Enhancing Traffic Flow Prediction via Adaptive Spatial-Temporal Encoding and Attention Refinement

1st Mingchao Zhang
Software School
Yunnan University
Kunming, China
zhangmingchao@stu.ynu.edu.cn

2nd Wei Wang
Software School
Yunnan University
Kunming, China
wangwei@ynu.edu.cn

***Abstract*—Traffic flow prediction (TFP) is a critical component of modern traffic management systems, yet it remains challenging due to complex spatial-temporal dependencies. Recent studies have attempted to address this challenge by leveraging various advanced neural networks, such as graph neural networks and transformers. While these models may achieve high accuracy, they remain susceptible to multiple challenges, including the cumulative effects of individually minor attention weights. Furthermore, existing research has primarily focused on developing intricate TFP architectures, overlooking the importance of traffic observation encoding. To address these issues, we propose PARformer, a novel approach that introduces an adaptive spatial-temporal embedding module to capture evolving dependency patterns. Additionally, PARformer leverages controlled perturbations to encourage attention mechanism to actively calibrate the weight distribution, aiming to mitigate the adverse effects of minor attention weights. We validate the effectiveness of PARformer through comparative experiments on four real-world datasets collected from the California Department of Transportation. The results indicate that: (1) PARformer outperforms several recently proposed baselines in predictive accuracy. (2) The proposed perturbation mechanism enhances the calibration of attention weight distribution.**

***Index Terms*—Traffic flow prediction, Attention mechanism, Perturbation, Self-supervised Learning.**

I. Introduction

Traffic flow prediction (TFP) concerns forecasting future traffic states—such as traffic volume, travel speed, and congestion level—over road networks [1]. As a key capability in modern intelligent transportation systems, it supports a variety of operational and planning tasks [2], [3], including adaptive signal control and infrastructure planning.

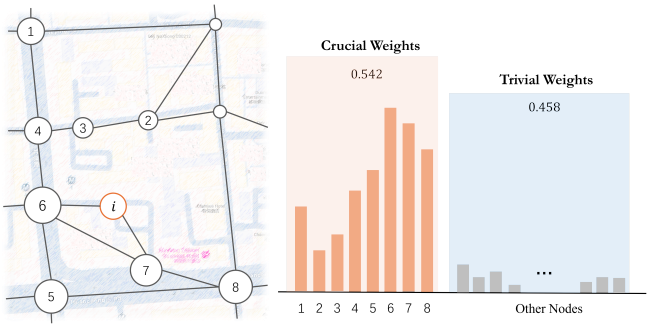
Traffic flow is inherently governed by complex spatial (e.g., congestion propagating across nearby roads) and temporal dependencies (e.g., rush-hour peaks, weekly cycles). However, modeling spatial-temporal dependencies is challenging due to the nonlinear dynamics and complex structure of traffic networks. Spatial dependencies often exhibit distance-decay effects, anisotropic connectivity patterns, and cross-regional interactions, while temporal dependencies introduce complexities such as irregular sampling intervals, long-term dependencies, and evolving patterns (e.g., rush hour vs. nighttime traffic). Moreover, traffic flow data is often affected by various noise sources, including sensor inaccuracies and external environmental factors, which can lead to unreliable predictions [4].

To address these challenges, researchers have proposed various sophisticated models. These models typically follow a common research paradigm that expands the parameter space to better capture complex spatial-temporal dependencies. Especially in recent years, inspired by the achievements of deep learning (DL) in multiple domains, numerous DL-based models have been proposed for TFP. Among these approaches, graph neural networks (GNNs) and attention-based models have become particularly popular due to their outstanding performance [5], [6], [7], [8].

GNNs encode the prior knowledge of traffic networks, e.g., link connectivity or spatial distance between locations into an adjacency matrix [9] and use it to filter out the irrelevant dependencies. Theoretical analysis [10] has shown that the function of adjacency matrix analogously to a first-order filter. However, dependencies in traffic flow are inherently dynamic, whereas the prior knowledge encoded in a fixed adjacency matrix cannot adequately capture such evolving spatial-temporal patterns. Although some TFP models with adaptive adjacency matrices have been proposed [11], which leverage historical similarities between locations to construct dynamic graphs, such similarities do not necessarily guarantee accurate representation of future dependencies between locations.

In contrast to the static priors of GNNs, attention mechanisms operate in a purely data-driven fashion by adaptively allocating higher weight to informative dependencies and lower weights to irrelevant ones. This dynamic weighting enables attention-based models to outperform traditional GNN-based methods and highlights their potential for modeling complex traffic dynamics.

Nonetheless, recent studies [12] have revealed that attention weights are not always precisely allocated. Attention techniques tend to allocate small but non-negligible weights to irrelevant features or noisy observations, whose cumulative



(a) (b)

Fig. 1. (a) The 8 important nodes for traffic node i; (b) Attention weight distribution across all nodes for traffic node i, showing the sum for both important and unimportant nodes.

effect can still rival that of informative dependencies. Fig. 1 illustrates this effect in TFP: although many non-critical traffic nodes are assigned only negligible weights individually, their aggregated contribution can approach that of truly critical nodes. This accumulation dilutes the attention distribution and reduces the model's ability to concentrate on the most informative nodes, which degrade the model performance and serve as carriers of noise, distorting the normal message passing between nodes [13], [14].

