Cross Space and Time: A Spatio-Temporal Unitized Model for Traffic Flow Forecasting

Weilin Ruan, Wenzhuo Wang, Siru Zhong, Wei Chen, Li Liu, and Yuxuan Liang

香港科技大学（广州）
THE HONG KONG
UNIVERSITY OF SCIENCE AND
TECHNOLOGY (GUANGZHOU)

Table of Content

01 Background and Motivation

02 Related work

03 Methodology

04 Experiments

05 Conclusion and Future work

THE HONG KONG
UNIVERSITY OF SCIENCE AND
TECHNOLOGY (GUANGZHOU)

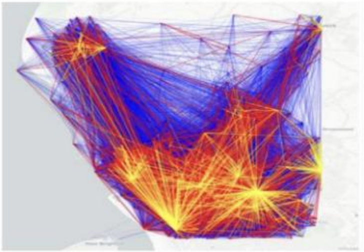
Background - Research Context

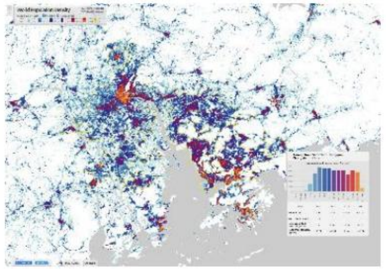
Spatio-temporal Forecasting

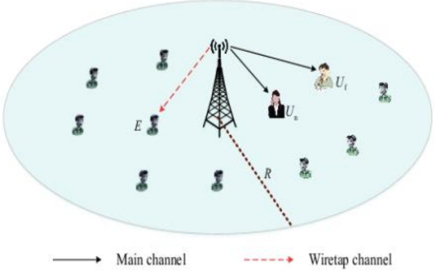
Given spatio-temporal graph $\mathcal{G}$ and input tensor X, we aim to learn function $\mathcal{F}$:

$$[X^{(t-s+1)}, \ldots, X^{(t)}; \mathcal{G}] \xrightarrow{\mathcal{F}(\cdot)} [X^{(t+1)}, \ldots, X^{(t+h)}].$$

- Urban growth → severe traffic congestion
- ITS enables data-driven traffic management
- Flow prediction is key for safety & efficiency





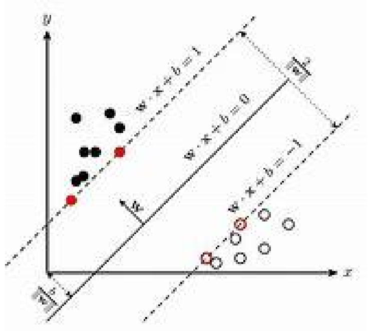


Background - Existing Methods & Limitations

$$y_t = C + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \dots + \phi_p y_{t-p} + \varepsilon_t$$

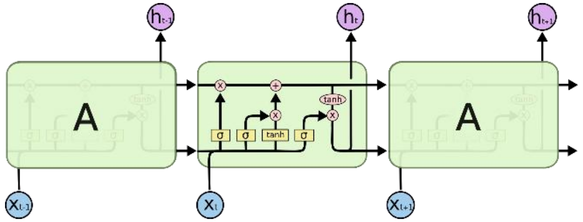
1ST TERM

Prediction using traditional **statistical** methods such as VAR, HI, and ARIMA, but ignoring spatial dependence.



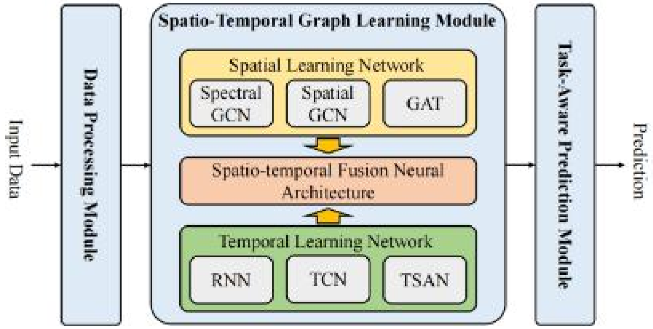
Using **machine learning** models such as SVMs to capture features, but it is difficult to deal with complex nonlinear relationships.

2ND TERM



3RD TERM

Deep learning models such as CNNs and LSTMs used to capture spatio-temporal dependencies a decade ago.



Input Data

Data Processing Module

Spatio-Temporal Graph Learning Module

Spatial Learning Network

Spectral GCN

Spatial GCN

GAT

Spatio-temporal Fusion Neural Architecture

Temporal Learning Network

RNN

TCN

TSAN

Task-Aware Prediction Module

Prediction

After 2017, **STGNNs** model performance continues to break through and combine with cutting-edge technologies such as big language modeling.

