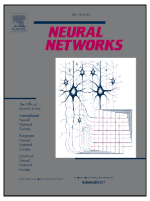
Contents lists available at ScienceDirect

Neural Networks

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



Full Length Article

Enhancing intelligent transportation systems with a more efficient model for long-term traffic predictions based on an attention mechanism and a residual temporal convolutional network

Selim Reza[a], Marta Campos Ferreira[b], J. J. M. Machado[c], João Manuel R.S. Tavares[c,*]

[a] Faculdade de Engenharia, Universidade do Porto, Rua Dr. Roberto Frias, s/n, 4200-465, Porto, Portugal
[b] INESC TEC, Departamento de Engenharia e Gestão Industrial, Faculdade de Engenharia, Universidade do Porto, Rua Dr. Roberto Frias, s/n, 4200-465, Porto, Portugal
[c] Instituto de Ciência e Inovação em Engenharia Mecânica e Engenharia Industrial, Departamento de Engenharia Mecânica, Faculdade de Engenharia, Universidade do Porto, Rua Dr. Roberto Frias, s/n, 4200-465, Porto, Portugal

ARTICLE INFO

Keywords:
Multi-step prediction
Dynamic feature embedding
Deep linear projection
Spatial-temporal relationship
Gating mechanism
Residual temporal convolutional network

ABSTRACT

Accurate traffic state prediction is fundamental to Intelligent Transportation Systems, playing a critical role in optimising traffic management, improving mobility, and enhancing the efficiency of transportation networks. Traditional methods often rely on feature engineering, statistical time-series approaches, and non-parametric techniques to model the inherent complexities of traffic states, incorporating external factors such as weather conditions and accidents to refine predictions. However, the effectiveness of long-term traffic state prediction hinges on capturing spatial-temporal dependencies over extended periods. Current models face challenges in dealing with (i) high-dimensional traffic features, (ii) error accumulation for multi-step prediction, and (iii) robustness to external factors effectively. To address these challenges, this study proposes a novel model with a Dynamic Feature Embedding layer designed to transform complex data sequences into meaningful representations and a Deep Linear Projection network that refines these representations through non-linear transformations and gating mechanisms. These two features make the model more scalable when dealing with high-dimensional traffic features. The model also includes a Spatial-Temporal Positional Encoding layer to capture spatial-temporal relationships, masked multi-head attention-based encoder blocks, and a Residual Temporal Convolutional Network to process features and extract short- and long-term temporal patterns. Finally, a Time-Distributed Fully Connected Layer produces accurate traffic state predictions up to 24 timesteps into the future. The proposed architecture uses a direct strategy for multi-step modelling to help predict timesteps non-autoregressively and thus circumvents the error accumulation problem. The model was evaluated against state-of-the-art baselines using two benchmark datasets. Experimental results demonstrated the model's superiority, achieving up to 21.17 % and 29.30 % average improvements in Root Mean Squared Error and 3.56 % and 32.80 % improvements in Mean Absolute Error compared to the baselines, respectively. The Friedman Chi-Square statistical test further confirmed the significant performance difference between the proposed model and its counterparts. The adversarial perturbations and random sensor dropout tests demonstrated its good robustness. On top of that, it demonstrated good generalizability through extensive experiments. The model effectively mitigates error accumulation in multi-step predictions while maintaining computational efficiency, making it a promising solution for enhancing Intelligent Transportation Systems.

1. Introduction

Traffic state prediction plays a critical role in intelligent transportation systems (ITS), serving as a foundation for optimising traffic management, mitigating congestion, and improving mobility. Accurate predictions of traffic states, such as flow, speed, and volume, are essential to addressing challenges posed by the increasing number of vehicles on roadways (Xing et al., 2022). These challenges include heightened accident rates, extended travel times, rising fuel consumption, and elevated greenhouse gas emissions (Miner et al., 2024). By anticipating

* Corresponding author.
E-mail addresses: up202003355@fe.up.pt (S. Reza), mferreira@fe.up.pt (M.C. Ferreira), jjmm@fe.up.pt (J. J. M. Machado), tavares@fe.up.pt (J.M.R.S. Tavares).

https://doi.org/10.1016/j.neunet.2025.107897
Received 2 February 2025; Received in revised form 21 May 2025; Accepted 21 July 2025
Available online 23 July 2025

traffic conditions, transportation planners can better organise networks and respond to critical events (Cao et al., 2024).

Traffic state prediction is inherently complex due to the stochastic, non-linear, and time-dependent nature of traffic conditions. Temporal dependencies arising from trends, cycles, seasonality, and disruptions caused by non-daily or extreme events, such as concerts or adverse weather, further complicate the task (Ansari Esfeh et al., 2020; Li et al., 2024a; Sharma et al., 2024). Traditional methods, such as Auto-Regressive Integrated Moving Average (ARIMA) and its variations, often fall short in handling such complexities (Ma et al., 2020; Xu et al., 2021). Similarly, while Deep Learning (DL) algorithms, such as Recurrent Neural Networks (RNNs), have proven effective for modeling short- and medium-term dependencies (Cheng et al., 2022; Yang et al., 2019), they suffer from vanishing or exploding gradients, limiting their ability to capture long-term temporal dependencies (Ribeiro et al., 2020).

Advances such as Long Short-Term Memory (LSTM) networks and their combinations with Convolutional Neural Networks (CNNs) or attention mechanisms have addressed some limitations, but challenges remain. LSTM networks, while capable of learning from moderately long sequences, are computationally expensive and struggle with long-term dependencies due to their sequential processing nature (Reza et al., 2022). Transformer-based models offer an alternative, leveraging attention mechanisms to capture global dependencies effectively. However, their architectural constraints, high computational cost, and lack of means to capture spatial-temporal correlations hinder their applicability to long-term traffic state prediction (Chen et al., 2022; Wen et al., 2023; Zhang et al., 2022).

Efforts to enhance prediction accuracy often rely on merging traffic state data with external factors, such as weather and accident information, alongside denoising techniques to remove data outliers (Chen et al., 2021, 2020; Essien et al., 2021; Guo et al., 2015). While these approaches yield competitive results for short- and medium-term predictions, their dependence on additional data sources and preprocessing limits their scalability. Moreover, their effectiveness in capturing long-term spatial-temporal dependencies remains limited.

Recent attention-based models have addressed some of these challenges. For instance, Tedjopurnomo et al. (2023) introduced a time-day embedding mechanism to enhance long-term traffic pattern learning, while Jiang et al. (2023) incorporated spatial self-attention to capture dynamic spatial relationships. On the other hand, Yu et al. (2024) proposed a spatial-temporal Autoformer leveraging autocorrelation within data sequences. Despite these advancements, these models are computationally demanding, and their performance for long-term predictions, particularly in mitigating error accumulation, requires improvement.

This study addresses the critical challenges associated with long-term traffic state prediction tasks, eliminating the dependence on weather data and data outlier removal techniques. To this end, a novel Dynamic Feature Embedding (DFE) mechanism was developed to facilitate meaningful transformations of feature representations. A Deep Linear Projection (DLP) block is proposed to refine these representations by applying non-linear transformations and a gating mechanism. Furthermore, a Spatial-Temporal Positional Encoding (STPE) mechanism is introduced to augment the data for each sensor at every time step with spatial-temporal positional information. The enhanced features are subsequently encoded using a masked multi-head attention-based encoder block, followed by their processing through a Residual Temporal Convolutional Network (RTCN) block designed to extract both short- and long-term temporal dependencies. Finally, a Time-Distributed Dense layer enables the model to generate predictions for up to 24 timesteps into the future.

The proposed methodology was validated using traffic flow data from the Caltrans Performance Measurement System (PeMS) database (Chen, 2002) and the PEMS-BAY traffic speed dataset (Li et al., 2018a). Baseline models representing the state-of-the-art in long-term traffic state prediction tasks were implemented and trained on the same datasets to ensure comprehensive and unbiased performance comparisons. Extensive experimental evaluations demonstrated that the proposed model consistently outperformed the baseline models across most evaluated metrics, with particularly noticeable improvements observed for longer prediction horizons. The main contributions of this study are:

- This research proposes a Dynamic Feature Embedding mechanism to facilitate meaningful representation of the normalised inputs and a Deep Linear Projection block to refine these representations by applying non-linear transformations and a gating mechanism. These techniques help the model to learn the relationship between traffic state features more effectively.
- It proposes a Spatial-Temporal Positional Encoding mechanism to augment the data for each sensor at every time step with spatial-temporal positional.
- It also incorporates a Residual Temporal Convolutional Network to efficiently process the encoded data representation and capture both short- and long-term temporal features. The residual connection within various stages of architecture helps the model to be deep enough to learn the outliers to achieve adequate prediction accuracy over extended time horizons.
- It eliminates dependency on external data sources, such as weather or accident information, and preprocessing steps, such as data outlier removal, while maintaining robust performance.

The remainder of this article is organised as follows: Section 2 provides a comprehensive summary of the selected state-of-the-art models, highlighting their performance and limitations. Section 3 introduces the proposed model in detail, while the experimental setup and corresponding results are presented in Section 4. Section 5 discusses the overall performance of the proposed model, with an emphasis on the impact of various model parameters. Finally, the conclusions are pointed out in Section 6.

2. Related works

This section presents state-of-the-art traffic state prediction models and discusses their main contributions based on the methods and datasets used. Traffic state features are stochastic and non-linear because of their spatial-temporal relationships and different seasonality granularity, such as monthly, weekly, and daily. Consequently, parametric approaches with linearity cannot provide high prediction performance, motivating greater attention to non-parametric strategies such as DL-based strategies. According to Yang et al. (2019), models based on DL usually perform better due to their ability to handle indeterministic and complex time-series traffic datasets. Therefore, many DL-based methods have been extensively tested, adapted, developed, and tuned for traffic prediction applications in the past few years.

Conventional DL-based models like RNNs and CNNs were used to address traffic state prediction problems with limited success (Lv et al., 2018; Polson & Sokolov, 2017; Xiao & Yin, 2019). With the introduction of the attention mechanism (Vaswani et al., 2017), these approaches gained importance among researchers to overcome their limitations. Du et al. (2020) focused on capturing short-term information and considered multiple modes to improve the prediction performance. The suggested model uses an attention-based auxiliary architecture focused on learning the spatial-temporal correlation and long-term interdependence of multimodal traffic data. The base module of their method consists of one-dimensional CNNs and GRUs with the attention mechanism, where the CNNs and GRUs capture the local trend features, such as the relationships between the data and the extended temporal dependencies. The proposed architecture achieved an MAE of 5% on the PeMS dataset. A similar attention mechanism was proposed by Vijayalakshmi et al. (2021) to improve the performance further by using the same dataset to implement a multi-step prediction model, combining CNNs with LSTMs and an attention mechanism. The CNNs helped to extract specific features and relationships of the data samples, and the LSTM networks allowed them to maintain the recurrence. The attention mechanism supported identifying near-term traffic data points, for example, daily seasonality, such as hours of the day, which are very important for predicting future traffic states. The results showed that the proposed model provided better accuracy than the considered baselines. Similar mechanisms were proposed to overcome the current shortcomings with some success (Grigsby et al., 2023; Li et al., 2019; Lim et al., 2021; Reza et al., 2022).

Chauhan et al. (2024) incorporated an attention mechanism with Bi-GRU networks, aiming to focus on the most recent relevant information of the data sequence to make future predictions. Conventional feature engineering was performed to extract weekdays or holidays and passed to the model as inputs. It needed comprehensive comparisons with recent models and could not effectively capture long-term dependencies due to its sequential processing of each element simultaneously, constraining its ability to model extended temporal relationships. Also, errors accumulate with the increase in prediction time horizons. Similar methods suffer from the same problems because of their architectures (Li et al., 2024b). Zou et al. (2024) presented a spatial-temporal transformer network to address the shortcomings of vanilla transformer models. The authors incorporated spatial-temporal tokens and a positional encoding mechanism to capture spatial long-term dependencies. A traffic accident prediction model using a transformer was presented in the works by Al-Thani et al. (2024), where an advanced multi-head attention mechanism was built to overcome the limitations of CNNs and LSTMs. However, it lacks comprehensive experiments for long-term prediction problems and suffers from the accumulation of errors. Moreover, the positional encoding layer lacks an efficient means to capture spatial-temporal positional relationships of the traffic states.

Graph Neural Networks (GNN) with attention mechanisms were explored to overcome the drawbacks of capturing spatial-temporal dependencies. Luo et al. (2024) presented a spatial-temporal GNN with a long-short term transformer network to capture the periodic and short-term features and long-term trends to perform future predictions. It obtained $5.63 - 16.78\,\%$ improvements compared to the baselines. However, the proposed model suffers from accumulation of error problems and is computationally demanding. Chen et al. (2024) designed a traffic flow matrix to represent the inter-node relationship better and used an attention mechanism-based Graph Convolutional Network (GCN) to make future predictions. Although the results demonstrated outperformance than the baselines, it suffered from predictive stability and lacked robustness. Aiming to enhance the model's robustness, particularly to non-recurrent traffic incidents, Geng et al. (2024) proposed a transformer-based GNN model. The authors developed a multidimensional embedding method to effectively fuse spatial-temporal features and a gated temporal self-attention technique to capture local and global temporal features. It was modelled to predict only 12 timesteps, and more experimental analyses are required to make efficient long-term predictions. Liu et al. (2024) incorporated prior physical knowledge into the adjacency matrix to improve the performance of GCN. Xu et al. (2024) also explored a similar model. However, these models are computationally demanding, and more performance improvements are necessary.