Therefore, how to design a TFP model that can capture complex spatial-temporal relationships through attention mechanisms and calibrate attention weight allocation remains a critical challenge. To address this issue, we propose a novel attention-based TFP model, denoted as PARformer in this paper. The contributions are as follows:

- **Attention-based traffic observation encoding**: To enhance PARformer's sensitivity to the dynamics of spatial-temporal dependency, we introduce an adaptive spatial-temporal dependency encoder and integrate it with raw traffic observations and periodicity embeddings as inputs for the following computation.
- **Perturbation-guided self-calibration**: We propose a controlled perturbation mechanism that encourages the attention mechanism to actively prioritize critical features and recalibrate its weight distribution.
- **Ablation and benchmark analysis**: Experiments on four real-world datasets demonstrate that PARformer achieves strong predictive performance. Ablation studies confirming the calibration efficacy of the perturbation mechanism on the attention module.

II. Preliminaries and Related Work

This section summarizes recent TFP methods and perturbation techniques, highlighting how our approach differs from existing work.

A. TFP Models

Early works usually abstract the TFP as a time series analysis problem, employing statistical models such as Autoregressive Integrated Moving Average (ARIMA) [15], Historical Average (HA) [16], Kalman filter [17]. With the advent of machine learning (ML) and deep learning (DL), various models such as Support Vector Regression (SVR) [18], XGBoost [19], and recurrent neural networks (RNNs) with their variants LSTM [20] and GRU [21] were introduced to better model temporal dependencies. To incorporate spatial dependencies, convolutional neural networks (CNNs) [22] and graph neural networks (GNNs) [23] were proposed, enabling DL-based models to learn complex, nonlinear, and high-dimensional spatial-temporal patterns.

Attention-based models have become increasingly prominent and highly competitive in recent TFP research [6], [7], [24], [25], [26]. The core idea of the attention mechanism is to assign different weights to input features, thereby enabling the model to selectively focus on the most relevant components of the spatial-temporal data. Unlike RNNs or GNNs, which process all input locations or temporal steps with relatively uniform importance, attention mechanisms explicitly learn a weighted representation that emphasizes critical information while suppressing irrelevant part in TFP.

However, recent studies have reported that attention mechanisms suffers from diluted weights [12], [27], where non-informative locations, time stamps, or noisy signals still receive small weights. These weights accumulate and flatten the distribution, obscuring truly critical information. Analysis [14] further shows that this degradation is intrinsically induced by the dot-product operation and the $\mathrm{SoftMax}(\cdot)$ normalization, representing an inherent and unavoidable limitation of the attention mechanism.

Building on these insights, our work extends the study of diluted attention weights to TFP. Unlike prior natural language processing (NLP) approach [12] that rely on stacking multiple attention modules, we propose a perturbation-based strategy to address this issue.

B. Perturbation

Perturbation generally refers to a small change or modification applied to a model. In the context of DL, perturbations often enhance data diversity through data augmentation [28], [29], interpret models by observing output changes [30], [31], and evaluate model robustness via adversarial attacks[32], [33]. In TFP, perturbation has been employed in [34] as a tool for robustness analysis. [35] perturbs both adjacency matrices and temporal inputs to enhance interpretability and improve predictive performance. However, its design does not directly target the intrinsic deficiencies of the attention mechanism itself. Overall, the applications of perturbation for improving the attention mechanism in TFP remain relatively underexplored. In this work, we leverage perturbation not merely as a diagnostic tool or external augmentation method, but as a means to provide independent perspective of feature importance, thereby driving the attention mechanism to actively recalibrate its own weight distribution.

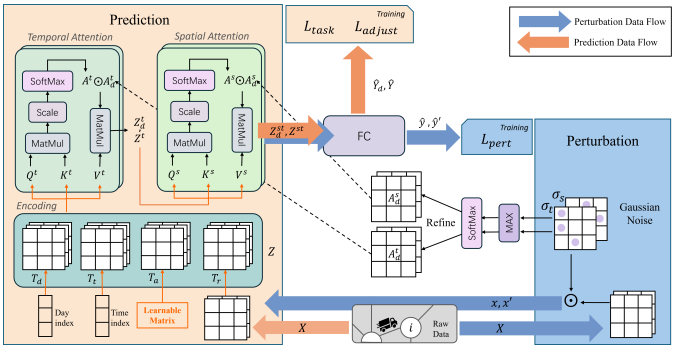


Fig. 2. The overall architecture of PARformer.

III. Proposed Method

PARformer involves three key modules: (1) **Spatial-temporal dependencies modeling:** leverage adaptive traffic observation encoding and transformer architecture to convert the raw traffic observations into the spatial-temporal embedding; (2) **Attention weights adjusting:** refines the attention adjusting matrix by injecting controlled perturbations into the inputs and adjusts the learned attention weights; (3) **Alternating training strategy:** alternates the optimization of the two modules above, enabling progressive enhancement of dependency modeling and attention adjustment.

A. Spatial-temporal dependencies model

In this section, we present the details of modeling spatial-temporal dependencies which are directly utilized by the following sections.

1) Adaptive traffic observation encoding: In this section, we encode the periodicity, raw traffic observation and the dynamics of spatial-temporal dependency separately.