4TH TERM

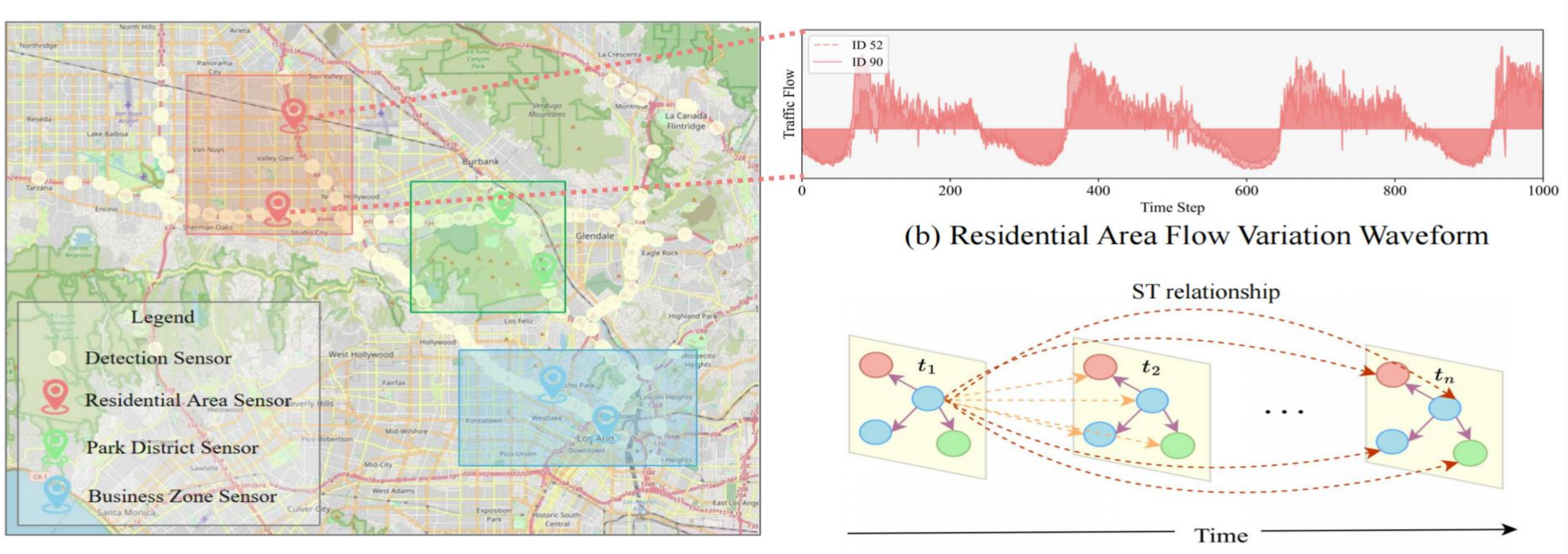
Motivation

Challenges

1. Space–time separation
2. Data heterogeneity

Need for

1. Unified spatial–temporal learning
2. Efficient heterogeneity modeling



(b) Residential Area Flow Variation Waveform

(a) Sensor Distribution Map

(c) Spatiotemporal Dependencies

Related Work

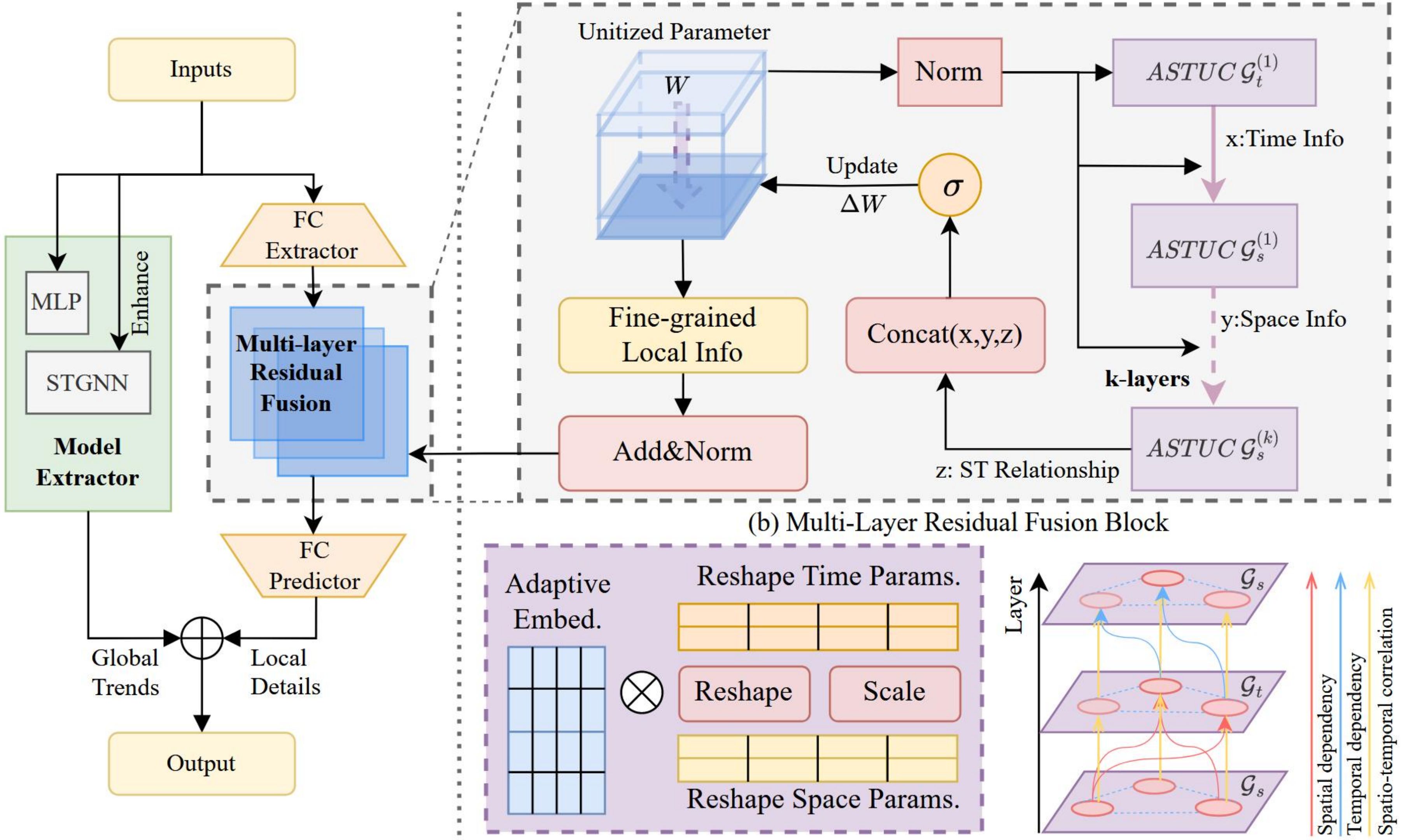
Spatio-temporal Forecasting

Spatio-temporal forecasting, aiming to predict future states from historical data, has evolved from traditional statistical/time-series methods (limited in capturing complex spatial - temporal relationships) to deep learning.

Low-rank Matrix Factorization

Low-rank matrix factorization addresses computational challenges by decomposing high - dimensional matrices into low - rank products, reducing complexity while preserving info. Applied in deep learning for network compression (via tensor networks or single - layer factorization, e.g., Tucker tensor layers), it now extends to graph data. For GNNs handling spatio-temporal data, it uncovers hidden info, enhancing performance, as seen in traffic forecasting to capture dependencies and cut computation.

Overview of STUM



(a) Spatio-temporal Unitized Model

(b) Multi-Layer Residual Fusion Block

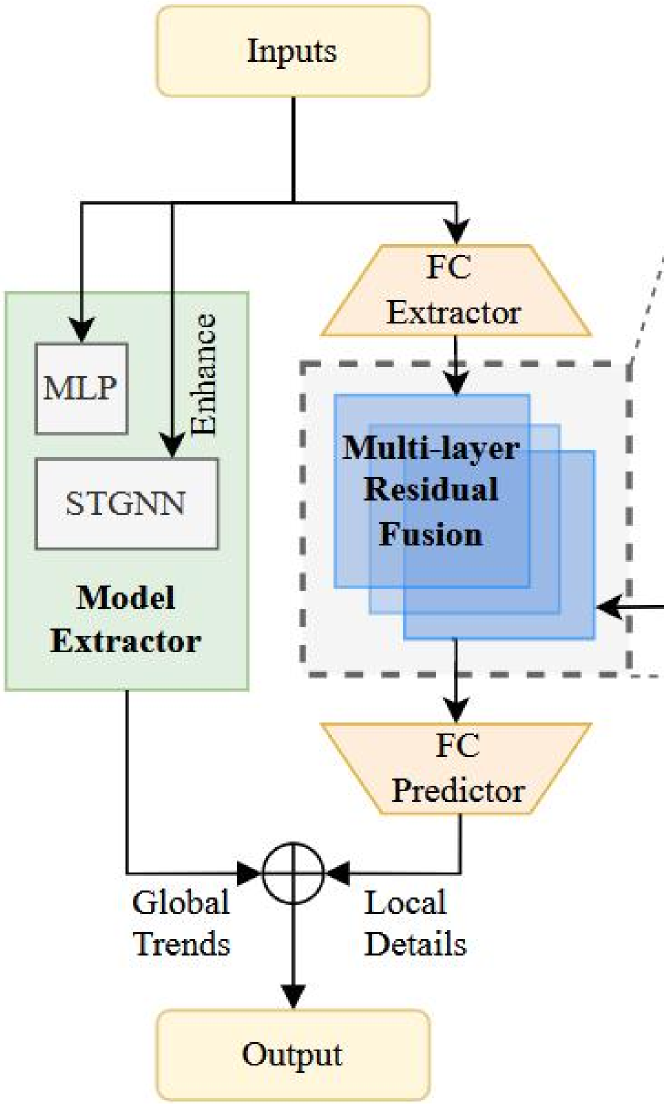
(c) Adative Spatio-temporal Unitized Cell

Overall Design

- **Backbone Extractor: Global spatio-temporal features.**

- **Multi - Layer Residual Fusion Blocks (MLRF): Fuses encoded info. with a predictor.**

Methodology - Global Enhancement and Local Refinement



Problem

Existing spatio-temporal models fail to balance global pattern capture and local detail refinement. They also lack flexibility to integrate with pre-trained architectures.

Key Idea

Global Enhancement: Backbone extractor captures large-scale spatio-temporal dependencies, providing a universal structure to integrate with other models.

Local Refinement: Modular design (plus fully-connected components) fine-tunes local features. Supports plug-and-play adaptation (e.g., MLP, CNN, or pre-trained baselines).

Outcome

Balances global context + local details → More robust to noise, more accurate predictions.

Methodology - Dual Feature Extraction of Spatio-temporal Data

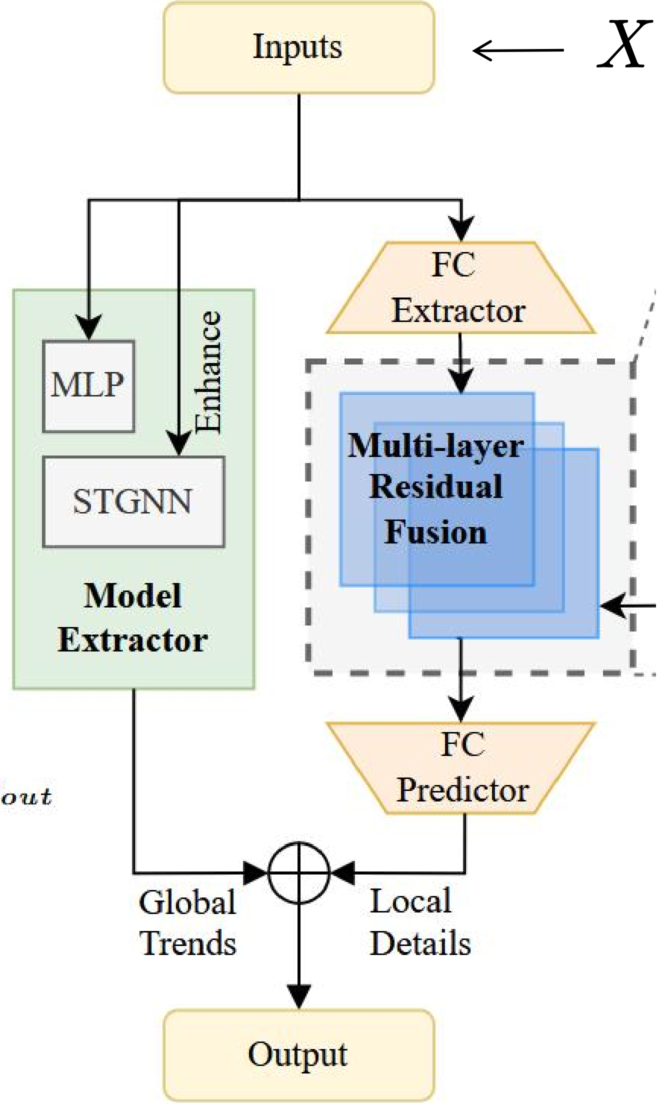
$$\longleftarrow \quad X = [X_1, X_2, \cdots, X_s]$$

