conduct dependencies discovery and information aggregation at the series level.
- FEDformer (Zhou et al., 2022) is a frequency-enhance Transformer for long-sequence time series forecasting. It proposes an attention mechanism with low-rank approximation in frequency and a mixture of experts decomposition to control the distribution shifting.
- Pyraformer (Liu et al., 2022) is a pyramidal attention-based model for long-sequence time series forecasting. Pyramidal attention can effectively describe short and long temporal dependencies.
- Crossformer (Zhang and Yan, 2023) is a Transformer-based model utilizing cross-dimension dependency for multivariate time-series (MTS) forecasting.
- PatchTST (Nie et al., 2023) proposes an effective design of Transformer-based models for time series forecasting tasks by introducing two key components: Patching and channel-independent structure.

Metrics. In this study, we evaluate the performances of all baselines by Mean Absolute Error (MAE) and Mean Absolute Percentage Error (MAPE) metrics. First, the MAE metric reflects the absolute prediction error, but is affected by the units of the dataset. For example, traffic speed datasets usually take values between 0 and 70 km/h, while traffic flow datasets usually take values between zero and hundreds. Thus, we also adopt MAPE, which can eliminate the impact of data units and reflects the relative error, helping to understand the accuracy more intuitively.

Implementation. For all datasets, we use historical $T_p = 288$ time steps (i.e., 1 day) to predict future $T_f = 288$ time steps. For HUTFormer, we set the segment length L to 12, and the number of segments $P = 24$ ($L \times P = 288$). We set the window size to 3. We set the hidden dimension of temporal embedding $\mathbf{T}^{TiD}$ to 8, while others d to 32. The depth of HUTFormer is set to 4. For baselines, we adopt the default settings. Moreover, as discussed before, STGNNs can not directly handle the long-term traffic forecasting task due to their high complexity. Therefore, we first apply the segment embedding to reduce the length of input tokens for them.[4] On the one hand, all baseline models are based on their official open-source code, with only modifications to the model input (introducing segment embedding technology), and we ensure that both HUTFormer and the baseline models use the same hyperparameters (segment size and stride size), thereby maintaining fairness in model structure. On the other hand, all models are trained using a unified and scalable pipeline (Shao et al., 2025c), ensuring fairness in the training process.

Optimization settings. For both encoding and decoding stages, we apply the optimization settings in Table 2. Specifically, we adopt Adam (Kingma and Ba, 2015) as our optimizer, and set learning rate and weight decay to 0.0005 and 0.0001, respectively. The batch size is set to 64. In addition, we use a learning rate scheduler, MultiStepLR, which adjusts the learning rate at epochs 1, 40, 80, and 120 with gamma 0.5. Moreover, the gradient clip is set to 5. All the experiments in Section 5 are running on an Intel(R) Xeon(R) Gold 5217 CPU @ 3.00 GHz, 128G RAM computing server, equipped with RTX 3090 graphics cards.

[4] Methods implemented with segment embeddings are marked with *.

Table 2
Optimization settings.

Config	Value
Optimizer	Adam
Learning rate	0.0005
Batch size	64
Weight decay	0.0001
Learning rate schedule	MultiStepLR
Milestones	[1, 40, 80, 120]
Gamma	0.5
Gradient clip	5

5.2. Main results

Settings. We follow the dataset division in previous works. Specifically, for traffic speed datasets (METR-LA and PEMS-BAY), we use 70%, 10%, and 20% of the data for training, validating, and testing, respectively. For traffic flow datasets (PEMS04 and PEMS08), we use 60%, 20%, and 20% of data for training, validating, and testing, respectively. We compare the performance at 1, 4, 8, 12, 16, and 24 h (horizon 12, 48, 96, 144, 192, and 288) of forecasting on the MAE and MAPE metrics.

Results. The results of traffic speed and flow forecasting are shown in Tables 3 and 4, respectively. In general, HUTFormer consistently outperforms all baselines, indicating its effectiveness. Notably, Crossformer (Zhang and Yan, 2023) suffers from out-of-memory issues due to its high complexity and is therefore ignored in Tables 3 and 4.

Long-sequence forecasting models do not perform well on traffic forecasting tasks. We conjecture that the main reason is that these models do not fit the characteristics of traffic data. First, there exist strong correlations between the time series of traffic data. For example, due to the constraint of road networks, time series from adjacent sensors or from similar geographical functional areas may be more similar (Pan et al., 2019). Understanding and exploiting the correlations between time series is essential for traffic forecasting. However, long-sequence forecasting models are usually not concerned with such spatial dependencies. Second, as discussed in Section 1, the long-term traffic forecasting task requires exploiting multi-scale representations to capture the complex dynamics of traffic data. However, most long-term sequence forecasting models mainly focus on capturing global dependencies based on self-attention mechanisms. For example, Informer (Zhou et al., 2021) optimizes the efficiency of the original self-attention mechanism through the ProbSparse mechanism. Autoformer (Wu et al., 2021) conducts the dependencies discovery at the series level. They can not generate and utilize multi-scale representations of traffic data. In summary, the above-mentioned uniqueness of long-term traffic forecasting tasks significantly affects the effectiveness of long-sequence forecasting models.

Compared to long-sequence forecasting models, traffic forecasting models achieve better performance. This is mainly because they model correlations between time series with the help of graph convolution. Most of them (Li et al., 2018; Shao et al., 2022c, 2022d; Wu et al., 2019, 2020) utilize diffusion convolution, a variant of graph convolution, to model the diffusion process at each time step. However, there is no free lunch. The graph convolution brings a high complexity (Shao et al., 2022c). As mentioned earlier, we had to implement these models with the segment embedding in HUTFormer to reduce the length of input tokens to make them runnable. Kindly note that although the latest baseline STEP (Shao et al., 2022c) can handle long-term historical data, it still requires a downstream STGNN as the backend, which can only make short-term future predictions. In summary, these models only focus on short-term traffic forecasting and do not consider the uniqueness of long-term traffic forecasting, i.e., exploiting multi-scale representations.

Compared to all baselines, HUTFormer achieves state-of-the-art performances by sufficiently addressing the issues of long-term traffic forecasting tasks. Specifically, on the one hand, HUTFormer efficiently handles the correlations between long-term time series with spatial-temporal positional encoding and segment embedding. On the other

Table 3
Long-term traffic forecasting on traffic speed datasets METR-LA and PEMS-BAY.