In general, the aforementioned methods have common drawbacks: (i) lack of efficiency in addressing the problems of prediction over longer time horizons, (ii) error accumulation for multi-step prediction, and (iii) high computational complexity. This research aims to address these particular problems.

3. Methodology

Traffic state prediction is usually modelled as a time-series prediction problem where attempts are taken to predict its future state based on a series of past samples at regular intervals. Let's assume that X represents the dataset, partitioned into overlapping data windows using a sliding window approach. Each window comprises input and label sequences spanning S timesteps. For a sequence of traffic state data samples $X_t \epsilon X$, X_t denotes the observed traffic state at the t-th time interval, where $t = 1, 2, \dots, T$. The model uses learned patterns from preceding observations across sliding windows to predict the traffic state X_{T+S} at the next S intervals. The prediction horizon, S, specifies the number of future timesteps for which the traffic states are predicted, enabling long-term multi-intersection traffic state prediction.

The main challenges of RNN-based models for traffic state prediction are to (i) capture long-range dependencies, (ii) address gradient vanishing and exploding phenomena, (iii) reduce the number of training steps, and (iv) enhance efficient parallel computation. An attention mechanism can address most of these shortcomings primarily because of its architecture.

3.1. Attention mechanism

From the X dataset, the v_i values can be retrieved based on a given query q_i and corresponding keys k_i, through a probabilistic approach. For a given query, the attention mechanism assesses the degree of alignment between itself and all keys and calculates the result as a weighted sum of values. The weights show the similarity between the query and each key, indicating the relative relevance of each value concerning the query. This process is formally expressed as:

$$\text{Attention}(q, k, v) = \sum_i \text{similarity}(q, k_i) \cdot v_i. \tag{1}$$

The output corresponds to the value associated with the key having the highest similarity. The similarity computation is represented as s_i, a function of the query and key. This function can have numerous forms, such as matrix dot product, $q_i^T \otimes k_i$, scaled dot product, $\frac{q_i^T \otimes k_i}{\sqrt{d}}$, general dot product, $q_i^T \otimes W k_i$, or additive similarity, $W_q^T \otimes q_i + W_k^T \otimes k_i$, as follows (Vaswani et al., 2017):

$$s_i = (q_i, k_i) = \begin{cases} q_i^T \otimes k_i, \\ \frac{q_i^T \otimes k_i}{\sqrt{d}}, \\ q_i^T \otimes W k_i, \\ W_q^T \otimes q_i + W_k^T \otimes k_i, \end{cases} \tag{2}$$

where d denotes the dimensionality of each key, and W is a collection of weights used to transform the query into a new space, allowing it to learn the appropriate content. The self-attention mechanism produces the following output:

$$S_A = Self\,Attention(Q, K, V) = \sum_i a_i \otimes v_i, \tag{3}$$

where the attention weights a_i is computed using $a_i = \frac{e^{s_i}}{\sum_j e^{s_j}}$. The multi-head attention is then derived by concatenating all the outputs of S_A across the h heads using:

$$M_A \in \mathbb{R}^{B \times T \times D} = Concat(S_{A_1}, S_{A_2}, \dots, S_{A_h}), \tag{4}$$

where h is the number of attention heads, and each S_{A_i} corresponds to the output of the i-th self-attention head. The proposed model incorporates a masked multi-head attention mechanism to enhance the encoder's performance. The fundamental concept is to mask certain values, effectively nullifying their probabilities and ensuring that the output at any step depends only on previous outputs and not on future ones. If M denotes the mask matrix consisting of elements 0 (zero) and $-\infty$, the masked attention can be computed by adapting one of the methods presented in Eq. (2) and following the previously described procedures. For example, using the scaled dot-product attention:

$$s_i = (q_i, k_i) = \frac{q_i^T \otimes k_i + M}{\sqrt{d}}. \tag{5}$$

3.2. Layer normalization

A layer normalisation (LayerNorm) helps to reduce the number of steps needed by the gradient descent algorithm to optimise the network. A network with multiple layers consists of weights in each layer, which are trained by gradient descent. The computation of the gradient of one set of weights depends on the outputs of the preceding and subsequent layers, which results in slow convergence. To address the problem, a LayerNorm operation is required, ensuring that the output of that layer, regardless of the setup of the weights, is the same by calculating the mean and variance. Let's suppose a mini-batch, B, of inputs, X, is $B = \{X_1, X_2, X_3, \ldots, X_m\}$, and each sample, X_i, contains N elements so that $flatten(X_i) = \{X_{i,1}, X_{i,2}, X_{i,3}, \ldots, X_{i,N}\}$, and their mean and variance are:

$$\mu_i = \frac{1}{N} \sum^{N} X_{i,N}, \tag{6}$$

$$\sigma_i^2 = \frac{1}{N} \sum^{N} (X_{i,N} - \mu_i)^2. \tag{7}$$

Then, each sample is normalised so that it possesses 0 (zero) mean and unit variance, such as:

$$\hat{X}_{i,N} = \frac{X_{i,N} - \mu_i}{\sqrt{\sigma_i^2 + \epsilon}}, \tag{8}$$

where ϵ is a stability factor that prevents any case of the denominator becoming 0 (zero). Finally, after performing some scaling and shifting, the output of the LayerNorm becomes (Ba et al., 2016):

$$L_N = \gamma \hat{X}_{i,N} + \beta, \tag{9}$$

where γ and β are learnable parameters. Now, when a model becomes very deep, most often, it tends to learn the statistical noise of the dataset, resulting in over-fitting problems. Dropout layers are used to address this phenomenon (Baldi & Sadowski, 2013).

3.3. Spatial-temporal positional encoding

Ordering traffic state data within a sequence is critical for prediction tasks, and the positional embedding (PE) mechanism is often used in attention techniques to record sequential ordering. However, the typical PE mechanism cannot capture the spatial and temporal correlations required for long-term prediction tasks. This research suggests an STPE mechanism to overcome this constraint. The primary goal is to supplement the data for each sensor at each time step with spatial and temporal location information.

A spatial embedding mechanism is built to capture a trainable representation of sensors in a vector space to accomplish it. Let's suppose that S is the number of sensors, then $S_e \in \mathbb{R}^{S\times D}$ represents the spatial embedding matrix, where each row corresponds to the embedding of a specific sensor. The spatial embedding for the i sensor can be retrieved as $s_{e_i} \in \mathbb{R}^D = S_e[i]$.

Now, the temporal encoding can be computed as a sinusoidal function of the position using (Vaswani et al., 2017):

$$T_e(t) \in \mathbb{R}^{T\times D} = \begin{cases} \sin\left(t \cdot \frac{1}{10000^{2i/D}}\right), & \text{for even indices of } i, \\ \cos\left(t \cdot \frac{1}{10000^{2i/D}}\right), & \text{for odd indices of } i, \end{cases} \tag{10}$$

where t and i represent the time step and dimension index, respectively, and the $10000^{2i/D}$ factor ensures that the encoding for different dimensions has varying frequencies. Therefore, the STPE can then be defined as:

$$E_{STPE} \in \mathbb{R}^{T\times S\times D} = S_e[i] + T_e(t). \tag{11}$$

To obtain an output of size $B \times T \times D$, the following steps are performed: (i) a reduction operation is applied throughout the spatial dimension, and (ii) the result is broadcast throughout the batch size, B. Thus, the final output of the STPE layer is computed as:

$$O_{STPE} \in \mathbb{R}^{B\times T\times D} = X + E_{STPE}, \tag{12}$$

where $X \in \mathbb{R}^{B\times T\times D}$ is the input.

3.4. Dynamic feature embedding

The input features of a usual traffic state dataset are complex and high-dimensional. To transform it into a meaningful representation, this research uses a DFE layer. The aim is to help the model capture complex spatial-temporal relationships within the data samples to improve long-term prediction performance. Let's suppose that S is the number of features of the raw input, $x \in \mathbb{R}^{B\times T\times S}$, then the DFE layer uses a learnable linear transformation to transform it into size $(B \times T \times D)$, where D is the model dimension using:

$$\hat{x} \in \mathbb{R}^{B\times T\times D} = x w_d + b_d, \tag{13}$$

where w_d and b_d represent the weight and bias matrices.

3.5. Deep linear projection

The proposed DPL technique works as an intermediate-level feature transformation platform to enhance their representation using non-linear transformations and gating mechanisms. It is used after the DFE layer to refine and project existing temporal feature representations into a desired dimension. Let's assume now that the input is $\hat{x} \in \mathbb{R}^{B\times T\times D}$. Thus, it does a non-linear transformation with ReLU at each time step, followed by a gating mechanism with the sigmoid activation function. Then, an element-wise gating operation is done to obtain $\hat{x}_2 \in \mathbb{R}^{B\times T\times d_{hidden}}$ as:

$$\hat{x}_1 \in \mathbb{R}^{B\times T\times d_{hidden}} = ReLU(\hat{x} w_1 + b_1), \tag{14}$$

$$g \in \mathbb{R}^{B\times T\times d_{hidden}} = \sigma(\hat{x} w_2 + b_2), \tag{15}$$

$$\hat{x}_2 \in \mathbb{R}^{B\times T\times d_{hidden}} = \hat{x}_1 \odot g, \tag{16}$$

where w_i and b_i represent the weight and bias matrices, and d_{hidden} is the hidden dimension. Now, $\hat{x}_2 \in \mathbb{R}^{B\times T\times d_{hidden}}$ is projected back to $\hat{x}_2 \in \mathbb{R}^{B\times T\times D}$ using a linear projection mechanism using:

$$\hat{x}_3 \in \mathbb{R}^{B\times T\times D} = \hat{x}_2 w_3 + b_3. \tag{17}$$

Subsequently, a learnable scaling is introduced to allow the model to optimise the amplitude of features dynamically based on the data and training process. A parameter, α, equal to the $\sqrt{D}$, is used to achieve the purpose. Also, a LayerNorm is introduced to make the model's training more stable and speed up its convergence. This ensures that the values are centred and have a consistent scale using:

$$\hat{x}_4 \in \mathbb{R}^{B\times T\times D} = \alpha \hat{x}_3, \tag{18}$$

$$\hat{x}_5 \in \mathbb{R}^{B\times T\times D} = L_N(\hat{x}_4). \tag{19}$$

Finally, a residual connection is used to allow the DPL layer to directly propagate the original input, $\hat{x} \in \mathbb{R}^{B\times T\times D}$, along with the transformed output using:

$$\hat{x}_{proj} \in \mathbb{R}^{B\times T\times D} = \hat{x} w_4 + b_4, \tag{20}$$

$$Y_{DPL} \in \mathbb{R}^{B\times T\times D} = \hat{x}_{proj} + \hat{x}_5. \tag{21}$$

3.6. Residual temporal convolutional network

This research incorporated an RTCN to preserve the temporal order of the sequences and efficiently learn both short and long-term dependencies. It applies a series of transformations on the data sequences using causal convolutions (Hamad et al., 2021), batch normalisation (Bjorck et al., 2018), ReLU activations, and dropout for regularisation.

It also contains a residual connection to allow deeper network construction. For an input $X \in \mathbb{R}^{B\times T\times D}$, the RTCN first applies a one dimensional convolution (1DCNN) with dilation using (Kwon et al., 2021):

$$C_t \in \mathbb{R}^{B\times T\times F} = \sum_{k=0}^{K-1} X_{t-k} \cdot w_k, \tag{22}$$

where F denotes the number of filters, $C_t \in \mathbb{R}^{B\times T\times F}$ represents the output at the t time step, K denotes the size of the convolutional kernel, and k indexes the kernel weights, specifying the contribution of the corresponding input sequence segment to the output at t. Furthermore, w_k refers to the weight of the k^{th} kernel filter, and X_{t-k} signifies the input element at the $t-k$ time step. The padding is causal to prevent information leakage from future timesteps, i.e., ensuring that predictions at t depend only on information from timesteps, $\le t$. It also helps to increase the receptive field without increasing the computational cost, which is key in modelling long-range dependencies. $C_t \in \mathbb{R}^{B\times T\times F}$ then undergoes batch normalization and ReLU activation followed by a spatial dropout layer to get $\bar{C}_t \in \mathbb{R}^{B\times T\times F}$. It helps stabilise the model's training and introduce non-linearity.

One of the most important parts of an RTCN is the residual connection, aiming to enhance model stability and facilitate gradient flow. Specifically, a skip connection from the input, $X_t \in \mathbb{R}^{B\times T\times D}$, is added to the output, $\bar{C}_t \in \mathbb{R}^{B\times T\times D}$, with F equals D. It is then passed through an additional ReLU activation function, ensuring a non-linear combination of the input and transformed features using:

$$R_t \in \mathbb{R}^{B\times T\times D} = \bar{C}_t + X_t, \tag{23}$$

$$Y_R \in \mathbb{R}^{B\times T\times D} = ReLU(R_t), \tag{24}$$

where $Y_R \in \mathbb{R}^{B\times T\times D}$ represents the residual output.