$\mathcal{F}_b$

backbone network

$$z_b = \mathcal{F}_b(X) = [f_1, ..., f_h] \in \mathbb{R}^{n \times c_{out}}$$

prediction - ready features



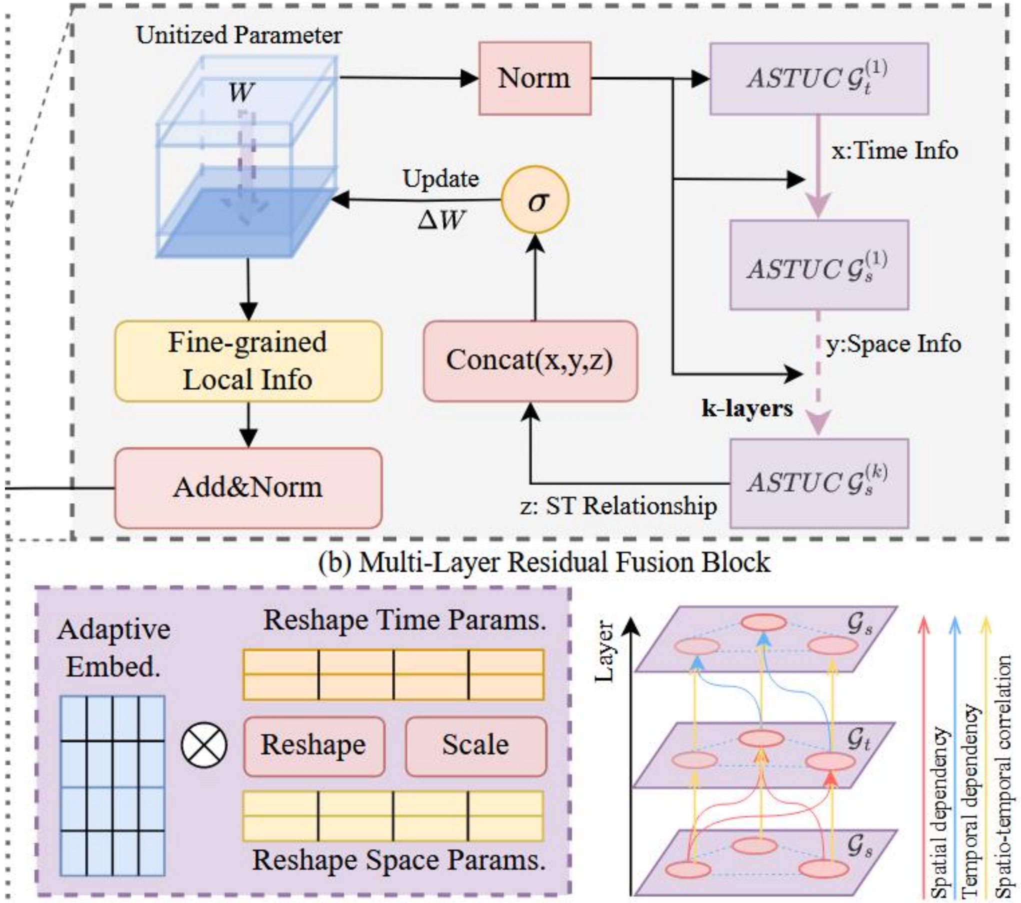
$\mathcal{F}_c$

adaptive low-rank linear layer

$$X' = \mathcal{F}_c(X) = [w_1, ..., w_h] \in \mathbb{R}^{n \times m}$$

hidden layer dimensions

Methodology - Unified Representations



(b) Multi-Layer Residual Fusion Block

(c) Adative Spatio-temporal Unitized Cell

What is ASTUC?

Adaptive Spatio-Temporal Unitized Cell: A core component unifying spatial + temporal info.

How It Works

Leverages low-rank matrix decomposition → Captures complex dependencies with fewer parameters.→ Adapts to scenarios without blowing up computation, handles heterogeneity better.

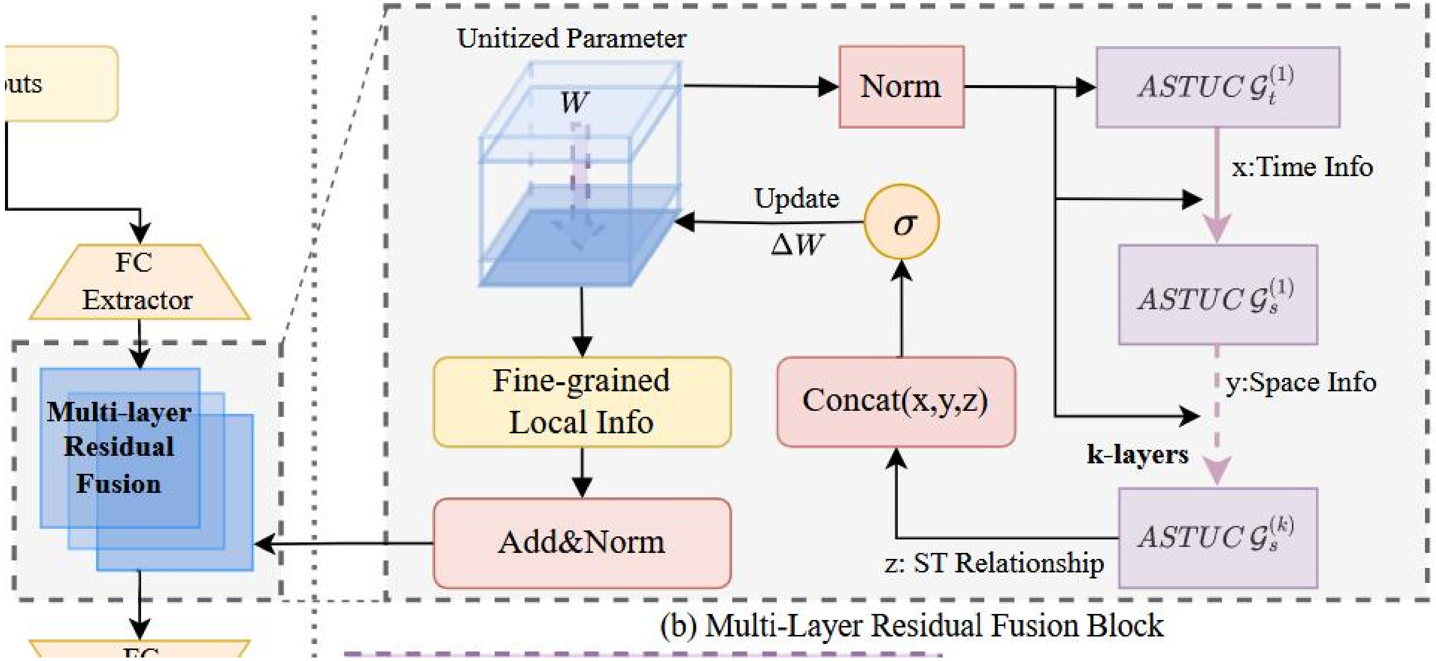
Key Idea

Stores/updates space - time info in a unified parameter matrix (via low-rank matrices).Iteratively processes temporal ($\mathcal{G}_t^{(i)}$) and spatial($\mathcal{G}_s^{(i)}$) info, fusing them with residual structures.

Formulation

$$\mathcal{G}_t^{(i)} = \text{Update}(X_{:t}, \mathcal{G}_s^{(i-1)}; W, b)$$
$$\mathcal{G}_s^{(i)} = \text{Update}(X_{:t}, \mathcal{G}_t^{(i-1)}; W, b)$$
$$W \leftarrow \Delta W = \text{Memory}(\mathcal{G}_t^{(i)} \oplus \mathcal{G}_s^{(i)}, b)$$

Methodology - Training and Optimization of STUM Framework



(b) Multi-Layer Residual Fusion Block

Novelty

Global Enhancement: MLRF + ASTUCs eliminate "temporal-spatial silos" in deep models.

Efficiency Boost: Low-rank decomposition cuts params $\mathcal{O}(N^2d) \rightarrow \mathcal{O}(Nr + Tr)$

Modular Flexibility: Plug-and-play design adapts to pre-trained models/tasks (swap backbones, reuse MLRF).

Problem

Single ASTUCs can't handle complex spatio-temporal heterogeneity.

Key Idea

MLRF alternates spatial/temporal info transmission across ASTUCs. Integrate normalization, low-rank decomposition, and adaptive parameter updates.

