PAANet: Patch-Aligned Attention Network for Spatio-Temporal Prediction on Low-Quality Data

Jiajun Ouyang, Yingchi Mao*, Hongliang Zhou, Ling Chen, Yi Rong
College of Computer Science and Software Engineering, Hohai University, Nanjing, China
{oyjiajun, yingchimao, zhouhongliang, 221307050018, 230207060002}@hhu.edu.cn

***Abstract*—Spatio-temporal prediction enables reliability in climate monitoring and traffic management. Presently, the deployment of sensor devices generates massive data, which provides support for spatio-temporal prediction by extracting spatio-temporal dependencies in the data. However, making accurate spatio-temporal prediction is still challenging due to the presence of low-quality data, which generally includes two characteristics: 1) missing data, *i.e.*, missing data disrupts the continuity of time series, and 2) noise data, *i.e.*, deviations are introduced into data due to noise. Missing and noise data make it difficult for the model to accurately capture temporal dependencies between timesteps and the true spatial relationships between time-varying variables. To ensure prediction accuracy on missing and noise data, we propose a novel framework for spatio-temporal prediction, namely *Patch-Aligned Attention Network* (PAANet). In particular, PAANet is composed of two main parts, called Temporal Feature Mining (TFM) and Spatial Feature Mining (SFM) modules. To jointly address missing and noise data issues in the temporal dimension, the TFM module is designed on the basis of multi-scale temporal feature encoder (MSTFE) and wavelet temporal decomposition encoder (WTDE). More specifically, MSTFE is used for the first time to capture both local and global dependencies under missing data conditions. Then, WTDE is introduced to suppress high-frequency noise and stabilize temporal trends. Meanwhile, to handle both noise and missing observations in the spatial dimension, the SFM module leverages a missing-aware graph imputation (MAGI) layer and a gated graph attention network (GGAtt), which imputes missing node information and dynamically filters noise. Experimental results on two real-world datasets with missing and noise data demonstrate that PAANet is superior to advanced benchmarks, as evidenced by the higher prediction precision.**

***Index Terms*—time series forecasting, low-quality time series, wavelet transform, cross-attention, graph neural network**

I. Introduction

Spatio-temporal prediction aims to model and forecast future trends across time and space based on historical observations, and has become an important research focus recently. The rapid development of sensing technologies and the deployment of large-scale sensors have produced massive data, enabling accurate forecasting for applications such as climate monitoring and traffic management [1]–[3]. In recent years, deep learning techniques have shown superior capability in capturing such dependencies. Graph-based approaches [4]–[6] leverage graph structures to represent spatial relationships, while temporal models such as RNN [7], [8] and Temporal Convolutional Network (TCN) [9] capture temporal dynamics. Recent attention-based architectures [10]–[12] further enhance the modeling of global and dynamic spatio-temporal dependencies through self-attention mechanisms. Nevertheless, most existing approaches assume complete and clean data, while real-world spatio-temporal datasets often encounter the following challenges due to the high missing rate and complex noise in historical data:

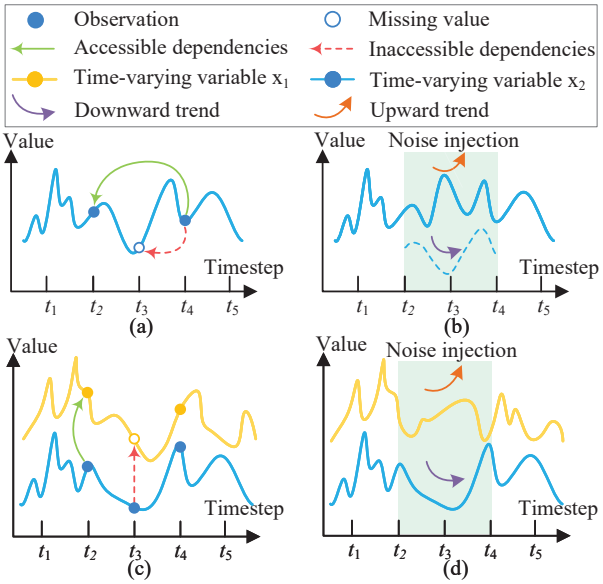

Fig. 1: (a) and (b) are examples demonstrating the effects of missing and noise values in temporal feature extraction. (c) and (d) are examples demonstrating missing and noise effects on spatial feature extraction.

Challenge I: Impact of missing data. In real-world environments, missing data arise when observations are absent at certain timesteps due to collection or transmission failures. As illustrated in Figure 1(a), such missing values interrupt temporal continuity, hindering the extraction of dependencies between adjacent timesteps. Meanwhile, when x_1 is missing at timestep t_3 (Figure 1(c)), its actual correlation with other spatially related variables (x_2) becomes obscured, leading to inaccurate spatial feature aggregation.

Challenge II: Impact of noise data. Noise occurs when sensor errors or environmental fluctuations cause deviations from true values. As shown in Figure 1(b) and 1(d), noise distorts temporal and spatial dependencies: it may alter intrinsic trends (e.g., changing a downward trend into an upward one)

*Corresponding author

and reverse the correlation between variables (x_1, x_2), resulting in unreliable pattern extraction.

To address these issues, we propose PAANet, which contains a temporal feature mining module and a spatial feature mining module. The temporal module aligns irregular sequences and suppresses temporal noise, while the spatial module imputes incomplete node features and models noise-resistant graph dependencies. The contributions are summarized as follows:

- We introduce PAANet, a unified spatio-temporal prediction framework capable of simultaneously handling missing and noise data, addressing two challenges in real-world time-series modeling.
- We design two key modules: the TFM module for robust temporal dependency extraction under low-quality data conditions, and the SFM module for reliable spatial representation learning through adaptive graph reasoning.
- We conduct extensive experiments on two real-world datasets(*i.e.*, USHCN and Air), demonstrating that PAANet significantly outperforms advanced benchmarks in prediction accuracy for the next 12 horizons.