Data	Method	@Horizon 12		@Horizon 48		@Horizon 96		@Horizon 144		@Horizon 192		@Horizon 288	
		MAE	MAPE (%)	MAE	MAPE (%)	MAE	MAPE (%)	MAE	MAPE (%)	MAE	MAPE (%)	MAE	MAPE(%)
METR-LA	HI	10.44	23.21	10.42	23.19	10.43	23.23	10.43	23.32	10.40	23.34	10.22	22.81
	DLinear	7.61	16.19	12.86	23.79	12.99	23.11	12.90	23.48	12.89	23.15	13.07	23.33
	Informer	4.65	15.52	4.86	16.54	4.98	17.16	5.07	17.41	5.07	17.30	5.06	17.14
	Autoformer	7.23	19.25	7.27	19.73	7.45	20.23	7.83	21.49	7.74	20.98	8.41	22.43
	FEDformer	8.78	22.29	9.11	22.69	9.12	22.75	9.54	24.18	9.81	24.76	10.13	25.56
	Pyraformer	4.22	12.84	4.55	14.93	4.75	15.81	4.80	15.89	4.81	15.68	4.62	14.79
	PatchTST	4.43	13.58	5.02	16.37	5.14	16.64	5.19	16.98	5.21	16.67	5.25	17.16
	DCRNN*	4.07	12.74	4.39	14.08	4.44	14.02	4.46	14.16	4.51	14.41	4.71	15.59
	GWNet*	3.87	12.18	4.19	13.60	4.25	13.62	4.42	14.56	4.58	15.40	4.51	15.09
	MTGNN*	4.01	12.31	4.31	13.84	4.53	14.85	4.59	14.77	4.57	15.18	4.75	15.93
	STID	3.84	12.17	4.13	14.11	4.04	13.05	4.11	13.65	4.15	14.07	4.17	13.83
	STEP*	3.74	11.60	4.14	13.24	4.22	13.52	4.38	14.07	4.34	13.96	4.43	14.42
	D^2STGNN*	3.71	11.24	3.96	12.84	3.99	13.26	4.05	13.17	4.05	13.36	4.09	12.78
	HUTFormer	**3.59**	**10.93**	**3.77**	**11.88**	**3.79**	**11.86**	**3.80**	**12.08**	**3.82**	**12.18**	**3.84**	**12.28**
PEMS-BAY	HI	3.37	7.84	3.36	7.80	3.36	7.77	3.36	7.76	3.36	7.74	3.38	7.79
	DLinear	2.70	6.28	3.14	7.75	3.13	7.77	3.15	7.76	3.15	7.78	3.23	7.90
	Informer	2.77	6.65	2.80	6.88	2.84	7.06	2.83	7.07	2.82	6.98	2.92	7.16
	Autoformer	3.15	7.48	3.24	7.85	3.30	8.00	3.37	8.10	3.39	8.15	4.35	11.25
	FEDformer	3.04	7.55	3.14	7.61	3.13	7.58	3.32	8.00	3.42	8.45	3.67	9.33
	Pyraformer	2.53	6.21	2.71	6.72	2.64	6.39	2.74	6.65	2.75	6.68	2.77	6.81
	PatchTST	2.35	5.94	2.92	7.45	2.96	7.52	3.00	7.62	3.01	7.67	3.10	7.73
	DCRNN*	2.18	5.49	2.52	6.49	2.54	6.43	2.66	6.79	2.67	6.80	2.66	6.62
	GWNet*	2.01	5.11	2.35	5.91	2.40	5.98	2.47	6.35	2.46	6.24	2.46	6.09
	MTGNN*	2.17	5.40	2.45	6.11	2.51	6.04	2.52	6.13	2.57	6.19	2.70	6.40
	STID	2.02	5.02	2.29	5.66	2.32	5.69	2.33	5.72	2.32	5.67	2.38	5.81
	STEP*	2.00	4.94	2.33	5.93	2.38	6.05	2.44	6.26	2.45	6.24	2.54	6.41
	D^2STGNN*	2.04	4.97	2.26	5.44	2.29	5.60	2.34	5.55	2.31	5.50	2.38	5.64
	HUTFormer	**1.93**	**4.62**	**2.18**	**5.16**	**2.21**	**5.24**	**2.22**	**5.24**	**2.23**	**5.25**	**2.28**	**5.35**

Table 4
Long-term traffic forecasting on traffic flow datasets PEMS04 and PEMS08.

Data	Method	@Horizon 12		@Horizon 48		@Horizon 96		@Horizon 144		@Horizon 192		@Horizon 288	
		MAE	MAPE (%)	MAE	MAPE (%)	MAE	MAPE (%)	MAE	MAPE (%)	MAE	MAPE (%)	MAE	MAPE(%)
PEMS04	HI	41.73	28.46	41.16	28.61	41.38	28.62	41.28	28.42	30.99	27.34	39.58	26.49
	DLinear	27.29	19.83	37.20	26.51	37.50	26.78	37.57	26.87	37.17	25.27	36.87	25.21
	Informer	25.94	17.56	25.72	18.05	25.60	18.27	25.98	17.81	26.42	17.67	27.42	18.57
	Autoformer	29.94	28.00	31.30	27.41	31.47	27.73	31.95	27.89	32.03	28.03	33.34	29.82
	FEDformer	34.94	34.33	32.24	37.23	33.90	34.33	35.12	41.26	35.16	34.08	41.83	51.01
	Pyraformer	23.40	17.18	25.40	18.80	26.45	19.89	26.22	19.01	26.51	19.18	26.58	20.57
	PatchTST	22.75	16.67	29.37	21.85	30.63	23.15	32.01	24.00	30.54	21.54	31.50	24.00
	DCRNN*	22.25	16.59	24.42	18.89	25.20	19.17	26.31	19.61	27.32	19.74	28.04	21.02
	GWNet*	22.24	16.51	23.50	18.29	24.08	18.07	24.85	18.21	25.83	18.98	31.17	21.00
	MTGNN*	21.75	15.93	23.04	17.81	24.33	17.80	25.56	17.68	25.80	17.85	26.78	20.64
	STID	21.01	15.24	22.77	16.61	23.39	16.87	24.06	17.08	24.43	17.22	25.19	17.49
	STEP*	20.82	15.56	22.23	17.11	22.87	17.21	24.46	17.97	24.89	17.40	26.18	18.47
	D^2STGNN*	21.55	16.03	22.98	17.04	24.16	17.57	24.50	17.93	24.59	17.19	24.79	17.97
	HUTFormer	**19.61**	**13.59**	**21.54**	**14.95**	**21.96**	**15.22**	**22.66**	**15.30**	**23.10**	**15.35**	**23.43**	**15.71**
PEMS08	HI	37.33	25.01	37.31	25.07	37.23	25.05	37.09	25.02	36.94	24.98	36.40	24.76
	DLinear	22.91	17.23	34.13	24.15	34.34	25.54	34.44	23.80	34.52	23.91	35.11	23.71
	Informer	24.55	14.76	24.80	15.03	24.72	15.03	25.07	15.11	24.82	14.91	25.09	15.61
	Autoformer	31.36	25.44	32.29	27.13	33.19	27.45	32.98	26.15	33.57	25.78	36.75	28.82
	FEDformer	24.62	20.01	26.76	21.85	28.56	23.02	30.33	24.47	29.11	23.14	29.91	24.47
	Pyraformer	21.92	14.43	23.00	14.70	23.80	15.46	24.45	16.88	24.34	16.17	22.71	14.79
	PatchTST	16.94	11.37	21.27	15.10	22.56	16.39	23.22	17.40	23.18	17.70	23.73	17.35
	DCRNN*	18.64	13.47	20.42	14.92	20.97	15.11	21.63	15.51	22.45	16.23	22.95	16.72
	GWNet*	17.07	11.57	19.55	11.93	20.38	14.33	20.49	14.82	20.00	14.68	20.29	15.20
	MTGNN*	17.75	12.61	19.27	13.35	19.99	13.85	20.68	15.00	20.95	14.65	22.16	15.68
	STID	16.40	11.42	18.53	13.26	19.17	13.66	19.59	13.78	19.59	14.03	20.23	15.35
	STEP*	16.67	11.34	19.05	14.00	19.74	14.74	20.15	14.88	19.80	14.84	20.37	15.54
	D^2STGNN*	17.27	11.47	18.45	12.35	18.97	12.63	19.33	12.81	19.09	12.34	19.55	12.93
	HUTFormer	**15.18**	**10.09**	**16.72**	**11.26**	**17.23**	**11.55**	**17.59**	**11.74**	**17.83**	**11.84**	**18.44**	**12.20**