Formulation

Low-Rank Matrix Adaptation:

$$W^{(i)} = Norm(X) = X \cdot \frac{W^{(i-1)}}{\frac{1}{d}\sum_{i=1}^{d} x_i^2 + \epsilon}$$

$$\hat{h_i} = \mathcal{G}_t^{(l)} \cdot \mathcal{G}_s^{(l)}(\sigma(...\mathcal{G}_t^{(1)} \cdot \mathcal{G}_s^{(1)}(W^{(i)})...))$$

Gated Prediction Fusion:

$$z_t = \mathrm{FC}(\sigma(\hat{h_i}))$$

$$Z = \mathcal{H}(z_b, z_t; X_{:t}, \alpha) = (1-\alpha) \odot \mathcal{F}_b(X_{:t}) + \alpha \odot z_t$$

End-to-End Training:

$$\nabla_\alpha \mathcal{L}_{\text{train}} = \nabla_Z \mathcal{L}_{\text{train}} \cdot \nabla_\alpha Z$$

Experimental Setup

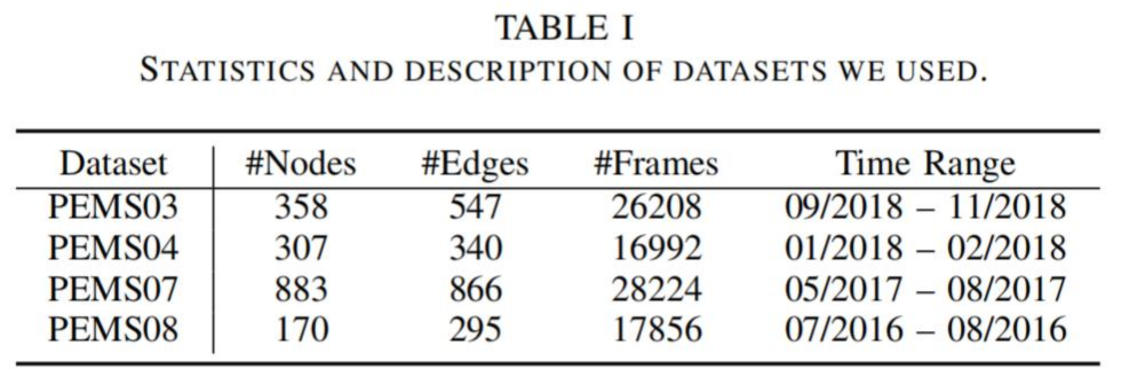
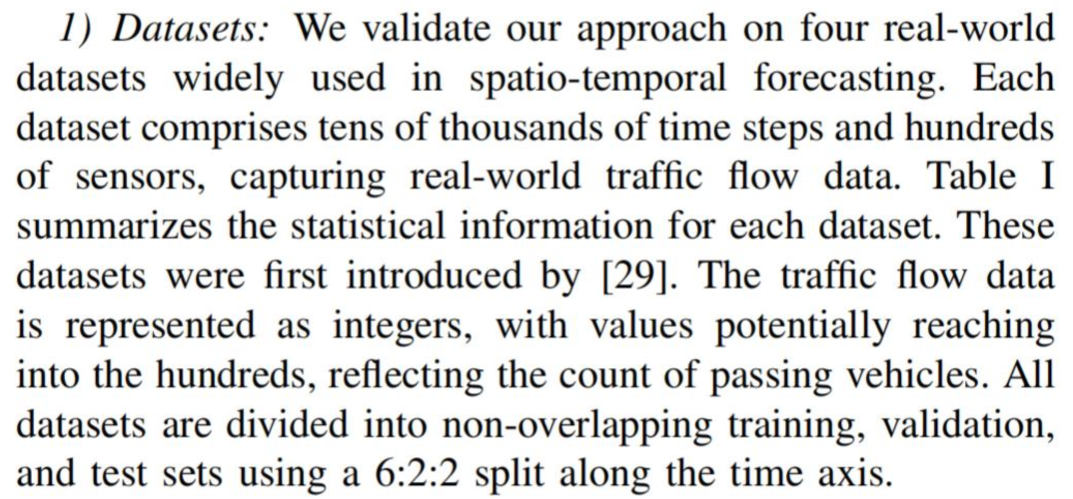
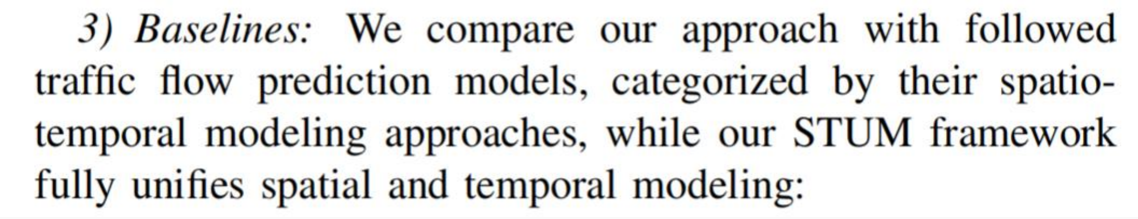


TABLE I
STATISTICS AND DESCRIPTION OF DATASETS WE USED.

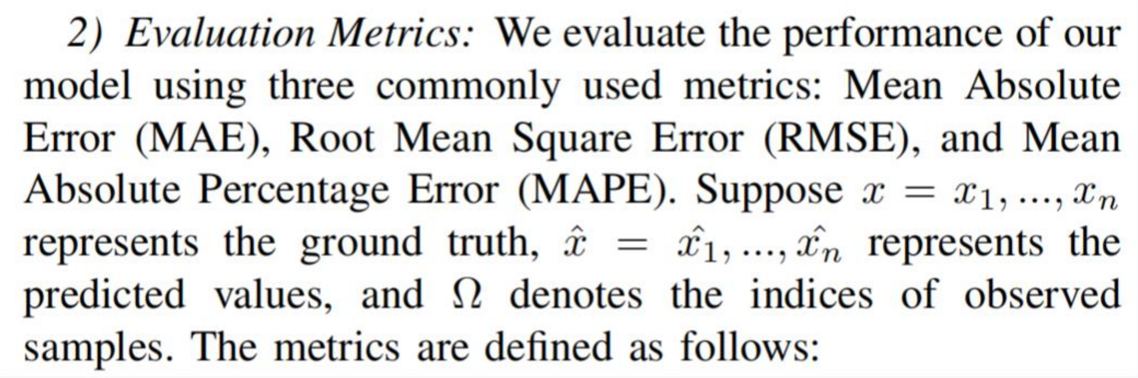
Dataset	#Nodes	#Edges	#Frames	Time Range
PEMS03	358	547	26208	09/2018 – 11/2018
PEMS04	307	340	16992	01/2018 – 02/2018
PEMS07	883	866	28224	05/2017 – 08/2017
PEMS08	170	295	17856	07/2016 – 08/2016



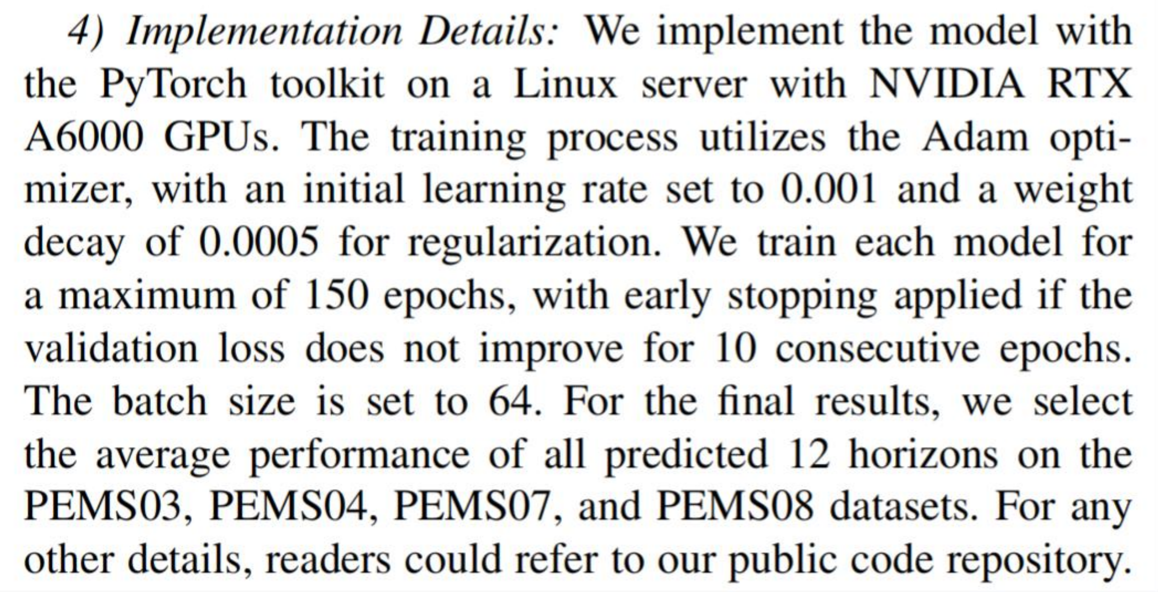
1) Datasets: We validate our approach on four real-world datasets widely used in spatio-temporal forecasting. Each dataset comprises tens of thousands of time steps and hundreds of sensors, capturing real-world traffic flow data. Table I summarizes the statistical information for each dataset. These datasets were first introduced by [29]. The traffic flow data is represented as integers, with values potentially reaching into the hundreds, reflecting the count of passing vehicles. All datasets are divided into non-overlapping training, validation, and test sets using a 6:2:2 split along the time axis.