II. Preliminaries

A. Definitions

1) Definition I. Time-varying network: We model N time-varying variables as a graph $G = (V, \mathcal{E}, A)$, where nodes V represent the variables and edges $\mathcal{E}$ denote their connections. The adjacency matrix $A \in \mathbb{R}^{N\times N}$ encodes spatial dependencies, with entries $a_{ij} \in [0, 1]$ indicating the positive correlation strength between the i-th and j-th variables.

2) Definition II. Time-varying variable: The multivariate time series is defined as $X = \{x_n^{t_i}\}_{n=1,i=1}^{N,L} \in \mathbb{R}^{N\times L\times C}$, where L is the time window length and C is the feature dimension. In real-world scenarios, missing data results in irregular lengths $L_n \leq L$ for each variable. Additionally, observations are often corrupted by additive Gaussian noise $\epsilon_n \sim \mathcal{N}(0, \sigma^2)$ controlled by an injection rate v. Therefore, the effective input under low-quality conditions is formulated as $\tilde{X} = \{\tilde{x}_n^{t_i}\}_{n=1,i=1}^{N,L_n} \in \mathbb{R}^{N\times L_n\times C}$, where $\tilde{x}_n^{t_i} = x_n^{t_i} + \epsilon_n^{t_i}$.

B. Problem Formulation.

Given the time-varying network $G = (V, \mathcal{E}, A)$, and the multiple time-varying variables $\tilde{X} = \{\tilde{x}_n^{t_i}\}_{n=1,i=1}^{N,L_n}$, the goal is to learn a function $f(\cdot)$ that performs spatio-temporal prediction on missing and noise data:

$$\hat{Y} = f(G, \tilde{X}), \tag{1}$$

where $\hat{Y} = \{\hat{q}_n^j\}_{n=1,j=1}^{N,Q,} \in \mathbb{R}^{N\times Q\times C}$ denotes the predicted sequence for N time-varying variables in the next Q timesteps. Note that C is set to 1 in this paper.

III. PAANet Design

A. System Overview

As shown in Figure 2, PAANet consists of a temporal feature mining module and a spatial feature mining module. TFM aligns irregular sequences and suppresses temporal noise, while SFM imputes missing spatial information and models noise-resistant graph dependencies.

B. TFM Module Design

To extract temporal dependencies on low-quality data, TFM module is designed. Comprised of three components (*e.g.*, patching alignment, MSTFE, and WTDE), TFM module forms a pipeline structure as displayed in Figure 2(b).

1) Patching Alignment: Due to missing values, the lengths of time-varying variables are irregular, making it difficult for time-series networks to learn temporal dependencies. To address this, a transformable patching scheme is designed to align variables along the temporal dimension. Specifically, given the input $\tilde{X} \in \mathbb{R}^{N\times L_n\times C}$, each variable $\tilde{X}_n$ is processed independently by grouping its neighboring timesteps into P equal-length patches of size L_p. This process is formulated as:

$$\tilde{\mathcal{X}}_n^{patch} = \text{patching}(\tilde{X}_n), \tag{2}$$

where $\tilde{\mathcal{X}}_n^{patch} \in \mathbb{R}^{L_p\times P\times C}$ are the aligned representations for the n-th variable. By applying Eq. 2 across all N variables, we directly obtain the unified representations $\tilde{\mathcal{X}}^{patch} \in \mathbb{R}^{N\times L_p\times P\times C}$.

To encode temporal information, we apply Time2Vec [13] to the timestamps $T \in \mathbb{R}^{N\times L_p\times P}$, where $T_p \in \mathbb{R}^{L_p}$ denotes the timestamps of the p-th patch. The time embedding of patch p is defined as:

$$\phi(T_p)[d_p] = \begin{cases} \omega_{0p} \cdot T_p + \alpha_{0p}, & d_p = 0 \\ \sin(\omega_{dp} \cdot T_p + \alpha_{dp}), & 0 < d_p < D_t \end{cases}, \tag{3}$$

where D_t is the embedding dimension, and ω and α are learnable parameters. The resulting time embedding $\phi(T) \in \mathbb{R}^{N\times L_p\times P\times D_t}$ is then concatenated with the aligned representation $\tilde{X}^{patch}$ to obtain

$$\mathcal{Z} = \phi(\mathcal{T}) \,\|\, \tilde{\mathcal{X}}^{patch}. \tag{4}$$

where $Z \in \mathbb{R}^{N\times L_p\times P\times (D_t+C)}$ denotes the unified representation.

2) Multi-scale Temporal Feature Encoder: To capture local temporal dependencies within each patch and global ones between patches, a multi-scale temporal feature encoder is proposed, consisting of an adaptively weighted TCN and a standard attention mechanism.

Adaptively Weighted TCN: Standard TCN models adjacent timestep temporal relationships via causal and dilated convolutions but uses fixed-shape convolution filters, failing to adapt to variable-length patches from missing values. Thus, we design the enhanced adaptively weighted TCN, whose attention-based meta-filter mechanism replaces fixed filters to dynamically adjust convolution filter shape for variable-length patches. This meta-filter design dynamically adjusts to irregular observation patterns caused by missing values. The unified representations $\mathcal{Z} \in \mathbb{R}^{N\times L_p\times P\times (D_t+C)}$ are treated as the input of adaptively weighted TCN, in which $Z_n^p = \{z_n^i\}_{i=1}^{L_p} \in \mathbb{R}^{L_p\times (D_t+C)}$ is the unified representations of the p-th patch for the n-th time-varying variable.