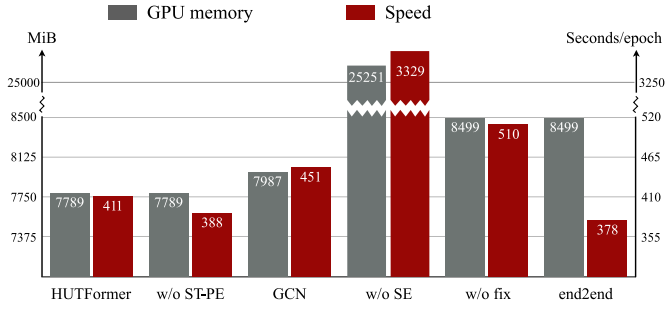


Fig. 6. Efficiency study.

hand, HUTFormer effectively generates and utilizes multi-scale representations based on the hierarchical U-Net.

5.3. Efficiency

In this section, we conduct more experiments to evaluate the efficiency of the HUTFormer variants in Section 5.5. We conduct experiments with a single NVIDIA V100 graphics card with 32 GB memory, and report the GPU memory usage and running time. Specifically, for the two-stage training variants, we report the largest GPU memory usage of the two stages and report the sum of the running time in the encoding and decoding stages. We conduct experiments on the METR-LA dataset.

The results are shown in Fig. 6. First, we can see that removing the segment embedding (i.e., w/o SE) will significantly increase the computational complexity, and require more GPU memory. Second, compared with applying GCN, HUTFormer is more efficient and effective by leveraging the spatial-temporal positional encoding, which does not increase much complexity.

5.4. Generalization

The ability of HUTFormer to generate and utilize multi-scale features should also be effective in many non-traffic data, since the multi-scale features widely exists in many domains. In order to verify the generalization of HUTFormer, in this part, we compare HUTFormer with more latest Transformer-based long time series forecasting models [10, 74] based on three commonly used long-sequence prediction datasets, ETTh1, ETTm1, and Weather. The details of Crossformer (Zhang and Yan, 2023) and Triformer (Cirstea et al., 2022) as well as the three datasets are neglected for simplicity. Interest readers can refer to their papers [10, 74]. We use the same setting as the other datasets in our paper. As shown in Table 5, HUTFormer still outperforms these models on these datasets, which further verifies the effectiveness and generalization of HUTFormer.

5.5. Ablation study

In this subsection, we conduct more experiments to evaluate the impact of some important components and strategies. Specifically, we evaluate from three aspects, including the effectiveness of the hierarchical U-Net structure, the input embedding strategy, and the two-stage training strategy. Due to space limitations, we only present the results on METR-LA datasets in Table 6.

The hierarchical U-Net structure is designed to generate and exploit multi-scale features. Specifically, the encoder combines window self-attention and segment merging to generate multi-scale features, while the decoder primarily utilizes extracted features based on cross-scale attention. Therefore, to evaluate their effectiveness, we set up three variants. First, we replace the decoder with a simple concatenation, named HUTFormer concat. The concatenation of features from different scales naturally preserves all information. Second, we set HUTFormer w/o decoder to remove the decoder and use the intermediate prediction as the final prediction. The above two variants are used to demonstrate

Table 5
Experiments on ETTh1, ETTm1, and Weather datasets.

Data	Method	@Horizon 12		@Horizon 48		@Horizon 96		@Horizon 144		@Horizon 192		@Horizon 288	
		MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
ETTh1	Informer	0.62	0.82	0.69	0.91	0.82	1.10	0.90	1.25	0.96	1.43	0.85	1.17
	Autoformer	0.45	0.44	0.47	0.47	0.48	0.50	0.48	0.52	0.50	0.53	0.51	0.53
	FEDformer	0.42	0.37	0.43	0.40	0.44	0.43	0.46	0.46	0.46	0.47	0.52	0.57
	Pyraformer	0.56	0.63	0.57	0.64	0.65	0.79	0.74	0.92	0.76	1.03	0.79	1.09
	Triformer	0.44	0.44	0.46	0.48	0.48	0.52	0.49	0.55	0.50	0.55	0.51	0.56
	Crossformer	0.39	0.35	0.40	0.38	0.44	0.44	0.45	0.46	0.45	0.47	0.48	0.49
	PatchTST	0.37	0.32	**0.38**	**0.35**	0.42	0.42	**0.43**	0.46	**0.45**	**0.47**	0.48	0.49
	HUTFormer	**0.36**	**0.31**	**0.38**	**0.35**	**0.41**	**0.41**	**0.43**	**0.44**	**0.45**	**0.47**	**0.47**	**0.47**
ETTm1	Informer	0.53	0.59	0.60	0.67	0.63	0.74	0.68	0.85	0.72	0.91	0.74	0.95
	Autoformer	0.49	0.53	0.53	0.61	0.53	0.62	0.54	0.63	0.54	0.63	0.62	0.76
	FEDformer	0.37	0.29	0.41	0.37	0.43	0.40	0.44	0.42	0.43	0.42	0.46	0.47
	Pyraformer	0.52	0.53	0.64	0.80	0.62	0.71	0.71	0.89	0.59	0.65	0.71	0.88
	Triformer	0.34	0.26	0.39	0.34	0.39	0.35	0.41	0.39	0.41	0.38	0.43	0.42
	Crossformer	0.32	0.23	0.41	0.37	0.42	0.37	0.51	0.51	0.53	0.52	0.58	0.61
	PatchTST	0.29	0.21	**0.35**	0.32	0.36	0.34	**0.38**	0.38	0.38	0.38	0.40	0.41
	HUTFormer	**0.28**	**0.20**	**0.35**	**0.31**	**0.35**	**0.31**	**0.38**	**0.36**	**0.36**	**0.35**	**0.38**	**0.39**
Weather	Informer	0.34	0.27	0.38	0.36	0.40	0.38	0.43	0.42	0.45	0.46	0.45	0.48
	Autoformer	0.36	0.29	0.38	0.33	0.39	0.35	0.41	0.39	0.42	0.41	0.44	0.44
	FEDformer	0.32	0.24	0.34	0.27	0.35	0.30	0.36	0.32	0.38	0.35	0.47	0.48
	Pyraformer	0.28	0.23	0.42	0.45	0.36	0.34	0.38	0.38	0.50	0.59	0.42	0.44
	Triformer	0.15	0.12	0.23	0.20	0.26	0.22	0.28	0.24	0.32	0.29	0.34	0.33
	Crossformer	0.14	0.11	0.22	0.19	0.25	0.21	0.27	**0.23**	0.31	0.27	0.32	0.31
	PatchTST	0.14	0.11	0.22	0.18	0.25	0.21	0.27	**0.23**	0.31	0.28	0.33	0.32
	HUTFormer	**0.12**	**0.10**	**0.20**	**0.16**	**0.24**	**0.20**	**0.26**	**0.23**	**0.29**	**0.27**	**0.31**	**0.30**

Table 6
Ablation study on the METR-LA dataset.