Periodicity, e.g., daily rush hours, weekends, or holidays is crucial for TFP, since it reflect human behavior and routine activities, which significantly influence traffic volumes. In this paper, traffic conditions are observed every five minutes, resulting in 288 observations at each location per day. We encode the day-of-week T_d and timestep-of-day T_t patterns explicitly. Given two learnable matrices $D_d \in \mathbb{R}^{7\times d_w}, D_t \in \mathbb{R}^{288\times d_w}$, each day of week and each time step in a day are represented as $T_d = I_d D_d$ and $T_t = I_t D_t$ respectively, where $I_d \in \mathbb{R}^7$ and $I_t \in \mathbb{R}^{288}$ are one-hot indexing matrices, using them to extract corresponding embedding from the learnable matrices D_d, and D_t. d_w is the dimension of embeddings.

Given a batch observations from N locations within T time steps, we model the dynamics of spatial-temporal dependency as a learnable matrix $T_a \in \mathbb{R}^{T\times N\times d_w}$.

The raw traffic observations are encoded by a fully connected layer, which can be formalized as $T_r = \mathrm{FC}\left(X_{t-T+1}, \cdots, X_t\right)$, $T_r \in \mathbb{R}^{T\times N\times d_w}$.

Thus, the input embeddings are represented as the concatenation of the embeddings above.

$$Z = T_d || T_t || T_a || T_r \tag{1}$$

where $Z \in \mathbb{R}^{T\times N\times d_{st}}$ and $d_{st} = 4d_w$.

2) Spatial-temporal dependencies encoding: In this section, we apply the vanilla transformer [36] along the ax (T) and the ax (N) of input embedding to capture spatial and temporal dependencies separately. The query, key and value temporal embeddings are defined as follows:

$$Q^t = ZW_Q^t, K^t = ZW_K^t, V^t = ZW_V^t \tag{2}$$

where $W_Q^t, W_K^t, W_V^t \in \mathbb{R}^{d_{st}\times d_{st}}$ are learnable matrics. The temporal attention weights are defined as:

$$A^t = \mathrm{SoftMax}\left(\frac{Q^t {K^t}^{\mathrm{T}}}{\sqrt{d_{st}}}\right) \tag{3}$$

where $A^t \in \mathbb{R}^{T\times T\times N}$ indicates the temporal attention weight matrix. Then the temporal dependencies with adjustment are defined as:

$$Z_d^t = (A^t \odot A_d^t) V^t \tag{4}$$

where $A_d^t \in \mathbb{R}^{T\times T\times N}$ is the temporal adjusting matrix which is defined in the following section and its elements are all initialized to 1.

Similarly, the spatial-temporal dependencies with adjustment is defined as:

$$Z_d^{st} = (A^s \odot A_d^s) V^s \tag{5}$$

where $Q^s = Z_d^t W_Q^s, K^s = Z_d^t W_K^s, V^s = Z_d^t W_V^s$ and $W_Q^s, W_K^s, W_V^s \in \mathbb{R}^{d_{st}\times d_{st}}$ are learnable matrix. The spatial attention weights are defined as $A^s = \mathrm{SoftMax}\left(\frac{Q_d^s {K_d^s}^{\mathrm{T}}}{\sqrt{d_{st}}}\right)$ and $A^s \in \mathbb{R}^{T\times N\times N}$. $A_d^s \in \mathbb{R}^{T\times N\times N}$ is the spatial adjusting matrix which is defined in the following section and its elements are all initialized to 1.

The output of this module with adjustment is $Z_d^{st} \in \mathbb{R}^{T\times N\times d_{st}}$. The unadjusted intermediate embedding and output are Z^t and Z^{st}, which do not apply the adjusting matrix and will be used in subsequent steps.

B. Attention adjusting matrix

In this section, we provide the details of perturbation-based temporal and spatial adjusting matrix.

1) Perturbation learning: In the context of traffic network, the connections (e.g., roads) between locations are fixed and does not change frequently. Injecting perturbation to connections may introduce unrealistic bias that harm for truly modeling the spatial-temporal dependencies. Comparing to connections, locations represent the physical components, e.g., sensors, road segments in the traffic network. Location features (e.g., speed, volume, occupancy at a sensor location) represent the real-time conditions of traffic which are often noisy or uncertain due to sensor errors, missing data, or unpredictable traffic patterns (e.g., accidents, weather changes). So, applying perturbations on location features helps simulate these uncertainties, enabling to make the model more efficiently e during inference.

Given the traffic observations at time t, $X_{t,..,} = x_t \in \mathbb{R}^{N\times d}$, the spatial perturbations on x_t are defined as $x_t' = x_t \odot \epsilon_t^s$. $\epsilon_t^s \sim \mathcal{N}(\mathbf{1}, \mathbf{\Sigma}_i = \sigma_i^2 \mathbf{I})$ is a Gaussian noise and $\odot$ is the Hadamard

product. Bigger deviation σ_t usually means more likely to get a larger perturbation. Similarly, the observations of the i-th location, $X_{.,i,.} = x_i \in \mathbb{R}^{T\times d}$, the temporal perturbations on x_i are defined as $x_i' = x_i \odot \epsilon_i^t$. $\epsilon_i^t \sim \mathcal{N}(\mathbf{1}, \mathbf{\Sigma}_i = \sigma_i^2\mathbf{I})$ is a Gaussian noise.