The attention-based meta-filter mechanism contains D convolution filters. Each convolution filter shares the same shape, but its parameters are independent, intending to learn various

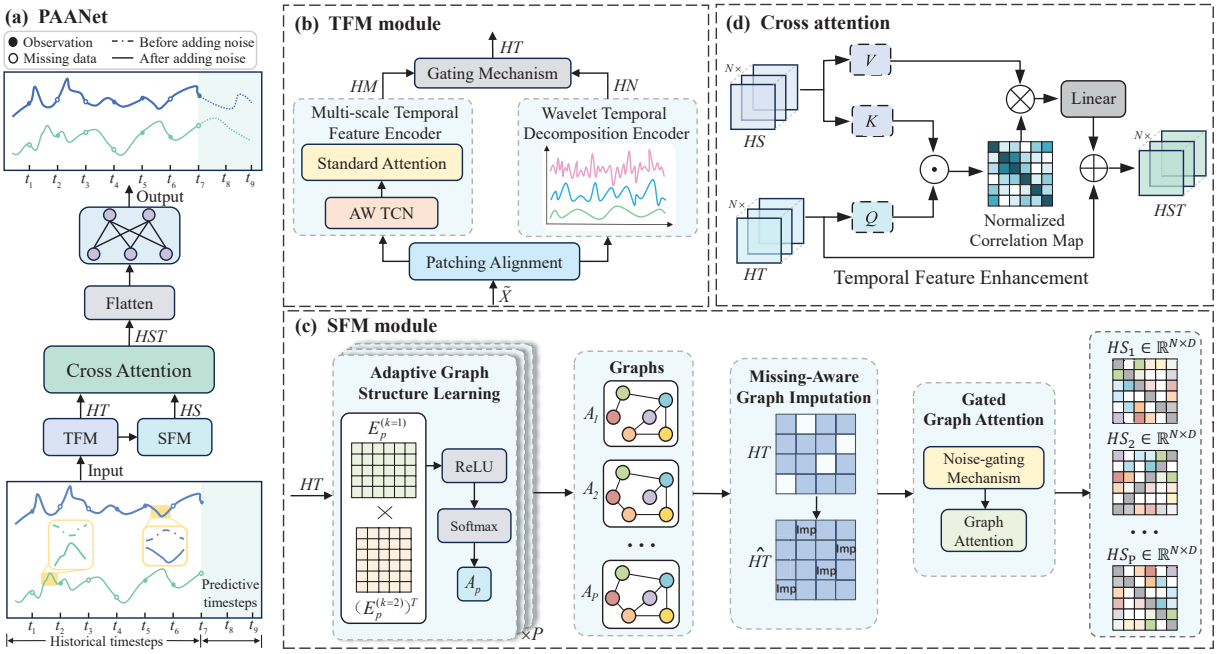

Fig. 2: Overview of our PAANet. (a) PAANet. (b) TFM module. (c) SFM module. (d) Cross attention.

local temporal dependencies. In detail, for the i-th unified representations $z_n^i \in \mathbb{R}^{D_t+C}$ in Z_n^p, we separately use D convolution filters to handle z_n^i :

$$h_{conv}^p = \left[\sum_{i=1}^{L_p} f_d[i]^\top \cdot z_n^i \right]_{d=1}^{D}, \tag{5}$$

where $h_{conv}^p \in \mathbb{R}^D$ is local temporal representations for the p-th patch. $[...]_{d=1}^D$ denotes the traversal operation by D convolution filters. The subscript $conv$ in h_{conv}^p stands for the convolution operation. $f_d[i] \in \mathbb{R}^{D_t+C}$ represents the i-th element in the d-th convolution filter, which is denoted as:

$$f_d[i] = \frac{\exp(F_d(z_n^i))}{\sum_{j=1}^{L_p} \exp(F_d(z_n^j))}. \tag{6}$$

where $F_d(\cdot)$ is a learnable meta-filter operation, aiming to generate parameters of the d-th convolution filter. After L_p operations using Eq.6, we obtain the d-th convolution filter $f_d = \{f_d[i]\}_{i=1}^{L_p} \in \mathbb{R}^{L_p \times (D_t+C)}$.

Leveraging Eqs.5-6, adaptively weighted TCN generates local temporal representations of P patches in the n-th time-varying variable, indicated as $H_{n,conv} = \{h_{conv}^p\}_{p=1}^P \in \mathbb{R}^{P \times D}$.

Standard Attention Mechanism: To capture global dependencies, the local temporal representations $H_{n,conv}$ are summed with their positional encodings $PE_n \in \mathbb{R}^{P \times D}$ to obtain $HX_n \in \mathbb{R}^{P \times D}$, serving as input to a single-layer multi-head self-attention mechanism. The computation for the b-th head is:

$$HM_n^b = Softmax\left(\frac{q_n^b \cdot (k_n^b)^\top}{\sqrt{D/B}} \right) \cdot v_n^b, \tag{7}$$

where $HM_n^b \in \mathbb{R}^{P \times (D/B)}$ represents the b-th head's output. The query q_n^b, key k_n^b, and value v_n^b are obtained via linear projections of HX_n using learnable weights $W_q^b, W_k^b, W_v^b \in \mathbb{R}^{D \times (D/B)}$. The representations from all B heads are then concatenated to formulate the temporal representation $HM_n \in \mathbb{R}^{P \times D}$:

$$HM_n = (||_{b=1}^B HM_n^b) \cdot W_o^t, \tag{8}$$

where $W_o^t \in \mathbb{R}^{D \times D}$ is a learnable parameter, $\|$ stands for the concatenation. Applying this operation across all variables yields the missing-aware temporal representations $HM \in \mathbb{R}^{N \times P \times D}$.