Variant	@Horizon 12		@Horizon 48		@Horizon 96		@Horizon 144		@Horizon 192		@Horizon 288	
	MAE	MAPE (%)	MAE	MAPE (%)	MAE	MAPE (%)	MAE	MAPE (%)	MAE	MAPE (%)	MAE	MAPE(%)
HUTFormer	**3.59**	**10.93**	**3.77**	**11.88**	**3.79**	**11.86**	**3.80**	**12.08**	**3.82**	**12.18**	**3.84**	**12.28**
concat	3.86	12.16	3.98	13.23	4.01	13.36	4.01	13.36	4.05	13.41	4.08	13.65
w/o decoder	3.80	11.94	3.85	12.33	3.90	12.64	3.88	12.91	3.96	12.91	3.97	12.93
w/o hierarchy	3.90	12.56	3.97	12.85	3.96	12.86	3.98	12.88	3.98	12.92	4.12	13.48
w/o ST-PE	4.11	12.68	4.78	15.80	4.90	16.44	5.00	16.81	5.13	17.47	5.25	17.56
GCN	3.79	11.87	4.23	14.14	4.28	14.32	4.30	14.21	4.32	14.28	4.35	14.40
w/o SE	3.76	11.83	3.86	12.39	3.85	12.35	3.91	12.73	3.92	12.75	4.03	13.12
end2end	3.72	11.60	3.95	12.59	3.97	12.83	3.95	12.58	3.95	12.58	4.00	12.73
w/o fix	3.64	11.28	3.85	12.11	3.88	12.49	3.90	12.40	3.93	12.57	3.91	12.57

that exploiting multi-scale features is a non-trivial challenge and our hierarchical decoder is effective. Third, we set HUTFormer w/o hierarchy to further remove segment merging and replace the window Transformer layer with a standard Transformer layer, to evaluate the effectiveness of hierarchical multi-scale representations. As shown in Table 6, HUTFormer significantly outperforms HUTFormer concat and HUTFormer w/o decoder, which shows that it is not an easy task to utilize the multi-scale features, and validates the effectiveness of our decoder. In addition, HUTFormer w/o hierarchy shows that hierarchical multi-scale features are crucial for accurate long-term traffic forecasting. The above results show that generating and utilizing hierarchical multi-scale features is important, and the designed hierarchical U-Net structure is effective.

The input embedding strategy aims to address the complexity issue from both spatial and temporal dimensions. Specifically, it consists of a Segment Embedding (SE) and a Spatial-Temporal Positional Encoding (ST-PE). To verify their effectiveness, we set up three variants. First, we set HUTFormer w/o ST-PE, which replaces the ST-PE with standard learnable positional encoding. Second, we set HUTFormer GCN, which replaces the spatial embeddings in ST-PE with graph convolution (Wu et al., 2019). Third, we remove the segment embedding to get HUTFormer w/o SE. As shown in Table 6, without ST-PE, the performance of HUTFormer decreases significantly. This is because modeling the correlations between time series is the basis of traffic forecasting. In addition, we can see that the ST-PE strategy is significantly better than performing graph convolution, indicating the superiority of ST-PE. Moreover, removing segment embedding not only leads to a significant decrease in performance but also increases the complexity due to the increased sequence length. These results indicate the effectiveness of the spatial-temporal positional encoding and segment merging.

Finally, we evaluate the two-stage training strategy of HUTFormer. To this end, we set two variants. First, we set HUTFormer end2end, which trains the HUTFormer in an end-to-end strategy. Second, we set HUTFormer w/o fix, which does not fix the parameter of the encoder when training the decoder. The results in Table 6 show that either the end-to-end strategy or the strategy without fixing the encoder leads to insufficient optimization and significant performance degradation. In addition, both strategies require more memory. In contrast, our two-stage strategy achieves the best performance and efficiency simultaneously.

5.6. Hyper-parameter and convergence study

In this subsection, we first conduct experiments to study the impact

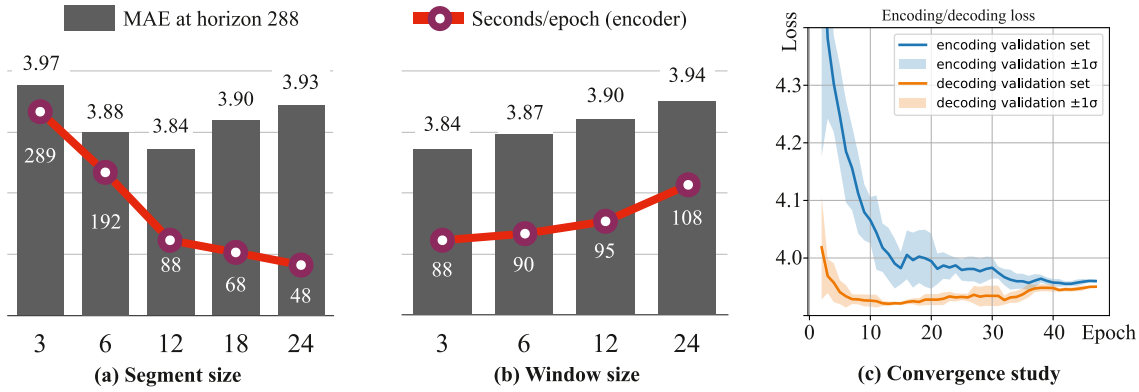


Fig. 7. Hyper-parameter and converagence study.

of two key hyper-parameters: segment size and window size. We conduct experiments on the METR-LA dataset and report the MAE at horizon 288. Moreover, we report the training speed of the encoder, since these hyper-parameters mainly affect the encoder. As shown in Fig. 7a, the segment size $L = 12$ achieves the best performance. Smaller segments cannot provide robust semantics, while larger segments ignore more local details. In addition, we can see that as the segment size increases, the encoder runs faster (s/epoch). Kindly note that changing the segment size may change the depth of the HUTFormer to ensure that the receptive field covers the entire sequence. The impact of the window size is shown in Fig. 7b, where larger window sizes perform worse. This is because the ability to extract multi-scale features is weakened as the window size increases. Moreover, the efficiency of HUTFormer will also decrease (Liu et al., 2021b) on larger window sizes.

