

# CCDSReFormer: Traffic Flow Prediction with a Criss-Crossed Dual-Stream Enhanced Rectified Transformer Model

Zhiqi Shao, Michael G.H. Bell, Ze Wang, D. Glenn Geers, Xusheng Yao, and Junbin Gao

**Abstract**—Accurate, effective, and rapid traffic forecasting is crucial for intelligent traffic systems, playing a significant role in urban traffic planning, management, and control. Despite the effectiveness of existing Spatio-Temporal Transformer models in predicting traffic flow, they face challenges in balancing computational efficiency with accuracy, a tendency to favor global over local time series information, and a segregated approach to spatial and temporal data, which limits understanding of complex spatio-temporal interactions. To address these limitations, we introduce the Criss-Crossed Dual-Stream Enhanced Rectified Transformer model (CCDSReFormer), featuring three novel modules: Enhanced Rectified Spatial Self-attention (ReSSA), Enhanced Rectified Delay Aware Self-attention (ReDASA), and Enhanced Rectified Temporal Self-attention (ReTSA). These modules collectively reduce computational demands through sparse matrix attention, prioritize local information for a comprehensive capture of traffic dynamics, and integrate spatial and temporal data through a criss-crossed learning approach. Our extensive experiments on six real-world datasets demonstrate the superior performance of CCDSReFormer. Additionally, an ablation study was conducted to evaluate the contribution of each component to the model’s predictive accuracy. The results confirm the effectiveness of our proposed model in accurately forecasting traffic flow.

**Index Terms**—Transformer; Traffic Flow; Prediction; Spatio-Temporal Data; Time-Series Prediction

## I. INTRODUCTION

At the core of advancing intelligent transportation systems (ITS) is the intricate challenge of traffic flow prediction, crucial for enhancing route efficiency, mitigating congestion, and reducing accident rates in urban settings. The rapid evolution of data acquisition technologies, coupled with the exponential increase in vehicle numbers, has facilitated a more comprehensive collection of traffic data, propelling traffic flow prediction to the forefront of urban transportation management research [1].

The quintessential hurdle in this domain is the accurate modeling and understanding of spatiotemporal dependencies within traffic data. These dependencies, reflecting the dynamic nature of traffic flow, involve complex patterns of spatial

Zhiqi Shao and Junbin Gao are with the Business Analytics Discipline, The University of Sydney, Camperdown, NSW 2006, Australia. (e-mail: zhiqi.shao@sydney.edu.au; junbin.gao@sydney.edu.au)

Michael G.H. Bell, Ze Wang, and D. Glenn Geers are with the Institute of Transport and Logistics, The University of Sydney, Camperdown, NSW 2006, Australia. (e-mail: michael.bell@sydney.edu.au; ze.wang@sydney.edu.au; glenn.geers@sydney.edu.au)

Xusheng Yao (Correspond Author) is with the College of Management and Economics, Tianjin University, Tianjin University, 300072, China. (e-mail: xsyao@tju.edu.cn)

and temporal interplay that are constantly evolving. Capturing these patterns is fundamental for accurate prediction and poses a significant computational challenge, necessitating a delicate balance between prediction accuracy and computational efficiency [2], [3].

Traditional methodologies, such as autoregressive integrated moving average (ARIMA) models [4], [5], [6], [7] and Kalman filters [8], [9], [10], [11], [12], have made strides in traffic prediction. Yet, their effectiveness is often limited by inherent assumptions of data stationarity and their inadequacy in capturing the complex spatiotemporal dependencies characteristic of traffic flow. Spatial regression methods [13], while introducing spatial awareness, remain constrained by the dynamic intricacies of traffic patterns and the computational burdens they impose.

With the increasing computational power, deep learning has become the leading methodology in traffic flow prediction, excelling in handling the complex and nonlinear patterns of traffic data. In the past, emerging studies have focused on employing recurrent neural networks (RNNs) and its variants, such as long short-term memory (LSTM) [14], [15], [16], which are particularly suited to grid-based data [17], [18]. These methods are crucial in modeling temporal aspects of traffic. Convolutional Neural Networks (CNNs), which are also suitable for grid data, have been widely used due to their efficacy in extracting temporal features [19], [20], [21].

More recently, graph convolutional networks (GCNs), aligning with the graph-structured nature of traffic data, have shown promise in identifying intricate spatial characteristics and dependencies in road networks [22], [23], [24], [25]. However, GNNs in traffic prediction encounter specific challenges: they struggle with dynamic spatial dependencies that change over time, have a limited capacity for long-range dependency modeling due to local interaction focus, and often overlook the time-delay impact in spatial information transfer [18]. While the attention mechanism has demonstrated effectiveness in capturing both spatial and temporal information by adapting to the input changes [2], [26], [27], there remain some limitations such as:

- 1) Lack of integration of both spatial and temporal information for learning the complex dynamics;
- 2) Less adept at making short-term predictions (or lack of local feature focus); and
- 3) Their use of dense matrices may cause high computational cost.

To fill these gaps, we propose a state-of-the-art technique call the Criss-Cross Dual-Stream Enhanced Rectified Transformer (**CCDSReFormer**) model which consists of a novel Criss-Crossed Dual-Stream for enriched spatial and temporal learning, an Enhanced Convolution (**EnCov**) method focusing on local traffic pattern nuances, and a rectified linear self attention (**ReLSA**) mechanism for dynamic and computationally efficient attention allocation in traffic flow prediction. We show that our new method out-performs extant techniques used for analyzing spatial and temporal data in traffic networks.

Our main contributions are summarized as follows:

- Our model introduces a criss-crossed dual stream, enabling simultaneous learning of spatial and temporal information to enhance performance. This dual approach effectively captures the complexities of traffic flow, offering a thorough understanding of spatial and temporal dynamics.
- We design a locally enhanced convolution within our attention mechanism, to make the model focus on local spatio-temporal features and capture the nuanced dynamics of traffic patterns influenced by localized conditions.
- To address the limitations of traditional softmax-based attention, our model is the first to employ Rectified Linear Self Attention (**ReLSA**) [28] in the traffic flow prediction field, offering a dynamic, adaptable approach that responds to the unique spatial-temporal characteristics of traffic data while reducing computational complexity.
- We conducted extensive experiments across six real-world datasets, demonstrating that our model outperforms the existing state-of-the-art in both performance and computational efficiency, with robust parameter tuning capabilities.

The rest of this paper is structured as follows: Section II provides a summary of previous related studies. Section III outlines the relevant definitions and problem statement. The proposed **CCDSReFormer** model is elaborated in Section IV. Section V discusses the experiments conducted and their results. The paper concludes with a summary and future research directions in Section VII.

## II. LITERATURE REVIEW

In previous studies, parameter-based approaches such as ARIMA models [4], [5], [6], [7], Kalman filters [8], [9], [10], [11], [12], and other regression techniques [29], [30] have been used for traffic flow prediction. However, they must be trained on large data sets to achieve higher precision and they can only extract the time features from traffic flow data effectively ignoring the spatial information which is important for predicting traffic flow [9]. Non-parametric machine learning methods such as artificial neural networks (ANNs) [31], [5], [32], [33], k-nearest neighbors (KNN) [15], [34], [35], [36] and support vector machines (SVM) [34] have also been used to predict traffic flow, with varying degrees of success.

Deep learning models have notably improved traffic prediction methodologies. This section reviews various methods that utilize deep learning for traffic prediction. Generally, applications of deep learning in this domain can be classified

into four major categories RNNs, CNNs, GNNs and the attention mechanism, each with its unique advantages.

RNNs including LSTM are commonly applied to sequence data because of their memorization capability, which can learn both long and short-term dependencies between parts of a data sequence. Previously they have been used for traffic flow prediction as exemplified in [14], [15], [16], [37], [38]. These papers demonstrated the ability of these methods to capture the temporal patterns in traffic data. However, by their ability to take in very long data sequences, these methods suffer from the vanishing gradient problem [39].

CNNs have shown proficiency in processing grid-based spatial data, particularly in capturing spatial dependencies within traffic data, making them a go-to choice for early traffic prediction tasks as evidenced by studies such as [19], [40], [41]. More recently, [42], [43] proposed a hybrid model that combines CNNs for spatial feature extraction and gated recurrent units (GRUs) for temporal feature analysis in traffic flow data. The research in [20] further harnesses CNNs, employing a spatio-temporal feature selection algorithm to optimize input data and extract pivotal traffic features, enhancing the predictive model.

Recently, the spotlight has turned to Graph Neural Networks (GNNs) for their adeptness in handling traffic data, effectively represented as graphs. Unlike the rigid structure of CNNs, GNNs embrace a flexible, graph-based approach suitable for the complex, non-Euclidean topology of traffic networks [44]. They've been increasingly utilized for traffic flow prediction, with innovations such as integrating learnable positional attention in GNNs for capturing spatial-temporal patterns [45], the Progressive Graph Convolutional Network (PGCN) enhancing adaptability to both training and testing phases [46], and models like the Dynamics Extractor and node connection strength index enhancing traffic flow characteristics analysis [47]. Other notable developments include DCRNN's use of bidirectional random walks for complex spatial-temporal dynamics [48], STGCN's efficient convolutional graph structures [49], GWNET's adaptive graph modeling [50], and MTGNN's novel graph learning techniques [51]. Additionally, STFGNN and STGCNDE offer advanced spatio-temporal forecasting by integrating differential equations with neural networks [52], [53]. The paper [54] uses an optimized GCN to preserve the spatial structure of road networks through a graph representation. AST-InceptionNet introduces multi-scale feature extraction and adaptive graph convolutions to address spatial heterogeneity and unknown adjacencies [55]. Despite their advancements, these models confront challenges with dynamic spatial dependencies, long-range dependency modeling, over-smoothing, and time-delay impacts, indicating ongoing areas for refinement.

Therefore, the growing adoption of attention mechanisms in traffic forecasting is becoming increasingly prevalent. The self-attention mechanism developed by [56], has significantly impacted data processing for complex tasks, including language translation, image recognition, and sequence prediction, by dynamically focusing on the most pertinent input data sections. This mechanism is increasingly acknowledged for its ability to concurrently process spatial and temporal information. In

traffic forecasting, models like the spatial-temporal transformer network model (STTN)[57] uses spatial transformers and long-range temporal dependencies that dynamically model directed spatial dependencies, a graph multi-attention network model (GMAN)[58] utilizing an encoder-decoder architecture with spatio-temporal attention blocks to model the impact of spatio-temporal factors on traffic conditions, and featuring an attention layer that effectively links historical and future time steps for long-term traffic prediction, an attention-based spatial-temporal graph neural network (ASTGNN)[59] integrating a graph convolutional recurrent module (GCRN) with a global attention module, designed to effectively model both long-term and short-term temporal correlations in traffic data, and propagation delay-aware dynamic long-range transformer for traffic flow prediction (PDFormer)[2] design multi-self-attention modules to capture the dynamic relations, a progressive space-time self-attention (ProSTformer)[21] focuses on spatial dependencies from local to global regions and a novel traffic flow prediction approach combining Vision Transformers (VTs) and Convolutional Neural Networks (CNN)[60] is used to accurately forecast urban congestion. These models collectively demonstrate the growing trend of incorporating advanced attention mechanisms to improve the accuracy and efficiency of spatio-temporal traffic forecasting. However, the complexity of these attention mechanisms can lead to increased computational costs, presenting a challenge that needs to be addressed. Finding a balance between computational efficiency and model accuracy remains a critical issue to resolve.

In summary, traffic flow prediction has progressed from traditional parameter-based methods to sophisticated deep learning approaches, offering a nuanced understanding of the complex spatio-temporal dynamics within transportation networks. While RNNs, CNNs, GNNs, and attention mechanisms have improved prediction accuracy, they continue to face challenges in balancing computational cost and efficiency. Additionally, achieving an optimal mix of local and global information is pivotal, as long-term forecasts tend to prioritize broader trends, potentially overlooking critical local insights. Addressing these issues is key to refining accuracy and computational practicality, steering future research toward more adept and streamlined forecasting techniques.

### III. PRELIMINARIES

In this section, we provide background information and introduce fundamental ideas that are pertinent to the traffic flow prediction scheme presented in Section III-A and the vanilla attention mechanism discussed in Section III-B.

#### A. Essential Notation and Definitions

To better understand our work, we first present several preliminaries and define our problem formally in this subsection.