3) Wavelet Temporal Decomposition Encoder: Traditional wavelet transforms (WT) [14] are limited by fixed filters for noise separation. Our Wavelet Temporal Decomposition Encoder addresses this by introducing a learnable and differentiable architecture that adaptively decomposes frequencies according to noise patterns. Given the patched sequence $z_n^p \in \mathbb{R}^{L_p \times (D_t+C)}$, a group of trainable filters $\{h^{(l)}\}_{l=1}^L$ extracts distinct frequency bands:

$$F_n^{p,(l)}(t) = \sum_{\tau=1}^{k} h^{(l)}(\tau)\, z_n^p(t-\tau+1), \tag{9}$$

where $F_n^{p,(l)} \in \mathbb{R}^{L_p \times (D_t+C)}$ represents the decomposed signal at the l-th frequency band, and $t \in [1, L_p]$ indexes the temporal position within the patch. For each frequency band, the band-level feature $H_n^{p,(l)}$ is obtained by temporal aggregation over all timesteps in $F_n^{p,(l)}$:

$$H_n^{p,(l)} = \frac{1}{L_p} \sum_{t=1}^{L_p} F_n^{p,(l)}(t), \tag{10}$$

Then, WTDE flexibly fuses all band-level features through noise-aware weighting:

$$HN_n^p = \sum_{l=1}^{L} \beta_n^{p,(l)} H_n^{p,(l)}, \tag{11}$$

$$\beta_n^{p,(l)} = \frac{\exp(w^\top H_n^{p,(l)} - \gamma s_n^{p,(l)})}{\sum_{l'} \exp(w^\top H_n^{p,(l')} - \gamma s_n^{p,(l')})}, \tag{12}$$

where $s_n^{p,(l)}$ estimates the noise intensity of the l-th band and is derived through normalized temporal variance calculation. γ controls the penalty on unreliable components. By performing the above operations (Eqs. 9–12) for all N time-varying variables and P patches, WTDE yields the complete denoised representation $HN = \{HN_n^p\}_{n=1,p=1}^{N,P} \in \mathbb{R}^{N\times P\times D}$, which preserves smooth trends while suppressing noisy fluctuations.

Finally, to enhance temporal robustness, HN is adaptively fused with the missing-aware representations HM from MSTFE through a gating mechanism:

$$HT = g \odot HN + (1 - g) \odot HM. \tag{13}$$

where $HT \in \mathbb{R}^{N\times P\times D}$ is the final temporal representations, $g \in \mathbb{R}^{N\times P\times D}$ is a learnable gate that balances the contributions of both representations.

C. SFM Module Design

Besides temporal feature extraction, a profound understanding of spatial dependencies on low-quality data is crucial to achieving accurate spatio-temporal prediction. To this end, SFM module is presented as shown in Figure 2(c), containing an adaptive graph structure learning layer, a MAGI layer and GGAtt.

1) Adaptive Graph Structure Learning Layer: Low-quality data alter the true dependencies between variables. Thus, we design an adaptive graph structure learning layer to dynamically model spatial relationships at the patch level. The layer takes $HT \in \mathbb{R}^{N\times P\times D}$ as input, with $HT_p \in \mathbb{R}^{N\times 1\times D}$ denoting the representations for the p-th patch. To construct time-varying adaptive graphs, we first employ K linear projections to map the temporal representations HT_p into graph space. Note that K is set to 2. The k-th ($k \in \{1, 2\}$) projection operation is formulated as:

$$E_p^{d,(k)} = HT_p \cdot W_d^{(k)}, \tag{14}$$

where $E_p^{d,(k)} \in \mathbb{R}^{N\times D_g}$ denotes the dynamic temporal projection of the p-th patch. D_g refers to the graph embedding dimension. $W_d^{(k)} \in \mathbb{R}^{D\times D_g}$ is learnable parameters.

Then, we compute the K gating vectors to adaptively control the injection of patch-wise information into HT_p. This enables the model to dynamically adjust the impact of temporal dependencies under varying noise conditions. The k-th gating vector $g_p^{(k)} \in \mathbb{R}^N$ is expressed as:

$$g_p^{(k)} = \text{ReLU}\left(\tanh\left(\left[HT_p \parallel E_s^{(k)}\right] \cdot W_g^{(k)}\right)\right), \tag{15}$$

where $E_s^{(k)} \in \mathbb{R}^{N\times D_g}$ denotes the learnable static node embeddings, representing the inherent properties of time-varying variables. $W_g^{(k)} \in \mathbb{R}^{D+D_g}$ is learnable parameters. These static embeddings provide a prior for relationships that persist even when observations are missing or noisy. Finally, we generate K time-varying node embeddings by combining the static node embeddings and gating vector, in which the k-th time-varying node embedding is calculated as:

$$E_p^{(k)} = E_s^{(k)} + g_p^{(k)} \odot E_p^{d,(k)}, \tag{16}$$

where $E_p^{(k)} \in \mathbb{R}^{N\times D_g}$ denotes the time-varying node embeddings of the p-th patch for N time-varying variables. $\odot$ denotes element-wise multiplication. Based on K time-varying node embeddings, the adaptive adjacency matrix $A_p \in \mathbb{R}^{N\times N}$ of the p-th patch is computed as:

$$A_p = \text{Softmax}\left(\text{ReLU}\left(E_p^{(k=1)} \cdot (E_p^{(k=2)})^\top\right)\right). \tag{17}$$

A_p contains dynamic spatial dependencies of the p-th patch between N time-varying variables.