3.7. Encoder layer

The encoder layer processes the input sequence, $X \in \mathbb{R}^{B\times T\times D}$, where B is the batch size, T is the sequence length, and D is the model's feature dimension. It also takes a padding mask, $M \in \mathbb{R}^{B\times 1\times 1\times T}$, to exclude invalid positions during the attention computation. The processing begins with a layer normalisation step:

$$X_{norm} \in \mathbb{R}^{B\times T\times D} = L_N(X), \tag{25}$$

where L_N is defined by Eq. 9. Next, multi-head attention is applied using the normalised inputs and the padding mask:

$$A \in \mathbb{R}^{B\times T\times D} = M_A(X_{norm}, M), \tag{26}$$

where M_A computes the attention scores using the mechanism defined in Eq. 4. A dropout layer is then applied to the attention output:

$$A_{dropout} \in \mathbb{R}^{B\times T\times D} = D_o(A), \tag{27}$$

where D_o represents the dropout function. A residual connection adds the original input, X, to the attention output:

$$A_{residual} \in \mathbb{R}^{B\times T\times D} = X + A_{dropout}. \tag{28}$$

The second part of the encoder layer consists of a fully connected network with a gating mechanism. First, the residual output is normalised:

$$A_{norm} \in \mathbb{R}^{B\times T\times D} = L_N(A_{residual}). \tag{29}$$

The gating mechanism is then applied as follows:

$$g_e = \sigma(A_{norm} W_g + b_g), \tag{30}$$

$$f_e = ReLU(A_{norm} W_f + b_f), \tag{31}$$

$$z_e = g_e \odot f_e, \tag{32}$$

where g_e is the gate vector, f_e is the feature vector, $\odot$ denotes element-wise multiplication, and W_g, b_g, W_f, and b_f are learnable parameters of the fully connected layers. A feedforward network projects the output back to the model's feature dimension, D:

$$O_e \in \mathbb{R}^{B\times T\times D} = D_o(z_e W_o + b_o), \tag{33}$$

where W_o and b_o are the projection parameters, and D_o applies dropout. Finally, a residual connection is applied after the feedforward network to produce the final output of the encoder layer:

$$O_E \in \mathbb{R}^{B\times T\times D} = A_{residual} + O_e, \tag{34}$$

where O_E is the output of the encoder layer.

3.8. Proposed model

The core component of the proposed model is an encoder block composed of several encoder layers. Unlike RNN-based models, which process one value at a time in a sequence, the encoder block processes all data samples within the sequence, i.e., the data window simultaneously, enabling parallel computations.

Let the input be $X \in \mathbb{R}^{B\times T\times D}$ and the padding mask $M \in \mathbb{R}^{B\times 1\times 1\times T}$ computed as:

$$M = create\_padding\_mask\left(\sum_{d=1}^{D} X_{b,t,d}\right), \tag{35}$$

where $\sum_{d=1}^{D} X_{b,t,d}$ reduces the feature dimension for each time step t and create_padding_mask$(\cdot)$ is a function that generates a mask to exclude invalid positions based on the reduced values. Now, a DFE layer maps input features to a learned dynamic embedding space and then joins the original features and the embeddings along the feature dimension using:

$$dynamic\_features \in \mathbb{R}^{B\times T\times D} = \hat{x}(X), \tag{36}$$

$$Y_1 \in \mathbb{R}^{B\times T\times (D+D)} = concat(X, dynamic\_features). \tag{37}$$

Afterward, $Y_1 \in \mathbb{R}^{B\times T\times (D+D)}$ is passed through a DLP layer to produce $Y_2 \in \mathbb{R}^{B\times T\times D}$ and then is scaled using the square root of the model dimension D using:

$$Y_2 \in \mathbb{R}^{B\times T\times D} = Y_{DPL}(Y_1), \tag{38}$$

$$Y_2 \in \mathbb{R}^{B\times T\times D} = Y_2 \cdot \sqrt{D}, \tag{39}$$

where $\sqrt{D}$ ensures that the projection values are scaled appropriately for subsequent processing. A STPE layer is then used to capture the spatial-temporal features and positional relationships using:

$$Y_3 \in \mathbb{R}^{B\times T\times D} = O_{STPE}(Y_2). \tag{40}$$

A LayerNorm layer and a dropout layer are then used to stabilise the model training process. The encoder block has N identical encoder layers. Each output of the encoder layer is repeatedly updated over N to get:

$$Y_4^{(i)} \in \mathbb{R}^{B\times T\times D} = O_E(Y_3^{(i-1)}, M), \tag{41}$$

where $i \in \{1, 2, \dots, N\}$, and the after final iteration $Y_4^{(N} \in \mathbb{R}^{B\times T\times D}$ contains the refined and deeply encoded features of the input sequence. The RTCN block uses it to extract the temporal correlation of the timesteps and to help the Time-Distributed Fully Connected layer to make the final predictions using:

$$Y_5 \in \mathbb{R}^{B\times T\times U} = Y_R(Y_4^N), \tag{42}$$

$$Y_6 \in \mathbb{R}^{B\times T\times U} = D_o(Y_5), \tag{43}$$

$$Y_7 \in \mathbb{R}^{B\times T\times U} = ReLU(Y_6 w_y + b_y), \tag{44}$$

$$Y_o \in \mathbb{R}^{B\times T\times O} = TimeDistributed(Y_7 w_o + b_o), \tag{45}$$

where U and O are the number of filters of the RTCN block and output size of the model. The complete architecture of the proposed model is depicted in Fig. 1. One should note that the aforementioned equations were established considering only the forward pass processes for simplicity.

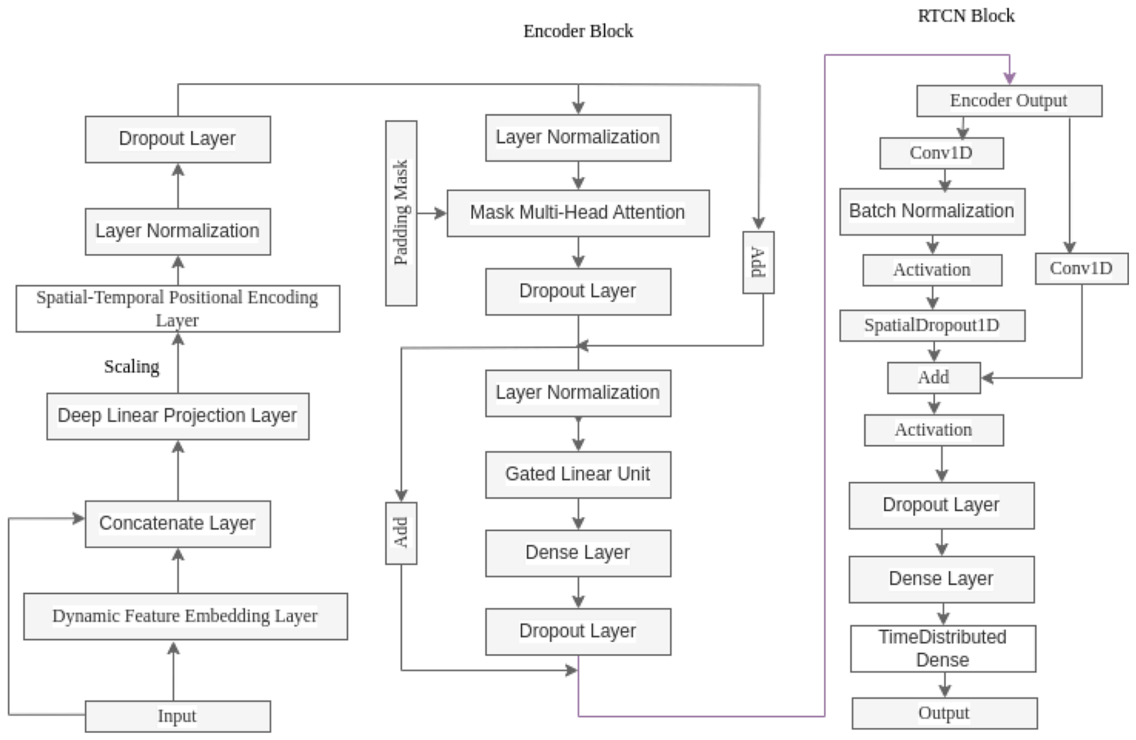


Fig. 1. Architecture of the proposed model.

3.9. *Weighted MSE-MAE loss function*

The proposed model calculates the loss with the help of a weighted MSE-MAE loss function described by:

$$loss(x_i, y_i) = MSE(x_i, y_i) + 0.5 \cdot MAE(x_i, y_i). \tag{46}$$

The goal is to minimise the loss value to support proper model learning effectively. Higher loss function values indicate that the model has not been learned properly. In backpropagation, the model uses gradient descent to update the weights based on an error, E, defined by the loss function. The aim is to calculate the gradient, $\frac{\partial E}{\partial w_i}$, where w_i represents the i^{th} weight of the network, and update the weight. Given the loss function, E can be expressed as:

$$E = \frac{1}{n}\sum_{i=1}^{n}\left[(x_i - y_i)^2 + 0.5 \cdot |x_i - y_i|\right], \tag{47}$$

where x_i and y_i represent the true and predicted values, respectively. The gradient $\frac{\partial E}{\partial w_i}$ is computed as:

$$\frac{\partial E}{\partial w_i} = -\frac{1}{n}\sum_{i=1}^{n}\left[2(x_i - y_i) + 0.5 \cdot sign(x_i - y_i)\right]\frac{\partial y_i}{\partial network_i}\frac{\partial network_i}{\partial w_i}, \tag{48}$$

where $sign$ represents the Signum function (Tatlıcıoğlu, 2024) and the term $\frac{\partial y_i}{\partial network_i}$ depends on the activation function. The $ReLU$ activation function is employed in the fully connected layers of the proposed model. The derivative of $ReLU$ for y_i can be computed as (Lederer, 2021):

$$\frac{d(ReLU)}{dy_i} = \begin{cases} 1 & \text{if } y_i > 0, \\ 0 & \text{if } y_i \leq 0. \end{cases} \tag{49}$$

If $\frac{\partial network_i}{\partial w_i} = k_i$, then the final gradient equation becomes:

$$\frac{\partial E}{\partial w_i} = -\frac{1}{n}\sum_{i=1}^{n}\left[2(x_i - y_i) + 0.5 \cdot sign(x_i - y_i)\right] \cdot ReLU'(y_i) \cdot k_i. \tag{50}$$

4. Experiments

The training of the proposed model requires less computational power compared to other transformer-based models, and hence, the training procedures can be completed on a local computer with an Intel Core i7-10750H CPU @ 2.60GHz × 12 processor and a memory of 16.0 GB. However, an NVIDIA Corporation TU117GLM [Quadro T1000 Mobile] Graphical Processing Unit (GPU) on a 64-bit Ubuntu 22.04.5 LTS operating system with a GNOME version of 42.9 was used to complete training and is recommended. The open-source Tensorflow machine learning library facilitated the coding tasks.

4.1. *Code implementation*

The proposed model was implemented in several stages, with the TensorFlow platform as the foundation. First, a scaled dot-product attention mechanism was built using the *tf.matmul* and *tf.nn.softmax* functions. The latter function computed the attention weights by scaling the dot product of the query and key tensors, normalised by the square root of the key's depth. An optional masking operation was integrated to handle padded sequences. Second, a multi-head attention module was implemented using the *tf.keras.layers.Dense* layer to project query, key, and value tensors. These tensors were split into multiple heads for parallel computation. The attention mechanism aggregated information from different subspaces and established residual connections for stability. The outputs of the attention module were concatenated and processed through a feedforward dense layer.

Third, a STPE layer was introduced to model spatial-temporal dependencies. Spatial embeddings for sensors were initialised as learnable parameters, and temporal encodings were generated with sinusoidal functions. These encodings were paired with input features to boost the model's representational power, using *tf.broadcast_to* and *tf.reduce_sum* for operations across batches. Fourth, the transformer encoder was designed using pre-normalisation with *tf.keras.layers.LayerNormalization*, multi-head attention, and a feedforward network with gating mechanisms. Gated linear units were implemented to control feature selection dynamically. Residual connections ensured stable gradient flow during training. The encoder block was stacked multiple times, and temporal convolutional blocks were added to refine temporal patterns using the *Conv1D* and *Add* layers.

Finally, a DFE layer and a DLP layer were used to embed input features into a higher-dimensional space. These embeddings were processed through a dense layer with a learnable scaling factor and combined with positional encodings. The entire architecture was trained using a composite loss function combining the MSE and MAE metrics. The Nadam optimiser with a cosine decay learning rate schedule ensured an efficient optimisation. Also, the Friedman Chi-Square function was developed using the *scipy.stats* module to facilitate the statistical testing mechanism.

4.2. Evaluation metrics

The state-of-the-art MAE and RMSE metrics were used to assess the accuracy of the proposed model. Also, the Friedman Chi-Square test (Pereira et al., 2015) was conducted to find the statistical significance of the prediction distributions of the models under comparison. If x and y are the actual and predicted values, n is the number of samples, and x_i and y_i are values of the i^{th} samples in x and y, respectively, then their formulations have the following forms.

The mean absolute error is simply an arithmetic average of the absolute errors:

$$MAE(x,y) = \frac{1}{n}\sum_{i=1}^{N}|x_i - y_i|. \tag{51}$$

The root mean squared error is the standard deviation of the residuals, i.e., prediction errors, and, in terms of mathematics, has the form of:

$$RMSE(x,y) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - y_i)^2}. \tag{52}$$

Let's suppose that there are M models that are to be evaluated on D datasets, then the test statistic can be formulated as (Rainio et al., 2024):

$$F_s = \frac{(D-1)\chi_F^2}{D(M-1) - \chi_F^2}, \tag{53}$$

where χ_F^2 represents the Friedman Chi-Square function. F_s follows the F-distribution with $(M-1)$ and $(M-1)(D-1)$ degrees of freedom under the null hypothesis.