According to the analysis above, we can define the optimal perturbation injection is the largest Gaussian noise added to traffic observation that keeps the prediction results remain unchanged. Therefore, the process of finding the optimal perturbation is equivalent to estimate parameters of Gaussian noise. In this paper, we formalize the loss function of the parameter estimation as follow:

$$L_{pert} = \|\hat{y'} - \hat{y}\|^2 - \lambda \sum_{i=1}^{n} \mathcal{H}(\epsilon) \tag{6}$$

where the $\hat{y'}$ and $\hat{y}$ are the predictions of x' and x, $\mathcal{H}(\epsilon)$ is the entropy of ϵ. Minimizing the discrepancy between $\hat{y'}$ and $\hat{y}$ ensures the predictions before and after perturbation are consistent, while minimizing $-\lambda \sum_{i=1}^{n} \mathcal{H}(\epsilon_i)$ aims to encourage adding as much noise as possible to each location feature. To further clarify the intuition behind the equation, we provide the following calculation:

$$\begin{aligned} &\text{Minimize}\left(-\mathcal{H}(\epsilon)\right) \\ =&\text{Minimize}\left(-\int p(\epsilon)\ln p(\epsilon) d\epsilon\right) \\ =&\text{Maxmize}\left(\frac{1}{2}(\ln(2\pi\sigma^2)+1)\right) \\ =&\text{Maxmize}\left(\alpha + \log\sigma\right) \\ =&\text{Minimize}\left(-\log(\sigma)\right) \end{aligned} \tag{7}$$

where $\alpha = \ln 2(\frac{1}{2}\log(2\pi e))$ is a constant. Then we can rewrite the loss function as follows:

$$L_{pert} = \|\hat{y'} - \hat{y}\|^2 + \lambda \sum_{i=1}^{n} \log(\sigma_i) \tag{8}$$

2) Adjusting matrix update: In this section, we provide the details of adjusting the matrix based on parameters σs which are obtained by optimizing Eq. 8.

According to the assumption of this paper, if the embedding of one location at certain time can tolerate more perturbation injection without changing the prediction for the future traffic, the corresponding σ_i would have larger value. Then, it means the i-th location is less important, deserving small attention weight. Conversely, small σ_i indicates that the corresponding embedding is more important, deserving bigger attention weight.

Given $\sigma_\mathbf{t} = [\sigma_1, \cdots, \sigma_N]$ at time t, the spatial adjustment for the locations j is defined as follows:

$$\begin{aligned} A_d^s(t,j,.) &= A_d^s(t,j,.) - \frac{\sigma_t}{\text{Max}\{\sigma_t\}} \\ A_d^s &= \text{Softmax}(A_d^s) \end{aligned} \tag{9}$$

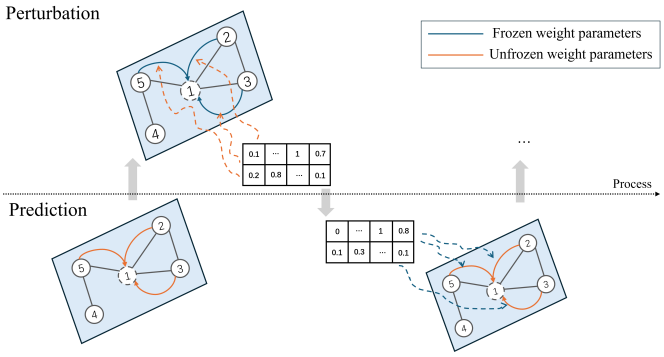


Fig. 3. The alternating training process of PARformer.

Similarly, given $\sigma_s = [\sigma_1, \cdots, \sigma_T]$ for the location s, the temporal adjustment for the i-th time step is defined as follows:

$$\begin{aligned} A_d^t(i,t,.) &= A_d^t(i,t,.) - \frac{\sigma_s}{\text{Max}\{\sigma_s\}} \\ A_d^t &= \text{Softmax}(A_d^t) \end{aligned} \tag{10}$$

For example, consider a prediction instance for location i. If a certain perturbation is applied to the traffic observations at location j, and this prediction for location i leads to a significant change, it indicates that location j plays a crucial role in the prediction of location i. In contrast, applying the same perturbation to the observations of location k, which fails to induce a noticeable change in the prediction of location i, suggests that location k is relatively unimportant. Such differences are expected to encode in the spatial adjusting matrix A_d^s, where the corresponding parameters satisfy $A_d^s(i,j) = \sigma_{i,j} < A_d^s(i,k) = \sigma_{i,k}$. Similarly, the temporal adjusting matrix is also defined by the different σ.

C. Alternating training strategy

The spatial-temporal dependencies module and adjusting matrix module are interdependent. When such two modules are trained simultaneously, the gradients from one can interfere with the other, leading to instability and slow convergence. To optimize both modules collaboratively, we adopt the alternating training strategy. As presented in Fig. 3, the training process is divided into three sub-processes, and the last two steps are performed alternately.

1) Initialization. The parameters of the spatial-temporal dependency modeling module are initialized randomly, while all elements in the spatial and temporal adjustment matrices are set to 1.
2) Prediction Module Training. In this phase, the parameters of perturbation module are frozen. The spatial-temporal dependency modeling module is trained to generate accurate predictions, using the fixed adjustment matrices A_d^t and A_d^s.
3) Perturbation Module Training. The parameters of spatial-temporal dependency modeling module are frozen and the perturbation module is then trained to learn an optimal σ.