2) Missing-Aware Graph Imputation: To handle spatially correlated missingness, we introduce a Missing-Aware Graph Imputation (MAGI) layer. MAGI performs imputation directly within the graph convolution using the learned adaptive adjacency A_p and estimates missing features through neighbor aggregation while preserving observed values. Specifically, given the temporal representations $HT_p \in \mathbb{R}^{N\times D}$ obtained from the TFM module for the p-th patch and the corresponding binary observation mask $M_p \in \{0, 1\}^{N\times D}$, MAGI first constructs the observed tensor as:

$$O_p = M_p \odot HT_p, \tag{18}$$

where each element of M_p indicates whether the corresponding variable is observed (1) or missing (0). Then, the available information from neighboring nodes is propagated through the learned adaptive adjacency matrix $A_p \in \mathbb{R}^{N\times N}$:

$$\text{Agg}_p = \widetilde{A}_p O_p, \tag{19}$$

where $\text{Agg}_p \in \mathbb{R}^{N\times D}$ is the aggregated neighborhood information, and $\widetilde{A}_p \in \mathbb{R}^{N\times N}$ denotes the row-normalized adjacency to ensure scale invariance. This aggregation effectively diffuses spatial context from observed nodes to those with missing values.

Next, MAGI fuses the aggregated neighbor features and self-node information through a learnable mapping to estimate candidate imputation values:

$$\text{Cand}_p = \phi(\text{Agg}_p W_1 + O_p W_2 + b), \tag{20}$$

where $W_1, W_2 \in \mathbb{R}^{D\times D}$ are trainable projection matrices, $b \in \mathbb{R}^D$ is the bias term, and $\phi(\cdot)$ denotes a nonlinear activation function such as GELU. Finally, the imputed representation $\hat{HT}_p \in \mathbb{R}^{N\times D}$ is constructed by filling the missing entries with the generated candidates while keeping the original observations:

$$\hat{HT}_p = O_p + (1 - M_p) \odot \text{Cand}_p. \tag{21}$$

Through this process, MAGI adaptively restores missing spatial representations by integrating information from correlated nodes in the learned graph structure.

By performing the above imputation on all patches across the N time-varying variables, we obtain the missing-aware unified representations $\hat{HT} \in \mathbb{R}^{N\times P\times D}$.

3) Gated Graph Attention: Spatial attention mechanisms [15], [16] like GAT [17] adaptively model dependencies by dynamically weighting neighboring nodes. However, observational noise can obscure true relationships, causing GAT to learn spurious correlations. In light of this, we design an improved variant of GAT, namely GGAtt. GGAtt replaces GAT's direct feature propagation [18] with a noise-gating mechanism, which dynamically filters out noise while capturing actual spatial dependencies. In particular, the p-th patch's temporal representations $\hat{HT}_p \in \mathbb{R}^{N\times D}$ of N time-varying variables are regarded as the input of GGAtt. For the variable s_r ($r \in [1, N]$), its temporal representations $\hat{HT}_p^r \in \mathbb{R}^D$ are first denoised through noise-gating mechanism, formulated as:

$$\tilde{HT}_p^r = ga_r \odot \hat{HT}_p^r, \tag{22}$$

where $\tilde{HT}_p^r \in \mathbb{R}^D$ is denoised representations. The noise-gating vector $ga_r \in \mathbb{R}^D$ of the r-th variable is defined as:

$$ga_r = \sigma(W_f \cdot \hat{HT}_p^r + b_f), \tag{23}$$

where $W_f \in \mathbb{R}^{D\times D}$ and $b_f \in \mathbb{R}^D$ are learnable parameters. σ stands for the sigmoid function.

After obtaining $\tilde{HT}_p^r$, GGAtt employs an improved one-layer attention mechanism to extract the true spatial dependencies. For the variable s_r and its neighboring variable s_q, the operation of the b-th head is described as:

$$HS_p^{r,b} = \text{Softmax}\left(\frac{e_{r,q}}{\sqrt{D/B}}\right) \cdot v^{r,b}, \tag{24}$$

where $HS_p^{r,b} \in \mathbb{R}^{D/B}$ stands for the b-th head's spatial representations. $v^{r,b} = \tilde{HT}_p^q \cdot W_{v'}^b$ is value vector. $W_{v'}^b \in \mathbb{R}^{D\times(D/B)}$ is learnable parameters. $e_{r,q}$ is the attention coefficient between variables s_r and s_q, which can be computed as:

$$e_{r,q} = \text{LeakyReLU}\left(W_e{}^\top \cdot \left(q^{r,b} \| k^{r,b}\right)\right) \cdot A_p[r, q], \tag{25}$$

where $W_e \in \mathbb{R}^{2D}$ is learnable parameters. $q^{r,b} = \tilde{HT}_p^r \cdot W_{q'}^b$, and $k^{r,b} = \tilde{HT}_p^q \cdot W_{k'}^b$ denote query vector, and key vector, respectively. $W_{q'}^b$, and $W_{k'}^b \in \mathbb{R}^{D\times(D/B)}$ are learnable parameters. $A_p[r, q] \in [0, 1]$ is the element located at the i-th row and j-th column of the adaptive adjacency matrix A_p.

Once $HS_p^{r,b}$, and $b \in [1, B]$, is obtained, the spatial representations $HS_p^r \in \mathbb{R}^D$ of time-varying variable s_r at the p-th patch is formulated as:

$$HS_p^r = (\|_{b=1}^B HS_p^{r,b}) W_o^s. \tag{26}$$

where $W_o^s \in \mathbb{R}^{D\times D}$ is learnable parameters. We perform multiple iterations using Eqs. 22–26, intending to acquire spatial representations $HS \in \mathbb{R}^{N\times P\times D}$ of N time-varying variables.