4.3. Datasets

The PEMS traffic flow dataset of the year 2012 was gathered. This dataset contains the measurements of date-time, the location identification (ID), and the number of vehicles passed by the sensors in a frequency of 5 minutes. Another Caltrans-PeMS dataset with data about the readers' locations was combined with the traffic information using SQL Server. The resultant dataset contained only two independent variables with missing values: Angle and City ID, which were not used in this study. Only the data acquired from readers installed in the Dwight D. Eisenhower Highway part of Interstate 80, between West Oakland and North San Francisco, in the USA, was used for this study. It was chosen based on high traffic flow peaks and the clear presence of seasonality. It consists of traffic flow measurements recorded over 23 days, 19 hours, and 5 minutes, from January 1, 2012, 00:00:00, to January 24, 2012, 19:05:00. It includes 6854 data entries (rows) and 154 variables (columns). The first column, *date_hour*, contains the timestamps of each data point in the format *YYYY-MM-DD HH:MM:SS*, with no missing values observed. The remaining 153 columns correspond to individual sensor IDs, i.e., 200, 201, and 300, representing traffic flow data. Notably, a few sensor columns contain minimal missing values, with the maximum being 1 (one) missing value per column. The data observations were taken regularly, providing high temporal resolution for traffic flow analysis.

Also, the PEMS-BAY dataset was used for the model's experimentation and validation. It contains traffic speed information collected over six months from January 1, 2017, to June 30, 2017, with 180 days and 23 hours, 55 minutes. It comprises 52,116 samples (rows) and 326 features (columns). The first column (*date_time*) represents the timestamps of the recordings, formatted as *YYYY-MM-DD HH:MM:SS*, with no missing values. The subsequent 325 columns represent sensor IDs, i.e., 400,001 and 400017, each containing continuous traffic speed data recorded. No missing values were identified across these columns, ensuring data completeness for analysis.

These two datasets were chosen because of their wide adaptability in traffic research, making them a community standard. Also, they are from different geographical locations with varying characteristics, demand profiles, and animalities, helping the model to test its robustness. Lastly, they are large-scale and thus can test the model's computational efficiency.

4.3.1. Data analysis

The potential outliers were investigated using the Interquartile range (IQR) method (Wan et al., 2014), testing both conservative, i.e., $1.5IQR$, and non-conservative, i.e., $3IQR$, variation measures. These variations were subtracted from the first quartile Q_1, i.e., 25% of the data, and added to the third quartile Q_3, i.e., 75% of the data, to define the lower inner fence (LIF), upper inner fence (UIF), lower outer fence (LOF), and upper outer fence (UOF), respectively, using:

$$\begin{aligned} IQR &= Q_3 - Q_1,\\ LIF &= Q_1 - 1.5IQR,\\ UIF &= Q_3 + 1.5IQR,\\ LOF &= Q_1 - 3IQR,\\ UOF &= Q_3 + 3IQR. \end{aligned} \tag{54}$$

A value may be classified as an outlier if it is not within these limits. Fig. 2 helps visualise the dataset's outliers using box plot analysis of traffic flow values hourly. In this figure, the x-axis and y-axis represent hours of the day and traffic flow (vehicles per 5 minutes), respectively.

Table 1 presents the obtained Q_1, Q_3, IQR, LIF, UIF, LOF, and UOF values of the traffic flow data samples of the whole dataset, respectively.

The traffic flow, i.e., the number of vehicles per unit of time, values for July 2012 possess a mean, median, and standard deviation of 65.62, 70.00, and 37.98, respectively. However, the mean, median, and standard deviation of their amplitude for December 2012 were equal to 74.71, 79.00, and 44.43, respectively. Again, for July 2013, their mean was equal to 56.28, and median and standard deviation were equal to 60.00 and 30.89, respectively. The traffic flow peak in December 2012 was 8.5% higher than that for July 2012. Interestingly, the traffic flow peaks in July 2013 were 19.2% lower than in July 2012.

Furthermore, the Isolation Forest algorithm (Liu et al., 2008) was applied to the dataset to double-check the presence of outliers. Fig. 3 helps to visualise the found dataset's outliers using this method.

In summary, the two analyses aforementioned reveal that the dataset contains some outliers, and this study aimed to propose a model that can efficiently deal with these outliers without sacrificing good accuracy.

Table 1
Summary of the PEMS dataset analysis to determine the outliers within the data samples based on the IQR method.

Q_1	Q_3	IQR	LIF	UIF	LOF	UOF
32.0	107.0	75.0	−80.5	219.5	−193	332

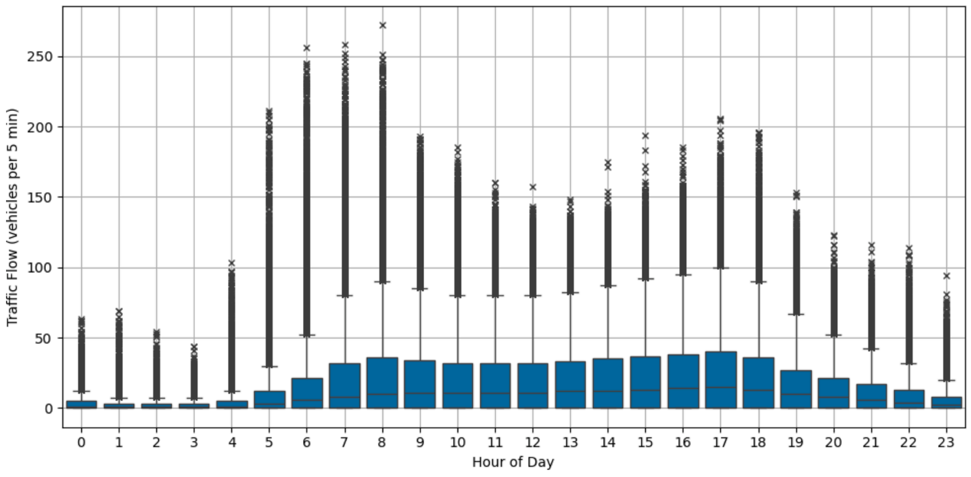


Fig. 2. Box plot analysis of hourly traffic of the PEMS traffic flow dataset (the whole dataset is grouped hourly to build these box plots, the × sign represents the outliers).

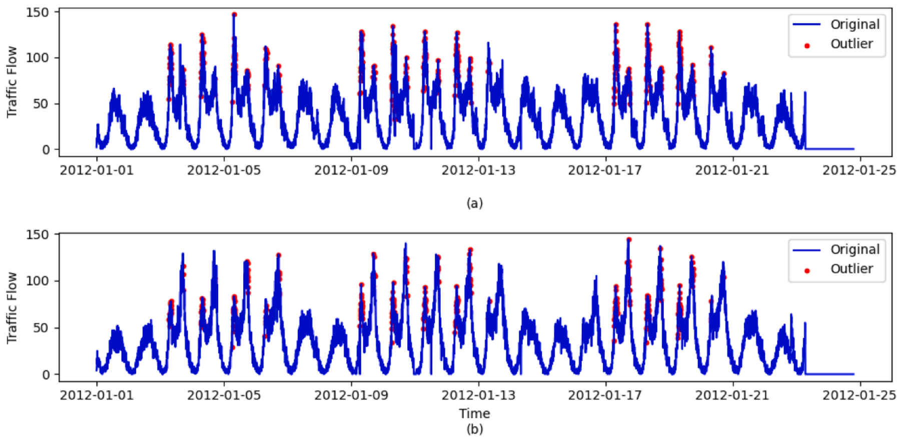


Fig. 3. Visualisation of outliers of the PEMS dataset for sensor ID (a) 24,001 and (b) 24,200 built after fitting it to the Isolation Forest algorithm.

4.4. Preprocessing

The traffic flow PEMS dataset contains only 6854 samples; hence, a data augmentation technique was used to generate two augmented samples per original sample. A random noise with a noise level of 0.01 generated synthetic samples, which were added to the original dataset. In this way, the data samples were increased to 20,454. However, the PEMS-BAY dataset does not require any data augmentation process. The missing values in the dataset were handled using two techniques. First, any missing values were forward-filled using the Forward fill method (Cenitta et al., 2021), propagating the last valid observation to fill subsequent missing entries. Then, any remaining missing values were interpolated using a linear method, which estimates missing values based on the values of neighbouring data points. This combination ensures that missing values are appropriately addressed before further processing. However, the PEMS-BAY dataset does not have any missing values; hence, these techniques do not apply to it. The PEMS-BAY and PEMS datasets have 325 and 153 sensors, respectively, and this research used all of them. The data was split into training, validation, and test datasets with a percentage ratio of $70 : 20 : 10$ and was scaled between values 0 (zero) to 1 (one) using MinMaxScaler (Patro & Sahu, 2015) to facilitate efficient model training. It was fit on the training set only to avoid data leakage. Applying the proposed model for prediction tasks requires creating appropriate data windows and correctly designating the inputs and labels. This research divided the training, validation, and test sets into input-output pairs for multi-step traffic state prediction. For example, it can be configured into 24 consecutive timesteps of input data features to predict the next 24 timesteps as the target, as shown in Fig. 4. Here, the first data window started with $t = 0$ timesteps and the next with $t = 1$ timesteps. This process continued until the training set could not have a sequence of 24 consecutive labels. Each input window overlaps with the preceding one by all but one timestep, ensuring temporal continuity.

The forward fill and interpolation methods were used to tackle the missing values. However, they can create artificial patterns, underestimate volatility, and fail in the case of prolonged missingness. Since these two datasets have a very low/zero number of missing values, these effects were avoided. Again, the MinMaxScaler may cause the amplification of sparse outliers, distorting relationships between multi-variate series. Lastly, the utilised data windowing technique eliminates trailing data points that do not conform precisely to the specified input-output windows, resulting in suboptimal utilisation of the dataset. Also, the model would find it challenging to learn the data patterns for a tiny pair of input-output windows.

4.5. Model training

The proposed model contains $2.64M$ and $816k$ trainable parameters for the PEMS-BAY and PEMS traffic flow datasets. Different hyperparameters can profoundly impact its outcomes. Hence, Keras Tuner provided a proper set of hyperparameters using the Random Search (Bergstra & Bengio, 2012) method, which ensured an efficient exploration of the hyperparameter space, as shown in Table 2, to identify optimal configurations. 89 and 83 trials were required for the PEMS and PEMS-BAY datasets to derive a final set of hyperparameters.

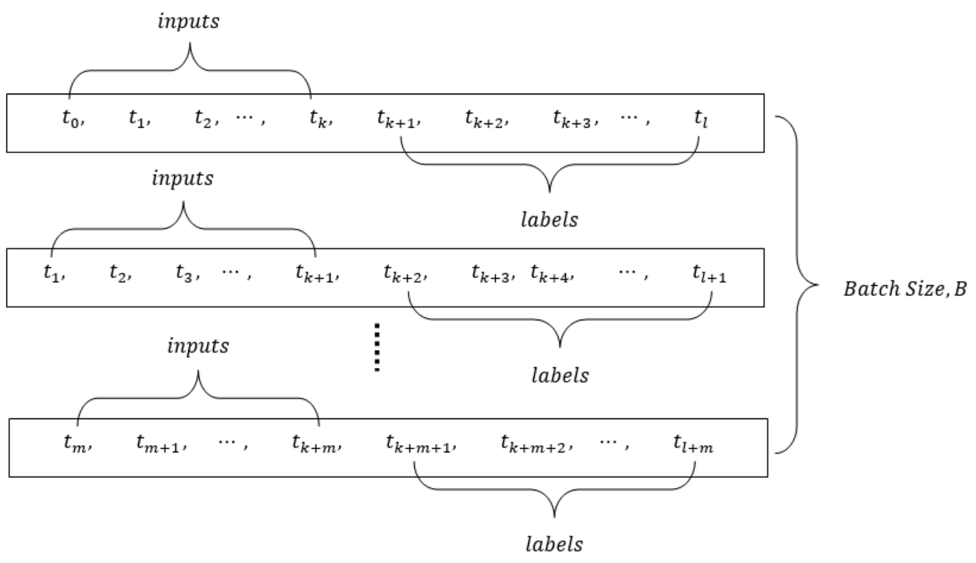

inputs
$t_0, t_1, t_2, \cdots, t_k, t_{k+1}, t_{k+2}, t_{k+3}, \cdots, t_l$
labels
inputs
$t_1, t_2, t_3, \cdots, t_{k+1}, t_{k+2}, t_{k+3}, t_{k+4}, \cdots, t_{l+1}$
labels
inputs
$t_m, t_{m+1}, \cdots, t_{k+m}, t_{k+m+1}, t_{k+m+2}, \cdots, t_{l+m}$
labels
Batch Size, B

Fig. 4. Visualisation of different data windows in a batch.

Table 2
Hyperparameter search space explored in this study (values in bold correspond to the optimal hyperparameters identified for the PEMS dataset, while values marked with an asterisk (*) represent the optimal hyperparameters for the PEMS-BAY dataset).

Hyperparameter	Values
Numb_heads (PEMS)	1, **3**, 9, 17, 51
Numb_heads (PEMS-BAY)	1, 3, 5, 13, 25*, 65
Numb_layers	1*, **2**, 3, ..., 15
Units	32, 64, 96, 128, **160**, 192, 224, 256*
Dropout rate	**0.1***, 0.2, 0.3, 0.4, 0.5
Learning rate	$1e^{-4}$, $\mathbf{5e^{-4}}$*, $1e^{-3}$

In Table 2, Numb_heads, Numb_layers, and Units represent the number of attention heads, encoder layers, and neurons of the fully connected layers, respectively. Two sets of optimal hyperparameters were derived for the two datasets used in this study. Given that the model dimension (D) corresponds to the number of sensors (153 for the PEMS and 325 for the PEMS-BAY datasets), the Numb_heads were selected from two distinct sets of values to ensure that the dimension per head is an integer. The other sets of values were selected as a common practice. The model was trained using the Nadam optimiser and a hybrid loss function, as described in Section 3.9. The traffic state values were normalised to the range $[0, 1]$ using the MinMax scaling technique to ensure stable and efficient training, thereby mitigating the risk of large gradients. Training the proposed model (fully tuned) to reach convergence (around 105 epochs) required approximately 10 to 17 minutes on a GPU with the mentioned configuration.