Specifically, in the second step, we leverage the output of spatial-temporal dependency modeling module Z_d^{st} and Z^{st} to generate predictions. The predictions can be formalized as follows:

$$\begin{aligned} \hat{Y}_d &= \mathrm{FC}(Z_d^{st}) \\ \hat{Y} &= \mathrm{FC}(Z^{st}) \end{aligned} \tag{11}$$

where $\hat{Y}_d \in \mathbb{R}^{\hat{T} \times N}$, $\hat{Y} \in \mathbb{R}^{\hat{T} \times N}$ and $\mathrm{FC}(\cdot)$ is a fully connected layer. The loss function of prediction module is defined as:

$$\begin{aligned} L &= L_{task} + L_{adjust} \\ L_{task} &= a\mathcal{L}(\hat{Y}, Y) \\ L_{adjust} &= b\mathcal{L}(\hat{Y}_d, Y) + c\mathcal{L}(\hat{Y}_d, \hat{Y}) \end{aligned} \tag{12}$$

where Y is the ground truth, L_{task} assesses the discrepancy between model's prediction and the ground truth, while L_{adjust} assesses the difference before and after the attention weight adjustment. In this study, the loss function $\mathcal{L}$ is defined as the mean absolute error (MAE), which is a technique that has been extensively employed in the TFP field. In the third step, the loss function of perturbation module is defined as Eq. 8.

IV. Experiments

To objectively assess the effectiveness of PARformer, we benchmark it against a set of representative baselines and report comparative results.

A. Datasets

We evaluate PARformer on four widely used datasets: PEMS03, PEMS04, PEMS07, and PEMS08, which published by the California Department of Transportation [7]. These datasets provide 5-minute aggregated traffic measurements collected from sensor networks deployed in multiple districts across California.

B. Experiment Setting

To evaluate the performance of PARformer objectively, we use the following models as the baselines: HA [37], ARIMA [38], LSTM [39], GRU [40], GCRN [41], STGCN [9], ASTGCN [7], STGODE [42], MGCN [43], Bi-STAT [26], AdpSTGCN [25], STAEformer[24]. This set includes representative baselines from statistical methods, recurrent models, graph spatial-temporal models, attention based architectures and mamba methods in TFP.

We use the widely adopted mean absolute error (MAE) and root mean square error (RMSE) metrics to evaluate the predictive performance of each model.

We partition each dataset into training, validation, and test subsets using a 6:2:2 split following the setup used in prior work [24]. All experiments are conducted on a computing server equipped with an Intel(R) Xeon(R) Gold 5218 CPU @ 2.30GHz, 128 GB of RAM, and two NVIDIA RTX 3090 GPUs.

We adopt the widely used prediction setting where the model forecasts the next 12 time steps based on the past 12 observations. All models are trained with Adam with a learning rate of 1×10^{-4} for 40 epochs, using a batch size of 16. For PARformer, the perturbation module is frozen for the first 30 epochs to stabilize the predictor. In the final 10 epochs, we alternate training the predictor and perturbation module every 3 epochs by freezing and unfreezing their parameters. The hyperparameters a, b, c, λ in the loss function are set to $3, 2, 1, 0.5$.

C. Results and analysis

The experimental results are presented as follows.

TABLE I
THE MAEs AND RMSEs OF PARFORMER AND BASELINES ON FOUR DATASETS

Model	PEMS03		PEMS04		PEMS07		PEMS08	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
HA	31.58	52.69	38.03	59.24	34.59	58.63	34.86	59.24
ARIMA	35.31	38.26	33.73	48.80	117.27	153.20	31.09	44.32
GRU	19.12	32.85	23.68	39.27	28.78	42.05	22.00	36.22
LSTM	21.33	35.11	26.77	40.65	26.16	38.51	23.09	35.17
GCRN	19.76	31.34	26.16	39.13	25.57	38.76	20.44	30.47
STGCN	19.76	33.87	23.90	36.43	24.89	*37.06*	18.79	28.23
ASTGCN	23.21	37.55	22.96	36.16	24.45	37.48	24.13	36.28
STGODE	19.18	*28.58*	21.45	32.61	24.94	37.49	17.85	26.73
MGCN	20.03	34.91	22.64	35.70	25.76	40.44	17.87	28.29
Bi-STAT	*17.21*	28.69	20.92	33.34	25.57	40.54	16.68	26.95
AdpSTGCN	**17.12**	30.13	20.53	32.45	*23.36*	37.32	17.15	27.48
STAEformer	17.38	30.08	*19.28*	*31.77*	24.33	40.39	*14.94*	*25.62*
PARformer	*17.21*	**25.46**	**19.09**	**28.17**	**22.19**	**31.45**	**14.82**	**21.49**

1) Prediction performance: The MAEs and RMSEs across the four datasets are presented in Table I. The best and second-best results are highlighted in bold and italic, respectively.

According the experimental results, we can see that PARformer achieves the best or near-best results across all datasets. Specifically, it achieves almost the lowest MAE and RMSE on all datasets, except for the MAE on PEMS03, where our method achieves 17.21 compared to the best score of 17.12. Statistical baselines like HA and ARIMA show the poor performance across all datasets. LSTM and GRU perform significantly better than statistical methods. Their memory structure enables them to maintain the historical long-term dependencies. GNN-based models like GCRN, STGCN, ASTGCN achieve better performance than LSTM and GRU. Among them, STGODE achieves the best performance. Attention-based approaches such as Bi-STAT, AdpSTGCN, and STAEformer achieve competitive performance. For example, STAEformer achieves the best MAE on PEMS08 and Bi-STAT also shows impressive results on PEMS03 and PEMS08. These results demonstrate that attention mechanism provides the positive contribution to the performance of GNN-based and DL-based TFP models. MGCN, the Mamba-based model designed to offer a lightweight alternative to attention mechanisms, fails to outperform attention-based models such as Bi-STAT, AdpSTGCN, STAEformer and our approach.