3) Baselines: We compare our approach with followed traffic flow prediction models, categorized by their spatio-temporal modeling approaches, while our STUM framework fully unifies spatial and temporal modeling:



2) Evaluation Metrics: We evaluate the performance of our model using three commonly used metrics: Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and Mean Absolute Percentage Error (MAPE). Suppose $x = x_1, ..., x_n$ represents the ground truth, $\hat{x} = \hat{x_1}, ..., \hat{x_n}$ represents the predicted values, and Ω denotes the indices of observed samples. The metrics are defined as follows:



4) Implementation Details: We implement the model with the PyTorch toolkit on a Linux server with NVIDIA RTX A6000 GPUs. The training process utilizes the Adam optimizer, with an initial learning rate set to 0.001 and a weight decay of 0.0005 for regularization. We train each model for a maximum of 150 epochs, with early stopping applied if the validation loss does not improve for 10 consecutive epochs. The batch size is set to 64. For the final results, we select the average performance of all predicted 12 horizons on the PEMS03, PEMS04, PEMS07, and PEMS08 datasets. For any other details, readers could refer to our public code repository.

Performance Comparisons(RQ1 and RQ2)

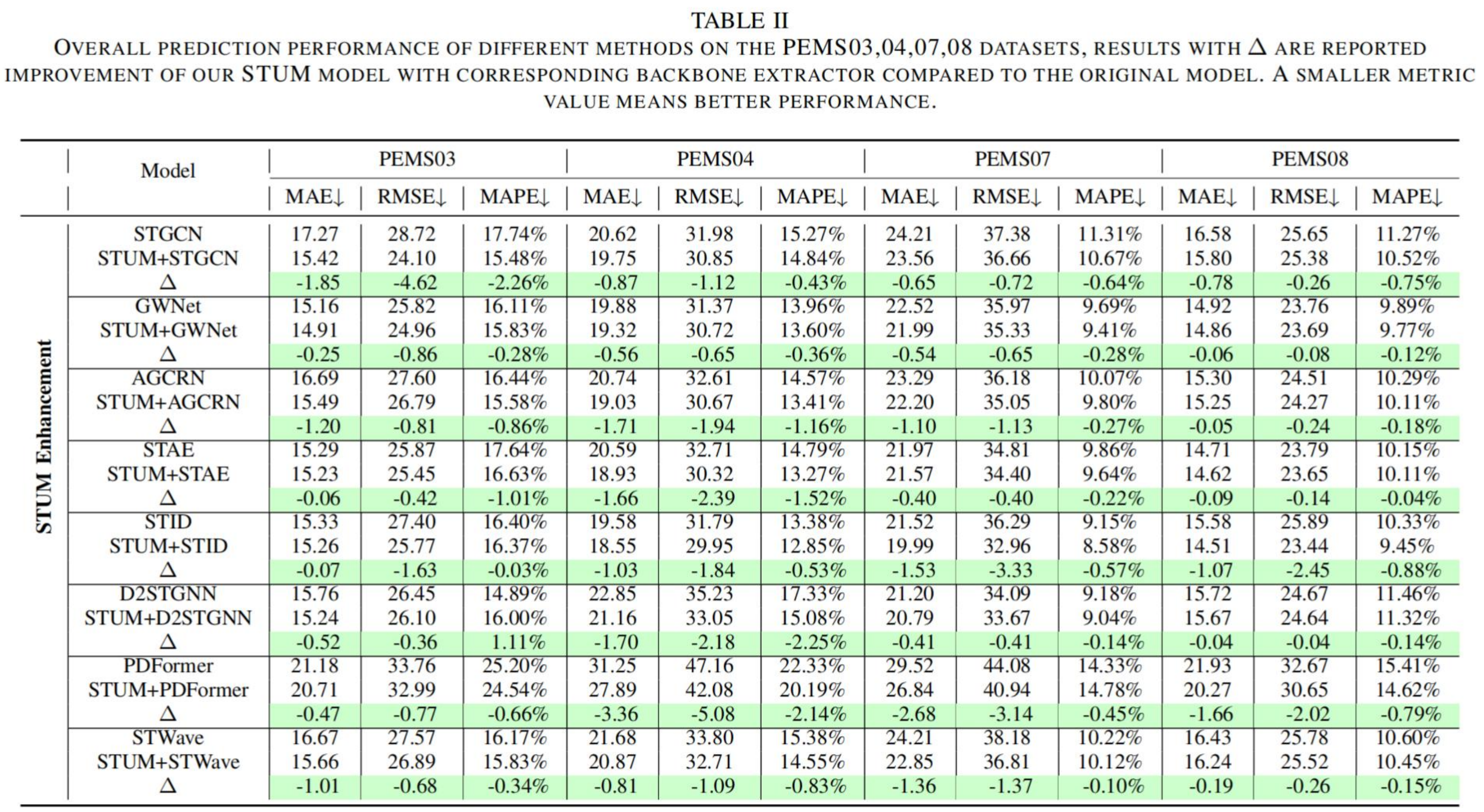


TABLE II
OVERALL PREDICTION PERFORMANCE OF DIFFERENT METHODS ON THE PEMS03,04,07,08 DATASETS, RESULTS WITH Δ ARE REPORTED IMPROVEMENT OF OUR STUM MODEL WITH CORRESPONDING BACKBONE EXTRACTOR COMPARED TO THE ORIGINAL MODEL. A SMALLER METRIC VALUE MEANS BETTER PERFORMANCE.

	Model	PEMS03			PEMS04			PEMS07			PEMS08		
		MAE↓	RMSE↓	MAPE↓	MAE↓	RMSE↓	MAPE↓	MAE↓	RMSE↓	MAPE↓	MAE↓	RMSE↓	MAPE↓
STUM Enhancement	STGCN	17.27	28.72	17.74%	20.62	31.98	15.27%	24.21	37.38	11.31%	16.58	25.65	11.27%
	STUM+STGCN	15.42	24.10	15.48%	19.75	30.85	14.84%	23.56	36.66	10.67%	15.80	25.38	10.52%
	Δ	-1.85	-4.62	-2.26%	-0.87	-1.12	-0.43%	-0.65	-0.72	-0.64%	-0.78	-0.26	-0.75%
	GWNet	15.16	25.82	16.11%	19.88	31.37	13.96%	22.52	35.97	9.69%	14.92	23.76	9.89%
	STUM+GWNet	14.91	24.96	15.83%	19.32	30.72	13.60%	21.99	35.33	9.41%	14.86	23.69	9.77%
	Δ	-0.25	-0.86	-0.28%	-0.56	-0.65	-0.36%	-0.54	-0.65	-0.28%	-0.06	-0.08	-0.12%
	AGCRN	16.69	27.60	16.44%	20.74	32.61	14.57%	23.29	36.18	10.07%	15.30	24.51	10.29%
	STUM+AGCRN	15.49	26.79	15.58%	19.03	30.67	13.41%	22.20	35.05	9.80%	15.25	24.27	10.11%
	Δ	-1.20	-0.81	-0.86%	-1.71	-1.94	-1.16%	-1.10	-1.13	-0.27%	-0.05	-0.24	-0.18%
	STAE	15.29	25.87	17.64%	20.59	32.71	14.79%	21.97	34.81	9.86%	14.71	23.79	10.15%
	STUM+STAE	15.23	25.45	16.63%	18.93	30.32	13.27%	21.57	34.40	9.64%	14.62	23.65	10.11%
	Δ	-0.06	-0.42	-1.01%	-1.66	-2.39	-1.52%	-0.40	-0.40	-0.22%	-0.09	-0.14	-0.04%
	STID	15.33	27.40	16.40%	19.58	31.79	13.38%	21.52	36.29	9.15%	15.58	25.89	10.33%
	STUM+STID	15.26	25.77	16.37%	18.55	29.95	12.85%	19.99	32.96	8.58%	14.51	23.44	9.45%
	Δ	-0.07	-1.63	-0.03%	-1.03	-1.84	-0.53%	-1.53	-3.33	-0.57%	-1.07	-2.45	-0.88%
	D2STGNN	15.76	26.45	14.89%	22.85	35.23	17.33%	21.20	34.09	9.18%	15.72	24.67	11.46%
	STUM+D2STGNN	15.24	26.10	16.00%	21.16	33.05	15.08%	20.79	33.67	9.04%	15.67	24.64	11.32%
	Δ	-0.52	-0.36	1.11%	-1.70	-2.18	-2.25%	-0.41	-0.41	-0.14%	-0.04	-0.04	-0.14%
	PDFormer	21.18	33.76	25.20%	31.25	47.16	22.33%	29.52	44.08	14.33%	21.93	32.67	15.41%
	STUM+PDFormer	20.71	32.99	24.54%	27.89	42.08	20.19%	26.84	40.94	14.78%	20.27	30.65	14.62%
	Δ	-0.47	-0.77	-0.66%	-3.36	-5.08	-2.14%	-2.68	-3.14	-0.45%	-1.66	-2.02	-0.79%
	STWave	16.67	27.57	16.17%	21.68	33.80	15.38%	24.21	38.18	10.22%	16.43	25.78	10.60%
	STUM+STWave	15.66	26.89	15.83%	20.87	32.71	14.55%	22.85	36.81	10.12%	16.24	25.52	10.45%
	Δ	-1.01	-0.68	-0.34%	-0.81	-1.09	-0.83%	-1.36	-1.37	-0.10%	-0.19	-0.26	-0.15%