4.6. Results

The proposed model was trained based on the previously mentioned experimental setup with a combination of optimal hyperparameters. Its performance was evaluated using state-of-the-art metrics on two datasets, as outlined in Section 4.2. The model's performance was compared with the results obtained by previously published state-of-the-art methods designed to address medium to long-term prediction problems. These methods were selected due to their recency and widespread recognition within the research community. The used baselines were:

- LSTM-BiLSTM (Ma et al., 2021): The authors proposed a combination of an LSTM, a bidirectional LSTM, and Dense layers to address multi-step traffic flow prediction tasks. A similar model was developed, trained, and tested on the two datasets used in this study.
- CNN-LSTM (Lee & Park, 2024): This approach used a CNN, an LSTM, and fully connected layers to achieve accurate predictions. An identical model was implemented, trained, and tested on the two datasets employed in this research.
- TrafFORMER (Tedjopurnomo et al., 2023): The authors introduced a spatio-temporal multi-head attention-based transformer model with a time-day embedding mechanism. A comparable model was implemented and trained on the same datasets in this study.
- Autoformer (Wu et al., 2021): This method proposed a decomposition architecture with an auto-correlation mechanism for detecting dependencies and aggregating representations at the sub-series level. An identical model was implemented, trained, and tested on the two datasets employed in this research.
- Pdformer (Jiang et al., 2023): The authors introduced a propagation delay-aware dynamic long-range transformer model that employs a spatial self-attention mechanism. A comparable model was implemented and trained on the same datasets in this study.
- ST-Autoformer (Yu et al., 2024): This work presented a spatio-temporal autoformer model based on spatial-temporal sequence autocorrelation. A similar model was implemented and trained on the same datasets in this study.

Table 3 summarises the obtained MAE(%) and RMSE(%) scores of the proposed model and the baselines under study for different prediction horizons (3–24 timesteps) on the traffic flow PEMS test dataset. For the 3-step prediction horizon, the Autoformer model performed best by obtaining a 6.25 % lower MAE(%) score than the 2nd best model. Also, the LSTM-BiLSTM outperformed the other models for 3- and 6-step prediction lengths regarding MAE(%) and RMSE(%). However, the proposed model comprehensively outperformed all the baselines for 18- and 24-step prediction horizons regarding the MAE(%) and RMSE(%) scores. For an 18-step prediction horizon, it demonstrated a 0.6 % lower MAE compared to the next best model, LSTM-BiLSTM (3.10), and a 34.4 % lower RMSE compared to CNN-LSTM (8.16). For the 24-step prediction horizon, the proposed model's MAE was 38.3 % lower than the worst-performing model (LSTM-BiLSTM, 5.19).

For shorter timesteps (3 and 6), the proposed model underperformed compared to Autoformer, which had a 39.3 % lower MAE (1.95 vs. 3.21) and 17.2 % lower RMSE (4.66 vs. 5.64) at step 3. Overall, the proposed model and Autoformer outperformed the other models in their

Table 3
Performance of the proposed and baseline models for different prediction timesteps (3 - 24) on the PEMS test dataset based on MAE and RMSE scores (best and second-best values are in bold and italic, respectively).

Models	MAE					RMSE				
	3	6	12	18	24	3	6	12	18	24
LSTM-BiLSTM	*2.08*	**2.28**	**2.77**	*3.10*	5.19	**4.45**	**5.57**	7.06	8.16	11.38
CNN-LSTM	2.26	2.76	3.27	4.30	4.42	4.91	6.07	7.41	9.15	9.82
TrafFORMER	3.75	3.55	3.60	3.57	*3.66*	5.99	*5.74*	*5.86*	*5.79*	*5.88*
Autoformer	**1.95**	*2.33*	3.02	3.83	4.51	*4.66*	5.83	7.24	9.21	10.24
Pdformer	2.24	2.61	*2.94*	4.07	4.43	4.74	5.97	7.48	8.74	10.31
ST-Autoformer	2.99	3.05	3.51	3.95	4.59	5.47	5.89	6.86	7.73	8.75
Proposed	3.21	3.37	3.25	**3.08**	**3.19**	5.64	5.86	**5.62**	**5.35**	**5.57**

Table 4
Performance of the proposed and baseline models for different prediction timesteps (3 - 24) on the PEMS-BAY test dataset based on MAE and RMSE scores (best and second-best values are in bold and italic, respectively).

Models	MAE					RMSE				
	3	6	12	18	24	3	6	12	18	24
LSTM-BiLSTM	3.60	3.81	4.00	4.07	4.09	6.92	7.44	8.02	8.17	8.28
CNN-LSTM	3.93	4.00	4.26	4.46	5.87	7.53	7.91	8.42	8.84	10.70
TrafFORMER	*2.65*	*3.33*	*3.02*	*3.06*	*3.57*	*4.33*	*5.39*	*5.19*	*5.34*	*6.18*
Autoformer	4.01	4.29	4.52	4.65	4.91	7.29	7.81	8.73	9.08	9.28
Pdformer	3.93	4.23	4.35	4.48	4.50	7.32	8.19	8.38	8.65	8.84
ST-Autoformer	3.29	3.51	4.02	4.16	4.26	5.92	6.42	7.29	7.69	8.13
Proposed	**2.64**	**2.65**	**2.55**	**2.62**	**2.86**	**5.15**	**5.22**	**5.06**	**5.17**	**5.47**

respective ranges, while TrafFORMER consistently showed the least effective performance across most timesteps.

From Table 4, it is possible to verify that the presented model is more capable of accurately predicting long-term future traffic speed compared to the baselines under consideration in terms of the MAE and RMSE scores on the PEMS-BAY dataset. As to the used statistic, it achieved an MAE and RMSE improvement of $0.4 - 51.3\,\%$ and $2.5 - 48.9\,\%$ compared to the studied baselines for a prediction time horizon of $3 - 24$ steps, respectively. The proposed model consistently outperformed all other models, achieving the lowest MAE and RMSE scores across all timesteps in this case. For example, at step 12, it achieved an MAE of 2.55, which was $15.6\,\%$ lower than the second-best model, TrafFORMER, and an RMSE of 5.06, which was $2.5\,\%$ lower than 5.19 of the TrafFORMER model. Overall, it demonstrated superior performance across both short and long-term horizons for the PEMS-BAY dataset, highlighting its robustness and accuracy compared to baselines.

It is a well-known fact that the prediction accuracy of the conventional models tends to degrade rapidly with the increase in time horizon (Jia et al., 2016) because of the accumulation of errors. Interestingly, the presented model demonstrated that its prediction accuracy is independent of that trend, as shown by both Tables 3 and 4.

This research applied the Friedman Chi-Square statistical test to evaluate whether there are significant differences in the prediction distributions of the proposed model. Two hypotheses were considered: the null hypothesis, indicating no significant differences, and the alternative hypothesis, suggesting the presence of differences. A significance level of $5\,\%$ was used. The null hypothesis holds if the p-value exceeds the significance level. However, as shown in Table 5, none of the p-values exceeded the significance level, indicating that the null hypothesis is rejected and that the prediction distributions exhibit significant differences.

Fig. 5 helps to visualise the good performance achieved by the proposed model on the PEMS test dataset for a prediction horizon of 24-steps. In this figure, the blue and red represent the ground truth and predictions of the traffic states, respectively. The green band represents the confidence intervals, i.e., ± 1 Standard Deviation (SD) surrounding the predicted values. The narrower the band, the more accurate the predictions.

Table 5
Friedman Chi-Square statistical test results of the proposed model for different prediction lengths on the PEMS dataset.

timesteps	Statistics	p-value
3	71.12	$3.5e^{-16}$
6	8.51	0.014
12	31.28	$1.6e^{-7}$
18	266.56	$1.3e^{-58}$
24	5.54	0.062

Fig. 6 presents four scatter plots comparing the actual and predicted traffic speeds for four sensors. Each plot includes a red dashed line representing the ideal case of perfect predictions, where the predicted values match the actual values. The blue dots represent the expected traffic speeds for each sensor, with the x-axis displaying the actual values and the y-axis displaying predicted values. The closeness of the points to the red line indicates how accurate the model's predictions are, with tighter clusters suggesting greater performance. The variation observed in the plots reflects sensor-specific prediction challenges and the proposed model's overall generalizability across different sensor locations.

5. Discussion

Capturing long-term spatial-temporal dependencies of traffic states is vital for achieving accurate predictions over longer time horizons. With a substantial amount of data, spanning many sensors, it is possible to address this problem. Then, another problem arises: Which models can effectively deal with the vast amount of data samples? DL-based models like LSTMs and GRUs are best suited to deal with this problem. These models process each data sample sequentially, one at a time, and, hence, tend to be less effective in parallel data processing. On top of that, these models suffer from gradient vanishing and exploding phenomena, resulting in ineffectiveness in capturing long-term dependencies. In addition, they need to complete many training steps to achieve reasonable accuracy. Attention mechanism-based models have

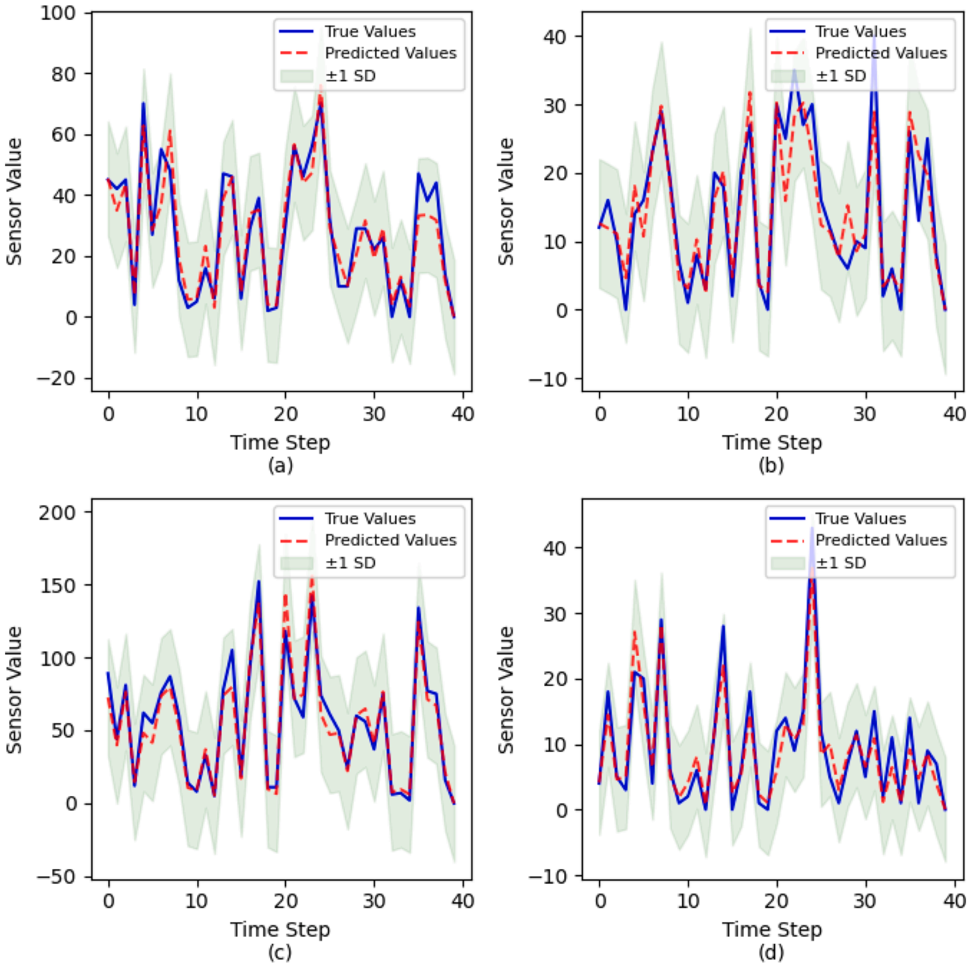


Fig. 5. Traffic speed prediction for a 24-step horizon on the PEMS test dataset achieved by the proposed model for four different sensors (denoted by (a), (b), (c), and (d)). The green band represents ±1 Standard Deviation (SD) surrounding the predicted values.

the potential to address these problems. However, current transformer models lack efficient means to capture spatial-temporal positional information at each time step. These models are computationally demanding and, on top of that, suffer from error accumulation problems for longer prediction timesteps.

Additionally, traffic states are affected by various events like rainy/snowy weather, summer/winter seasons, office peak/off-peak hours, and accidents. Hence, some state-of-the-art models include weather/accident information data with the raw dataset to increase the prediction accuracy over extended timesteps (Liu et al., 2022). Also, some methods rely on denoising mechanisms to remove outliers from the input datasets before passing them into the models for good accuracy (Chen et al., 2021). However, if the dataset contains substantial samples over extended periods, those effects are likely already hidden within the dataset. The proposed model addresses these problems by combining the attention mechanism with an RTCN block. The DFE mechanism facilitated the meaningful transformation of raw feature representations. A DLP block enhanced this representation using a non-linear transformation and a gating mechanism. The STPE layer augmented these transformed features with spatial-temporal positional information at each time step. Multiple encoder layers were used to deeply encode the data representations, and an RTCN block extracted both short and long-term temporal features. Finally, a Time Distributed Dense layer predicted the future traffic states up to 24 output timesteps. Table 6 depicts the main architectural differences between the proposed model and three other recently proposed models. In this context, out-of-distribution (OOD) denotes a generalisation method whereby a model is assessed on an unknown dataset that was excluded from the training, validation, and testing phases.