2) Ablation study: To confirm the contribution of the proposed perturbation mechanism, we conducted ablation experiments to examine whether the perturbation strategy effectively improves the predictive performance of the PARformer and whether it helps calibrate the attention weight distribution. For this purpose, we compare PM-ON, the full model with

the perturbation module enabled, and PM-OFF, denoting the variant with the perturbation mechanism removed.

TABLE II
THE MAES AND RMSES OF PM-ON AND PM-OFF ON FOUR DATASETS

Model	PEMS03		PEMS04		PEMS07		PEMS08	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
PM-OFF	17.23	29.75	20.03	32.31	24.12	39.86	16.07	26.65
PM-ON	**17.21**	**25.46**	**19.09**	**28.17**	**22.19**	**31.45**	**14.82**	**21.49**

The prediction MAEs and RMSEs of PM-ON and PM-OFF are presented in Table II. It can be seen that incorporating the perturbation module consistently improves the predictive accuracy, indicating its positive impact on model performance.

A good attention weight distribution typically should assign higher weights to informative locations or time steps and lower weights to irrelevant counterparts.

To further quantify the degree of separation in the attention weight distribution, we define an index called the separation index (SI). Given normalized attention weight $\{a_k\}_{k=1}^{N}$, SI is formulated as

$$\mathrm{SI} = \frac{\mathbb{E}[a_i \mid i \in \mathcal{H}]}{\mathbb{E}[a_j \mid j \in \mathcal{L}]} \tag{13}$$

where $\mathbb{E}[\cdot]$ denotes the average operator, $\mathcal{H}$ corresponds to the set of attention weights above a predefined threshold, and $\mathcal{L}$ represents the set of weights below this threshold. A higher SI indicates a clearer separation between important and unimportant positions in the attention distribution.

TABLE III
MINIMUM, MEDIAN, MAXIMUM ATTENTION WEIGHTS OF PM-ON AND PM-OFF MODELS ON FOUR DATASETS

Values	PEMS03		PEMS04		PEMS07		PEMS08	
	PM-ON	PM-OFF	PM-ON	PM-OFF	PM-ON	PM-OFF	PM-ON	PM-OFF
Minimum	0.00001	0.00031	0.00063	0.00148	0.00051	0.00061	0.00334	0.00379
Median	0.00064	0.00124	0.00318	0.00302	0.00075	0.00080	0.00395	0.00428
Maximum	0.02382	0.02107	0.00586	0.00605	0.02274	0.02298	0.06056	0.07392

TABLE IV
SI FOR PM-ON AND PM-OFF MODELS

	PEMS03		PEMS04		PEMS07		PEMS08	
	PM-ON	PM-OFF	PM-ON	PM-OFF	PM-ON	PM-OFF	PM-ON	PM-OFF
SI	**658.88**	59.62	**2.77**	2.69	**20.83**	16.42	**13.30**	10.75

As shown in Table III, the median of the distribution serves as an optimal threshold, with PM-ON and PM-OFF SIs under this setting listed in Table IV.

According to the results, the SI values of the PM-ON model are consistently higher than those of PM-OFF model across all dataset. For example, the SI increases from 59.62 to 658.88 on PEMS03 and from 16.42 to 20.83 on PEMS07, indicating a greater separation in the attention weight distribution. The results are consistent with our hypothesis and we can confirm that our perturbation module effectively encourages attention calibrate the weight distribution.

V. CONCLUSION AND FUTURE WORKS

In this work, we propose PARformer, a novel attention-based TFP model that integrates a perturbation mechanism to calibrate attention weights and mitigate the impact of misleading attention patterns. Extensive experiments on multiple real-world datasets show that PARformer achieves superior or comparable prediction accuracy. For future work, we plan to explore alternative perturbation strategies to further optimize model performance and reduce the computational overhead of the perturbation process.

VI. ACKNOWLEDGMENT

This work was supported in part by the National Natural Science Foundation of China under Grant 62262071. The authors used ChatGPT only for language polishing grammar correction, and improvement of readability. All suggested edits were reviewed and validated by the authors, who take full responsibility for the final manuscript.

REFERENCES

[1] D. A. Tedjopurnomo, Z. Bao, B. Zheng, F. M. Choudhury, A. K. Qin, A survey on modern deep neural network for traffic prediction: Trends, methods and challenges, IEEE Transactions on Knowledge and Data Engineering 34 (4) (2022) 1544–1561. doi:10.1109/TKDE.2020.3001195.

