A Foundation Model for Unified Urban Spatio-Temporal Flow Prediction

Yuan Yuan, Jingtao Ding, Chonghua Han, Depeng Jin, *Member, IEEE* and Yong Li *Senior Member, IEEE*

Abstract—Urban spatio-temporal flow prediction, encompassing traffic flows and crowd flows, is crucial for optimizing city infrastructure and managing traffic and emergency responses. Traditional approaches have relied on separate models tailored to either grid-based data, representing cities as uniform cells, or graph-based data, modeling cities as networks of nodes and edges. In this paper, we build UniFlow, a foundational model for general urban flow prediction that unifies both grid-based and graph-based data. We first design a multi-view spatio-temporal patching mechanism to standardize different data into a consistent sequential format and then introduce a spatio-temporal transformer architecture to capture complex correlations and dynamics. To leverage shared spatio-temporal patterns across different data types and facilitate effective cross-learning, we propose Spatio-Temporal Memory Retrieval Augmentation (ST-MRA). By creating structured memory modules to store shared spatio-temporal patterns, ST-MRA enhances predictions through adaptive memory retrieval. Extensive experiments demonstrate that UniFlow outperforms existing models in both grid-based and graph-based flow prediction, excelling particularly in scenarios with limited data availability, showcasing its superior performance and broad applicability. The datasets and code implementation have been released on https://github.com/YuanYuan98/UniFlow.

Index Terms—Spatio-temporal prediction, prompt learning, universal model.

I. Introduction

Urban flows refer to the dynamic movement patterns within a city, such as crowd flow and traffic flow [1]. Predicting these flows is essential for optimizing infrastructure and enabling efficient control and management of traffic, public transport, and emergency responses. To this end, various data collection approaches, including sensor networks [2], GPS tracking [3], and mobile networks [4], [5], are widely employed.

Urban spatio-temporal flow prediction has been a long-standing research problem, attracting significant attention from researchers [3], [6]–[13]. The data collected for this purpose are typically organized into two categories: grid-based and graph-based data. Grid-based data represents the city as a grid of uniform cells, each containing aggregated flow information, making it suitable for capturing local dynamics and interactions. Various models, primarily employing Convolutional Neural Network (CNN)-based techniques, have been proposed to handle grid-based data [3], [14], [15]. In contrast, graph-based data models the city as a network of nodes (e.g., intersections) and edges (e.g., roads), which better represents connectivity across the city. Numerous Graph Neural Network

Y. Yuan, J. Ding, C. Han, D. Jin, and Y. Li are with Beijing National Research Center for Information Science and Technology (BNRist), and Department of Electronic Engineering, Tsinghua University, Beijing 100084, China. E-mail: y-yuan20@mails.tsinghua.edu.cn
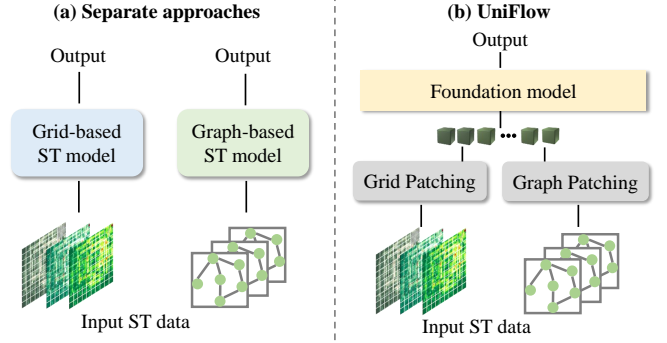

Fig. 1: High-level comparison between our proposed model and separate modeling for spatio-temporal data. Our solution unifies grid and graph data formats.

(GNN)-based models have been developed for prediction in this context [2], [10], [16]–[18].