D. Prediction Layer Design

To fuse spatial and temporal dependencies and infer prediction results, the prediction layer incorporates a standard multi-head cross-attention mechanism. Given the temporal representations $HT \in \mathbb{R}^{N\times P\times D}$ and spatial representations $HS \in \mathbb{R}^{N\times P\times D}$, the mechanism adaptively integrates them by treating HT as the query and HS as the key and value. To reinforce the criticality of temporal dependencies, we employ a residual connection with HT, generating the final fused spatio-temporal representations $HST \in \mathbb{R}^{N\times P\times D}$.

Followed by cross attention, the flatten operation and fully connected layer are leveraged in the prediction layer, thus generating prediction results based on the spatio-temporal representations $HST \in \mathbb{R}^{N\times P\times D}$.

During training, we employ the Mean Squared Error (MSE) loss with L2 regularization to optimize the parameters of PAANet. The MSE loss function is defined as follows:

$$\mathcal{L} = \frac{1}{Q \times N} \sum_{t_j=1}^{Q} \sum_{n=1}^{N} \left(x_n^{t_j} - \hat{q}_n^{t_j}\right)^2 + \frac{\lambda}{2}\|W\|^2, \tag{27}$$

where $x_n^{t_j} \in X$ and $\hat{q}_n^{t_j} \in \hat{Y}$ denote the observed and predicted values at timestep t_j for the n-th variable, respectively. W denotes the learnable parameters. λ represents as the regularization coefficient.

IV. Experiments

A. Experimental Setting

1) Dataset Description: We evaluate our model on two real-world datasets. **USHCN** [19] is a daily temperature dataset from 1114 U.S. meteorological stations (Jan 2000–Apr 2025) with a 77.98% missing rate. **Air** [20] comprises hourly $PM_{2.5}$ pollution readings from 1976 stations across 342 Chinese cities (Mar 2022–Jan 2025) with an 18.16% missing rate. Both datasets are Z-score normalized and chronologically split into training, validation, and test sets with a ratio of 7:1:2. To evaluate robustness, we inject Gaussian noise $\epsilon \sim \mathcal{N}(0, 0.1^2)$ into m random positions based on a noise rate v. Additionally, we construct a mixed noise scenario by combining the Gaussian noise with impulse noise, which replaces 5% of randomly selected values ($p = 0.05$) with extreme magnitudes of ± 2.0 (≈ 2 standard deviations of the normalized data).

2) Benchmarks for Comparison: PAANet is compared with representative baselines across five categories: 1) *Classical statistical approaches*: **ARIMA** [21] and **SVR** [22]; 2) *CNN-based approaches*: **MGSTC** [23]; 3) *GCN-based approaches*: **STGCN** [6]; 4) *Transformer-based approaches*: **CityCAN** [12] and **AirFormer** [20]; and 5) *Missing-data forecasting approaches*: **ODEs** [24].

3) Evaluation Metrics: Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE) are used to evaluate the performance of PAANet, where a lower metric indicates a higher prediction accuracy. In addition, we include training time (TT) and GPU memory (GPUM) usage of each model as metrics for evaluating training

TABLE I: Experimental results for different models on two real-world datasets under Gaussian noise and mixed noise.

Dataset	Model	Gaussian Noise						Mixed Noise						TT	GPUM
		1-6 horizon			7-12 horizon			1-6 horizon			7-12 horizon				
		MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE		
USHCN	ARIMA	30.29	60.58	34.75	33.16	63.42	36.32	33.15	63.92	37.28	36.42	66.75	39.15	0.56h	-
	SVR	29.83	59.66	33.82	32.71	62.45	35.48	32.57	62.89	36.45	35.92	65.68	38.32	0.64h	-
	MGSTC	20.80	41.62	24.60	23.65	44.92	26.21	23.45	44.98	27.30	26.89	48.35	29.15	2.23h	4.83G
	STGCN	16.55	33.10	19.52	19.35	36.85	21.35	19.23	36.45	22.18	22.68	40.23	24.10	3.27h	6.23G
	CityCAN	16.34	32.68	19.18	19.15	36.38	21.08	18.92	35.95	21.85	22.42	39.75	23.82	3.71h	6.73G
	AirFormer	16.15	32.33	18.89	18.95	35.95	20.81	18.68	35.62	21.54	22.18	39.23	23.45	4.04h	7.23G
	ODEs	15.12	26.71	14.96	18.22	29.52	16.53	17.85	29.92	17.68	21.54	32.85	19.21	4.87h	8.03G
	PAANet	**14.35**	**23.08**	**12.11**	**17.05**	**25.20**	**14.12**	**16.92**	**26.45**	**14.85**	**20.32**	**28.54**	**16.92**	4.92h	6.67G
Air	ARIMA	27.85	55.70	31.52	33.16	58.62	32.82	30.42	58.95	34.28	36.45	61.92	35.75	0.47h	-
	SVR	27.42	54.84	30.89	32.71	57.73	32.15	30.05	57.92	33.65	35.98	60.85	34.92	0.54h	-
	MGSTC	19.13	38.26	21.81	23.65	41.30	23.20	21.85	41.54	24.52	26.92	44.65	26.18	1.81h	4.21G
	STGCN	15.22	30.44	17.23	19.35	33.86	18.91	17.95	33.78	19.92	22.68	37.21	21.65	2.73h	5.71G
	CityCAN	15.03	30.06	16.90	19.15	33.38	18.64	17.72	33.35	19.58	22.42	36.75	21.32	3.23h	6.21G
	AirFormer	14.86	29.72	16.58	18.95	32.96	18.33	17.50	33.05	19.23	22.21	36.32	21.05	3.47h	6.71G
	ODEs	12.75	23.85	13.50	16.22	26.42	14.50	15.42	27.18	16.23	19.55	29.78	17.23	4.23h	7.43G
	PAANet	**12.11**	**20.15**	**10.42**	**15.12**	**23.75**	**12.39**	**14.78**	**23.49**	**13.15**	**18.45**	**27.09**	**15.12**	4.48h	6.29G

efficiency. For these two metrics, lower values indicate better computational efficiency.