Two state-of-the-art PEMS traffic flow and speed datasets were used for model training and testing. Five recent state-of-the-art baseline models were trained and tested using identical experimental setups for comprehensive performance comparisons. The proposed architecture outperformed the studied baselines on the two used datasets regarding the MAE and RMSE scores, particularly for higher prediction lengths. Also, the Friedman Chi-Square statistical test revealed the worthiness of the presented model by exhibiting a significant difference in prediction distributions compared to the baselines under consideration. Other statistical test methods are available in the literature, for example, the Wilcoxon signed-rank test (Demšar, 2006) was also performed. However, the proposed model demonstrated less impressive outcomes in this case.

Comprehensive ablation studies were performed to investigate the effect of different proposed components on the model's architecture. Table 7 presents the results of the ablation study conducted on the PEMS traffic flow dataset for 24 timesteps prediction length. In this table, w/o denotes the absence of specific components. The configuration without the DFE achieved a 3.75 % increase in MAE. At the same time, the exclusion of the DLP led to a 9.38 % and 18.16 % increase in MAE and RMSE, respectively. However, the STPE demonstrated little influence on the model's outcomes. On the other hand, additional studies indicated that the absence of the STPE marginally reduced the model's generalizability. The absence of the RTCN resulted in a 15.61 % increase in MAE; without data augmentation, the model showed a 5.61 % increase in MAE.

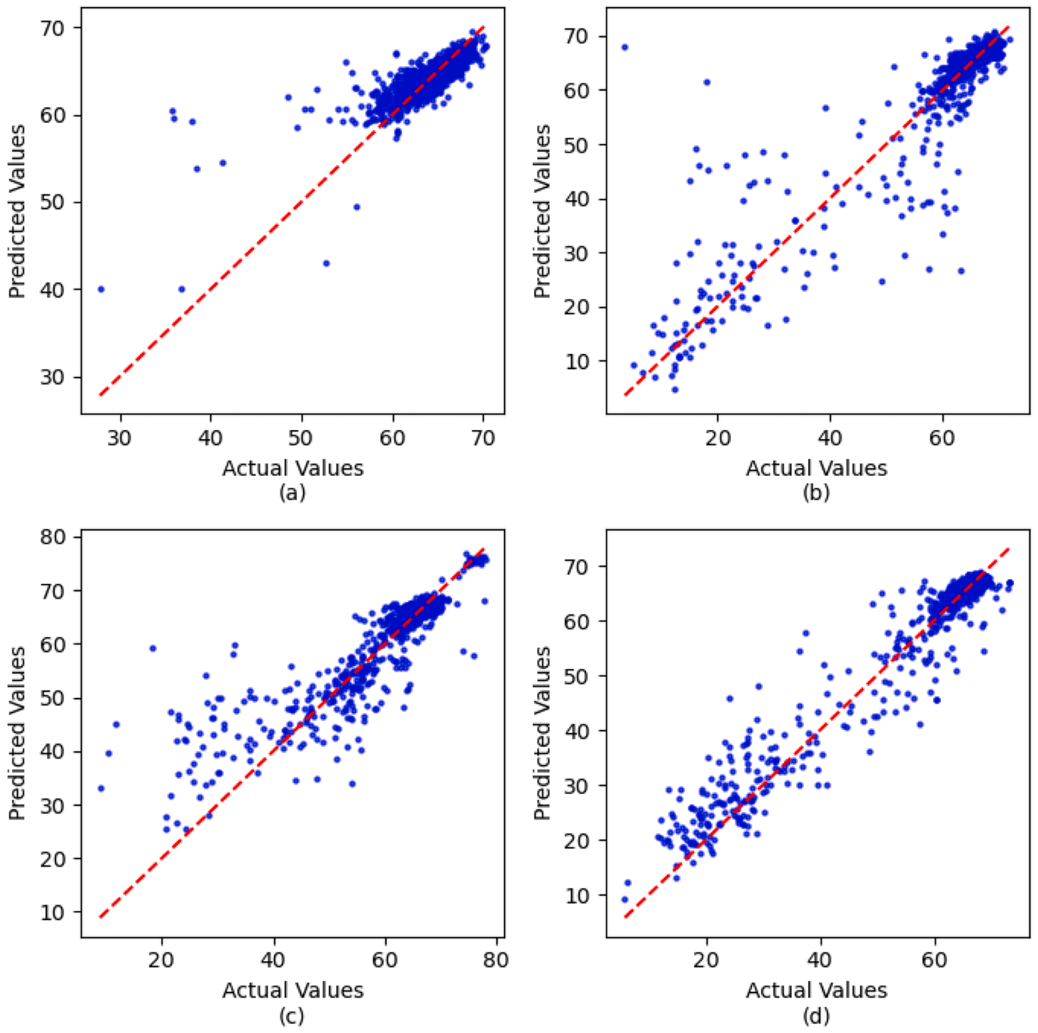


Fig. 6. Scatter plots comparing the actual vs. predicted traffic speeds for a 24-step horizon on the PEMS-BAY test dataset, achieved by the proposed model: the plots represent predictions for four different sensors (denoted by (a), (b), (c), and (d)) with the red dashed line indicating the ideal case of perfect predictions.

Table 6
Architectural innovations of the proposed model compared to recent state-of-the-art.

Tasks	uTransformer (Li et al., 2025)	RPConvformer (Wen et al., 2023)	HDT (Feng et al., 2025)	Proposed
Architecture	Unified transformer	Convolution and transformer	Hierarchical discrete transformer	Attention mechanism and RTCN
Scalability in high dimensions	Not addressed	Not addressed	Discrete token representations	DFE and DLP
Spatial-temporal dependencies	3D spatial-temporal correlation map	Partially addressed	Not Addressed	STPE
Short- and long-term dependencies	Self-attention	Self-attention	Hierarchical prediction approach	Attention and RTCN
Error accumulation	Not addressed	Not addressed	Indirectly using hierarchy	Using non-autoregressive decoding
Computational costs	Less	Not available	high	Moderate
Generalizability test	Partial	Partial	No OOD generalization	OOD generalization
Robustness test	Partial	Missing data robustness	Not addressed	Random noise, missing data robustness

This study also investigated the effects of the number of encoder layers and attention heads on the model's performance. Table 8 summarises these findings for the PEMS traffic flow dataset. Two encoder layers with three attention heads led to the best overall outcomes in this case. The increased number of encoder layers increased the computational cost and did not lead to the best outcomes. On the contrary, configurations with fewer encoder layers and attention heads generally achieved better performance with lower computational costs. The investigation also found that a lower number of encoder layers and attention heads resulted in late convergence, as can be observed in Fig. 7. These plots were obtained by training the proposed model for different numbers of encoder layers and attention heads. The model was permitted to train for a maximum of 150 epochs; however, training was halted upon achieving convergence. These findings suggest the importance of

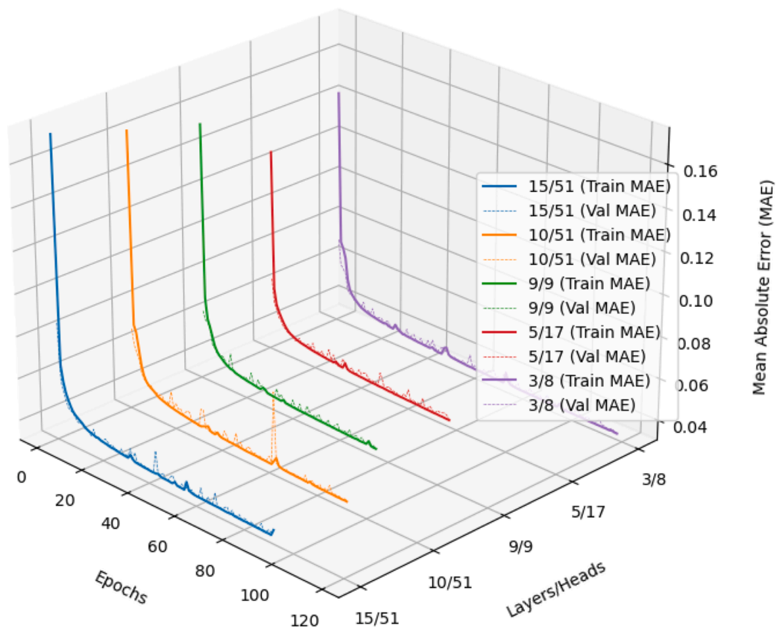


Fig. 7. Evolution of the MAE convergence for different encoder layers/attention heads combinations.

Table 7
Ablation study on the PEMS traffic flow dataset for 24 prediction steps using the proposed model (w and w/o represent with and without, best values in bold).

Attribute	Parameters	MAE	RMSE
w/o DFE	676*K*	3.31	5.72
w/o DLP	581	3.49	6.60
w/o STPE	792*K*	3.19	5.52
w/o RTCN	576*K*	3.77	6.58
w/o Data Augmentation	816*K*	3.38	5.79
Proposed	816*K*	**3.19**	**5.57**

Table 8
Performance of the proposed model on the number of encoder layers and attention heads on the PEMS test dataset for a 24-step prediction horizon (the bold values correspond to the best performance).

Num_layers	Num_heads	Parameters	MAE	Train Time (sec/epoch)
15	51	3.01M	3.41	30.63
10	51	2.17M	3.39	20.43
4	17	1.15M	3.44	8.05
9	9	1.99M	3.41	14.68
8	3	1.83M	3.34	13.64
7	1	1.66M	3.51	11.79
5	17	1.32M	3.36	9.19
2	3	**816K**	**3.19**	**5.29**

selecting an appropriate balance between the model's complexity and parameterisation.

While the proposed model took only 9.24 seconds per epoch for the PEMS-BAY dataset to train with the previously mentioned GPU configuration, the state-of-the-art models under consideration require higher computational resources, as is shown in Table 9. The Diffusion Convolutional Recurrent Neural Networks (DCRNN), Graph Multi-attention Network (GMAN), Multivariate Time-series Graph Convolutional Networks (MTGCN), Spatio-temporal Graph Convolutional Networks (STGCN), Lightweight Spatio-temporal Neural Network (LSTNN), and Hybrid Spatial-temporal Gated Convolution (HSTGCNN)

Table 9
Computational efficiency comparisons between the different models under study on the PEMS-BAY dataset (the values with bold correspond to the best performances found). NA represents the unavailability of corresponding values.

Model	Param	Training	Memory (MB)
Proposed	2.64M	**9.24**	**2697**
DCRNN (Li et al., 2018b)	372K	246.06	9603
GMAN (Zheng et al., 2020)	900K	87.67	7867
MTGCN (Wu et al., 2020)	573K	53.12	2837
STGCN (Yu et al., 2018)	**320K**	21.54	4765
LSTNN (Wang et al., 2025)	NA	114	NA
HSTGCNN (Zhang et al., 2025)	503K	134.7	NA

models were used to compare the computational demands of the presented model. It requires less training time and GPU memory usage compared to its counterparts. This feature would allow it to present excellent results with a much lower training cost, which helps deployment into edge devices, such as intelligent car systems or smartphones, for serving traffic prediction. Based on this comparison, it can be argued that the presented model is computationally lighter than its counterparts. It is to be noted that these models were not implemented/trained in this research. Instead, the computational statistics were taken from their respective works, and each required a different GPU configuration and experimental setup.

The accuracy score for the conventional state-of-the-art models starts reducing quickly with the increase in prediction length because of the error accumulation problem. This research aimed to reduce this trend by incorporating residual connections with the DLP and RTCN blocks. It also used a direct strategy where it predicts all future timesteps simultaneously, instead of forecasting one step at a time and incorporating the output into subsequent predictions. Fig. 8 illustrates the evolution of the MAE and RMSE scores with variations of the prediction time length. The accuracy of the CNN-LSTM model was quickly reduced with increasing time length. However, the TrafFORMER model provided the second-best performance compared to the others. Overall, the proposed model showed more capability in addressing this problem.

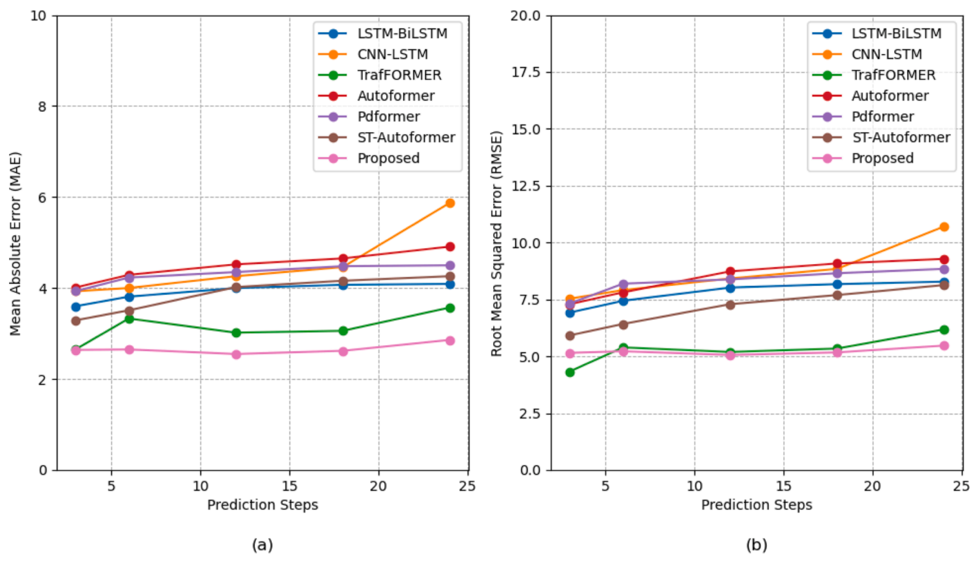


Fig. 8. Prediction accuracy in terms of (a) MAE and (b) RMSE for the studied models concerning the output timesteps on the PEMS-BAY test dataset.