Although existing works have made significant progress in this task due to the development of deep learning techniques, they still focus on developing and training specialized models for different types of data. In this work, we aim to answer a crucial research question: ***Is it possible to build a foundation model to unify urban spatio-temporal flow prediction?*** By doing so, we can simplify this classical urban task by building a single, one-for-all model without the need for multiple specialized models. The intuition behind this vision lies in the intrinsic connection between different data types, both representing the spatio-temporal rhythms of urban life and reflecting various facets of human activity. Grid-based data capture local, regular spatial relationships, while graph-based data describe complex, irregular connections. These aspects are interconnected; for example, traffic flow involves both regular spatial relationships (grid) and complex congestion connections between roads (graph). Moreover, universal spatio-temporal laws, such as periodicity, are not constrained by specific data structures. Addressing these data types separately hinders the ability to fully capture the spatio-temporal dynamics of a city.

However, achieving such a unified foundation model is not a trivial task. The first challenge lies in integrating different data structures. Grid-based data, with its uniform distribution and local flow information, contrasts sharply with the network-like structure of graph-based data, which emphasizes connectivity. Unifying these different types of data into a cohesive model requires bridging their structural differences. The second challenge is extracting and utilizing the underlying commonalities between these disparate data types, as

they capture different aspects of spatio-temporal dynamics. Directly applying a single model to fit all data without prior spatio-temporal knowledge makes it challenging to leverage shared commonalities, potentially undermining the objective of creating a truly unified model.

To address these challenges, we have developed UniFlow, a foundation model for unified urban spatio-temporal flow prediction that handles both gird-based and graph-based data. First, we design a multi-view spatio-temporal patching mechanism to process different types of input data in parallel, unifying them into a consistent sequential structure. Next, we develop a spatio-temporal transformer architecture to capture complex dynamics over time and space. To extract fundamental spatio-temporal patterns independent of specific data and facilitate cross-learning across different data types, we propose Spatio-Temporal Memory Retrieval Augmentation (ST-MRA). The core idea is to generate prompts that are most beneficial for the different spatio-temporal patterns encountered. Leveraging well-established spatio-temporal domain knowledge, we create explicit and structured memory modules to store spatio-temporal patterns. The transformer is enhanced by dynamically interacting with this memory, improving its utilization of stored knowledge. For each input sample, this process involves vectorizing queries based on extracted spatio-temporal patterns, retrieving relevant content from the memory, and generating personalized prompt vectors to guide the model's predictions. Unlike the non-parametric Retrieval-Augmented Generation (RAG) [19] approach widely used in large language models, ST-MRA is fully learnable, with both the memory modules and retrieval process being trainable. In summary, our contributions are as follows:

- We pioneer the unification of different types of urban flow data into a single model for universal prediction.
- We propose UniFlow, a transformer-based unified model that leverages multi-view spatio-temporal patching and spatio-temporal memory retrieval augmentation to capture and utilize complex spatio-temporal dynamics in urban flow data.
- Extensive experiments on nine real-world datasets demonstrate the superiority of UniFlow. Our one-for-all foundation model consistently outperforms specialized, individually trained baseline models, achieving an average performance improvement exceeding 10%. It highlights the significant potential of foundation models in this field.

II. RELATED WORK

A. Urban Spatio-Temporal Prediction

Urban spatio-temporal flow prediction [1], [6] aims to model and forecast the dynamic patterns of urban activities over space and time, such as crowd flows and traffic flows. It provides crucial insights for urban decision-makers to optimize infrastructure and improve public services. Generally, these predictions rely on two types of data: (i) grid-based flow data and (ii) graph-based flow data. Deep learning methods have achieved significant advancements in this task [6]–[8], [20], [21].

TABLE I: Comparison between existing works and UniFlow from four aspects.