STUM enhances diverse spatio-temporal baselines (STGCN, GWN, etc.) as a backbone—all outperform originals on datasets (STGCN +19.17% top gain). Even standalone on PEMS data, it beats baselines in short/long-term tasks, proving strong pattern capture for forecasting.

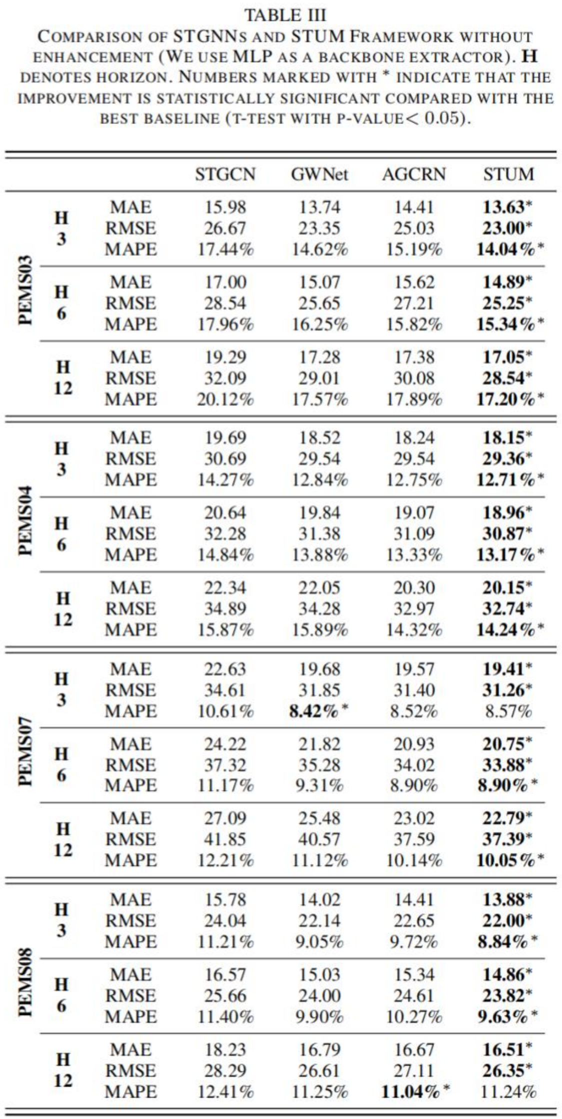


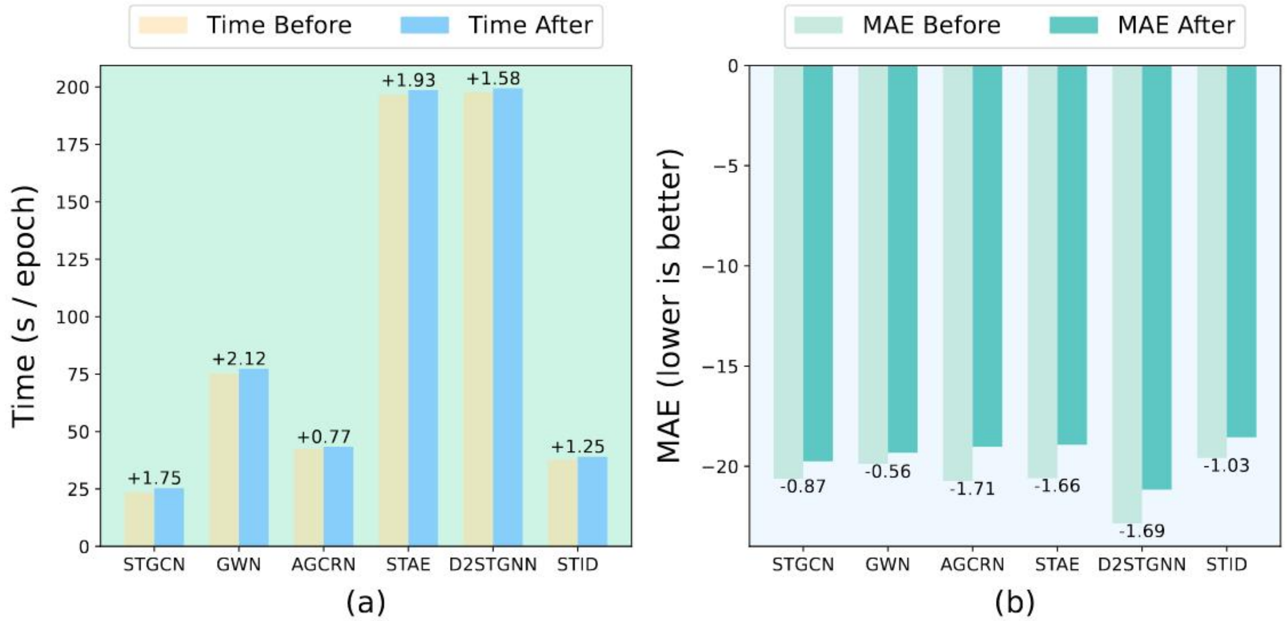
TABLE III
COMPARISON OF STGNNS AND STUM FRAMEWORK WITHOUT ENHANCEMENT (WE USE MLP AS A BACKBONE EXTRACTOR). **H** DENOTES HORIZON. NUMBERS MARKED WITH * INDICATE THAT THE IMPROVEMENT IS STATISTICALLY SIGNIFICANT COMPARED WITH THE BEST BASELINE (T-TEST WITH P-VALUE< 0.05).

			STGCN	GWNet	AGCRN	STUM
PEMS03	**H 3**	MAE	15.98	13.74	14.41	**13.63***
		RMSE	26.67	23.35	25.03	**23.00***
		MAPE	17.44%	14.62%	15.19%	**14.04%***
	H 6	MAE	17.00	15.07	15.62	**14.89***
		RMSE	28.54	25.65	27.21	**25.25***
		MAPE	17.96%	16.25%	15.82%	**15.34%***
	H 12	MAE	19.29	17.28	17.38	**17.05***
		RMSE	32.09	29.01	30.08	**28.54***
		MAPE	20.12%	17.57%	17.89%	**17.20%***
PEMS04	**H 3**	MAE	19.69	18.52	18.24	**18.15***
		RMSE	30.69	29.54	29.54	**29.36***
		MAPE	14.27%	12.84%	12.75%	**12.71%***
	H 6	MAE	20.64	19.84	19.07	**18.96***
		RMSE	32.28	31.38	31.09	**30.87***
		MAPE	14.84%	13.88%	13.33%	**13.17%***
	H 12	MAE	22.34	22.05	20.30	**20.15***
		RMSE	34.89	34.28	32.97	**32.74***
		MAPE	15.87%	15.89%	14.32%	**14.24%***
PEMS07	**H 3**	MAE	22.63	19.68	19.57	**19.41***
		RMSE	34.61	31.85	31.40	**31.26***
		MAPE	10.61%	**8.42%***	8.52%	8.57%
	H 6	MAE	24.22	21.82	20.93	**20.75***
		RMSE	37.32	35.28	34.02	**33.88***
		MAPE	11.17%	9.31%	8.90%	**8.90%***
	H 12	MAE	27.09	25.48	23.02	**22.79***
		RMSE	41.85	40.57	37.59	**37.39***
		MAPE	12.21%	11.12%	10.14%	**10.05%***
PEMS08	**H 3**	MAE	15.78	14.02	14.41	**13.88***
		RMSE	24.04	22.14	22.65	**22.00***
		MAPE	11.21%	9.05%	9.72%	**8.84%***
	H 6	MAE	16.57	15.03	15.34	**14.86***
		RMSE	25.66	24.00	24.61	**23.82***
		MAPE	11.40%	9.90%	10.27%	**9.63%***
	H 12	MAE	18.23	16.79	16.67	**16.51***
		RMSE	28.29	26.61	27.11	**26.35***
		MAPE	12.41%	11.25%	**11.04%***	11.24%

Efficiency Analysis (RQ3)

MLRF ← 4

ASTUC ← 8(Q_s&Q_t)



(a)

(b)

✓ It is observed that even when multiple ASTUCs are used, the low - rank adaptive part of the framework can keep the training time stable. These improvements are achieved with minimal additional training time, highlighting the efficiency of the framework in balancing accuracy and computational cost.