Additionally, we conduct a convergence analysis experiment. Fig. 7c illustrates the validation set loss during the two-phase process, showing the convergence behavior of the encoding and decoding stages. Combined with Table 6 and Fig. 9, it can be observed that the predictions during the encoding phase already achieve good accuracy. The decoding phase, by introducing multi-scale information, further improves the prediction results, especially in the details (particularly in periods of traffic congestion), leading to a further reduction in loss.

5.7. Visualization

5.7.1. Spatial-temporal positional encoding

To further understand the HUTFormer in modeling the correlations between multiple time series in traffic data, we analyze the spatial-temporal positional encoding layer. Modeling correlations between multiple time series have been widely discussed in multivariate time series forecasting (Shao et al., 2022c; Wu et al., 2019, 2020). Previous works usually utilize Graph Convolution Networks (GCN), which conduct message passing in a pre-defined graph. GCN is a powerful model, but it has high complexity of $\mathcal{O}(N^2)$. Very recent works, STID (Shao et al., 2022b) and ST-Norm (Deng et al., 2021), identify that graph convolution in multivariate time series forecasting is essentially used for addressing the indistinguishability of samples on the spatial dimension. Based on such an observation, STID proposes a simple but effective baseline of attaching spatial and temporal identities, achieving a similar performance of GCN but high efficiency. The Spatial-Temporal Positional Encoding (ST-PE) is designed based on such an idea (Shao et al., 2022b).

The ST-PE contains three learnable positional embeddings, $\mathbf{E} \in \mathbb{R}^{N \times d}$, $\mathbf{T}^{TiD} \in \mathbb{R}^{N_D \times d}$, and $\mathbf{T}^{DiW} \in \mathbb{R}^{N_W \times d}$, where N is the number of time series, N_D is the number of time slots of a day (determined by the sensor's sampling frequency), and $N_W = 7$ is the number of days in a week. We utilize t-SNE (van der Maaten and Hinton, 2008) to visualize these three embedding matrices. Kindly note that $\mathbf{T}^{DiW}$ only have 7 embeddings, which is significantly less than the hidden dimension 32, making it hard to get correct visualizations. Therefore, we additionally train a HUTFormer with the embedding size of $\mathbf{T}^{DiW}$ to 2 to get a more accurate visualization.

The results are shown in Fig. 8. First, as shown in Fig. 8a, the spatial embeddings are likely to cluster. For example, traffic conditions observed by sensors that are connected or have similar geographical functionality are more likely to be similar. However, it is not as apparent as in the results in STID (Shao et al., 2022b). We conjecture this is because the impact of the indistinguishability of the samples becomes weaker as the length of the historical data increases. Second, Fig. 8b shows the embeddings of 288 time slots, where the daily periodicity is very obvious. Third, Fig. 8c visualizes the embeddings of each day in a week, where weekdays are closer and weekends' are different.

5.7.2. Prediction visualization

In order to further intuitively evaluate HUTFormer, in this subsection, we visualize the prediction of HUTFormer and other baselines on the METR-LA dataset. Specifically, we select sensor 12 and displayed its data from June 05th, 2012 to June 06th, 2012 (located in the test dataset).

Fig. 9 shows two consecutive days from the METR-LA dataset: The first day is used as historical input, while the second day is the prediction target. This design serves two purposes. First, the strong similarity between the two days illustrates the dataset's pronounced periodicity. Second, the series captures sharp, localized fluctuations during the morning and evening rush hours, revealing the onset and dissipation of traffic congestion.

Capturing global patterns is essential for modeling overall cyclical trends, whereas capturing local patterns is crucial for accurately forecasting fine-grained changes such as rapid, short-term spikes. As discussed previously, models like Autoformer (Wu et al., 2021), Graph WaveNet (Wu et al., 2019), and HUTFormer w/o hierarchy focus primarily on global dynamics and largely overlook local feature extraction. Consequently, although they reproduce broad periodic behavior reasonably well, their accuracy falls during intervals of rapid change—specifically, the congestion periods marked by the red background in the figure. By contrast, HUTFormer effectively models multi-scale features and thus maintains high accuracy even in these volatile segments. This result underscores the importance of integrating features at multiple scales when modeling complex, periodic traffic data.

5.8. Limitations

Although HUTFormer demonstrates strong performance, it still has several issues that need to be addressed in future work. First, the most significant issue with HUTFormer is its lack of generalization to new sensors. Graph-based spatial dependency modeling methods (Kipf and

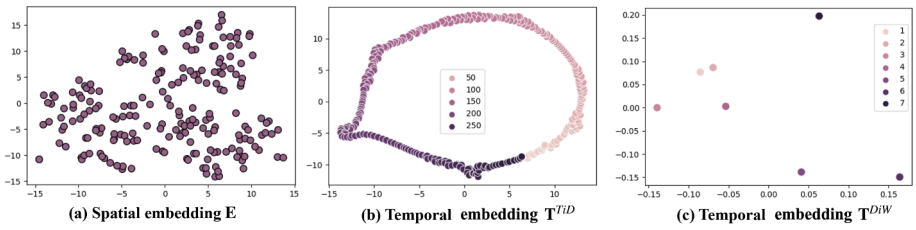


Fig. 8. Visualization of the spatial and temporal embeddings.

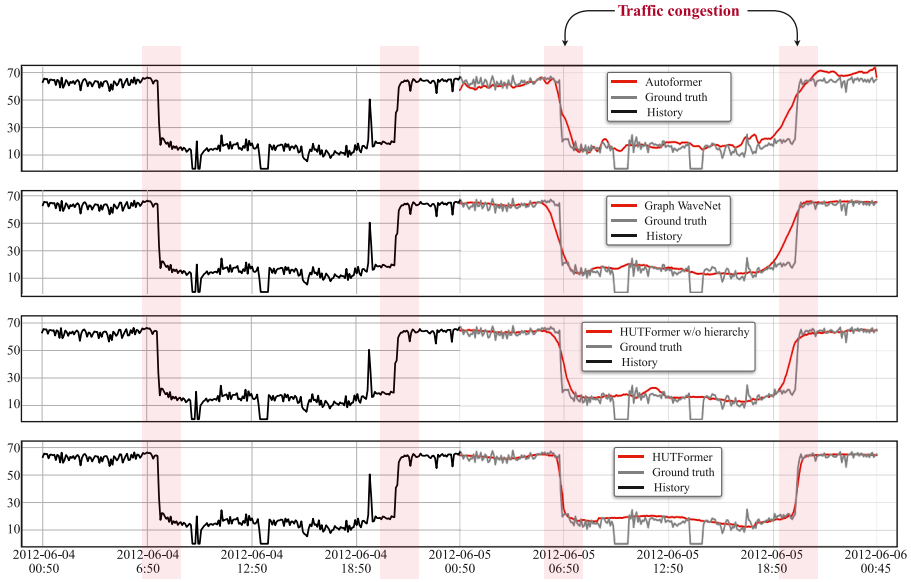


Fig. 9. Visualization of prediction results of HUTFormer and some baseline models.