Approach	Model[1]	Data[2]	Zero-shot
TrafficBERT [41]	S	Graph-O	✗
GPT-ST [42]	S	Graph-O	✗
ST-SSL [43]	S	Graph-O	✗
STEP [44]	S	Graph-O	✗
STG-LLM [45]	S	Graph-O	✗
UrbanGPT [36]	U	Grid-M	✓
PromptST [46]	S	Grid-O	✗
UniST [45]	U	Grid-M	✓
UniFlow	U	Graph-Grid-M	✓

[1] S: specific model; U: universal model.
[2] Data diversity: O denotes one dataset and M denotes multiple datasets.

For grid-based flow data, CNNs [3], [9], [15] and ResNets [3] have been introduced to model the regular adjacency spatial correlations. For graph-based flow data, various GNNs [7], [10], [11], [22], [23] are utilized to capture more complicated spatial relationships. Besides, RNNs [24]–[26] and transformers [12], [13], [27], [28] are also utilized to model temporal dynamics. However, most approaches remain constrained by the need to train separate models for different datasets. In contrast, our proposed model allows to train across different data types and provides a unified solution. While transformer-based approaches [28] can, in theory, handle both grid and graph data, they face scalability issues when applied to large-scale graphs or grid with massive nodes or locations.

B. Foundation Models for Spatio-Temporal Data

Building on the significant advancements in foundation models for NLP [29], [30] and CV [31], [32], recent efforts have introduced foundation models for urban tasks. These research have explored the potential of large language models (LLMs) in this domain. Intelligent urban systems like CityGPT [33], [34], CityBench [35], and UrbanGPT [36] have shown their proficiency in handling language-based urban reasoning tasks. There has also been great progress in foundation models for time series prediction [8], [37]–[39]. Spatio-temporal data, unlike the sequential nature of time series data, involves a more intricate setup due to its dependencies across space and time. The complexity deepens with different types of spatial correlations, including those arranged in structured grids and those mapped onto unstructured graphs, adding layers of difficulty to the modeling process. UniST [40] provides a universal solution for multiple spatio-temporal scenarios but is still limited to grid-based data. In Table I, we contrast the key characteristics of UniFlow against other related works. UniFlow stands out as a universal model proficient in handling a diverse range of data types. Additionally, it excels in both few-shot and zero-shot learning scenarios.

C. Retrieval-Augmented Generation

Retrieval-Augmented Generation (RAG) [19] enhances model performance by incorporating external knowledge into
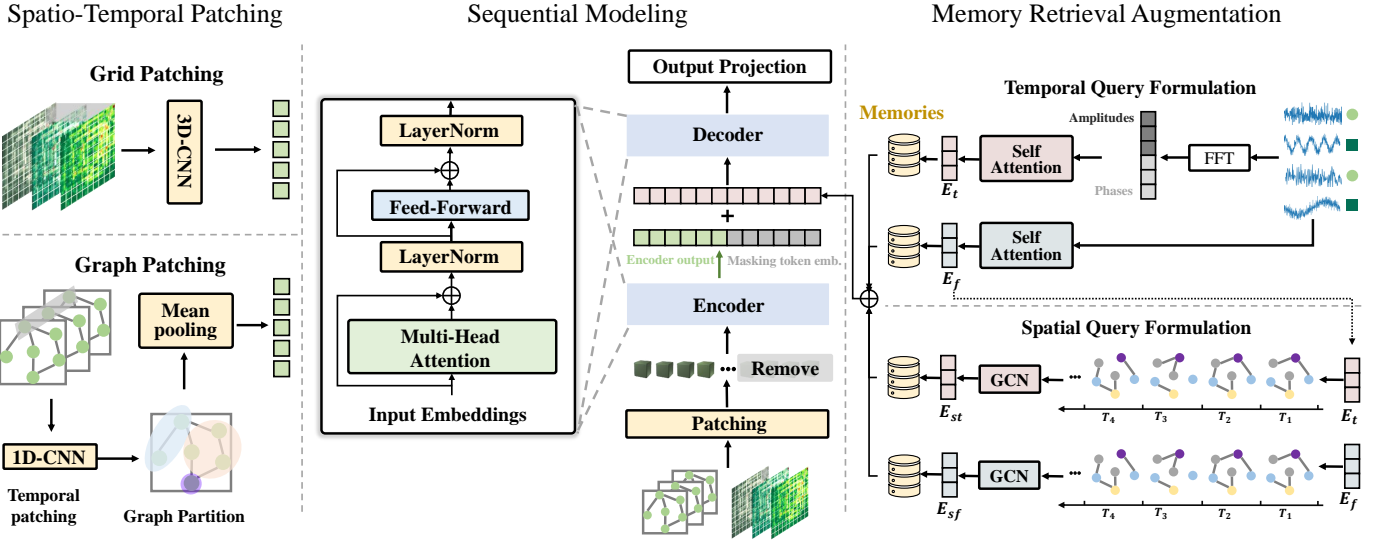
Fig. 2: The overall architecture of UniFlow. For the input data, we first perform spatio-temporal patching to construct sequential structures. Then the transformer model performs sequential modeling, which is prompted by spatio-temporal memory retrieval augmentation.