Ablation Study (RQ4)

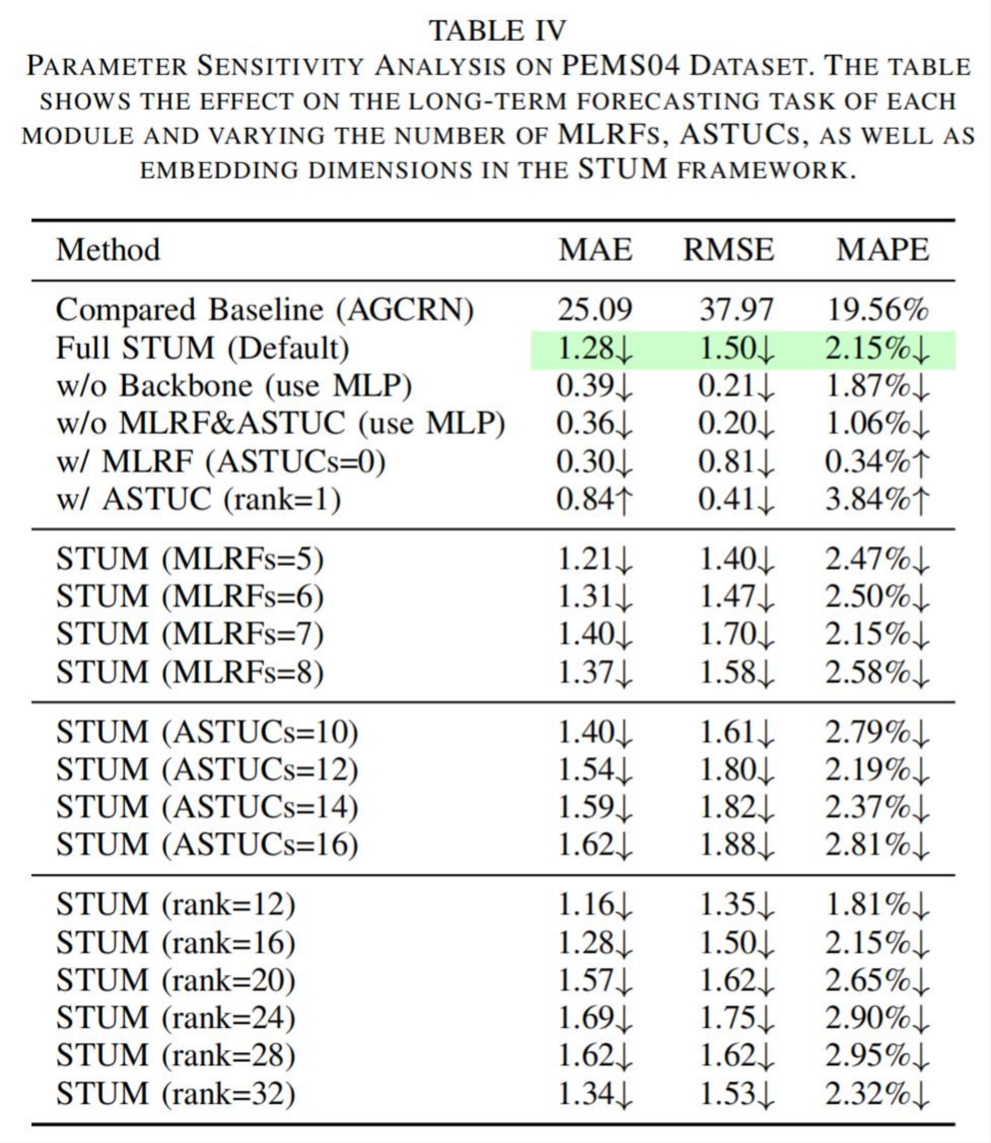


TABLE IV

PARAMETER SENSITIVITY ANALYSIS ON PEMS04 DATASET. THE TABLE SHOWS THE EFFECT ON THE LONG-TERM FORECASTING TASK OF EACH MODULE AND VARYING THE NUMBER OF MLRFS, ASTUCS, AS WELL AS EMBEDDING DIMENSIONS IN THE STUM FRAMEWORK.

Method	MAE	RMSE	MAPE
Compared Baseline (AGCRN)	25.09	37.97	19.56%
Full STUM (Default)	1.28↓	1.50↓	2.15%↓
w/o Backbone (use MLP)	0.39↓	0.21↓	1.87%↓
w/o MLRF&ASTUC (use MLP)	0.36↓	0.20↓	1.06%↓
w/ MLRF (ASTUCs=0)	0.30↓	0.81↓	0.34%↑
w/ ASTUC (rank=1)	0.84↑	0.41↓	3.84%↑
STUM (MLRFs=5)	1.21↓	1.40↓	2.47%↓
STUM (MLRFs=6)	1.31↓	1.47↓	2.50%↓
STUM (MLRFs=7)	1.40↓	1.70↓	2.15%↓
STUM (MLRFs=8)	1.37↓	1.58↓	2.58%↓
STUM (ASTUCs=10)	1.40↓	1.61↓	2.79%↓
STUM (ASTUCs=12)	1.54↓	1.80↓	2.19%↓
STUM (ASTUCs=14)	1.59↓	1.82↓	2.37%↓
STUM (ASTUCs=16)	1.62↓	1.88↓	2.81%↓
STUM (rank=12)	1.16↓	1.35↓	1.81%↓
STUM (rank=16)	1.28↓	1.50↓	2.15%↓
STUM (rank=20)	1.57↓	1.62↓	2.65%↓
STUM (rank=24)	1.69↓	1.75↓	2.90%↓
STUM (rank=28)	1.62↓	1.62↓	2.95%↓
STUM (rank=32)	1.34↓	1.53↓	2.32%↓

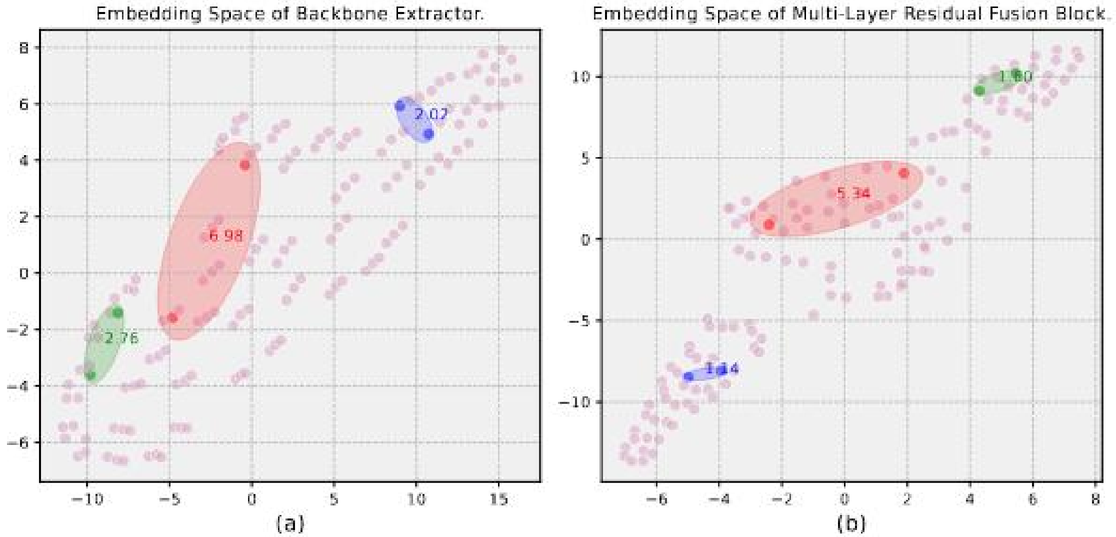
1) **Component Necessity:** STUM’s MLRFs, ASTUCs, and backbone are critical; removing any hurts optimization, yet even with simple MLP replacements, gains persist → strong generalization.