Table 10
Performance of the proposed model for various noise levels based on the PEMS-BAY test dataset for a 24-step prediction horizon (the bold values correspond to the best performance).

Noise Level	MAE	RMSE
0.00 (Original)	**2.86**	**5.47**
0.10	9.13	11.84
0.20	17.37	22.01
0.30	25.41	33.99

Table 12
The adversarial perturbations test of the proposed model for various perturbation levels based on the PEMS test dataset for a 24-step prediction horizon.

Perturbation Magnitude (ϵ)	MAE Change (%)	RMSE Change (%)
0.05	0.46	3.19
0.10	0.79	4.92
0.20	1.38	7.46
0.30	1.87	9.20

Table 11
Results after performing a random sensor dropout test of the proposed model based on the PEMS test dataset for a 24-step prediction horizon (the bold values correspond to the best performance).

Sensor Dropout (%)	MAE	RMSE
Original	**3.19**	**5.57**
10	3.75	8.36
20	3.81	8.04
30	5.12	10.18
40	5.51	10.82
50	6.87	13.32

This study also investigated the robustness of the proposed model by simulating the arrival of non-recurrent events, such as accidents and abrupt weather changes, using the PEMS-BAY test dataset, as illustrated in Fig. 9. The original MAE and RMSE scores for a prediction time step of 24 were 2.86 and 5.47, respectively. However, the experimental results reveal a minimal change in the MAE and RMSE scores due to the sudden spikes, with values of 2.87 and 5.50, respectively. This investigation demonstrated the model's enhanced robustness capability.

On the other hand, with the addition of various levels of noise, the proposed model failed to maintain original prediction accuracy, as presented in Table 10. It presents the performance of the proposed model under varying noise levels based on the PEMS-BAY test dataset, specifically for a 24-step prediction horizon. As the noise level increases from 0.00 to 0.30, the model's performance deteriorates, with both MAE and RMSE values rising significantly. These results demonstrate that the model's predictive accuracy declines as noise levels rise, indicating its sensitivity to noise in the data samples. So, it is one of the main drawbacks of the proposed model. Also, it was not tested on another category of time-series datasets; hence, applying only to traffic state datasets is recommended.

Also, the random sensor dropout test was conducted to examine the proposed model's robustness further. It assessed the resilience of the trained model in the event of random sensor failures, simulating a situation where a portion of sensors becomes inactive, resulting in the omission of their values, and quantified the consequent decline in model performance using MAE and RMSE. Table 11 tabulates the results of this experiment concerning different sensor dropout percentages ranging from 10 % to 50 %. The MAE and RMSE scores increased with the increased percentage of sensor dropout. However, the changes are in an acceptable range, though, and room for further improvements persists.

In addition, the adversarial perturbations test was performed to check the model's robustness. For different perturbation magnitudes, ϵ, the MAE and RMSE deviations were computed by taking the difference between the original and perturbed scores. Table 12 tabulates the obtained MAE and RMSE deviation for different perturbation magnitudes. The proposed model demonstrated resilience in the lower and higher magnitudes of adversarial attacks regarding MAE. However, regarding RMSE, the proposed model failed under strong attack, i.e., for ϵ of 0.30.

This research used another state-of-the-art dataset named METR-LA (Li et al., 2018a) to examine the generalizability of the proposed model. This dataset was split into train and test sets, and the test set was split weekly. The saved proposed model, which had never seen this dataset, was evaluated based on the weekly split test sets. Table 13 presents the

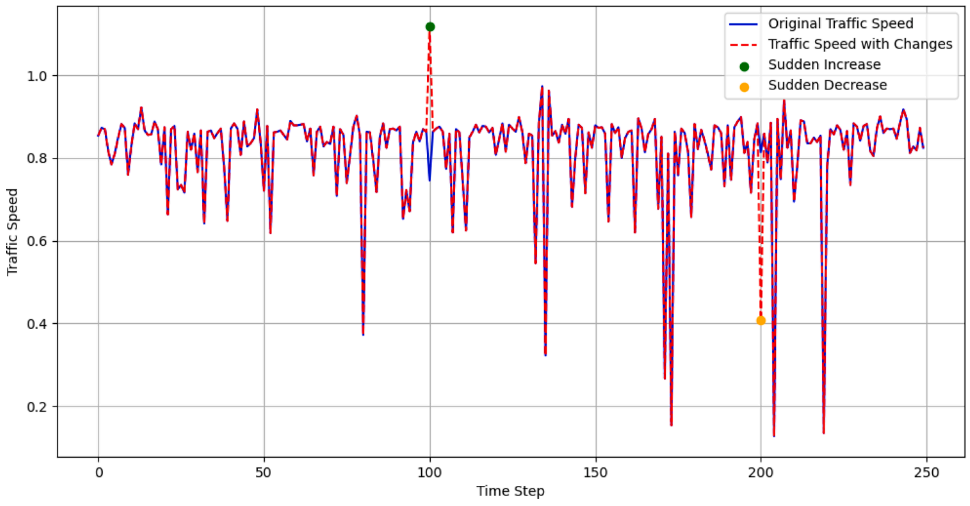


Fig. 9. Traffic speed with sudden increase and decrease on the PEMS-BAY test dataset to mimic the arrival of non-recurrent events.

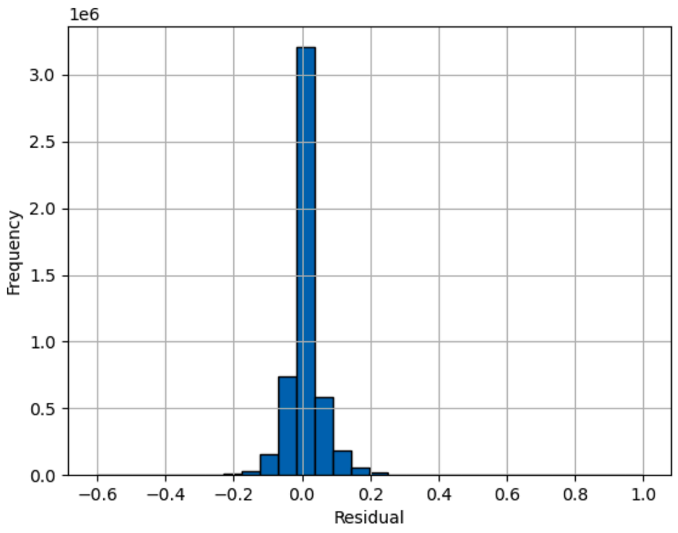


Fig. 10. Residual distribution between the true and predicted values based on the PEMS test dataset.

Table 13
The generalizability test of the proposed model on the METR-LA test dataset for a 24-step prediction horizon.

Test Set	MAE	RMSE
2012-03-07 to 2012-03-13	5.19	5.88
2012-03-14 to 2012-03-20	5.61	6.12
2012-03-21 to 2012-03-27	4.89	5.68
2012-03-28 to 2012-04-03	5.66	7.18
2012-04-04 to 2012-04-10	4.80	5.66
2012-04-11 to 2012-04-17	5.65	6.15
2012-04-18 to 2012-04-24	5.54	6.08
2012-04-25 to 2012-05-01	5.22	5.93
2012-05-02 to 2012-05-08	5.61	6.14
2012-05-09 to 2012-05-15	5.61	6.13
2012-05-16 to 2012-05-22	5.58	6.11
2012-05-23 to 2012-05-29	5.75	6.25
2012-05-30 to 2012-06-05	5.19	5.89
2012-06-06 to 2012-06-12	5.18	5.86
2012-06-13 to 2012-06-19	4.23	5.27

results of this experiment for different weekly test sets. It demonstrated good generalizability of the proposed model, although the obtained MAE and RMSE were higher than those of the PEMS-BAY and PEMS datasets. Nevertheless, it needs to be further increased, which is a potential future work.

On top of that, the investigation also covered the Wilcoxon sign-rank (Rainio et al., 2024) and the Analysis of Variance (ANOVA) (Kennedy & Wang, 2025) tests to find out if there are statistically significant differences in the model's performance. In the case of the Wilcoxon test, first, the residual was computed from the difference between the true and predicted values and then the test was performed based on the residual values. The test statistic was very high, and the p_value was zero, indicating the presence of model bias. Although the residual contained tiny entities, the null hypothesis was rejected because they were not centred around the zero (median), as is shown in Fig. 10. In the case of ANOVA, the test was performed on the MAE and RMSE values of the models. The results indicated higher test statistics and p_value, indicating no significant differences between them. These tests indicated the drawbacks of the proposed model.

This study additionally used confidence intervals between runs to completely validate the performance disparities. In 10 runs, the average MAE was 3.10 ± 0.20 while the mean RMSE was 5.37 ± 0.35. The ± values denote 95 % confidence intervals calculated from the standard error of the mean across the runs. Over 10 trials, MAE changes by only 0.20 % and RMSE by 0.35 %. Since long-term prediction is necessary for real-world applications, the proposed model bears great potential in this regard. Its direct multi-step approach is ideal for cases requiring long-term prediction accuracy and stability. It is capable of fast inference as well. Nonetheless, additional research is required to assess its viability as a backend service in urban traffic control centres or its incorporation into navigation applications for rerouting based on predictions.

6. Conclusion

This study proposed a hybrid model combining attention mechanisms and RTCN structures to address long-term traffic state prediction tasks. State-of-the-art transformer-based methods often fall short in effectively integrating spatial-temporal positional information into data features and are computationally intensive, especially for multi-step ahead prediction tasks prone to error accumulation. The proposed model mitigates these issues by introducing innovative components tailored for transportation applications.

The Dynamic Feature Embedding layer transforms raw traffic data into meaningful feature representations. At the same time, the Deep Linear Projection further enhances these representations using non-linear transformations and a gating mechanism. The Spatial-Temporal Positional Encoding layer enriches these features by embedding spatial-temporal information for each sensor at every time step. These enriched features are encoded through a masked multi-head attention mechanism, followed by an RTCN block to capture both short- and long-term temporal dependencies. Finally, a Time-Distributed Dense layer produces accurate long-term predictions.

Performance evaluations demonstrated the superiority of the proposed model over six state-of-the-art baseline models using two benchmark traffic datasets. On the PEMS-BAY test dataset, the model improved in MAE and RMSE, ranging from 0.4 to 51.3 % and 2.5 to 48.9 %, respectively, across 3 to 24-step prediction horizons. These results were achieved without incorporating additional contextual factors like weather or accident data and after removing outliers, showcasing the model's robustness. Moreover, the computational cost analysis highlighted a significant reduction in resource requirements compared to recently published transformer-based models, making it a practical solution for large-scale traffic state prediction tasks. The model also demonstrated resilience under scenarios of sudden fluctuations in traffic conditions, adversarial perturbation, and random sensor dropouts, which are critical for real-world transportation systems.

However, the study acknowledges certain limitations. The model's performance is sensitive to external noise, highlighting the need for future work on noise-resilient architectures. Furthermore, the requirement for substantial data samples to learn long-term dependencies suggests potential avenues for optimising data efficiency. Another critical consideration is the model's dependency on the inherent characteristics of the training data, such as trends, seasonality, and extreme events. The model performs well when applied to regions with traffic conditions similar to the training data. Extending its applicability to heterogeneous traffic networks with varying characteristics remains an important area for future research.

By addressing these challenges, the proposed model is a promising solution for enhancing Intelligent Transportation Systems, enabling transportation stakeholders to anticipate traffic dynamics more accurately, optimise resource allocation, and enhance mobility across diverse urban environments.

Source code

The source code can be found from: GitHub Repository

CRediT authorship contribution statement

Selim Reza: Validation, Investigation, Writing – original draft, Methodology, Software; **Marta Campos Ferreira:** Writing – review & editing, Supervision; **J. J. M. Machado:** Writing – review & editing, Supervision, Validation; **João Manuel R.S. Tavares:** Supervision, Writing – review & editing, Conceptualization, Funding acquisition.

Data availability

The used experimental datasets are publicly available at Caltrans PeMS and PEMS-BAY.

Declaration of competing interest

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

Acknowledgments

The authors would like to thank “Fundação para a Ciência e Tecnologia” (FCT) for the PhD grant (with reference 2022.12391.BD) awarded to the first author, which this work is a part of. This article partially results from the project “Sensitive Industry”, co-funded by the European Regional Development Fund (ERDF) through the Operational Programme for Competitiveness and Internationalization (COMPETE 2020) under the PORTUGAL 2020 Partnership Agreement.

References

Al-Thani, M. G., Sheng, Z., Cao, Y., & Yang, Y. (2024). Traffic transformer: Transformer-based framework for temporal traffic accident prediction. *AIMS Mathematics*, *9*(5), 12610–12629.

Ansari Esfeh, M., Kattan, L., Lam, W. H. K., Ansari Esfe, R., & Salari, M. (2020). Compound generalized extreme value distribution for modeling the effects of monthly and seasonal variation on the extreme travel delays for vulnerability analysis of road network. *Transportation Research Part C: Emerging Technologies*, *120*, 102808.

Ba, J. L., Kiros, J. R., & Hinton, G. E. (2016). Layer normalization. https://arxiv.org/abs/1607.06450.

Baldi, P., & Sadowski, P. J. (2013). Understanding dropout. *Advances in Neural Information Processing Systems*, *26*.

Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, *13*(2).

Bjorck, N., Gomes, C. P., Selman, B., & Weinberger, K. Q. (2018). Understanding batch normalization. *Advances in Neural Information Processing Systems*, *31*.