[2] A. Miglani, N. Kumar, Deep learning models for traffic flow prediction in autonomous vehicles: A review, solutions, and challenges, Vehicular Communications 20 (2019) 100184. doi:https://doi.org/10.1016/j.vehcom.2019.100184.
URL https://www.sciencedirect.com/science/article/pii/S2214209619302311

[3] A. Boukerche, J. Wang, Machine learning-based traffic prediction models for intelligent transportation systems, Computer Networks 181 (2020) 107530. doi:https://doi.org/10.1016/j.comnet.2020.107530.
URL https://www.sciencedirect.com/science/article/pii/S1389128620311877

[4] X. Zhang, W. W. Y. Ng, T. Wang, Robust self-attention convlstm-based traffic flow prediction model, in: 2022 12th International Conference on Information Science and Technology (ICIST), 2022, p. 224–230. doi:10.1109/icist55546.2022.9926853.
URL http://dx.doi.org/10.1109/icist55546.2022.9926853

[5] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, Y. Bengio, Graph attention networks, in: International Conference on Learning Representations, 2018.
URL https://openreview.net/forum?id=rJXMpikCZ

[6] J. Jiang, C. Han, W. X. Zhao, J. Wang, 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, 2023. doi:10.1609/aaai.v37i4.25556.
URL https://doi.org/10.1609/aaai.v37i4.25556

[7] S. Guo, Y. Lin, N. Feng, C. Song, H. Wan, Attention based spatial-temporal graph convolutional networks for traffic flow forecasting, Proceedings of the AAAI Conference on Artificial Intelligence (2019) 922–929doi:10.1609/aaai.v33i01.3301922.
URL http://dx.doi.org/10.1609/aaai.v33i01.3301922

[8] C. Zheng, X. Fan, C. Wang, J. Qi, Gman: A graph multi-attention network for traffic prediction, Proceedings of the AAAI Conference on Artificial Intelligence (2020) 1234–1241doi:10.1609/aaai.v34i01.5477.
URL http://dx.doi.org/10.1609/aaai.v34i01.5477

[9] B. Yu, H. Yin, Z. Zhu, Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting, in: Proceedings of the 27th International Joint Conference on Artificial Intelligence, 2018, pp. 3634–3640.

[10] H. NT, T. Maehara, T. Murata, Revisiting graph neural networks: Graph filtering perspective, in: 2020 25th International Conference on Pattern Recognition (ICPR), 2021, pp. 8376–8383. doi:10.1109/ICPR48806.2021.9412278.

[11] Z. Shi, Y. Zhang, J. Wang, J. Qin, X. Liu, H. Yin, H. Huang, Dagcrn: Graph convolutional recurrent network for traffic forecasting with dynamic adjacency matrix, Expert Systems with Applications 227 (2023) 120259. doi:https://doi.org/10.1016/j.eswa.2023.120259.
URL https://www.sciencedirect.com/science/article/pii/S0957417423007613

[12] T. Ye, L. Dong, Y. Xia, Y. Sun, Y. Zhu, G. Huang, F. Wei, Differential transformer, in: The Thirteenth International Conference on Learning Representations, 2025.
URL https://openreview.net/forum?id=OvoCm1gGhN

[13] S. K. Maurya, X. Liu, T. Murata, Feature selection: Key to enhance node classification with graph neural networks, CAAI Transactions on Intelligence Technology 8 (1) (2023) 14–28. doi:10.1049/cit2.12166.
URL https://doi.org/10.1049/cit2.12166

[14] M. Shi, Y. Tang, X. Zhu, Y. Zhuang, M. Lin, J. Liu, Feature-attention graph convolutional networks for noise resilient learning, IEEE Transactions on Cybernetics (2022) 7719–7731doi:10.1109/tcyb.2022.3143798.
URL http://dx.doi.org/10.1109/tcyb.2022.3143798

[15] M. Ahmed, A. Cook, Analysis of freeway traffic time-series data by using box-jenkins techniques, Transportation Research Record,Transportation Research Record (Jan 1979).

[16] B. Pan, U. Demiryurek, C. Shahabi, Utilizing real-world transportation data for accurate traffic prediction, in: 2012 ieee 12th international conference on data mining, IEEE, 2012, pp. 595–604.

[17] I. Okutani, Y. J. Stephanedes, Dynamic prediction of traffic volume through kalman filtering theory, Transportation Research Part B: Methodological 18 (1) (1984) 1–11. doi:https://doi.org/10.1016/0191-2615(84)90002-X.
URL https://www.sciencedirect.com/science/article/pii/019126158490002X

[18] A. J. Smola, B. Schölkopf, A tutorial on support vector regression, Statistics and Computing (2004) 199–222doi:10.1023/b:stco.0000035301.49549.88.
URL http://dx.doi.org/10.1023/b:stco.0000035301.49549.88

[19] X. Dong, T. Lei, S. Jin, Z. Hou, Short-term traffic flow prediction based on xgboost, in: 2018 IEEE 7th Data Driven Control and Learning Systems Conference (DDCLS), 2018, pp. 854–859. doi:10.1109/DDCLS.2018.8516114.

[20] Y. Tian, L. Pan, Predicting short-term traffic flow by long short-term memory recurrent neural network, in: 2015 IEEE International Conference on Smart City/SocialCom/SustainCom (SmartCity), 2015, pp. 153–158. doi:10.1109/SmartCity.2015.63.

[21] D. Zhang, M. R. Kabuka, Combining weather condition data to predict traffic flow: A gru based deep learning approach, in: 2017 IEEE 15th Intl Conf on Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence and Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech), 2017, pp. 1216–1219. doi:10.1109/DASC-PICom-DataCom-CyberSciTec.2017.194.

[22] S. Sun, H. Wu, L. Xiang, City-wide traffic flow forecasting using a deep convolutional neural network, Sensors 20 (2) (2020). doi:10.3390/s20020421.
URL https://www.mdpi.com/1424-8220/20/2/421

[23] L. Zhao, Y. Song, C. Zhang, Y. Liu, P. Wang, T. Lin, M. Deng, H. Li, T-gcn: A temporal graph convolutional network for traffic prediction, IEEE Transactions on Intelligent Transportation Systems 21 (9) (2020) 3848–3858. doi:10.1109/TITS.2019.2935152.