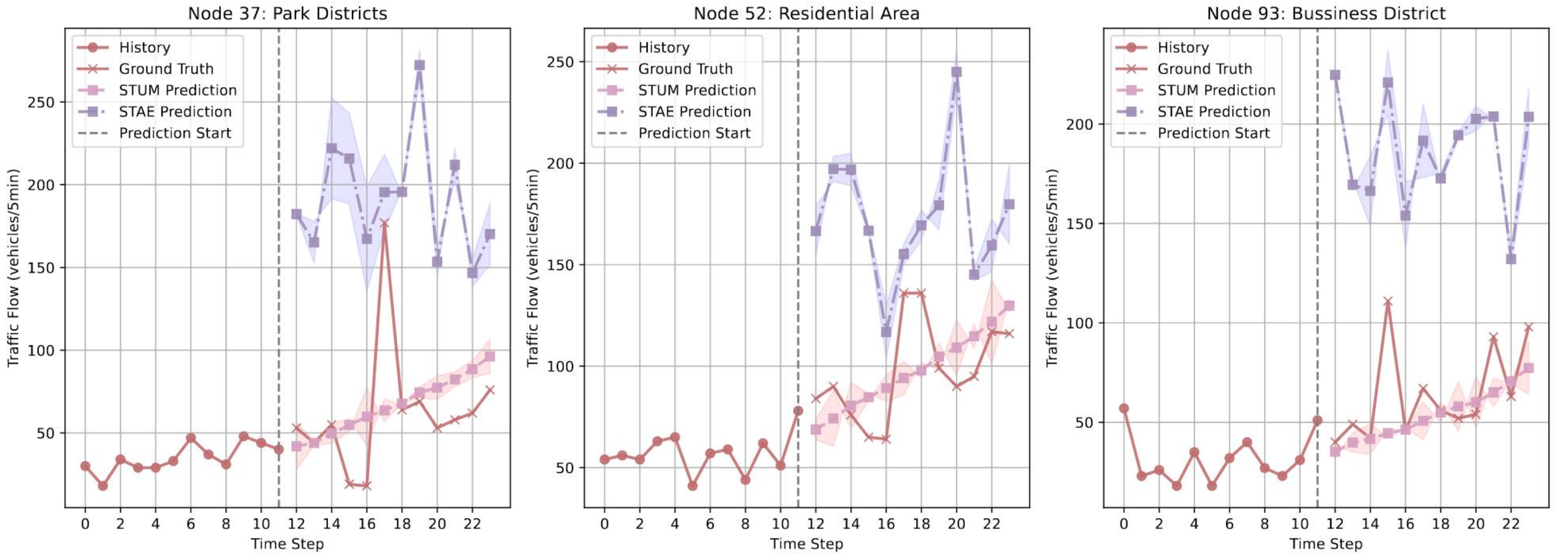
2) **Parameter Sensitivity:** Increasing MLRFs, ASTUCs, or embedding dim boosts accuracy (ASTUCs → largest gain), but excessive layers/dimensions cause diminishing returns (gradient issues, overcomplexity).

3) **Efficiency Balance:** Low-rank factorization in ASTUCs controls compute cost when adding layers; balance ASTUCs with other params for optimal performance.

Visualization Case Study (RQ5)

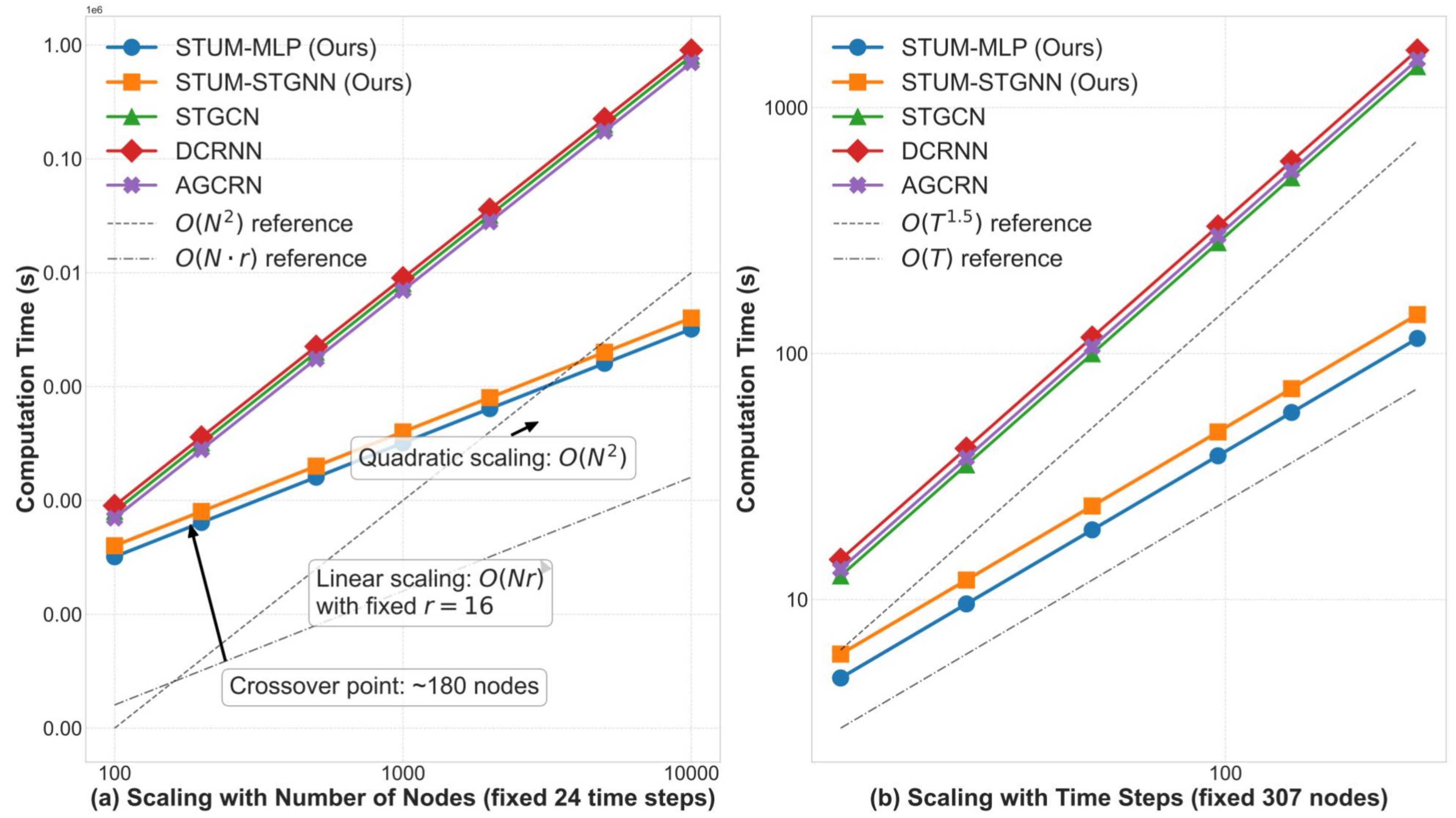
- T-SNE plots (Fig 5) show STUM refines region embeddings: similar traffic regions (residential, park, business) cluster closer, improving spatio-temporal pattern capture vs. backbone extractors.





- STUM outperforms baselines (Fig 6), accurately predicting traffic trends (e.g., downward flows in districts) and aligning with ground truth in volatile regions → validates robust modeling.

Scalability for Large Networks (RQ6)



(a) Scaling with Number of Nodes (fixed 24 time steps)

(b) Scaling with Time Steps (fixed 307 nodes)

STUM uses fixed rank r = 16, achieving linear $\mathcal{O}(Nr)$ scaling.

Baselines (e.g., STGCN) show $\mathcal{O}(N^2)$ quadratic scaling. STUM gains clear advantage beyond ~180 nodes.

Conclusion

In this paper, we address critical spatio-temporal forecasting challenges like data heterogeneity, separated spatial-temporal modules, and low combination efficiency. To solve these, we introduce STUM, unifying spatial and temporal processing. It leverages ASTUCs for capturing complex dependencies, tackling inefficiencies from module separation, and uses multi-layer residual fusion to balance computation and performance. Experiments show it outperforms baselines.

Contributions:

- Propose STUM to unify spatial-temporal learning, breaking from traditional separated-module designs.;
- Design ASTUC (with low-rank matrices) and MLRF dual-extraction to handle heterogeneity and complex interactions.;
- Show consistent outperformance on real datasets, balancing accuracy and efficiency.

Future work:

Extend STUM to multi-domain applications, enhance robustness to missing data/anomalies, and develop optimizations to reduce computational overhead further.

GitHub

WeChat

Thanks for listening

Q&A

Now I am looking for RA and **PhD opportunities.**
If you are interested in my work, please feel free to talk to me!

rwlinno@gmail.com