4) Implementation details: Unless otherwise specified, we set $Q = 24$, $L = 108$, $L_p = 12$, and $P = 9$. PAANet is implemented in PyTorch and optimized with SGD using a batch size of 32 and a learning rate of 0.001. Besides, the hyperparameters of PAANet consist of the number of convolution filters D, the layer of standard attention mechanism L_{SAM}, the layer of attention in GGAtt L_{GGA}, the layer of cross attention L_{CA}, the number of attention heads B, and the dimension of each head d_e. We fine-tune these hyperparameters on the validation set. After repeated trials, it is observed that PAANet achieves the highest prediction accuracy when $D = 16$, $L_{SAM} = 1$, $L_{GGA} = 1$, $L_{CA} = 1$, $B = 2$, and $d_e = 8$.

B. Model Comparison

The prediction accuracy of PAANet and benchmarks for 6, and 12 horizons ahead prediction on USHCN and Air datasets are compared in Table I. We can find that PAANet is superior to all benchmarks on MAE, and RMSE metrics. Compared with the second-best benchmark (*e.g.*, ODEs), PAANet significantly reduces MAE, and RMSE on 6-, and 12-horizon future prediction for USHCN and Air datasets.

From Table I, we draw three main conclusions. First, deep learning models consistently outperform classical statistical methods (e.g., ARIMA and SVR), reflecting their stronger representation ability. Second, GCN-based and Transformer-based methods generally surpass CNN-based methods, likely due to their more effective modeling of spatial dependencies. Third, PAANet achieves the best overall performance, outperforming ODEs under both Gaussian and mixed noise settings, which highlights its robustness to noisy observations. In addition, PAANet maintains competitive training time and memory usage, demonstrating a good trade-off between accuracy and efficiency.

TABLE II: The results of ablation study.

Model	USHCN			Air		
	MAE	RMSE	MAPE	MAE	RMSE	MAPE
Basic	30.29	59.98	34.75	25.85	51.70	29.99
+TFM	21.15	36.38	21.08	19.95	32.96	18.33
+SFM	18.50	27.71	15.96	17.86	29.72	16.58
PAANet	**14.35**	**23.08**	**12.11**	**12.11**	**20.15**	**10.42**
-AW TCN	17.02	26.41	14.64	16.02	26.41	14.64
-WTDE	15.32	25.46	13.98	14.63	24.60	12.98
-MAGI	16.50	24.00	14.30	12.50	24.80	14.20
-GGAtt	15.00	25.46	13.98	14.00	24.46	12.98

C. Ablation Study

We conduct ablations under mixed noise by progressively adding TFM, SFM, and cross attention, and by removing AW TCN, WTDE, MAGI, or GGAtt from the full PAANet model.

For fairness, PAANet and its variants share the same parameter settings except for the specified components. The results in Table II show that +TFM significantly improves prediction accuracy by capturing local and global temporal dependencies, while +SFM further enhances performance by incorporating spatial information. The full PAANet achieves the best results, demonstrating the effectiveness of adaptively fusing spatial and temporal features.

Removing key components leads to consistent performance degradation: -AW TCN weakens robustness to missing data, -WTDE reduces the ability to suppress high-frequency noise, -MAGI impairs spatial representation under incomplete observations, and -GGAtt degrades noise-resistant spatial dependency modeling. These results confirm the necessity of each module in PAANet.

D. Hyperparameter Analysis

The average accuracy of PAANet under various hyperparameter values over the future 12 horizons forecasting on the USHCN dataset is illustrated in Figure 3. When one hyperparameter is changed, the remaining hyperparameters

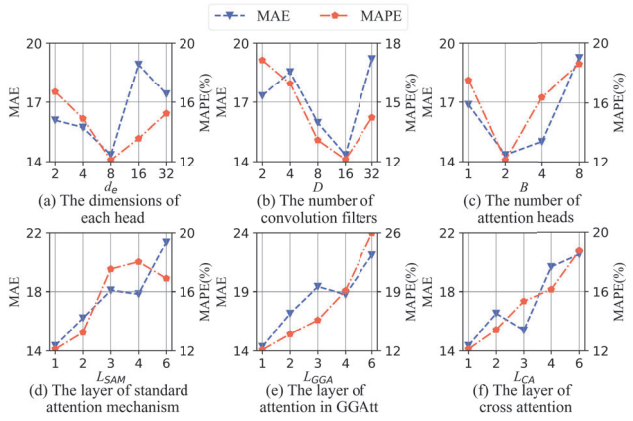

Fig. 3: Experimental results under different hyperparameter settings on USHCN dataset.

are kept unchanged. Bayesian optimization [25] is used to acquire optimal hyperparameters. As displayed in Figures 3. (a), (b), and (c), we observe that smaller models are more prone to overfitting, while larger models are more likely to underfit. On the contrary, Figures 3. (d), (e), and (f) reveal that fewer layers attain higher prediction accuracy. The main reason is that more layers lead to error accumulation.