[24] H. Liu, Z. Dong, R. Jiang, J. Deng, J. Deng, Q. Chen, X. Song, Spatio-temporal adaptive embedding makes vanilla transformer sota for traffic forecasting, in: Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, CIKM '23, Association for Computing Machinery, New York, NY, USA, 2023, p. 4125–4129. doi:10.1145/3583780.3615160.
URL https://doi.org/10.1145/3583780.3615160

[25] X. zhang, X. Chen, H. Tang, Y. Wu, H. Shen, J. Li, Adpstgcn: Adaptive spatial–temporal graph convolutional network for traffic forecasting, Knowledge-Based Systems 301 (2024) 112295. doi:https://doi.org/10.1016/j.knosys.2024.112295.
URL https://www.sciencedirect.com/science/article/pii/S0950705124009298

[26] C. Chen, Y. Liu, L. Chen, C. Zhang, Bidirectional spatial-temporal adaptive transformer for urban traffic flow forecasting, IEEE Transactions on Neural Networks and Learning Systems (2022).

[27] N. F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, P. Liang, Lost in the middle: How language models use long contexts, Transactions of the Association for Computational Linguistics 12 (2024) 157–173.

[28] C. Luo, H. Mobahi, S. Bengio, Data augmentation via structured adversarial perturbations, ArXiv abs/2011.03010 (2020).
URL https://api.semanticscholar.org/CorpusID:226254079

[29] J. Chen, D. Tam, C. Raffel, M. Bansal, D. Yang, An empirical survey of data augmentation for limited data learning in nlp, Transactions of the Association for Computational Linguistics 11 (2023) 191–211. arXiv:https://direct.mit.edu/tacl/article-pdf/doi/10.1162/tacl_a_00542/2074871/tacl_a_00542.pdf, doi:10.1162/tacl_a_00542.
URL https://doi.org/10.1162/tacl_a_00542

[30] Z. Carmichael, W. J. Scheirer, Unfooling perturbation-based post hoc explainers, Proceedings of the AAAI Conference on Artificial Intelligence 37 (6) (2023) 6925–6934. doi:10.1609/aaai.v37i6.25847.
URL https://ojs.aaai.org/index.php/AAAI/article/view/25847

[31] R. Ying, D. Bourgeois, J. You, M. Zitnik, J. Leskovec, Gnnexplainer: Generating explanations for graph neural networks, Cornell University - arXiv,Cornell University - arXiv (Mar 2019).

[32] W. E. Zhang, Q. Z. Sheng, A. Alhazmi, C. Li, Adversarial attacks on deep-learning models in natural language processing: A survey, ACM Transactions on Intelligent Systems and Technology (TIST) 11 (3) (2020) 1–41.

[33] F. Liu, H. Liu, W. Jiang, Practical adversarial attacks on spatiotemporal traffic forecasting models, Advances in Neural Information Processing Systems 35 (2022) 19035–19047.

[34] J. Bai, J. Zhu, Y. Song, L. Zhao, Z. Hou, R. Du, H. Li, A3t-gcn: Attention temporal graph convolutional network for traffic forecasting, ISPRS International Journal of Geo-Information 10 (7) (2021). doi:10.3390/ijgi10070485.
URL https://www.mdpi.com/2220-9964/10/7/485

[35] L. Kong, H. Yang, W. Li, Y. Zhang, J. Guan, S. Zhou, Traffexplainer: A framework toward gnn-based interpretable traffic prediction, IEEE Transactions on Artificial Intelligence 6 (3) (2025) 559–573. doi:10.1109/TAI.2024.3459857.

[36] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Advances in neural information processing systems 30 (2017).

[37] I. Welch, A. Goyal, A comprehensive look at the empirical performance of equity premium prediction, The Review of Financial Studies 21 (4) (2008) 1455–1508.

[38] C. Chen, J. Hu, Q. Meng, Y. Zhang, Short-time traffic flow prediction with arima-garch model, in: 2011 IEEE Intelligent Vehicles Symposium (IV), IEEE, 2011, pp. 607–612.

[39] B. Yang, S. Sun, J. Li, X. Lin, Y. Tian, Traffic flow prediction using lstm with feature enhancement, Neurocomputing 332 (2019) 320–327.

[40] D. Zhang, M. R. Kabuka, Combining weather condition data to predict traffic flow: a gru-based deep learning approach, IET Intelligent Transport Systems 12 (7) (2018) 578–585.

[41] Y. Seo, M. Defferrard, P. Vandergheynst, X. Bresson, Structured sequence modeling with graph convolutional recurrent networks, in: International Conference on Neural Information Processing, Springer, 2018, pp. 362–373.

[42] Z. Fang, Q. Long, G. Song, K. Xie, Spatial-temporal graph ode networks for traffic flow forecasting, in: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, 2021, pp. 364–373.

[43] W. Lin, Z. Zhang, G. Ren, Y. Zhao, J. Ma, Q. Cao, Mgcn: Mamba-integrated spatiotemporal graph convolutional network for long-term traffic forecasting, Knowledge-Based Systems 309 (2025) 112875. doi:https://doi.org/10.1016/j.knosys.2024.112875.
URL https://www.sciencedirect.com/science/article/pii/S0950705124015090