**Definition 1 (Graphical Representation of a Road Network):** A *Road Network* is modeled as a graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A})$ , where  $\mathcal{V} = \{v_1, \dots, v_N\}$  is a set of  $N (= |\mathcal{V}|)$  nodes composed from points of interest in the network such as traffic sensors, intersections, or subway stations,  $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$  is a set of edges

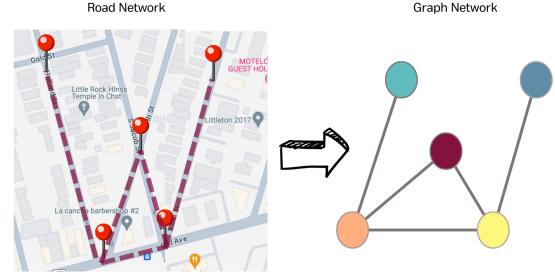


Fig. 1. The graph network is structured to reflect the local road network topology. The figure on the left illustrates the geographical locations of a subset of traffic sensor nodes as marked on the chosen benchmark dataset. On the right, the corresponding graph network is depicted, composed of these sensor nodes interconnected according to their locations and the road network links they monitor.

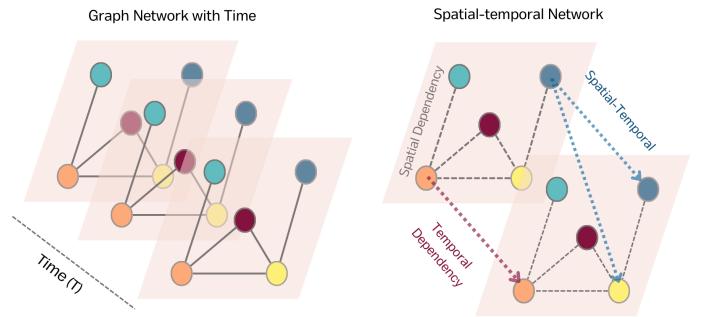


Fig. 2. The figure on the left represents the structure of graph data, and that on the right illustrates the spatio-temporal network of the blue node. Grey lines represent the network's connections, showing spatial influence. The red arrow reflects the self-influence of the blue node over time, while the blue arrows demonstrate the combined spatial and temporal influence between the blue node and its neighboring nodes at the subsequent time step. It is similar applies to the grid data.

typically determined by roads that connect the nodes, and  $\mathcal{A}$  signifies the (square) adjacency matrix associated with the network graph.

Figure 1 illustrates an example of a traffic network depicted as a graph network. Although the graph is not changing, the node information may change during the time when data is collected at the sensor nodes. Thus, the network can be regarded as a spatio-temporal graph network as shown in Figure 2, showcasing the dynamic nature of traffic data over time. This gives rise to the following:

**Definition 2 (Tensor Representation of Traffic Flow):** Given a road network  $\mathcal{G}$  with  $N$  nodes, the traffic flow information on the road network at timestamp  $t$  is denoted by  $\mathbf{X}_t \in \mathbb{R}^{N \times d}$  where  $d$  is the dimensionality of the flow features. For any given  $T > 0$ , the overall traffic flow can be represented by a rank-3 tensor, denoted by

$$\mathbf{X} = (\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_T) \in \mathbb{R}^{T \times N \times d},$$

where the first mode (or dimension) of the tensor corresponds to the timestamps, the second mode to the network nodes and the third mode corresponds to the flow channels (or dimensions).

Given a traffic flow tensor  $\mathbf{X} \in \mathbb{R}^{T \times N \times d}$  over a road network  $\mathcal{G}$ , the  $t$ -slice over the first mode is denoted by  $\mathbf{X}_{t,:} \in \mathbb{R}^{N \times d}$  which contains the spatial information for all nodes at time  $t$ , and the  $n$ -slice over the second mode is denoted by  $\mathbf{X}_{:,n} \in \mathbb{R}^{T \times d}$  which is the temporal data at node  $n$  for all timestamps  $[1, \dots, T]$ .

a) *Problem Statement:* The objective of traffic flow forecasting is to predict future traffic flow based on historical flow data. Specifically, for a given section of traffic flow tensor  $\mathbf{X}$  recorded on the roadnetwork  $\mathcal{G}$ , the aim is to compute a function  $f$  that takes traffic flow values from the past  $m$  timestamps as input and outputs an estimate of the traffic flow for the following  $n$  timestamps:

$$f([\mathbf{X}_{t-h+1}, \dots, \mathbf{X}_t]; \mathcal{G}) \mapsto [\mathbf{X}_{t+1}, \dots, \mathbf{X}_{t+h'}]$$

for any given initial timestamp  $t$ .

### B. The Attention Mechanism

In order to introduce the new attention modules used in our proposed model we first describe the attention mechanism in detail by explaining scaled dot-product or ‘vanilla’ self-attention [56].

a) *Vanilla Self-Attention:* The principle of the attention mechanism, originating in natural language processing (NLP) [61], is to map query terms to a weighted sum of known keys and output the best matched values. A commonly used attention mechanism in various machine learning models is the so-called self-attention framework.

The primary objective of self-attention is to seek a more information-rich representation  $\mathbf{Y}$  of a given set of feature data  $\bar{\mathbf{X}} \in \mathbb{R}^{N \times d}$  by exploiting so-called ‘self-attention’ information. To realize this goal, we first construct three components: query ( $\mathbf{Q}$ ), key ( $\mathbf{K}$ ), and value ( $\mathbf{V}$ ), from  $\bar{\mathbf{X}}$  by,

$$\mathbf{Q} = \bar{\mathbf{X}}\mathbf{W}_Q, \quad \mathbf{K} = \bar{\mathbf{X}}\mathbf{W}_K, \quad \mathbf{V} = \bar{\mathbf{X}}\mathbf{W}_V \quad (1)$$

where  $\mathbf{W}_Q \in \mathbb{R}^{N \times d_q}$ ,  $\mathbf{W}_K \in \mathbb{R}^{N \times d_k}$ , and  $\mathbf{W}_V \in \mathbb{R}^{N \times d_v}$  are weight matrices that must be learned from the input data; and  $d_q$ ,  $d_k$  and  $d_v$  are the dimension of the query, key and value vectors respectively. For convenience it is usual to set  $d_q = d_k = d_v$ .

The new representation  $\mathbf{Y}$  is calculated in the following way based on the attention mechanism principle. Once we have  $\mathbf{Q} \in \mathbb{R}^{N \times d_q}$ ,  $\mathbf{K} \in \mathbb{R}^{N \times d_k}$ , and  $\mathbf{V} \in \mathbb{R}^{N \times d_v}$  according to (1), we are able to determine how the features in  $\bar{\mathbf{X}}$  pay influence or “attention” to all other features through the query features  $\mathbf{Q}$  and key features  $\mathbf{K}$  by the following scaled attention score  $\mathbf{A}$  given by

$$\mathbf{A} = \frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}} \in \mathbb{R}^{N \times N}, \quad (2)$$

where we have assumed  $d_k = d_q$ . The attention score is scaled by the square root of the dimension of the key vectors to stabilize numerical calculations during the training process.

Clearly the  $ij$ th element of  $\mathbf{A}$  is the scaled dot-product of the  $i$ th and  $j$ th feature of  $\bar{\mathbf{X}}$  because they correspond to the query and key vectors respectively. Thus the  $i$ th row of  $\bar{\mathbf{X}}$  contains the attention score of the  $i$ th feature with respect to all other features including itself.

The attention score matrix is then passed through a softmax function (a generalization of the logistic function to dimensions greater than one) to obtain the attention weight  $\mathbf{O}$ , in which each element is positive and every row sums to unity,

$$\mathbf{O} = \text{softmax}(\mathbf{A}). \quad (3)$$

The new representation  $\mathbf{Y}$  is then computed by,

$$\mathbf{Y} = \mathbf{OV}, \quad (4)$$

where the dimensions of  $\mathbf{Y}$  are the same as the dimensions of the value matrix  $\mathbf{V}$ .

We call the implementation (1) – (4) one-head attention, offering one way of exploration. In multi-head attention, we can utilize multiple sets (or “heads”) of  $\mathbf{Q}$ ,  $\mathbf{K}$ , and  $\mathbf{V}$  by learning different weight matrices  $\mathbf{W}_Q$ ,  $\mathbf{W}_K$ ,  $\mathbf{W}_V$ , producing multiple outputs  $\mathbf{Y}$  that can be concatenated and linearly transformed to produce the final output.

*Remark 1:* In this paper we set  $d_q = d_k = d_v = d_0$  for a given  $d_0$ .

## IV. THE METHOD

In this section, we introduce the proposed **CCDSReFormer** in detail. The framework of **CCDSReFormer** is shown as Figure 3.

### A. Data Embedding Layer

Similar to the study [2], the data embedding layer morphs the input into a higher-dimensional representation,  $\mathbf{X}_{\text{data}} \in \mathbb{R}^{T \times N \times d}$ , via a fully connected layer where  $d$  is the embedding dimension.

To further explore and incorporate spatio-temporal network dynamics, the output of the data embedding layer is used as input for computing a spatial embedding of the road network structure as described below.

In similar vein, a temporal periodic embedding is used to capture recurrent variations in traffic flow such as the morning and afternoon rush hours.

a) *Spatial Embedding:* For each timestamp  $t$ , the spatial embedding  $\mathbf{X}_{t,\text{spe}} \in \mathbb{R}^{N \times d}$  is derived from the graph Laplacian eigenvectors. The process begins with the computation of the normalized Laplacian matrix  $\Delta$  which is calculated using the adjacency matrix  $\mathcal{A}$  and the diagonal degree matrix  $\mathbf{D}$  where  $\mathbf{D}(i,i) = \sum_{j=1}^N \mathcal{A}(i,j)$ . The symmetrically normalized Laplacian matrix is given by  $\Delta = \mathbf{I} - \mathbf{D}^{-1/2} \mathcal{A} \mathbf{D}^{-1/2}$ , where  $\mathbf{I}$  is the identity matrix of appropriate dimension. Secondly an eigenvalue decomposition of  $\Delta$  is performed, resulting in  $\Delta = \mathbf{U}^\top \Lambda \mathbf{U}$  where  $\Lambda = \text{diag}(\lambda_0, \lambda_1, \dots, \lambda_{N-1})$  is the ordered matrix of eigenvalues, satisfying  $0 = \lambda_0 \leq \lambda_1 \leq \dots \leq \lambda_{N-1}$ , and  $\mathbf{U} = (u_0, u_1, \dots, u_{N-1})$  is the corresponding matrix of eigenvectors. We then select the  $k$  eigenvectors from  $\mathbf{U}$  corresponding to the  $k$  smallest nontrivial eigenvalues to construct the  $k$ -dimensional embedding for all nodes  $V$  at time  $t$ , denoted by  $\mathbf{U}_{t,\text{spe}} = [u_1, u_2, \dots, u_k] \in \mathbb{R}^{N \times k}$ . In the following the parameter  $k = 8$  as used in the baseline model [2]. Subsequently,  $\mathbf{U}_{t,\text{spe}}$  is subjected to a linear transformation, mapping it into a new  $d$ -dimensional space. This process culminates in the formation of the spatial graph

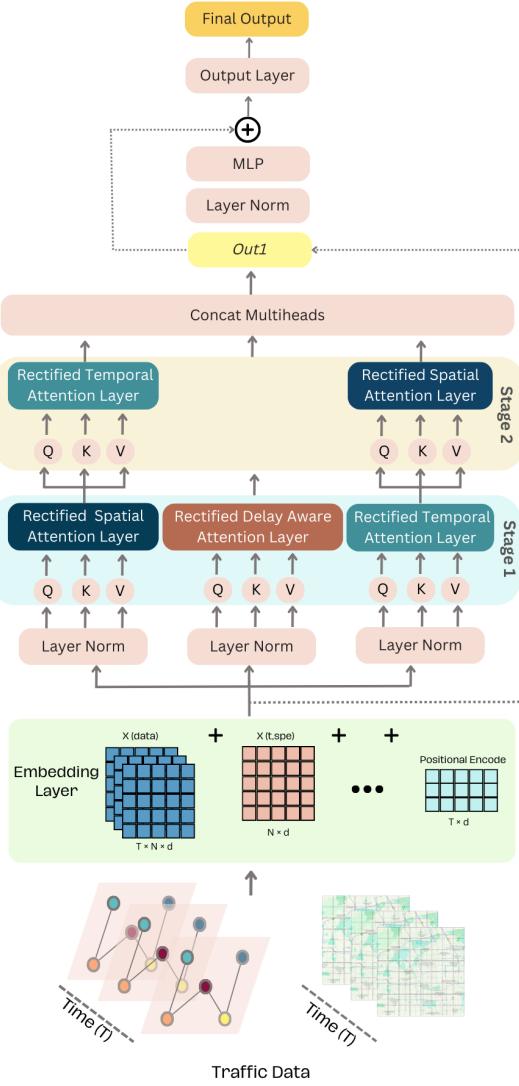


Fig. 3. The figure shows the process of **CCDSReFormer** which begins with the input traffic data being directed to the embedding layers. Subsequently, a layer normalization (Lay Norm) is applied. Following this, the model engages in various attention mechanisms including **ReSSA**, **ReTSA**, and **ReDASA**, facilitating cross-learning between **ReSSA**, **ReTSA** to assimilate information from various sources. The subsequent concatenation of these results yields the initial output, denoted as  $Out_1$ . This is followed by another layer normalization and a multi-layer perceptron (MLP) process. The output thus obtained is combined with the solution by concatenating the output from the data embedding layer and  $Out_1$ . Finally, this composite output is fed into the output layer to generate the final output.