Cao, Q., Yuan, J., Ren, G., Qi, Y., Li, D., Deng, Y., & Ma, W. (2024). Tracking the source of congestion based on a probabilistic sensor flow assignment model. *Transportation Research Part C: Emerging Technologies*, *165*, 104736.

Cenitta, D., Arjunan, R. V., & Prema, K. V. (2021). Missing data imputation using machine learning algorithm for supervised learning. In *2021 international conference on computer communication and informatics (ICCCI)* (pp. 1–5). IEEE.

Chauhan, N. S., Kumar, N., & Eskandarian, A. (2024). A novel confined attention mechanism driven bi-GRU model for traffic flow prediction. *IEEE Transactions on Intelligent Transportation Systems*, *25*(8), 9181–9191.

Chen, C. (2002). Freeway performance measurement system (PeMS). University of California, Berkeley.

Chen, C., Liu, Y., Chen, L., & Zhang, C. (2022). Bidirectional spatial-temporal adaptive transformer for urban traffic flow forecasting. *IEEE Transactions on Neural Networks and Learning Systems*, *34*(10), 6913–6925.

Chen, J., Zheng, L., Hu, Y., Wang, W., Zhang, H., & Hu, X. (2024). Traffic flow matrix-based graph neural network with attention mechanism for traffic flow prediction. *Information Fusion*, *104*, 102146.

Chen, X., Chen, H., Yang, Y., Wu, H., Zhang, W., Zhao, J., & Xiong, Y. (2021). Traffic flow prediction by an ensemble framework with data denoising and deep learning model. *Physica A: Statistical Mechanics and its Applications*, *565*, 125574.

Chen, X., Wu, S., Shi, C., Huang, Y., Yang, Y., Ke, R., & Zhao, J. (2020). Sensing data supported traffic flow prediction via denoising schemes and ANN: A comparison. *IEEE Sensors Journal*, *20*(23), 14317–14328.

Cheng, Z., Lu, J., Zhou, H., Zhang, Y., & Zhang, L. (2022). Short-term traffic flow prediction: An integrated method of econometrics and hybrid deep learning. *IEEE Transactions on Intelligent Transportation Systems*, *23*(6), 5231–5244.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine learning research*, *7*, 1–30.

Du, S., Li, T., Gong, X., & Horng, S.-J. (2020). A hybrid method for traffic flow forecasting using multimodal deep learning. *International Journal of Computational Intelligence Systems*, *13*(1), 85.

Essien, A., Petrounias, I., Sampaio, P., & Sampaio, S. (2021). A deep-learning model for urban traffic flow prediction with traffic events mined from twitter. *World Wide Web*, *24*(4), 1345–1368.

Feng, S., Zhao, P., Liu, L., Wu, P., & Shen, Z. (2025). Hdt: Hierarchical discrete transformer for multivariate time series forecasting. *arXiv preprint arXiv: 2502.08302*.

Geng, Z., Xu, J., Wu, R., Zhao, C., Wang, J., Li, Y., & Zhang, C. (2024). Stgaformer: Spatial–temporal gated attention transformer based graph neural network for traffic flow forecasting. *Information Fusion*, *105*, 102228.

Grigsby, J., Wang, Z., Nguyen, N., & Qi, Y. (2023). Long-range transformers for dynamic spatiotemporal forecasting. https://arxiv.org/abs/2109.12218.

Guo, J., Huang, W., & Williams, B. M. (2015). Real time traffic flow outlier detection using short-term traffic conditional variance prediction. *Transportation Research Part C: Emerging Technologies*, *50*, 160–172.

Hamad, R. A., Kimura, M., Yang, L., Woo, W. L., & Wei, B. (2021). Dilated causal convolution with multi-head self attention for sensor human activity recognition. *Neural Computing and Applications*, *33*, 13705–13722.

Jia, Y., Wu, J., & Du, Y. (2016). Traffic speed prediction using deep learning method. In *2016 IEEE 19th international conference on intelligent transportation systems (ITSC)* (pp. 1217–1222). IEEE.

Jiang, J., Han, C., Zhao, W. X., & Wang, J. (2023). Pdformer: Propagation delay-aware dynamic long-range transformer for traffic flow prediction. In *Proceedings of the thirty-seventh AAAI conference on artificial intelligence and thirty-fifth conference on innovative applications of artificial intelligence and thirteenth symposium on educational advances in artificial intelligence* AAAI'23/IAAI'23/EAAI'23. AAAI Press.

Kennedy, A., & Wang, S. (2025). Analysis of variance. In *Translational urology* (pp. 121–124). Elsevier.

Kwon, S. et al. (2021). Mlt-dnet: Speech emotion recognition using 1d dilated cnn based on multi-learning trick approach. *Expert Systems with Applications*, *167*, 114177.

Lederer, J. (2021). Activation functions in artificial neural networks: A systematic overview. https://arxiv.org/abs/2101.09957.

Lee, H.-J., & Park, D.-J. (2024). Collision evasive action timing for MASS using CNN–LSTM-based ship trajectory prediction in restricted area. *Ocean Engineering*, *294*, 116766.

Li, C., Liu, W., & Yang, H. (2024a). Deep causal inference for understanding the impact of meteorological variations on traffic. *Transportation Research Part C: Emerging Technologies*, *165*, 104744.

Li, J., Dong, W., & Gui, X. (2025). Utransformer: Unified spatial-temporal transformer with external factors for traffic flow forecasting. *The Journal of Supercomputing*, *81*(1), 1–34.

Li, S., Jin, X., Xuan, Y., Zhou, X., Chen, W., Wang, Y.-X., & Yan, X. (2019). Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems* Curran Associates Inc.

Li, Y., Yu, R., Shahabi, C., & Liu, Y. (2018a). Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *International conference on learning representations (ICLR '18)*.

Li, Y., Yu, R., Shahabi, C., & Liu, Y. (2018b). Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. https://arxiv.org/abs/1707.01926.

Li, Z., Xu, H., Gao, X., Wang, Z., & Xu, W. (2024b). Fusion attention mechanism bidirectional LSTM for short-term traffic flow prediction. *Journal of Intelligent Transportation Systems*, *28*(4), 511–524.

Lim, B., Arık, S. Ö., Loeff, N., & Pfister, T. (2021). Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, *37*(4), 1748–1764.

Liu, F. T., Ting, K. M., & Zhou, Z.-H. (2008). Isolation forest. In *2008 eighth IEEE international conference on data mining* (pp. 413–422). IEEE.

Liu, J., Wu, N., Qiao, Y., & Li, Z. (2022). Short-term traffic flow forecasting using ensemble approach based on deep belief networks. *IEEE Transactions on Intelligent Transportation Systems*, *23*(1), 404–417.

Liu, Z., Ding, F., Dai, Y., Li, L., Chen, T., & Tan, H. (2024). Spatial-temporal graph convolution network model with traffic fundamental diagram information informed for network traffic flow prediction. *Expert Systems with Applications*, *249*, 123543.

Luo, Q., He, S., Han, X., Wang, Y., & Li, H. (2024). Lsttn: A long-short term transformer-based spatiotemporal neural network for traffic flow forecasting. *Knowledge-Based Systems*, *293*, 111637.

Lv, Z., Xu, J., Zheng, K., Yin, H., Zhao, P., & Zhou, X. (2018). Lc-rnn: A deep learning model for traffic speed prediction. In *Proceedings of the twenty-seventh international joint conference on artificial intelligence, IJCAI-18* (pp. 3470–3476). International Joint Conferences on Artificial Intelligence Organization.

Ma, C., Dai, G., & Zhou, J. (2021). Short-term traffic flow prediction for urban road sections based on time series analysis and LSTM_BILSTM method. *IEEE Transactions on Intelligent Transportation Systems*, *23*(6), 5615–5624.

Ma, T., Antoniou, C., & Toledo, T. (2020). Hybrid machine learning algorithm and statistical time series model for network-wide traffic forecast. *Transportation Research Part C: Emerging Technologies*, *111*, 352–372.

Miner, P., Smith, B. M., Jani, A., McNeill, G., & Gathorne-Hardy, A. (2024). Car harm: A global review of automobility's harm to people and the environment. *Journal of Transport Geography*, *115*, 103817.

Patro, S. G. K., & Sahu, K. K. (2015). Normalization: A preprocessing stage. https://arxiv.org/abs/1503.06462.

Pereira, D. G., Afonso, A., & Medeiros, F. M. (2015). Overview of friedman's test and post-hoc analysis. *Communications in Statistics-Simulation and Computation*, *44*(10), 2636–2653.

Polson, N. G., & Sokolov, V. O. (2017). Deep learning for short-term traffic flow prediction. *Transportation Research Part C: Emerging Technologies*, *79*, 1–17.

Rainio, O., Teuho, J., & Klén, R. (2024). Evaluation metrics and statistical tests for machine learning. *Scientific Reports*, *14*(1), 6086.

Reza, S., Ferreira, M. C., Machado, J., & Tavares, J. M. R. S. (2022). A multi-head attention-based transformer model for traffic flow forecasting with a comparative analysis to recurrent neural networks. *Expert Systems with Applications*, *202*, 117275.

Ribeiro, A. H., Tiels, K., Aguirre, L. A., & Sch́on, T. (2020). Beyond exploding and vanishing gradients: Analysing RNN training using attractors and smoothness. In *International conference on artificial intelligence and statistics* (pp. 2370–2380). PMLR.

Sharma, S., Nayak, R., & Bhaskar, A. (2024). Multi-view feature engineering for day-to-day joint clustering of multiple traffic datasets. *Transportation Research Part C: Emerging Technologies*, *162*, 104607.

Tatlıcıoğlu, B. E. (2024). A simple chaotic system using signum function. *Mathematics and Computers in Simulation*, *225*, 1072–1088.

Tedjopurnomo, D. A., Choudhury, F. M., & Qin, A. K. (2023). Trafformer: A transformer model for predicting long-term traffic. https://arxiv.org/abs/2302.12388.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems*. Curran Associates, Inc. (*vol. 30*).

Vijayalakshmi, B., Ramar, K., Jhanjhi, N. Z., Verma, S., Kaliappan, M., Vijayalakshmi, K., Vimal, S., & Ghosh, U. (2021). An attention-based deep learning model for traffic flow prediction using spatiotemporal features towards sustainable smart city. *International Journal of Communication Systems*, *34*(3), e4609.

Wan, X., Wang, W., Liu, J., & Tong, T. (2014). Estimating the sample mean and standard deviation from the sample size, median, range and/or interquartile range. *BMC Medical Research Methodology*, *14*(1), 1–13.

Wang, D., Guo, G., Ouyang, T., Yu, D., Zhang, H., Li, B., Jiang, R., Xu, G., & Deng, S. (2025). A lightweight spatio-temporal neural network with sampling-based time series decomposition for traffic forecasting. *IEEE Transactions on Intelligent Transportation Systems*, *26*(6), 8682–8693. https://doi.org/10.1109/TITS.2025.3552010.

Wen, Y., Xu, P., Li, Z., Xu, W., & Wang, X. (2023). Rpconvformer: A novel transformer-based deep neural networks for traffic flow prediction. *Expert Systems with Applications*, *218*, 119587.

Wu, H., Xu, J., Wang, J., & Long, M. (2021). Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in Neural Information Processing Systems*, *34*, 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).

Xiao, Y., & Yin, Y. (2019). Hybrid LSTM neural network for short-term traffic flow prediction. *Information*, *10*(3), 105.

Xing, J., Wu, W., Cheng, Q., & Liu, R. (2022). Traffic state estimation of urban road networks by multi-source data fusion: Review and new insights. *Physica A: Statistical Mechanics and its Applications*, *595*, 127079.

Xu, Q., Pang, Y., Zhou, X., & Liu, Y. (2024). Pigat: Physics-informed graph attention transformer for air traffic state prediction. *IEEE Transactions on Intelligent Transportation Systems*, *25*(9), 12561–12577.

Xu, X., Jin, X., Xiao, D., Ma, C., & Wong, S. C. (2021). A hybrid autoregressive fractionally integrated moving average and nonlinear autoregressive neural network model for short-term traffic flow prediction. *Journal of Intelligent Transportation Systems*, (pp. 1–18).

Yang, B., Sun, S., Li, J., Lin, X., & Tian, Y. (2019). Traffic flow prediction using LSTM with feature enhancement. *Neurocomputing*, *332*, 320–327.

Yu, B., Yin, H., & Zhu, Z. (2018). Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. In *Proceedings of the 27th international joint conference on artificial intelligence* (pp. 3634–3640).

Yu, S., Peng, J., Ge, Y., Yu, X., Ding, F., Li, S., & Ma, C. (2024). A traffic state prediction method based on spatial–temporal data mining of floating car data by using autoformer architecture. *Computer-Aided Civil and Infrastructure Engineering*, *39*(18), 2774–2787.

Zhang, H., Zou, Y., Yang, X., & Yang, H. (2022). A temporal fusion transformer for short-term freeway traffic speed multistep prediction. *Neurocomputing*, *500*, 329–340.

Zhang, Y., Yang, S., Wang, H., Cheng, Y., Wang, J., Cao, L., & An, Z. (2025). A traffic flow forecasting method based on hybrid spatial–temporal gated convolution. *International Journal of Machine Learning and Cybernetics*, *16*(3), 1805–1817.

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). (*vol. 34*).

Zou, D., Wang, S., Li, X., Peng, H., Wang, Y., Liu, C., Sheng, K., & Zhang, B. (2024). Multispans: A multi-range spatial-temporal transformer network for traffic forecast via structural entropy optimization. In *Proceedings of the 17th ACM international conference on web search and data mining* (pp. 1032–1041).