Welling, 2017; Li et al., 2018; Wu et al., 2019) are inherently inductive (Hamilton et al., 2017), meaning they can make predictions on graphs with changing nodes and relationships. However, HUTFormer relies on spatial positional encoding to capture spatial dependencies. For newly introduced nodes, the positional encoding requires training, which means HUTFormer cannot naturally perform inductive reasoning. Second, the training of HUTFormer is more complex than that of end-to-end methods. Although the two-phase training strategy is effective, it objectively makes the model's training process less convenient compared to end-to-end models. Therefore, exploring methods for end-to-end generation and utilization of multi-scale information is a promising avenue for future research. In addition, the reliability of longer-term forecasting is crucial. The longer the forecast period, the greater the uncertainty. Only by systematically quantifying these uncertainties and incorporating known external events into the model can more accurate and practically valuable results be achieved.

6. Conclusions

In this study, we make the first attempt to explore the long-term traffic forecasting problem. To this end, we reveal its unique challenges in exploiting the multi-scale representations of traffic data, and propose a novel Hierarchical U-Net TransFormer (HUTFormer) to efficiently and effectively address them. The HUTFormer mainly consists of a hierarchical encoder and decoder. On the one hand, the hierarchical encoder generates multi-scale representations based on the window self-attention mechanism and segment merging. On the other hand, the hierarchical decoder effectively utilizes the extracted multi-scale features based on the cross-scale attention mechanism. In addition, HUTFormer adopts segment embedding and spatial-temporal positional encoding as the input embedding strategy to address the complexity issue. Extensive experiments on four commonly used traffic datasets show that the proposed HUTFormer significantly outperforms state-of-the-art traffic forecasting and long-sequence time series forecasting baselines.

Replication and data sharing

The source code is available at https://drive.google.com/file/d/1GA_wFv71P3mk2OVpM-PPlNYBmX7f4Y4d/view. Follow the detailed instructions in the README. md (included later in the document) to set up the environment and data, and you will be able to train HUTFormer with ease.

CRediT authorship contribution statement

Zezhi Shao: Writing – review & editing, Software, Project administration, Investigation, Data curation, Writing – original draft, Validation, Visualization, Supervision, Resources, Methodology, Formal analysis, Conceptualization. **Fei Wang:** Funding acquisition, Methodology, Formal analysis. **Tao Sun:** Writing – review & editing. **Chengqing Yu:** Writing – review & editing. **Yuchen Fang:** Validation. **Guangyin Jin:** Formal analysis. **Zhulin An:** Writing – review & editing. **Yang Liu:** Writing – review & editing. **Xiaobo Qu:** Writing – original draft. **Yongjun Xu:** Funding acquisition, Writing – review & editing.

Conflict of interest statement

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.

Acknowledgements

This work is supported by National Natural Science Foundation of China (Grant Nos. 62502505 and 62372430), and the Youth Innovation Promotion Association, Chinese Academy of Sciences (Grant No.2023112).