Laplacian embedding  $\mathbf{X}_{t,\text{spe}} \in \mathbb{R}^{N \times d}$  at time  $t$ , effectively embedding the graph in a Euclidean space and thus preserving its global structure [2].

*b) Temporal Embedding:* Each timestamp  $t$ , can be converted to either a weekly index  $w(t)$  or a daily index  $d(t)$ . Concretely, the weekly index  $w(t)$  translates the timestamp  $t$  into a day-of-week representation (1 to 7), while the daily index  $d(t)$  maps it to the specific minute of the day (1 to 1440). All the indices are then converted into trainable temporal embeddings by a way of the embedding layers. The weekly and daily embeddings for all the timestamps are denoted by  $\mathbf{X}_w \in \mathbb{R}^{T \times d}$  and  $\mathbf{X}_d \in \mathbb{R}^{T \times d}$ , respectively, where  $d$  is the

same as the spatial embedding dimension.

*c) Temporal Positional Encoding:* For generating the temporal positional encoding  $\mathbf{X}_{t,\text{pe}}$ , we draw inspiration from the study by Vaswani et al. (2017) [56]. In the context of traffic network prediction, it's crucial to account for another dynamic attribute: the temporal position relative to the input. To address this, we define the temporal input positions as  $t = \{1, 2, \dots, T\}$ , representing the sequence of timestamps. To capture this temporal aspect effectively, we define the temporal positional encoding as  $\mathbf{X}_{t,\text{pe}}$  for each timestamp  $t$  as a  $d$ -dimensional vector in  $\mathbb{R}^d$ . The components of  $\mathbf{X}_{t,\text{pe}}$  are computed as follows:

$$\mathbf{X}_{t,\text{pe}}(i) = \begin{cases} \sin\left(\frac{t}{10000^{2i/d}}\right) & \text{if } i \text{ is even,} \\ \cos\left(\frac{t}{10000^{2(i-1)/d}}\right) & \text{if } i \text{ is odd.} \end{cases}$$

Finally we use  $\mathbf{X}_{\text{pe}} \in \mathbb{R}^{T \times d}$  to collect all the temporal positional encoding.

This ensures that each dimension of the positional encoding varies according to a sinusoidal function of a different wavelength, providing a unique and continuous encoding for each time step  $t$ . Such an encoding is instrumental in enabling the model to capture the nuances of the temporal dynamics inherent in road network data.

*d) Final Output:* The final output from the data embedding layer, represented as  $\mathbf{X}_{\text{emb}}$ , is simply the element-wise sum of the various components:

$$\mathbf{X}_{\text{emb}} = \mathbf{X}_{\text{data}} \oplus_1 \mathbf{X}_{\text{spe}} \oplus_2 \mathbf{X}_w \oplus_2 \mathbf{X}_d \oplus_2 \mathbf{X}_{\text{pe}},$$

where  $\oplus_k$  denotes the broadcasting summation along the  $k$ th mode to ensure dimensional consistency. This concept of broadcasting summation is derived from the functionality provided by Python's NumPy library<sup>1</sup>. The resulting  $\mathbf{X}_{\text{emb}}$  is then passed to the spatio-temporal encoders. To simplify the notation the subscript emb will be dropped in the following sections, i.e.,  $\mathbf{X} \equiv \mathbf{X}_{\text{emb}}$ .

#### B. Workflow of Enhanced Rectified Linear Self-Attention

To provide a clear and logical introduction to the **CCDSReFormer**, we begin with an overview of our newly designed attention which is named Enhanced Rectified Linear Self-Attention (**EnReLSA**), as depicted in Figure 4. Understanding this concept is crucial for grasping the spatial, temporal, and delay-aware modules that are central to the architecture of **CCDSReFormer**.

The **EnReLSA** approach adapts the standard self-attention mechanism by initially determining the query, key, and value components through specific matrix operations. It modifies these matrices with learnable parameters to tailor the model's focus. Further, to enhance the handling of the computational demand posed by the attention mechanism, **EnReLSA** incorporates a summation of Rectified Self-Attention (**ReLSA**) and enhanced convolution **EnCov**.

The (**ReLSA**) approach introduces sparsity into the attention matrix, effectively reducing its complexity. Specifically,

<sup>1</sup>Broadcasting summation refers to the capability in Python's NumPy library to perform element-wise binary operations on arrays of different sizes

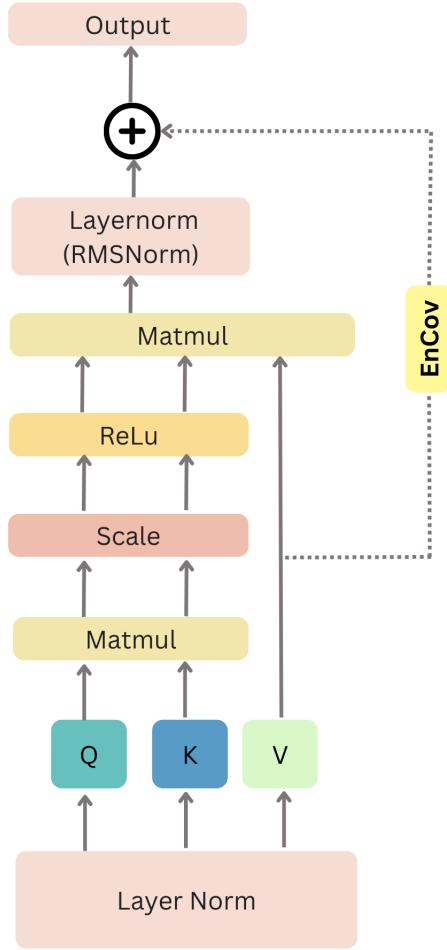


Fig. 4. The figure shows the proposed the Enhance Rectified Linear Self Attention(ReLSA) with **EnCov** workflow.

**ReLSA** employs a rectified linear unit (ReLU) for selecting positive values. Additionally, to further stabilize the attention distribution, **ReLSA** leverages a layer normalization technique known as **RMSNorm**. Then, an enhanced convolutional step (**EnCov**) is designed to enhance the localizes attention, allowing the model to hone in on adjacent features, which enhances its ability to discern dynamic traits in the data. Thus, the final out of **EnReLSA** is able to capture both local and global dependencies in the data.

The subsequent sections of the text promise a deeper exploration of the components of the **CCDSReFormer**, offering the workings of **EnReLSA** with different inputs and its contribution to the model's performance.

### C. Rectified Spatial Self-Attention Module (**ReSSA**)

Rectified spatial self-attention (**ReSSA**) is used to capture the dynamic spatial dependence of traffic data at reduced computational cost. As previously introduced the attention mechanisms in Section III-B, at each time  $t$ , the query, key, and value of self-attention in the rectified spatial self-attention module can be written as, referring from Figure 4,

$$\mathbf{Q}_t^{(S)} = \mathbf{X}_{t::} \cdot \mathbf{W}_Q^S, \mathbf{K}_t^{(S)} = \mathbf{X}_{t::} \cdot \mathbf{W}_K^S \text{ and } \mathbf{V}_t^{(S)} = \mathbf{X}_{t::} \cdot \mathbf{W}_V^S$$

where  $\mathbf{W}_Q^S$ ,  $\mathbf{W}_K^S$  and  $\mathbf{W}_V^S \in \mathbb{R}^{d \times d_0}$  are learnable parameters and  $d_0$  is the dimension of the query, key, and value matrix in this work. Then, we apply self-attention operations in the spatial dimension to model the interactions between nodes and obtain the spatial dependencies (attention scores) among all nodes at time  $t$  as:

$$\mathbf{A}_t^{(S)} = \frac{(\mathbf{Q}_t^{(S)}) (\mathbf{K}_t^{(S)})^\top}{\sqrt{d_0}}. \quad (5)$$

It can be seen that the spatial dependencies  $\mathbf{A}_t^{(S)} \in \mathbb{R}^{N \times N}$  between nodes are different in different time slices, i.e., dynamic. This variability necessitates a self-attention mechanism capable of adapting to these dynamic spatial dependencies. Traditional self-attention models, which assume fully connected graphs, do not always align with the more complex relationships found in real-world scenarios. In particular, the interactions between nodes that are geographically proximate or share certain functional similarity, regardless of their physical distance, are crucial.

To address this issue, our model incorporates a binary geographic masking matrix  $\mathbf{M}_{\text{geo}}$ , as proposed in study [2]. This matrix is specifically designed to account for geographic contiguity, giving priority to nodes within a predefined distance threshold. Such a focus allows for a more nuanced representation of spatial dependencies that are geographically grounded.

Additionally, to enhance computational efficiency, our model integrates the concept of rectified attention as detailed in the work [28]. The introduction of the Rectified Linear Spatial Self-Attention module (ReLSA) circumvents the re-centering constraint, resulting in a methodology that is not only more flexible but also computationally less demanding.

The final output of the ReLSA module is obtained by multiplying the attention scores with the value matrix. The formulation for this output is as follows:

$$\begin{aligned} \text{ReLSA}_{\text{geo}}(\mathbf{Q}_t^{(S)}, \mathbf{K}_t^{(S)}, \mathbf{V}_t^{(S)}) \\ = \text{LN}(\text{ReLU}(\mathbf{A}_t^{(S)} \odot \mathbf{M}_{\text{geo}}) \mathbf{V}_t^{(S)}), \end{aligned} \quad (6)$$

where  $\mathbf{M}_{\text{geo}}$  is binary geographic masking matrix as we mentioned earlier, the  $\odot$  indicates the Hadamard product.  $\text{ReLU}(\cdot) = \max(0, \cdot)$  is the rectified linear unit and **LN** means ‘layer normalization’ [62] which in this work is chosen to be **RMSNorm**. So

$$\text{LN}(\text{ReLU}(\cdot)) = \text{RMSNorm}(\text{ReLU}(\cdot)) \quad (7)$$

$$= \frac{\text{ReLU}(\cdot)}{\text{RMS}(\text{ReLU}(\cdot))} \odot g, \quad (8)$$

where  $\odot$  signifies the Hadamard (element-wise) product;  $g$ , the gain parameter, is an appropriately sized matrix typically initialized with all elements set to unity and **RMS**( $\cdot$ ) calculates the root mean square statistic, contributing to the stabilization of attention.

In the self-attention mechanism, the matrices  $\mathbf{Q}_t^{(S)}$ ,  $\mathbf{K}_t^{(S)}$ , and  $\mathbf{V}_t^{(S)}$  correspond to the query, key, and value components, respectively. These components are essential in computing the pairwise similarities between the queries and keys, where the

value matrix  $\mathbf{V}_t^{(S)}$  represents a 2D feature map encapsulating spatial and temporal features. To further refine the focus on spatial characteristics and to enhance local feature representation, we implement an Enhanced Convolution (**EnCov**) with a 3x3 2D convolution, to the value matrix  $\mathbf{V}_t^{(S)}$ . The **EnCov** is specifically designed to amplify the spatial information within the self-attention framework. Consequently, the output of the **ReSSA**, which integrates the benefits of rectified attention and enhanced local feature processing, is given as:

$$\begin{aligned} \mathbf{ReSSA}(\mathbf{Q}_t^{(S)}, \mathbf{K}_t^{(S)}, \mathbf{V}_t^{(S)}) \\ = \mathbf{EnCov}(\mathbf{V}_t^{(S)}) + \mathbf{ReLU}(\mathbf{Q}_t^{(S)}, \mathbf{K}_t^{(S)}, \mathbf{V}_t^{(S)}). \end{aligned}$$