V. Conclusion

We proposed PAANet for spatio-temporal prediction on missing and noise data. The TFM module combines patch alignment, multi-scale encoding, and wavelet decomposition to learn robust temporal dependencies, while the SFM module imputes incomplete node features and filters noisy graph interactions. Experiments on USHCN and Air demonstrate consistent improvements over strong baselines. Future work will investigate more adaptive and efficient extensions of the framework.

VI. Acknowledgment

This work was supported in part by the National Natural Science Foundation of China under Grant 62572171; in part by the 16th Batch of Science and Technology Development Programs (Innovation Consortium Projects) of Suzhou under Grant LHT202404; in part by the Technology Project of Huaneng Group under Grant HNKJ24-H167; and in part by the High Performance Computing Platform, Hohai University.

References

[1] H. Miao, Y. Zhao, C. Guo, B. Yang, K. Zheng, and C. S. Jensen, "Spatio-temporal prediction on streaming data: A unified federated continuous learning framework," *IEEE Transactions on Knowledge and Data Engineering*, 2025.

[2] G. Zou, Z. Lai, T. Wang, Z. Liu, and Y. Li, "Mt-stnet: A novel multi-task spatiotemporal network for highway traffic flow prediction," *IEEE Transactions on Intelligent Transportation Systems*, 2024.

[3] R. Kumar, M. Bhanu, J. Mendes-Moreira, and J. Chandra, "Spatio-temporal predictive modeling techniques for different domains: a survey," *ACM Computing Surveys*, vol. 57, no. 2, pp. 1–42, 2024.

[4] H. Chen and H. Eldardiry, "Graph time-series modeling in deep learning: a survey," *ACM Transactions on Knowledge Discovery from Data*, vol. 18, no. 5, pp. 1–35, 2024.

[5] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *International Conference on Learning Representations*, 2018.

[6] B. Yu, H. Yin, and 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.

[7] P. Kidger, J. Morrill, J. Foster, and T. Lyons, "Neural controlled differential equations for irregular time series," *Advances in neural information processing systems*, vol. 33, pp. 6696–6707, 2020.

[8] A. Pareja, G. Domeniconi, J. Chen, T. Ma, T. Suzumura, H. Kanezashi, T. Kaler, T. Schardl, and C. Leiserson, "Evolvegcn: Evolving graph convolutional networks for dynamic graphs," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 04, 2020, pp. 5363–5370.

[9] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, "Convolutional sequence to sequence learning," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 2017, pp. 1243–1252.

[10] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[11] A. Ali, I. Ullah, S. Ahmad, Z. Wu, J. Li, and X. Bai, "An attention-driven spatio-temporal deep hybrid neural networks for traffic flow prediction in transportation systems," *IEEE Transactions on Intelligent Transportation Systems*, 2025.

[12] C. Wang, Y. Liang, and G. Tan, "Citycan: Causal attention network for citywide spatio-temporal forecasting," in *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, 2024, pp. 702–711.

[13] S. M. Kazemi, R. Goel, S. Eghbali, J. Ramanan, J. Sahota, S. Thakur, S. Wu, C. Smyth, P. Poupart, and M. Brubaker, "Time2vec: Learning a vector representation of time," 2019. [Online]. Available: https://arxiv.org/abs/1907.05321

[14] S. Mallat, "A theory for multiresolution signal decomposition: the wavelet representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 7, pp. 674–693, 1989.

[15] C. Zheng, X. Fan, C. Wang, and J. Qi, "Gman: A graph multi-attention network for traffic prediction," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 01, 2020, pp. 1234–1241.

[16] X. Shi, H. Qi, Y. Shen, G. Wu, and B. Yin, "A spatial–temporal attention approach for traffic prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 8, pp. 4909–4918, 2021.

[17] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio *et al.*, "Graph attention networks," *stat*, vol. 1050, no. 20, pp. 10–48 550, 2017.

[18] E. Rossi, H. Kenlay, M. I. Gorinova, B. P. Chamberlain, X. Dong, and M. M. Bronstein, "On the unreasonable effectiveness of feature propagation in learning on graphs with missing node features," in *Learning on graphs conference*. PMLR, 2022, pp. 11–1.

[19] M. J. Menne and C. N. Williams Jr, "Homogenization of temperature series via pairwise comparisons," *Journal of Climate*, vol. 22, no. 7, pp. 1700–1717, 2009.

[20] Y. Liang, Y. Xia, S. Ke, Y. Wang, Q. Wen, J. Zhang, Y. Zheng, and R. Zimmermann, "Airformer: Predicting nationwide air quality in china with transformers," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 37, 2023, pp. 14 329–14 337.

[21] P. Duan, G. Mao, W. Liang, and D. Zhang, "A unified spatio-temporal model for short-term traffic flow prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 9, pp. 3212–3223, 2018.

[22] W. Xu, H. Binbin, Y. Xiao, Cirenluobu, KanAike, and L. Jinji, "A spatio-temporal series simulation and prediction method of geography based on svr-ca model," in *Proceedings of the 2nd International Conference on Intelligent Information Processing*, 2017, pp. 1–7.

[23] C. Chen, K. Li, S. G. Teo, X. Zou, K. Li, and Z. Zeng, "Citywide traffic flow prediction based on multiple gated spatio-temporal convolutional neural networks," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 14, no. 4, pp. 1–23, 2020.

[24] M. Han and Q. Wang, "Adaptive graph convolution neural differential equation for spatio-temporal time series prediction," *IEEE Transactions on Knowledge and Data Engineering*, 2024.

[25] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. De Freitas, "Taking the human out of the loop: A review of bayesian optimization," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, 2016.