References

Bai, L., Yao, L., Li, C., Wang, X., Wang, C., 2020. Adaptive graph convolutional recurrent network for traffic forecasting. In: Advances in Neural Information Processing Systems, pp. 17804–17815.
Belhadi, A., Djenouri, Y., Djenouri, D., Lin, J.C.-W., 2020. A recurrent neural network for urban long-term traffic flow forecasting. Appl. Intell. 50, 3252–3265.
Bogaerts, T., Masegosa, A.D., Angarita-Zapata, J.S., Onieva, E., Hellinckx, P., 2020. A graph CNN-LSTM neural network for short and long-term traffic forecasting based on trajectory data. Transport. Res. C Emerg. Technol. 112, 62–77.
Cao, D., Wang, Y., Duan, J., Zhang, C., Zhu, X., Huang, C., et al., 2020. Spectral temporal graph neural network for multivariate time-series forecasting. In: Advances in Neural Information Processing Systems, pp. 17766–17778.
Cao, H., Wang, Y., Chen, J., Jiang, D., Zhang, X., Tian, Q., et al., 2022. Swin-unet: unet-like pure transformer for medical image segmentation. In: Computer Vision - ECCV 2022 Workshops, pp. 205–218.
Chen, C., Petty, K., Skabardonis, A., Varaiya, P., Jia, Z., 2001. Freeway performance measurement system: mining loop detector data. Transp. Res. Rec. 1748, 96–102.
Cho, K., van Merrienboer, B., Bahdanau, D., Bengio, Y., 2014. On the properties of neural machine translation: encoder-decoder approaches. In: SSST@EMNLP 2014, pp. 103–111.
Chu, L., Hou, Z., Jiang, J., Yang, J., Zhang, Y., 2024. Spatial-temporal feature extraction and evaluation network for citywide traffic condition prediction. IEEE Trans Intell Veh 9, 5377–5391.
Chu, P., Zhang, J.A., Wang, X., Fang, G., Wang, D., 2019. Semi-persistent resource allocation based on traffic prediction for vehicular communications. IEEE Trans Intell Veh 5, 345–355.
Cirstea, R., Guo, C., Yang, B., Kieu, T., Dong, X., Pan, S., 2022. Triformer: triangular, variable-specific attentions for long sequence multivariate time series forecasting. In: Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, pp. 1994–2001.
Cui, Y., Xie, J., Zheng, K., 2021. Historical inertia: a neglected but powerful baseline for long sequence time-series forecasting. In: The 30th ACM International Conference on Information and Knowledge Management, pp. 2965–2969.
Defferrard, M., Bresson, X., Vandergheynst, P., 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In: Advances in Neural Information Processing Systems, pp. 3837–3845.
Deng, J., Chen, X., Jiang, R., Song, X., Tsang, I.W., 2021. St-norm: spatial and temporal normalization for multi-variate time series forecasting. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, pp. 269–278.
Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., et al., 2021. An image is worth 16x16 words: transformers for image recognition at scale. In: International Conference on Learning Representations, pp. 1–24.
Guo, S., Lin, Y., Feng, N., Song, C., Wan, H., 2019. Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In: The Thirty-First Innovative Applications of Artificial Intelligence Conference, pp. 922–929.
Guo, S., Lin, Y., Wan, H., Li, X., Cong, G., 2022. Learning dynamics and heterogeneity of spatial-temporal graph data for traffic forecasting. IEEE Trans. Knowl. Data Eng. 34, 5415–5428.
Guo, X., Zhang, Q., Jiang, J., Peng, M., Zhu, M., Yang, H.F., 2024. Towards explainable traffic flow prediction with large language models. Commun Transp Res 4, 100150.
Hamilton, W.L., Ying, Z., Leskovec, J., 2017. Inductive representation learning on large graphs. In: Advances in Neural Information Processing Systems, pp. 1024–1034.
Han, L., Du, B., Sun, L., Fu, Y., Lv, Y., Xiong, H., 2021. Dynamic and multi-faceted spatio-temporal deep learning for traffic speed forecasting. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, pp. 547–555.
He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778.
Huang, J., Xu, Y., Wang, Q., Wang, Q.C., Liang, X., Wang, F., et al., 2025. Foundation models and intelligent decision-making: progress, challenges, and perspectives. Innovation 6, 100948.
Jagadish, H.V., Gehrke, J., Labrinidis, A., Papakonstantinou, Y., Patel, J.M., Ramakrishnan, R., et al., 2014. Big data and its technical challenges. Commun. ACM 57, 86–94.
Jin, G., Li, F., Zhang, J., Wang, M., Huang, J., 2022. Automated dilated spatio-temporal synchronous graph modeling for traffic prediction. IEEE Trans. Intell. Transport. Syst. 24, 8820–8830.
Jin, G., Liang, Y., Fang, Y., Shao, Z., Huang, J., Zhang, J., et al., 2023. Spatio-temporal graph neural networks for predictive learning in urban computing: a survey. IEEE Trans. Knowl. Data Eng. 36, 5388–5408.
Kingma, D.P., Ba, J., 2015. Adam: a method for stochastic optimization. In: International Conference on Learning Representations, pp. 1–15.
Kipf, T.N., Welling, M., 2017. Semi-supervised classification with graph convolutional networks. In: International Conference on Learning Representations, pp. 1–14.
Kumar, S.V., Vanajakshi, L., 2015. Short-term traffic flow prediction using seasonal ARIMA model with limited input data. Eur Transp Res Rev 7, 1–9.
Li, F., Feng, J., Yan, H., Jin, G., Yang, F., Sun, F., et al., 2023. Dynamic graph convolutional recurrent network for traffic prediction: benchmark and solution. ACM Trans. Knowl. Discov. Data 17, 1–9, 21.
Li, Y., Bai, F., Lyu, C., Qu, X., Liu, Y., 2025. A systematic review of generative adversarial networks for traffic state prediction: overview, taxonomy, and future prospects. Inf 117, 102915.
Li, Y., Shao, Z., Xu, Y., Qiu, Q., Cao, Z., Wang, F., 2024. Dynamic frequency domain graph convolutional network for traffic forecasting. In: ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 5245–5249.
Li, Y., Yu, R., Shahabi, C., Liu, Y., 2018. Diffusion convolutional recurrent neural network: data-driven traffic forecasting. In: International Conference on Learning Representations, pp. 1–16.
Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S., 2017. Feature pyramid networks for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2117–2125.
Liu, S., Yu, H., Liao, C., Li, J., Lin, W., Liu, A.X., et al., 2022. Pyraformer: low-complexity pyramidal attention for long-range time series modeling and forecasting. In: International Conference on Learning Representations, pp. 1–20.
Liu, Y., Liu, Z., Jia, R., 2019. DeepPF: a deep learning based architecture for metro passenger flow prediction. Transport. Res. C Emerg. Technol. 101, 18–34.
Liu, Y., Lyu, C., Zhang, Y., Liu, Z., Yu, W., Qu, X., 2021a. DeepTSP: deep traffic state prediction model based on large-scale empirical data. Commun Transp Res 1, 100012.
Liu, Y., Wu, F., Liu, Z., Wang, K., Wang, F., Qu, X., 2023. Can language models be used for real-world urban-delivery route optimization? Innovation 4, 100520.
Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., et al., 2021b. Swin transformer: hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 10012–10022.
Lv, Y., Lv, Z., Cheng, Z., Zhu, Z., Rashidi, T.H., 2023. TS-STNN: spatial-Temporal neural network based on tree structure for traffic flow prediction. Transp. Res. Part E Logist Transp Rev 177, 103251.
van der Maaten, L., Hinton, G., 2008. Visualizing data using t-SNE. J. Mach. Learn. Res. 9, 2579–2605.
Nie, Y., Nguyen, N.H., Sinthong, P., Kalagnanam, J., 2023. A time series is worth 64 words: long-Term forecasting with transformers. In: International Conference on Learning Representations, pp. 1–24.
Pan, Z., Liang, Y., Wang, W., Yu, Y., Zheng, Y., Zhang, J., 2019. Urban traffic prediction from spatio-temporal data using deep Meta learning. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 1720–1730.
Park, C., Lee, C., Bahng, H., Tae, Y., Jin, S., Kim, K., et al., 2020. ST-GRAT: a novel spatio-temporal graph attention networks for accurately forecasting dynamically changing road speed. In: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, pp. 1215–1224.
Ronneberger, O., Fischer, P., Brox, T., 2015. U-net: convolutional networks for biomedical image segmentation. In: International Conference on Medical Image Computing and computer-assisted Intervention, pp. 234–241.
Shabani, M.A., Abdi, A.H., Meng, L., Sylvain, T., 2023. Scaleformer: iterative multi-scale refining transformers for time series forecasting. In: International Conference on Learning Representations, pp. 1–23.
Shang, C., Chen, J., Bi, J., 2021. Discrete graph structure learning for forecasting multiple time series. In: International Conference on Learning Representations, pp. 1–14.
Shao, Z., Li, Y., Wang, F., Yu, C., Fu, Y., Qian, T., et al., 2025a. BLAST: balanced sampling time series corpus for universal forecasting models. In: Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2, pp. 2502–2513.
Shao, Z., Qian, T., Sun, T., Wang, F., Xu, Y., 2025b. Spatial-temporal large models: a super hub linking multiple scientific areas with artificial intelligence. Innovation 6, 100763.
Shao, Z., Wang, F., Xu, Y., Wei, W., Yu, C., Zhang, Z., et al., 2025c. Exploring progress in multivariate time series forecasting: comprehensive benchmarking and heterogeneity analysis. IEEE Trans. Knowl. Data Eng. 37, 291–305.
Shao, Z., Xu, Y., Wei, W., Wang, F., Zhang, Z., Zhu, F., 2022a. Heterogeneous graph neural network with multi-view representation learning. IEEE Trans. Knowl. Data Eng. 35, 11476–11488.
Shao, Z., Zhang, Z., Wang, F., Wei, W., Xu, Y., 2022b. 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.
Shao, Z., Zhang, Z., Wang, F., Xu, Y., 2022c. Pre-training enhanced spatial-temporal graph neural network for multivariate time series forecasting. In: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 1567–1577.
Shao, Z., Zhang, Z., Wei, W., Wang, F., Xu, Y., Cao, X., et al., 2022d. Decoupled dynamic spatial-temporal graph neural network for traffic forecasting. Proc VLDB Endow 15, 2733–2746.
Shuman, D.I., Narang, S.K., Frossard, P., Ortega, A., Vandergheynst, P., 2013. The emerging field of signal processing on graphs: extending high-dimensional data