#### D. Rectified Temporal Self-Attention Module (ReTSA)

In traffic data, various dependencies exist, such as periodic or trending patterns, among traffic conditions observed in different time slices. The nature of these dependencies can vary depending on the specific situation. Therefore, we have incorporated a Temporal Self-Attention module to effectively identify and capture dynamic temporal patterns. More formally, for a given node, we initially derive the query, key, and value matrices as follows:

$$\mathbf{Q}_n^{(T)} = \mathbf{X}_{::n} \cdot \mathbf{W}_Q^\top, \quad \mathbf{K}_n^{(T)} = \mathbf{X}_{::n} \cdot \mathbf{W}_K^\top, \quad \mathbf{V}_n^{(T)} = \mathbf{X}_{::n} \cdot \mathbf{W}_V^\top \quad (9)$$

where  $\mathbf{Q}_n^{(T)}$ ,  $\mathbf{K}_n^{(T)}$ , and  $\mathbf{V}_n^{(T)}$  represent the query, key, and value matrices for node  $n$ , respectively. Here  $\mathbf{W}_Q^\top$ ,  $\mathbf{W}_K^\top$  and  $\mathbf{W}_V^\top \in \mathbb{R}^{d \times d_0}$  are learnable parameters and  $d_0$  is the dimension of the query, key, and value matrices. Then, we apply self-attention operations in the spatial dimension to model the interactions between nodes and obtain the spatial dependencies (attention scores) among all nodes at time  $t$  as:

$$\mathbf{A}_n^{(T)} = \frac{(\mathbf{Q}_n^{(T)})(\mathbf{K}_n^{(T)})^\top}{\sqrt{d_0}}. \quad (10)$$

It can be seen that temporal self-attention can discover the dynamic temporal patterns in traffic data that are different for each node. Since the temporal self-attention has a global receptive field to model the long-range temporal dependencies among all time slices. Thus, the usage of **EnCov** can increase the attention to local features. This locality ensures that even if two different queries have the same weight under self-attention, they can obtain different outputs from different local features (time, space), thereby better capturing dynamic characteristics. Hence, the rectified function **ReLU**( $\cdot$ ) also can reduce the computational cost. Finally, we can obtain the the output of the temporal self-attention module as follows:

$$\begin{aligned} \mathbf{ReTSA}(\mathbf{Q}_n^{(T)}, \mathbf{K}_n^{(T)}, \mathbf{V}_n^{(T)}) \\ = \mathbf{EnCov}(\mathbf{V}_n^{(T)}) + \mathbf{ReLU}(\mathbf{A}_n^{(T)}) \mathbf{V}_n^{(T)}. \end{aligned} \quad (11)$$

#### E. Rectified Delay Aware Self Attention (ReDASA)

In the real world, when an accident happening in one area, it might take a few minutes before the traffic in adjacent areas is affected. To model this aspect effectively, we draw inspiration from the concept of Delay Aware Self Attention, as elaborated in [2]. This approach is adept at integrating delay-related

information into the key matrix, thereby capturing the essence of temporal lags in the propagation of spatial information. For a detailed exposition of this concept, the reader is referred to [2].

Our implementation extends this idea by amalgamating Delay Aware Self Attention with Rectified Self Attention, as depicted in Figure 4. In this hybrid model, normal Delay Aware Self Attention is used to enrich the key matrix with temporal information. We denote this modified key matrix as  $\hat{\mathbf{K}}_t^{(S)}$ . The primary operation in this model involves the computation of the product of the query matrix and  $\hat{\mathbf{K}}_t^{(S)}$ , which leads to the derivation of spatial dependencies at the specific time slice  $t$ . The equation for this computation is as follows:

$$\hat{\mathbf{A}}_t^{(S)} = \frac{(\mathbf{Q}_t^{(S)})(\hat{\mathbf{K}}_t^{(S)})^\top}{\sqrt{d_0}}. \quad (12)$$

As we mentioned, the self-attention is assumed to be fully connected attention graph. Hence here we employed a graph mask matrix  $\mathbf{M}_{\text{sem}}$  alongside the mask matrix  $\mathbf{M}_{\text{geo}}$  as utilized in Eq. (6). These matrices enable the simultaneous capture of both short-range and long-range spatial dependencies in traffic data. Then, the ReLSA can be written as:

$$\begin{aligned} \mathbf{ReLSA}_{\text{sem}}(\mathbf{Q}_t^{(S)}, \mathbf{K}_t^{(S)}, \mathbf{V}_t^{(S)}) \\ = \mathbf{LN}(\mathbf{ReLU}(\hat{\mathbf{A}}_t^{(S)} \odot \mathbf{M}_{\text{sem}}) \mathbf{V}_t^{(S)}). \end{aligned} \quad (13)$$

where the  $\odot$  indicates the Hadamard product. .

With the sum of **EnCov**, the final **ReDASA** can be formulate as:

$$\begin{aligned} \mathbf{ReDASA}(\mathbf{Q}_t^{(S)}, \mathbf{K}_t^{(S)}, \mathbf{V}_t^{(S)}) \\ = \mathbf{EnCov}(\mathbf{V}_t^{(S)}) + \mathbf{ReLU}(\hat{\mathbf{A}}_t^{(S)}) \mathbf{V}_t^{(S)}, \end{aligned} \quad (14)$$

In this manner, the rectified spatial self-attention module seamlessly integrates and enhances both short-range geographic proximity by **ReSSA** and long-range semantic context with **ReDASA** simultaneously, with the bonus of rectified attention [28], the computation complexity is greatly reduced.

#### F. Criss-Crossed Dual-Stream Learning (CCDS)

Criss-Crossed learning is designed as crossed-learning for temporal and spatial attention which is depicted in Figure 3. In the proposed framework, a novel criss-crossed learning approach is employed to harness both spatial and temporal dynamics from time series data. Initially, the input of spatial information is directed through the spatial attention module which is in Stage 1, then the output is fed into the temporal attention module which is in Stage 2. The output, which follows the spatial-temporal attention sequence, is denoted as  $\mathcal{O}^{\text{ReSSA}}$  and is defined as:

$$\mathcal{O}^{\text{ReSSA}} = \mathbf{ReTSA}(\mathbf{ReSSA}(\mathbf{Q}_t^{(S)}, \mathbf{K}_t^{(S)}, \mathbf{V}_t^{(S)})) \quad (15)$$

Simultaneously, the input on time-related information traverses through Stage 1 via the temporal attention module, with its output channeled into Stage 2 with the spatial attention

module. The result for transitions from temporal to spatial processing, is represented as  $\mathcal{O}^{ReTSA}$ , formulated as:

$$\mathcal{O}^{ReTSA} = \text{ReSSA}(\text{ReTSA}(\mathbf{Q}_n^{(T)}, \mathbf{K}_n^{(T)}, \mathbf{V}_n^{(T)})) \quad (16)$$

This integrated learning architecture ensures a comprehensive understanding of spatial-temporal relationships, thereby enhancing the model's learning capability by allowing it to capture complex patterns and dependencies inherent in the data, fostering a more robust representation.

#### G. Attention Mixer and Output layer

a) *Attention Mixer*: In our model, we integrate three types of attention- **ReSSA**, **ReDASA**, and **ReTSA** using a multi-head self-attention block. The Rectified Spatial-Temporal Self-Attention block (**ReSTSA**) simultaneously processes spatial and temporal information. This integration is achieved by concatenating outputs from each attention head  $h_{ReSSA}$ ,  $h_{ReDASA}$  and  $h_{ReTSA}$ , then projecting these concatenated results to obtain the final output. For simplicity, we denote the output of **ReSSA** (two stages), **ReDASA**, and **ReTSA** (two stages) with  $\mathcal{O}^{ReSSA}$ ,  $\mathcal{O}^{ReDASA}$  and  $\mathcal{O}^{ReTSA}$ . Then, the **ReSTSA** block is formally defined as:

$$\text{ReSTSA} = \bigoplus (\mathcal{O}_1^{ReSSA}, \dots, \mathcal{O}_{h_{ReSSA}}^{ReSSA}, \mathcal{O}_1^{ReDASA}, \dots, \mathcal{O}_{h_{ReDASA}}^{ReDASA}, \mathcal{O}_1^{ReTSA}, \dots, \mathcal{O}_{h_{ReTSA}}^{ReTSA}) \widehat{\mathbf{W}}$$

Here,  $\bigoplus$  signifies the concatenation operation, and  $\mathcal{O}_i^{ReSSA}$ ,  $\mathcal{O}_i^{ReDASA}$ , and  $\mathcal{O}_i^{ReTSA}$  represent the outputs from the geographical, semantic, and temporal heads, respectively.  $\widehat{\mathbf{W}} \in \mathbb{R}^{d \times d}$  is the projection matrix. We define the dimension  $d_0$  as a function of the total number of heads in our enhanced rectified spatial-temporal self-attention model, calculated by dividing the original dimension  $d$  by the sum of the **ReSSA**, **ReDASA**, and **ReTSA** heads:

$$d_0 = \frac{d}{h_{ReSSA} + h_{ReDASA} + h_{ReTSA}}.$$

Furthermore, a position-wise fully connected feed-forward network is employed on the output of the multi-head self-attention block, resulting in outputs denoted by:

$$\mathcal{Y} \in \mathbb{R}^{T \times N \times d}.$$

b) *Output Layer*:: For the final output layer, a skip connection with 1x1 convolutions is utilized after each spatial-temporal encoder layer. This process transforms the outputs  $\mathcal{Y}$  into a skip dimension  $\mathcal{Y}_{sk}$  within the space  $\mathbb{R}^{T \times N \times d_{sk}}$ , where  $d_{sk}$  represents the skip dimension. The final hidden state  $\mathcal{Y}_{hid} \in \mathbb{R}^{T \times N \times d_{sk}}$  is then derived by aggregating the outputs from each skip connection layer. For multi-step forecasting, the output layer linearly transforms the final hidden state  $\mathbf{X}_{hid}$  into the required dimensions as:

$$\widehat{\mathcal{Y}} = \text{Conv2}(\text{Conv1}(\mathcal{Y}_{hid})),$$

where  $\widehat{\mathcal{Y}}$  in  $\mathbb{R}^{T_0 \times N \times d}$  represents the prediction results for  $h'$  steps, and both Conv1 and Conv2 are  $1 \times 1$  convolutions. This direct approach is preferred over recursive methods for multi-step prediction to minimize cumulative errors and enhance model efficiency.

## V. EXPERIMENTS

Before the experimental description, we first introduce the datasets to be used in Section V-A. This is followed by a description of the baseline models in Section V-B. Detailed information about the experimental settings is provided in Section V-C. Subsequently, the experiment results are presented in Section V-E and the evaluation metrics are described in Section V-D. The results of the ablation study <sup>2</sup> are discussed in Section V-F, followed by a comprehensive discussion in Section V-G.

#### A. Dataset description

In our study, we evaluate **CCDSReFormer** using six real-world datasets in both graph-based and grid-based data structures. Specifically, the graph-based datasets include **PeMS04**, **PeMS07**, and **PeMS08**, while the grid-based datasets comprise **CHIBike**, **TDrive**, and **NYTaxi**. These datasets are publicly accessible and can be found in the LibCity repository, as referenced in Wang et al. (2023)[63]. Detailed information about each of these datasets is provided in Table I.

**PeMS04** [64]: Representing traffic data from the San Francisco Bay Area, this dataset was accumulated by the Caltrans Performance Measurement Systems (PeMS). Data from one sensor is condensed into 5-minute intervals, incorporating traffic flow, average speed, and average occupancy. It encompasses records from 307 sensors, spanning from Jan 1, 2018, to Feb 28, 2018.

**PeMS07** [64]: Representing traffic data from the San Francisco Bay Area, this dataset was accumulated by the Caltrans Performance Measurement Systems (PeMS). Data from one sensor is condensed into 5-minute intervals, incorporating traffic flow, average speed, and average occupancy. It encompasses records from 883 sensors, spanning from May 1, 2017, to Aug 31, 2017.

**PeMS08** [64]: This is the highway traffic flow data collected by the California Department of Transportation (Caltrans). The data range is from Jul 1, 2016 to Aug 31, 2016. The flow data is sampled every 5 minutes. The number of sensors is 170.

**NYTaxi** [65]: The data was made available by the NYCTaxi & Limousine Commission (TLC) and built on data from ride-hailing companies such as Uber and Lyft. The records cover New York from Jan 1, 2014 to Dec 31, 2014. For each demand record, the data provides information such as the pick-up time, drop-off time, pick-up zone, drop-off zone, etc. The traffic zones are predetermined by the TLC.

**CHIBike** [63]: The CHIBike dataset comprises bicycle-sharing data from Chicago, capturing the period from Jul 1, 2020, to Sep 30, 2019. This dataset includes detailed records of bike rentals and returns across various stations in Chicago. Each record encompasses information such as rental and return times, originating and destination stations, and trip duration.

**T-Drive** [66]: The T-Drive dataset is derived from a comprehensive collection of taxi trajectory data in Beijing, spanning one week of continuous operation. It contains over

<sup>2</sup>An ablation study is a method of evaluating a system's performance by sequentially removing its components to identify their individual impacts on the overall effectiveness.

TABLE I  
DATASET INFORMATION

Dataset	#Nodes	#Edges	#Timestamps	Time Interval	Time Range
PeMS04	307	340	16992	5 min	01/01/2018 - 02/28/2018
PeMS07	883	866	28224	5 min	05/01/2017 - 08/31/2017
PeMS08	170	295	17856	5 min	07/01/2016 - 08/31/2016
NYCTaxi	75 (15x5)	484	17520	30 min	01/01/2014 - 12/31/2014
CHIBike	270 (15x18)	1966	4416	30 min	07/01/2020 - 09/30/2020
T-Drive	1024 (32x32)	7812	3600	60 min	02/01/2015 - 06/30/2015

15 million GPS records from thousands of taxis, providing detailed insights into urban traffic flow. The data encapsulates information such as GPS coordinates, timestamps, and taxi operation statuses (e.g., occupied, vacant).

### B. Tested frameworks and baseline methods

We compare **CCDSReFormer** with the following 11 baselines in three categories.

#### (1) Graph Neural Network based Models:

- DCRNN [48]: Is a deep learning framework for traffic forecasting that addresses the complexities of spatial dependencies on road networks and non-linear temporal dynamics, using bidirectional random walks and an encoder-decoder architecture.
- STGCN [49]: Is a novel deep learning framework for traffic time series prediction, utilizing complete convolutional structures on graphs for faster training and fewer parameters, effectively capturing spatio-temporal correlations and outperforming baselines on various real-world traffic datasets.
- GWNET [50]: Is a novel graph neural network for spatial-temporal graph modeling, which overcomes the limitations of fixed graph structures and ineffective temporal trend capture in existing methods by using an adaptive dependency matrix and stacked dilated 1D convolutions.
- MTGNN [51]: Introduces a general graph neural network framework tailored for multivariate time series data, which automatically extracts uni-directed relations among variables through a graph learning module and incorporates novel mix-hop propagation and dilated inception layers.
- STFGNN [52]: Is a model that effectively learns hidden spatial-temporal dependencies through a novel fusion of various spatial and temporal graphs, and integrates a gated convolution module for handling long sequences.
- STGCNDE [53]: Is a breakthrough method in traffic forecasting, combining graph convolutional networks and recurrent neural networks with neural controlled differential equations for spatial and temporal processing.

#### (2) Self-Attention based Models:

- STTN [57]: Is a novel approach for long-term traffic forecasting that dynamically models directed spatial dependencies using a spatial transformer and long-range temporal dependencies using a temporal transformer, offering enhanced accuracy and scalable training.

- GMAN [58]: Is a long-term traffic prediction model, utilizing an encoder-decoder architecture with spatio-temporal attention blocks to model the impact of spatio-temporal factors on traffic conditions, featuring a transform attention layer that effectively links historical and future time steps, and demonstrating superior performance in real-world traffic volume and speed prediction tasks.

- ASTGNN [59]: Is a novel spatial-temporal neural network framework integrating a graph convolutional recurrent module (GCRN) with a global attention module, designed to effectively model both long-term and short-term temporal correlations in traffic data, and demonstrating superior predictive performance on five real traffic datasets compared to baseline methods.

- PDFFormer [2]: Uses self-attention to capture dynamic spatial dependencies and explicitly model time delays in traffic systems, demonstrating state-of-the-art performance.

- STAEformer [67]: A state of art transformer model that leveraging Spatio-Temporal Adaptive Embedding, achieves top performance in traffic forecasting by effectively capturing complex spatio-temporal patterns, marking a significant advance over previous models.

#### (3) Other based Models:

- VAR [68]: Vector Auto-Regression used in traffic flow prediction for its ability to capture the linear interdependencies among multiple time series, making it suitable for forecasting traffic conditions based on historical data.
- SVR [69]: Support Vector Regression utilizes historical data to predict future traffic conditions by employing a linear kernel function, offering reliable forecasts even with high-dimensional data.

### C. Experiment Settings

a) *Data Processing*: In line with contemporary practices of all the baselines [2], we partition the three graph-based datasets into training, validation, and test sets using a 6:2:2 split. For these datasets, we predict traffic flow over the next hour (12 time steps) based on the data from the preceding hour (12 time steps), thus employing a multi-step prediction approach. For the grid-based datasets, we adopt a 7:1:2 split ratio which is aligned with the baseline models. In this case, the prediction model uses data from the previous six time steps to forecast the next step's traffic inflow and outflow. Before training, we standardize the inputs across all datasets

by applying Z-score normalization. The code is available in: <https://github.com/superca729/CCDSReFormer>.

*b) Parameter Settings:* All experiments are conducted on a machine with the GPU RTX 3090(24GB) CPU 15 vCPU Intel(R) Xeon(R) Platinum 8358P CPU @ 2.60GHz memory 150GB. We use the same configurations with [2] for the hidden dimension  $d$ , tuning its values within {16, 32, 64}, and assess different depths for the encoder layers  $L$  with an option set {2, 4, 6, 8}. Unlike the baselines, we fix the same number of attention heads to {2, 3} for  $h_{ReSSA}$  and  $h_{ReTSA}$ , respectively, to ensure the Criss-Cross performance well. We set the number of  $h_{ReDASA}$  in 4. The selection of the optimal model configuration was based on their performance on the validation set. For training a model, we employ the same Adam optimizer as [2] with a learning rate of 0.001. The models were trained with a batch size of 16 over 200 epochs.

#### D. Evaluation Metrics

Three common metrics, the mean absolute error (MAE), the mean absolute percentage error (MAPE) and the root mean square error (RMSE); are used to measure the traffic forecasting performance of the tested methods.

$$\begin{aligned} \text{MAE} &= \frac{1}{n} \sum_{i=1}^n |\hat{y}_i - y_i|, \\ \text{MAPE} &= \frac{1}{n} \sum_{i=1}^n \left| \frac{\hat{y}_i - y_i}{y_i} \right| \times 100\%, \\ \text{RMSE} &= \sqrt{\frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2} \end{aligned}$$

where  $y = \{y_1, y_2, \dots, y_n\}$  is the ground-truth value,  $\hat{y} = \{\hat{y}_1, \hat{y}_2, \dots, \hat{y}_n\}$  is the prediction value.

In our evaluations, we exclude missing values when calculating metrics. For grid-based datasets, samples with flow values below 10 were filtered out, as the method described in [70], [2]. Additionally, for these datasets, we compute the final result by taking the average of the inflow and outflow evaluation metrics which is the same as work [2].

#### E. Experiment Results

The performance results on two types of datasets are presented in Table II and III, where the best results are marked in **First**, **Second** and **Third**. To facilitate a straightforward comparison with the baseline models, we have presented the results in three decimal places, consistent with the precision used for the baseline models. We also further present a visualization comparing the prediction and ground truth of **CCDSReFormer** showing as in Figures 5, 6, 7, 8 and 9.

*a) Performance Results on Various Datasets:* To ensure an equitable assessment of model efficacy, we have chosen our baseline models from among the leading contenders in Traffic Flow prediction as identified at the time of drafting this paper. We include comparisons with the current state-of-the-art (SOTA) models (eg. STAEFormer[67]) with the time when we worked on the paper.

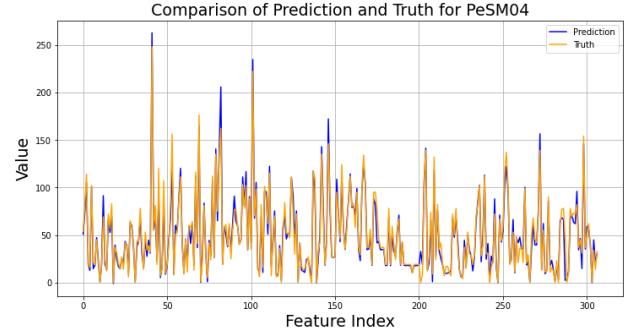


Fig. 5. The plot illustrates a comparison between the actual data (ground truth) and the predicted values for the PeMS04 dataset using the **CCDSReFormer** model. The data points are indexed along the x-axis, which represents as time.

Based on the results presented in Tables II and III, our **CCDSReFormer** demonstrates superior performance over all baseline models in terms of MAE (Mean Absolute Error) and RMSE (Root Mean Squared Error) across all datasets, as confirmed by the Student's t-test at a significance level of 0.01. For the PeMS04 dataset, **CCDSReFormer** outperforms other models with the lowest MAE of 18.176, the lowest MAPE of 12.096%, and the lowest RMSE of 29.844. For the PeMS07 dataset, **CCDSReFormer** again shows superior performance with the lowest MAE of 19.475, although it had a higher MAPE (32.473%) compared to PDFormer. While **CCDSReFormer** still achieves the best MAPE of 8.529%. The RMSE was recorded as the lowest for **CCDSReFormer** at 32.499. For the PeMS08 dataset, both **CCDSReFormer** and PDFormer are comparable in MAE and MAPE metrics. Specifically, **CCDSReFormer** achieves the best MAE of 13.564 and the best RMSE of 23.250, while PDFormer records the best MAPE of 9.046%. Consistent with previous observations, **CCDSReFormer** outperforms in MAE and RMSE metrics. However, its performance in MAPE was not strong enough to compete with PDFormer. This pattern suggests that while **CCDSReFormer** is generally reliable in terms of absolute error, it might struggle with relative error, especially in datasets where small actual values are more prevalent, thus leading to higher MAPE scores.

*b) Visualization of CCDSReFormer Performance:* We further present visualization to compare the ground truth and the prediction based on the **CCDSReFormer** model on the datasets in Figure 5, 6, 7, 8 and 9, respectively. The comparison between the ground truth (depicted in tan) and predicted values (shown in blue) reveals key insights into the model's performance. Notably, the predicted values generally aligned well with the actual data, indicating the model's capability to capture overall trends. However, specific deviations were observed: an overestimation in the PeMS04 (Figure 5), and TDrive (Figure 8) datasets, where the blue lines rose above the tan ground truth lines, and a slight underestimation in the PeMS08 (Figure 6) CHIBike (Figure 7) and NYTaxi (Figure 9) dataset, as indicated by the blue line falling just slightly below the actual values.

TABLE II  
PERFORMANCE METRICS FOR DIFFERENT MODELS ACROSS GRAPH DATASETS, WITH RANKING HIGHLIGHTS.

Models	PeMS04			PeMS07			PeMS08		
	MAE	MAPE	RMSE	MAE	MAPE	RMSE	MAE	MAPE	RMSE
VAR	23.750	18.090	36.660	101.200	39.690	155.140	22.320	14.470	33.830
SVR	28.660	19.150	44.590	32.970	15.430	50.150	23.250	14.710	36.150
DCRNN	22.737	14.751	36.575	23.634	12.281	36.514	18.185	11.235	28.176
STGCN	21.758	13.874	34.769	22.898	11.983	35.440	17.838	11.211	27.122
STFGNN	19.830	13.021	31.870	22.072	9.212	35.805	16.636	10.547	26.206
STGNCDE	19.211	12.772	31.088	20.620	8.864	34.036	15.455	9.921	24.813
STTN	19.478	13.631	31.910	21.344	9.932	34.588	15.482	10.341	24.965
GMAN	19.139	13.192	31.601	20.967	9.052	34.097	15.307	10.134	24.915
ASTGNN	18.601	12.630	31.028	20.616	8.861	34.017	14.974	9.489	24.710
PDFormer	18.321	12.103	29.965	19.832	8.012	32.870	13.583	9.046	23.505
STAFormer	18.224	12.301	30.832	19.343	8.012	32.603	13.462	8.889	23.254
CCDSReFormer	18.176	12.096	29.544	19.305	32.473	32.499	13.424	9.067	23.250

TABLE III  
PERFORMANCE METRICS FOR DIFFERENT MODELS ACROSS GRID DATASETS.

Models	CHIBike			TDrive			NYTaxi		
	MAE	MAPE	RMSE	MAE	MAPE	RMSE	MAE	MAPE	RMSE
DCRNN	22.737	14.751	36.575	23.634	12.281	36.514	18.185	11.235	28.176
STGCN	21.758	13.874	34.769	22.898	11.983	35.440	17.838	11.211	27.122
STFGNN	21.938	17.566	38.411	21.143	17.261	37.836	19.553	16.560	36.179
STGNCDE	19.211	18.601	31.088	19.478	12.772	31.910	19.139	13.631	31.601
STTN	20.620	8.864	34.036	21.344	9.932	34.588	20.967	10.134	34.097
GMAN	15.455	9.921	24.813	15.482	10.341	24.965	15.307	10.134	24.915
ASTGNN	13.279	13.926	21.675	13.366	13.984	21.834	13.270	13.893	21.661
PDFormer	19.289	16.504	36.118	20.513	16.659	37.143	19.104	16.449	36.053
CCDSReFormer	11.572	12.751	18.359	12.180	20.89	23.668	3.786	29.693	5.328

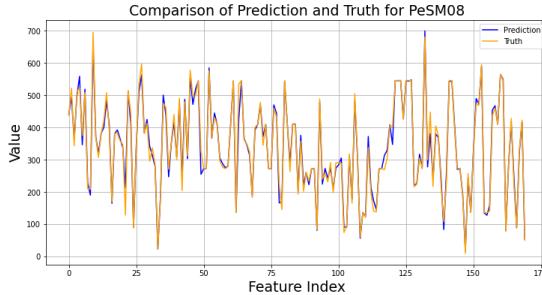


Fig. 6. The plot illustrates a comparison between the actual data (ground truth) and the predicted values for the PeMS08 dataset using the **CCDSReFormer** model. The data points are indexed along the x-axis, which represents time.