analysis to networks and other irregular domains. IEEE Signal Process. Mag. 30, 83–98.
Sutskever, I., Vinyals, O., Le, Q.V., 2014. Sequence to sequence learning with neural networks. In: Advances in Neural Information Processing Systems, pp. 3104–3112.
Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., et al., 2017. Attention is all you need. In: Advances in Neural Information Processing Systems, pp. 5998–6008.
Wang, F., Yao, D., Li, Y., Sun, T., Zhang, Z., 2023a. AI-enhanced spatial-temporal data-mining technology: new chance for next-generation urban computing. Innovation 4, 100405.
Wang, X., Ma, Y., Wang, Y., Jin, W., Wang, X., Tang, J., et al., 2020a. Traffic flow prediction via spatial temporal graph neural network. In: Proceedings of the Web Conference 2020, pp. 1082–1092.
Wang, Y., Shao, Z., Sun, T., Yu, C., Xu, Y., Wang, F., 2023b. Clustering-property matters: a cluster-aware network for large scale multivariate time series forecasting. In: Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, pp. 4340–4344.
Wang, Z., Su, X., Ding, Z., 2020b. Long-term traffic prediction based on lstm encoder-decoder architecture. IEEE Trans. Intell. Transport. Syst. 22, 6561–6571.
Wu, H., Xu, J., Wang, J., Long, M., 2021. Autoformer: decomposition transformers with auto-correlation for long-term series forecasting. In: Advances in Neural Information Processing Systems, pp. 22419–22430.
Wu, Z., Pan, S., Long, G., Jiang, J., Chang, X., Zhang, C., 2020. Connecting the dots: multivariate time series forecasting with graph neural networks. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 753–763.
Wu, Z., Pan, S., Long, G., Jiang, J., Zhang, C., 2019. Graph WaveNet for deep spatial-temporal graph modeling. In: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, pp. 1907–1913.
Xu, M., Di, Y., Ding, H., Zhu, Z., Chen, X., Yang, H., 2023. AGNP: network-Wide short-term probabilistic traffic speed prediction and imputation. Commun Transp Res 3, 100099.
Yu, B., Yin, H., Zhu, Z., 2018. Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting. In: IJCAI, pp. 3634–3640.
Yu, C., Wang, F., Shao, Z., Qian, T., Zhang, Z., Wei, W., et al., 2024. GinAR: an end-to-end multivariate time series forecasting model suitable for variable missing. In: Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 3989–4000.
Yu, C., Wang, F., Shao, Z., Sun, T., Wu, L., Xu, Y., 2023. DSformer: a double sampling transformer for multivariate time series long-term prediction. In: Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, pp. 3062–3072.
Yu, F., Koltun, V., 2016. Multi-scale context aggregation by dilated convolutions. In: International Conference on Learning Representations, pp. 1–13.
Zeng, A., Chen, M., Zhang, L., Xu, Q., 2023. Are transformers effective for time series forecasting?. In: Thirty-Seventh AAAI Conference on Artificial Intelligence, pp. 11121–11128.
Zhang, Y., Yan, J., 2023. Crossformer: transformer utilizing cross-dimension dependency for multivariate time series forecasting. In: International Conference on Learning Representations, pp. 1–21.
Zheng, C., Fan, X., Wang, C., Qi, J., 2020. Gman: a graph multi-attention network for traffic prediction. In: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 1234–1241.
Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., et al., 2021. Informer: beyond efficient transformer for long sequence time-series forecasting. In: Thirty-Fifth AAAI Conference on Artificial Intelligence, pp. 11106–11115.
Zhou, T., Ma, Z., Wen, Q., Wang, X., Sun, L., Jin, R., 2022. Fedformer: frequency enhanced decomposed transformer for long-term series forecasting. In: International Conference on Machine Learning, pp. 27268–27286.

Zezhi Shao is an Assistant Professor in the Institute of Computing Technology, Chinese Academy of Sciences. He received the Ph.D. degree from the Institute of Computing Technology, Chinese Academy of Sciences, China, in 2024. His research interests include traffic condition forecasting, multivariate time series forecasting, graph neural networks, and spatial-temporal data mining. He has published several papers as the first author in top journals and conferences such as KDD, VLDB, TKDE, CIKM.

Fei Wang received the Ph.D. degree in computer architecture from the Institute of Computing Technology, Chinese Academy of Sciences in 2017. From 2017 to 2020, he was a Research Assistant with the Institute of Technology, Chinese Academy of Sciences. Since 2020, he has been working as an Associate Professor in the Institute of Computing Technology, Chinese Academy of Sciences. His main research interest includes spatiotemporal data mining, Information fusion, graph neural networks. He has published several papers in top journals and conferences, such as KDD, VLDB, ICDE, TKDE, and ACM MM.

Tao Sun is an Assistant Professor in the Institute of Computing Technology, Chinese Academy of Sciences. He received the Ph. D. degree in computer architecture from the Institute of Computing Technology, Chinese Academy of Sciences in 2022. His research interest falls in the area of spatial-temporal data mining and trajectory data analysis. He has published several papers in journals and conferences, such as CIKM, DASFAA, ICPR, and IJCNN.

Chengqing Yu received the B.S. degree in transport equipment and control engineering from Central South University, Changsha, China, in 2019, and the M.S. degree in traffic and transportation engineering from Central South University, Changsha, China, in 2022. He is currently working toward the Ph.D. degree with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China. His main research interests include deep learning, reinforcement learning, and time series forecasting.

Yuchen Fang is currently pursuing the Ph.D. degree at the University of Electronic Science and Technology of China. His general research interests are in spatio-temporal data mining, graph neural networks, and urban computing, with a special focus on traffic forecasting. He has published several papers in top journals and conference proceedings, such as SIGKDD, ICDE, SIGIR, AAAI, and TITS.

Guangyin Jin is an Visiting Scholar in the Department of Planning, Design, and Technology of Architecture at Sapienza University of Rome. His research interest falls in the area of spatial-temporal data mining, graph neural networks and urban computing. So far, he has published more than ten papers in JCR Q1-level international journals such as TITS, TIST, TRC, INS, and top international conferences such as AAAI, CIKM, SIGSPATIL. He also serves as the PC member or reviewer for top international conferences or journals such as AAAI, Ww, ECML-PKDD, TITS, TKDD, and TRC.

Zhulin An is currently with the Institute of Computing Technology, Chinese Academy of Sciences, where he became an Associate Researcher in 2014. He received the B.E. and M.S. degrees in computer science from the Hefei University of Technology, Hefei, China, in 2003 and 2006, respectively, and the Ph.D. degree from the Chinese Academy of Sciences, Beijing, China, in 2010. His current research interests include deep neural network acceleration and continual learning.

Yang Liu earned the Ph.D. degree at Southeast University, Nanjing, China, in 2021. Currently, he is an Associate Research Fellow at the School of Vehicle and Mobility, Tsinghua University, China. His research interests include intelligent transportation systems, artificial intelligence techniques, and data mining, with applications in complex real-world problems such as autonomous vehicles and urban mobility. He is a recipient of the National High-Level Young Talents Program and a Marie Curie Fellow of the European Union. He serves as an Editorial Board Member of *Transportation Research Part E* and an Associate Editor of *IEEE Transactions on Intelligent Vehicles*.

Xiaobo Qu is a Professor in intelligent transportation at Tsinghua University, China. His research is focused on improving large, complex, and interrelated urban mobility systems by integrating with emerging technologies. He has published over 180 journal articles published at top tier journals in the area of transportation. He is an elected member of Academia Europaea-The Academy of Europe since 2020.

Yongjun Xu is a Professor at Institute of Computing Technology, Chinese Academy of Sciences (ICT-CAS) in Beijing, China. He received the B.Eng. and Ph.D. degree in computer communication from Xi'an Institute of Posts & Telecoms (China) in 2001 and Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China in 2006, respectively. His current research interests include artificial intelligence systems, and big data processing.