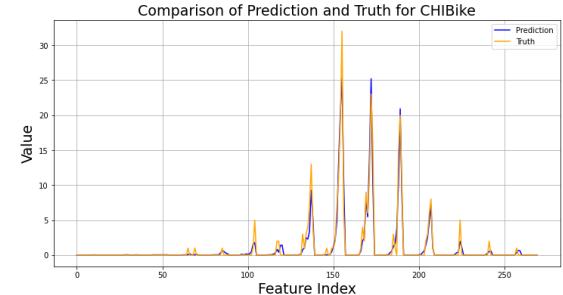


Fig. 7. The plot illustrates a comparison between the actual data (ground truth) and the predicted values for the CHIBike dataset using the **CCDSReFormer** model.

#### F. Ablation Study

To further investigate the efficacy of **CCDS**, **EnCov**, and **ReLSA** components in the **CCDSReFormer**, we conduct an ablation study on the PeMS04 dataset as shown in Table IV, examining different variants of the proposed model. We initially reproduced the model and with adding of each component, we obtain a better performance of the model.

This study primarily focuses on key performance metrics:

MAE, MAPE, and RMSE. The original PDFormer, as detailed in the paper[2], is used as a baseline. Our reproduced version exhibits slightly higher values in MAE, MAPE, and RMSE. Variants incorporating additional components specifically **CCDS**, **EnCov**, and **ReLSA** show diverse improvements. Notably, the **CCDSReFormer**, integrating all three of these components, achieves the most superior performance across all metrics. In particular, the **CCDSReFormer** variant excelled,

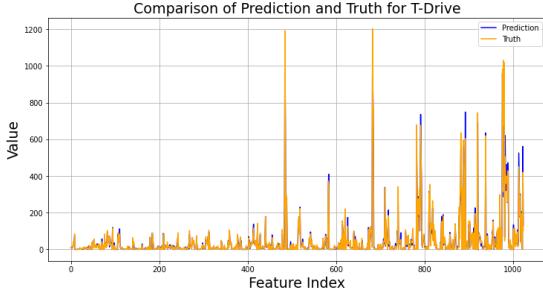


Fig. 8. The plot illustrates a comparison between the actual data (ground truth) and the predicted values for the TDrive dataset using the **CCDSReFormer** model. The data points are indexed along the x-axis, which represents as time.

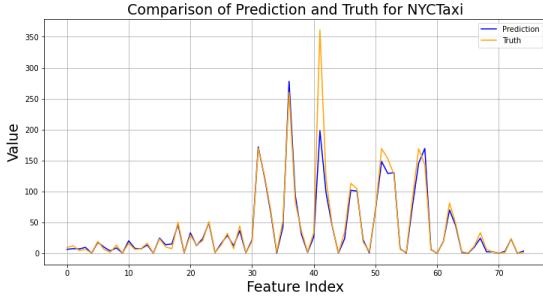


Fig. 9. The plot illustrates a comparison between the actual data (ground truth) and the predicted values for the NYCTaxi dataset using the **CCDSReFormer** model. The data points are indexed along the x-axis, which represents as time.

recording the lowest MAE (18.176), MAPE (12.096%), and RMSE (29.844). These results underscore the synergistic effect of combining **CCDS**, **EnCov**, and **ReLSA**, significantly enhancing the model's predictive accuracy on the PeMS04 dataset.

TABLE IV  
ABLATION STUDY FOR PEAMS04 DATASET

Models	MAE	MAPE	RMSE
<b>CCDSReFormer</b>	<b>18.176</b>	<b>12.096</b>	<b>29.844</b>
w/o ReLSA	18.332	12.472	29.948
w/o EnCov	18.269	12.121	30.022
w/o CCDS	18.355	12.217	30.055

To verify the effectiveness on **ReLSA**, we further test the average training time on **CCDSReFormer**, and ensure all models operated on the same device. Based on the result shown in Table V, our model showcases a shorter running time compared to PDFFormer, ASTGNN, and GMAN, indicating superior performance efficiency. It also remains competitive with STTN, underscoring its effectiveness. Given that the STAEFormer has fewer parameters, it's reasonable to expect better performance in terms of both training and inference times.

TABLE V  
MODEL PERFORMANCE METRICS ON THE PEAMS04 DATASET

Model	Training	Inference
	Time (s)	Time (s)
GMAN	493.578	39.824
ASTGNN	205.223	50.112
PDFFormer	131.871	8.420
CCDSReFormer	112.73	7.871
STTN	100.398	12.596
STAEFormer	83.099	7.156

### G. Discussion/Case Study

To gain a deeper understanding of the functionalities of **CCDSReFormer**, we further explore how attention mechanisms are applied to different types of information - spatial, temporal, integration of both and delay-aware attention. The structure of each attention module is divided into two distinct stages which include **ReSSA** and **ReTSA** as depicted in Figure 3. One given example for the spatial information **ReSSA** on the PeSM04 dataset is shown in Figure 10. The first stage is characterized by the implementation of **ReSSA**, while the second stage involves criss-cross learning or what we call integration of temporal information using **ReTSA**. For **ReDASA**, we generate visualizations that directly correspond to the increase in heads and layers, aligning closely with the experimental settings described in Section V-C. By examining the attention score visualizations, as displayed in Figures 10, 11 and 12, we can draw several insightful conclusions as below:

a) *Rectified self-attention with increasing of stages*:: The self-attention mechanism, when applied in increasing stages (or Criss-Cross learning), evenly distributes attention across integrated spatial and temporal information. This phenomenon is clearly visible when contrasting Figure 11a with Figure 11c. Initially, in **ReSSA** (Figure 11a), the heatmap displays attention scores from 0.005 to 0.025, predominantly accentuating spatial details with elevated attention scores. However, upon integrating **ReTSA**, the heatmap in Figure 11c exhibits a more evenly distribution of attention, with scores approximating 0.2, including both spatial and temporal aspects.

A similar trend is observable in the initial **ReTSA** attention scores (Figure 12a), where an increase in stage (as compared with Figure 12c) leads to an average trend in attention scores. Additional examples are provided in Figures 12e, 12g, 12i, and 12k, among others. Such observations suggest that the integration of the Criss-Cross methodology equips the **CCDSReFormer** with an enhanced capability to capture and delineate both spatial and temporal information with refined attention.

b) *Rectified self-attention in different number of heads*:: As the number of attention heads increases from 1 to 2, each head can potentially focus on distinct segments or attributes of the input data. This concept is illustrated through a comparison on **ReSSA** (as in Figure 10) between Figure 11a and Figure 11b. Notably, Figure 11b displays higher attention scores

Fig. 10. Attention scores in ReSSA: horizontal axis for stage and heads growth, vertical axis for layer depth. Note:  $H_n$  = Head  $n$ ,  $L_n$  = Layer  $n$

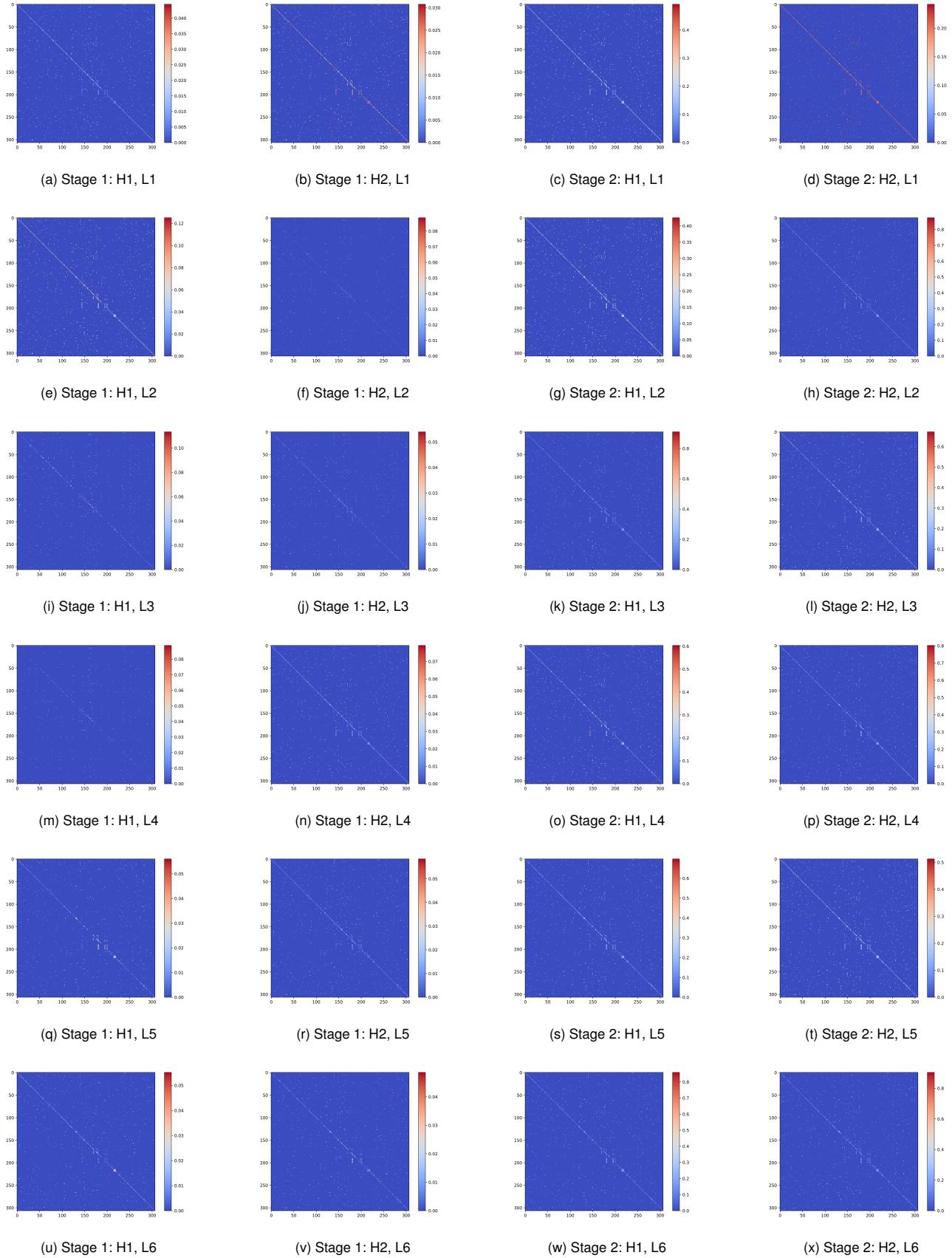


Fig. 11. Attention scores in **ReTSA** by layers: horizontal axis shows stage progression and more heads; vertical axis reflects deeper layers. Note: Hn = Head n, Ln = Layer n

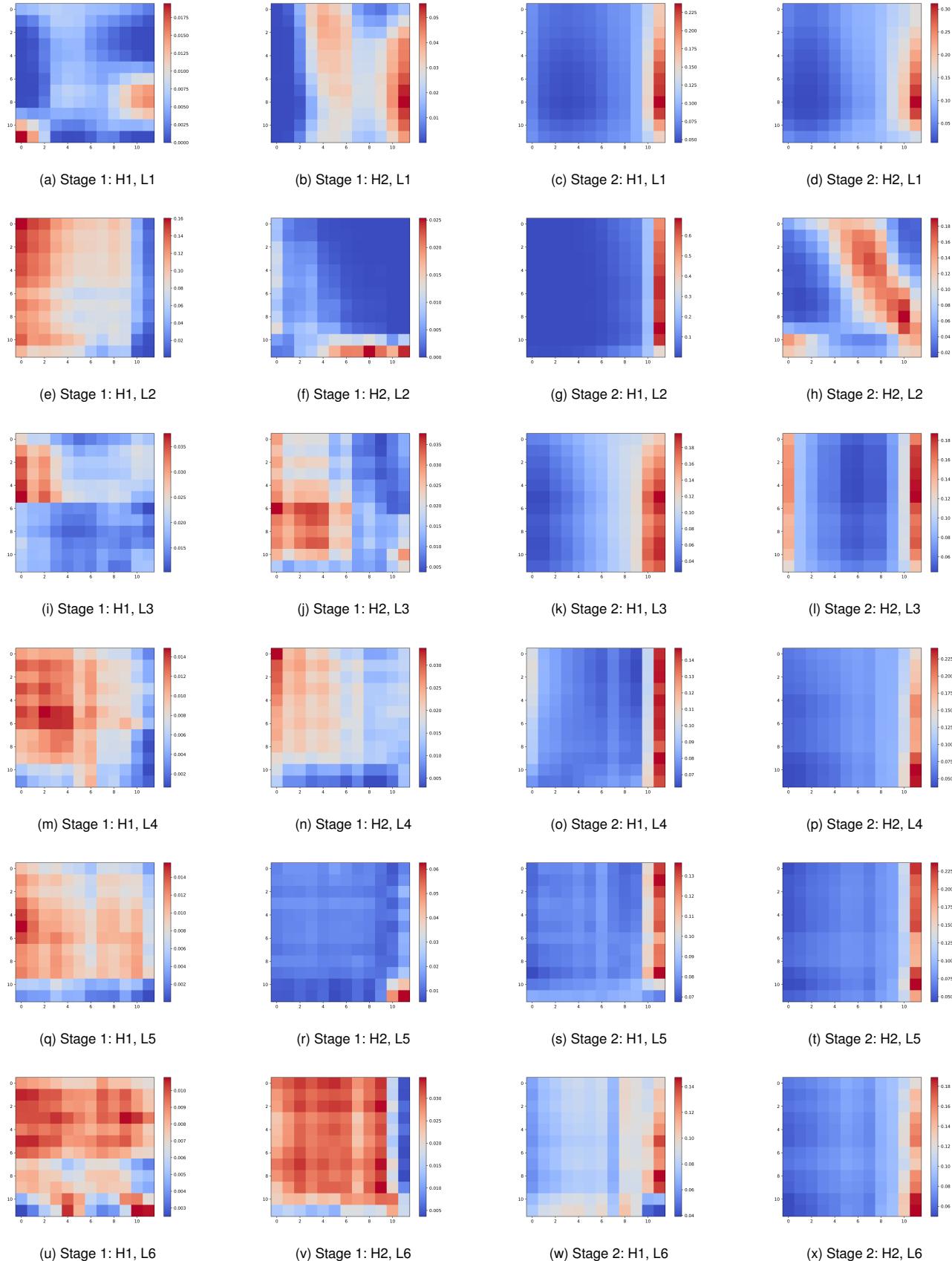
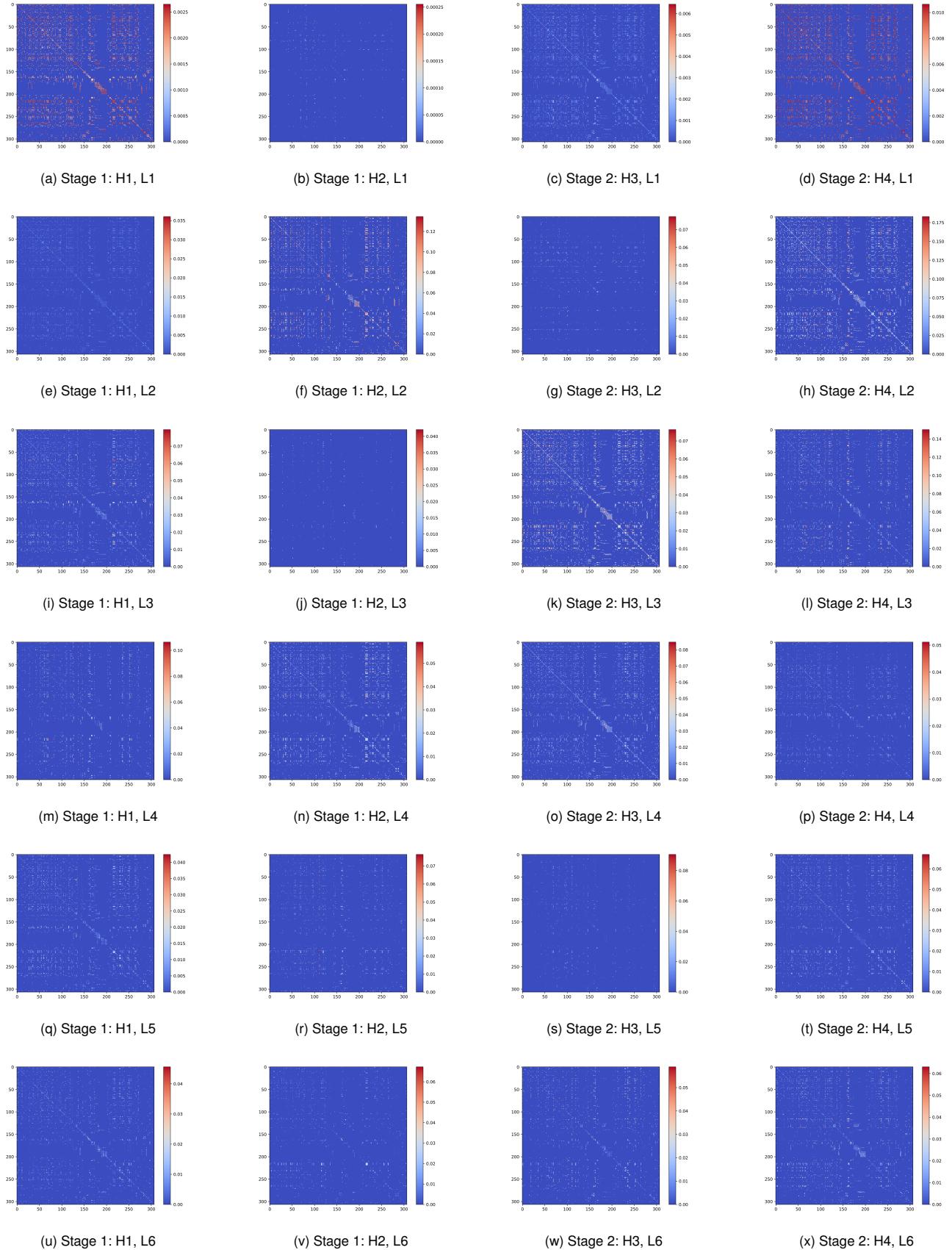


Fig. 12. Attention scores in **ReDASA** across layers: horizontal axis shows more heads, vertical indicates deeper layers. Note:  $H_n$  = Head  $n$ ,  $L_n$  = Layer  $n$



compared to Figure 11a, indicating a more comprehensive capture of data features and relationships.

Furthermore, in **ReTSA**, as illustrated in Figure 11, an increase in the number of heads during stage 1 (as shown in the first two columns, e.g., Figures 12a and 12b) results in significant changes in attention scores compared to stage 2 (as depicted in the last two columns, e.g., Figures 12c and 12d).

Similarly, in **ReDASA** (see Figure 12), setting the number of attention heads to 4 reveals that, with an increasing number of heads, the attention scores in **ReDASA** closely resemble those when the head count is one. This can be specifically observed in comparisons such as in Figures 13a and 13d; however, there are notable exceptions with significant differences, as seen when comparing Figures 13e and 13h. This diversity in attention distribution enhances our model's ability to analyze input data more effectively, allowing for the recognition of a wider range of patterns and connections.

c) *Rectified self-attention with increasing of layers::* As layers increase, attention scores are dynamically adjusted. This is evident in **ReSSA** (as in Figure 10) stage 1 and head 1, where the scores slightly rise from a ceiling of 0.04 to 0.05 across layers, as shown from Figure 11a to Figure 11t. The change becomes slighter at higher layers, such as layer 5 (Figure 11p) and layer 6 (Figure 11t). Similarly, in stage 2 and head 2, the maximum attention score rises from 0.5 to 0.8, as illustrated in Figure 11c and Figure 11x which means even with Criss-Cross learning, with the increasing of layers can further enhance the score of attention distribution.

Additionally, with **ReTSA** (referenced in Figure 11), a progression is evident when observing the visualization from the top row (Figures 13a, 13b, 13c, and 13d) to the bottom row (Figures 13u, 13v, 13w, and 13x). As we examine the subsequent layers, from Layer 1 through Layer 6, there is a noticeable trend towards a more uniform distribution of attention. This suggests that as the layers deepen, the model may be gaining a more nuanced perception of the data's interrelations.

## VI. LIMITATION

The current limitation is that the computational results do not surpass all the baselines, primarily due to the significant computational load of the entire model framework. However, we have demonstrated that even in models with substantial computational requirements, our method can effectively reduce computational costs. Any attention mechanism can utilize the **EnReLSA** attention to decrease computational expenses.

## VII. CONCLUSION

In this study, we introduce the Dual-Stream Criss-Cross Enhanced Rectified Transformer (**CCDSReFormer**), an innovative model designed for accurate and computationally efficient traffic flow prediction. The model uniquely combines spatial and temporal information through a dual Criss-Cross stream, effectively capturing the intricate dynamics of traffic patterns. A locally enhanced convolution within the attention mechanism is implemented, sharpening the model's focus on local spatial-temporal features and nuanced traffic dynamics

influenced by localized conditions. This ensures a nuanced exploration of traffic patterns influenced by localized conditions, contributing to a more accurate prediction. Additionally, we incorporate Rectified Linear Attention (ReLA) into the traffic forecasting domain, which adapts dynamically to the specific spatial-temporal characteristics of traffic data while simultaneously reducing computational demands. This novel approach introduces dynamism and adaptability, responding effectively to the unique spatial-temporal characteristics. Comparative analysis of various **CCDSReFormer** configurations demonstrates each component's positive impact on prediction accuracy. Our method has undergone extensive testing across six diverse real-world datasets, establishing its superiority over existing state-of-the-art models in terms of both performance and computational efficiency. Additionally, our model exhibits robust parameter tuning capabilities, further emphasizing its versatility and applicability in the dynamic domain of traffic flow prediction. Looking ahead, we plan to explore the application of the **CCDSReFormer** model to other spatial-temporal datasets, such as weather forecasting.

## REFERENCES

- [1] J. Zhang, F.-Y. Wang, K. Wang, W.-H. Lin, X. Xu, and C. Chen, "Data-driven intelligent transportation systems: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 4, pp. 1624–1639, 2011.
- [2] J. Jiang, C. Han, W. X. Zhao, and J. Wang, "PDFormer: Propagation delay-aware dynamic long-range transformer for traffic flow prediction," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 4, pp. 4365–4373, Jun. 2023.
- [3] B. Gomes, J. Coelho, and H. Aidos, "A survey on traffic flow prediction and classification," *Intelligent Systems with Applications*, vol. 20, p. 200268, 2023.
- [4] M. S. Ahmed and A. R. Cook, "Analysis of freeway traffic time-series data by using Box-Jenkins techniques," *Journal of the Transportation Research Board*, vol. 1979-2, no. 722, pp. 1–9, 1979.
- [5] D. Zeng, J. Xu, J. Gu, L. Liu, and G. Xu, "Short term traffic flow prediction using hybrid ARIMA and ANN models," in *2008 Workshop on Power Electronics and Intelligent Transportation System*. IEEE, 2008, pp. 621–625.
- [6] C. Chen, J. Hu, Q. Meng, and Y. Zhang, "Short-time traffic flow prediction with ARIMA-GARCH model," in *IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2011, pp. 607–612.
- [7] M. Tong and H. Xue, "Highway traffic volume forecasting based on seasonal ARIMA model," *Journal of Highway and Transportation Research and Development (English Edition)*, vol. 3, no. 2, pp. 109–112, 2008.
- [8] S. V. Kumar, "Traffic flow prediction using Kalman filtering technique," *Procedia Engineering*, vol. 187, pp. 582–587, 2017.
- [9] A. Emami, M. Sarvi, and S. A. Bagloee, "Short-term traffic flow prediction based on faded memory Kalman filter fusing data from connected vehicles and bluetooth sensors," *Simulation Modelling Practice and Theory*, vol. 102, p. 102025, 2020, special Issue on IoT, Cloud, Big Data and AI in Interdisciplinary Domains. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1569190X1930156X>
- [10] D.-w. Xu, Y.-d. Wang, L.-m. Jia, Y. Qin, and H.-h. Dong, "Real-time road traffic state prediction based on ARIMA and Kalman filter," *Frontiers of Information Technology & Electronic Engineering*, vol. 18, pp. 287–302, 2017.
- [11] T. Zhou, D. Jiang, Z. Lin, G. Han, X. Xu, and J. Qin, "Hybrid dual Kalman filtering model for short-term traffic flow forecasting," *IET Intelligent Transport Systems*, vol. 13, no. 6, pp. 1023–1032, 2019.
- [12] J. W. Gao, Z. W. Leng, B. Zhang, X. Liu, and G. Q. Cai, "The application of adaptive Kalman filter in traffic flow forecasting," *Advanced Materials Research*, vol. 680, pp. 495–500, 2013.

- [13] J. Zheng and L. M. Ni, "Time-dependent trajectory regression on road networks via multi-task learning," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 27, no. 1, pp. 1048–1055, Jun. 2013. [Online]. Available: <https://ojs.aaai.org/index.php/AAAI/article/view/8577>
- [14] D. D. Oliveira, M. Ricipinelli, G. Z. Tozatto, R. V. Andreão, and S. M. Müller, "Forecasting vehicular traffic flow using MLP and LSTM," *Neural Computing and Applications*, vol. 33, pp. 17245–17256, 2021.
- [15] X. Luo, D. Li, Y. Yang, and S. Zhang, "Spatiotemporal traffic flow prediction with KNN and LSTM," *Journal of Advanced Transportation*, vol. 2019, p. 4145353, 2019.
- [16] C. Ma, G. Dai, and J. Zhou, "Short-term traffic flow prediction for urban road sections based on time series analysis and LSTM-BILSTM method," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 6, pp. 5615–5624, 2021.
- [17] R. Fu, Z. Zhang, and L. Li, "Using LSTM and GRU neural network methods for traffic flow prediction," in *Proceedings of the 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC)*. IEEE, 2016, pp. 324–328.
- [18] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [19] J. Zhang, Y. Zheng, and D. Qi, "STResNet: Deep spatial-temporal residual networks for citywide crowd flows prediction," *Proceedings of AAAI Conference on Artificial Intelligence*, 2017.
- [20] W. Zhang, Y. Yu, Y. Qi, F. Shu, and Y. Wang, "Short-term traffic flow prediction based on spatio-temporal analysis and CNN deep learning," *Transportmetrica A: Transport Science*, vol. 15, no. 2, pp. 1688–1711, 2019.
- [21] X. Yan, X. Gan, J. Tang, D. Zhang, and R. Wang, "Prostformer: Progressive space-time self-attention model for short-term traffic flow forecasting," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–15, 2024.
- [22] Y. Zheng, S. Wang, C. Dong, W. Li, W. Zheng, and J. Yu, "Urban road traffic flow prediction: A graph convolutional network embedded with wavelet decomposition and attention mechanism," *Physica A: Statistical Mechanics and its Applications*, vol. 608, p. 128274, 2022. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0378437122008329>
- [23] Z. Chen, Z. Lu, Q. Chen, H. Zhong, Y. Zhang, J. Xue, and C. Wu, "Spatial-temporal short-term traffic flow prediction model based on dynamical-learning graph convolution mechanism," *Information Sciences*, vol. 611, pp. 522–539, 2022. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0020025522009902>
- [24] H. Xing, A. Chen, and X. Zhang, "RL-GCN: Traffic flow prediction based on graph convolution and reinforcement learning for smart cities," *Displays*, vol. 80, p. 102513, 2023. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0141938223001464>
- [25] L. Zhao, Y. Song, C. Zhang, Y. Liu, P. Wang, T. Lin, M. Deng, and H. Li, "T-gcn: A temporal graph convolutional network for traffic prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 9, pp. 3848–3858, 2020.
- [26] Y. Lin, Z. Lv, J. Lu, Z. Pan, Y. Wang, and J. Li, "Adaptively spatial-temporal attention network for traffic flow prediction," *IEEE Transactions on Intelligent Transportation Systems*, 2020.
- [27] J. Yan and L. Ma, "TFormer: Traffic forecasting transformer," *IEEE Transactions on Intelligent Transportation Systems*, 2021.
- [28] B. Zhang, I. Titov, and R. Sennrich, "Sparse attention with linear units," *arXiv:2104.07012*, pp. 1–10, 2021. [Online]. Available: <https://arxiv.org/abs/2104.07012>
- [29] I. Alam, D. M. Farid, and R. J. Rossetti, "The prediction of traffic flow with regression analysis," in *Proceedings of Emerging Technologies in Data Mining and Information Security (IEMIS)*, vol. 2. Springer, 2019, pp. 661–671.
- [30] B. Priambodo and A. Ahmad, "Predicting traffic flow based on average speed of neighbouring road using multiple regression," in *Advances in Visual Informatics: 5th International Visual Informatics Conference (IVIC)*, vol. 5. Springer, 2017, pp. 309–318.
- [31] V. Topuz, "Hourly traffic flow prediction using different ANN models," in *Urban Transport and Hybrid Vehicles*. IntechOpen, 08 2010, p. 192.
- [32] B. Sharma, S. Kumar, P. Tiwari, P. Yadav, and M. I. Nezhurina, "Ann based short-term traffic flow forecasting in undivided two lane highway," *Journal of Big Data*, vol. 5, no. 1, pp. 1–16, 2018.
- [33] A. Cohen and S. Dalyot, "Pedestrian traffic flow prediction based on ANN model and OSM data," in *Proceedings of the International Cartographic Association*, vol. 2. Copernicus Publications Göttingen, Germany, 2019, pp. 20–27.
- [34] F. I. Rahman, "Short term traffic flow prediction using machine learning KNNs, SVM and ANN with weather information," *International Journal for Traffic & Transport Engineering*, vol. 10, no. 3, p. 371, 2020.
- [35] L. Cai, Y. Yu, S. Zhang, Y. Song, Z. Xiong, and T. Zhou, "A sample-rebalanced outlier-rejected k-nearest neighbor regression model for short-term traffic flow forecasting," *IEEE Access*, vol. 8, pp. 22686–22696, 2020.
- [36] L. Yang, Q. Yang, Y. Li, and Y. Feng, "K-nearest neighbor model based short-term traffic flow prediction method," in *Proceedings of the 18th International Symposium on Distributed Computing and Applications for Business Engineering and Science (DCABES)*. IEEE, 2019, pp. 27–30.
- [37] J. Li, F. Guo, A. Sivakumar, Y. Dong, and R. Krishnan, "Transferability improvement in short-term traffic prediction using stacked LSTM network," *Transportation Research Part C: Emerging Technologies*, vol. 124, p. 102977, 2021. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0968090X21000140>
- [38] J.-M. Yang, Z.-R. Peng, and L. Lin, "Real-time spatiotemporal prediction and imputation of traffic status based on LSTM and graph laplacian regularized matrix factorization," *Transportation Research Part C: Emerging Technologies*, vol. 129, p. 103228, 2021. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0968090X21002412>
- [39] S. Hochreiter, "The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 06, no. 02, pp. 107–116, Apr. 1998.
- [40] J. Ke, H. Zheng, H. Yang, and X. M. Chen, "Short-term forecasting of passenger demand under on-demand ride services: A spatio-temporal deep learning approach," *Transportation Research Part C: Emerging Technologies*, vol. 85, pp. 591–608, 2017. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0968090X17302899>
- [41] Y. Duan, Y. Lv, Y.-L. Liu, and F.-Y. Wang, "An efficient realization of deep learning for traffic data imputation," *Transportation Research Part C: Emerging Technologies*, vol. 72, no. 11, pp. 168–181, 2016. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0968090X16301826>
- [42] Y. Wu, H. Tan, L. Qin, B. Ran, and Z. Jiang, "A hybrid deep learning based traffic flow prediction method and its understanding," *Transportation Research Part C: Emerging Technologies*, vol. 90, pp. 166–180, 2018. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0968090X18302651>
- [43] J. Kong, X. Fan, X. Jin, S. Lin, and M. Zuo, "A variational bayesian inference-based en-decoder framework for traffic flow prediction," *IEEE Transactions on Intelligent Transportation Systems*, 2023.
- [44] N. A. Asif, Y. Sarker, R. K. Chakrabortty, M. J. Ryan, M. H. Ahamed, D. K. Saha, F. R. Badal, S. K. Das, M. F. Ali, S. I. Moyeen, M. R. Islam, and Z. Tasneem, "Graph Neural Network: A Comprehensive Review on Non-Euclidean Space," *IEEE Access*, vol. 9, pp. 60588–60606, 2021.
- [45] X. Wang, Y. Ma, Y. Wang, W. Jin, X. Wang, J. Tang, C. Jia, and J. Yu, "Traffic flow prediction via spatial temporal graph neural network," in *Proceedings of the Web Conference*, 2020, pp. 1082–1092.
- [46] Y. Shin and Y. Yoon, "Pgen: Progressive graph convolutional networks for spatial-temporal traffic forecasting," *IEEE Transactions on Intelligent Transportation Systems*, 2024.
- [47] J. Chen, W. Wang, K. Yu, X. Hu, M. Cai, and M. Guizani, "Node connection strength matrix-based graph convolution network for traffic flow prediction," *IEEE Transactions on Vehicular Technology*, 2023.
- [48] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting," Feb. 2018.
- [49] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," *Proceedings of International Joint Conference on Artificial Intelligence*, 2018.
- [50] Z. Wu, S. Pan, G. Long, J. Jiang, X. Chang, and C. Zhang, "Graph WaveNet for deep spatial-temporal graph modeling," *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pp. 1907–1913, 2019.
- [51] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "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*, 2020, pp. 753–763.
- [52] Z. Li and W. Zhu, "Spatial-temporal fusion graph neural networks for traffic flow forecasting," *arXiv:2101.11174*, 2021.
- [53] J. Choi, H. Choi, J. Hwang, and N. Park, "Graph neural controlled differential equations for traffic forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, 2022, pp. 6367–6374.

- [54] K. Guo, Y. Hu, Z. Qian, H. Liu, K. Zhang, Y. Sun, J. Gao, and B. Yin, "Optimized graph convolution recurrent neural network for traffic prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 2, pp. 1138–1149, 2020.
- [55] Y. Wang, C. Jing, W. Huang, S. Jin, and X. Lv, "Adaptive spatiotemporal inceptionnet for traffic flow forecasting," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 4, pp. 3882–3907, 2023.
- [56] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30, 2017. [Online]. Available: [https://proceedings.neurips.cc/paper\\_files/paper/2017/file/3f5ee243547dee91bd053c1c4a845aa-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91bd053c1c4a845aa-Paper.pdf)
- [57] M. Xu, J. Gao, H. Yao, Z. Ye, B. Liu, and F. Fan, "Spatial-temporal transformer networks for traffic flow forecasting," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, 2020, pp. 1234–1240.
- [58] C. Zheng, X. Fan, C. Wang, and J. Qi, "GMAN: A graph multi-attention network for traffic prediction," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 1, pp. 1234–1241, 2020.
- [59] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, "Attention based spatial-temporal graph convolutional networks for traffic flow forecasting," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 1, pp. 922–929, Jul. 2019. [Online]. Available: <https://ojs.aaai.org/index.php/AAAI/article/view/3881>
- [60] K. Ramana, G. Srivastava, M. R. Kumar, T. R. Gadekallu, J. C.-W. Lin, M. Alazab, and C. Iwendi, "A vision transformer approach for traffic congestion prediction in urban areas," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 4, pp. 3922–3934, 2023.
- [61] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," May 2016.
- [62] B. Zhang and R. Sennrich, "Root mean square layer normalization," *Advances in Neural Information Processing Systems*, vol. 32, 2019.
- [63] J. Wang, W. Jiang, and J. Jiang, "LibCity-Dataset: A standardized and comprehensive dataset for urban spatial-temporal data mining," *Intelligent Transportation Infrastructure*, p. liad021, 2023.
- [64] C. Song, Y. Lin, S. Guo, and H. Wan, "Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, 2020, pp. 914–921.
- [65] L. Liu, J. Zhen, G. Li, G. Zhan, Z. He, B. Du, and L. Lin, "Dynamic spatial-temporal representation learning for traffic flow prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 11, pp. 7169–7183, 2020.
- [66] Z. Pan, Y. Liang, W. Wang, Y. Yu, Y. Zheng, and J. Zhang, "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*, 2019, pp. 1720–1730.
- [67] H. Liu, Z. Dong, R. Jiang, J. Deng, J. Deng, Q. Chen, and X. Song, "Staeformer: Spatio-temporal adaptive embedding makes vanilla transformer sota for traffic forecasting," 2023.
- [68] J. D. Hamilton, *Time Series Analysis*. Princeton, NJ: Princeton University Press, 1994, vol. 2.
- [69] H. Drucker, C. J. Burges, L. Kaufman, A. J. Smola, and V. Vapnik, "Support vector regression machines," in *Advances in Neural Information Processing Systems*, 1997, pp. 155–161.
- [70] H. Yao, F. Wu, J. Ke, X. Tang, Y. Jia, S. Lu, P. Gong, J. Ye, and Z. Li, "Deep Multi-View Spatial-Temporal Network for Taxi Demand Prediction," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, Apr. 2018.