the generation process. Initially popularized in the context of natural language processing [47], [48], RAG has shown promise in improving the quality and relevance of generated text by retrieving pertinent information from a vast corpus of documents. Recently, RAG techniques are also utilized in the recommender system [49], which designs semantic user behavior retrieval to improve the data quality. Applying RAG to spatio-temporal data presents unique opportunities but lacks explorations. In this work, inspired by the principles of RAG, we propose to utilize retrieval-augmented spatio-temporal knowledge. In this way, we can enhance the model's ability to draw on relevant spatio-temporal patterns, thereby improving prediction accuracy and robustness.

III. PRELIMINARY

Flow Data: Spatio-temporal Flow data, denoted as X, can be defined as a tensor with dimensions $T \times N \times C$, where T represents the temporal period length, N is the number of locations, and C is the number of variables. Typically, urban spatio-temporal flow data are organized in two main types: grid-based and graph-based.

For the grid-based data, X can be expressed as $T \times H \times W \times C$, where H and W represent the longitude and latitude partitions, respectively, and $H \times W = N$.

For the graph-based data, X is represented as $T \times N \times C$, where N corresponds to the number of nodes in the topology. Additionally, graph-based data includes a spatial topology denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A})$, where $\mathcal{V} = (v_1, v_2, ..., v_N)$ is the node set, $\mathcal{E} = \{e_{ij}\}$ is the edge set, and $\mathcal{A} = \{a_{ij} \in \{0,1\}\} \in \mathbb{R}^{N \times N}$ denotes the adjacency matrix of the spatial graph. In this context, $a_{ij} = 1$ indicates an edge between node i and node j, while $a_{ij} = 0$ indicates no edge.

Spatio-Temporal Flow Prediction: A common scenario involves using H historical records to predict future P flows. Formally, given a set of historical spatio-temporal observations

$\{X_t\}_{t=1}^{H}$, the objective is to predict the future spatio-temporal states $\{X_t\}_{t=H+1}^{H+P}$, where H denotes the number of history time steps and P is the number of prediction time steps. This involves learning a mapping $\mathcal{F} : \{X_t\}_{t=1}^{H} \rightarrow \{X_t\}_{t=H+1}^{H+P}$, which can be represented as:

$$\hat{X}_{H+1:H+P} = \mathcal{F}(X_{1:H}), \tag{1}$$

where $\hat{X}_{H+1:H+P}$ denotes the predicted values for the prediction period, and $X_{1:H}$ represents the historical records. The goal is to predict future data points from $H + 1$ to $H + P$ based on the historical data spanning the period 1 to H.

IV. METHODOLOGY

Figure 2 illustrates the overall architecture of UniFlow, which consists of three main components: (i) spatio-temporal patching, which converts spatio-temporal flow data with varied types into a standard sequential format, (ii) sequential modeling with transformers to process the patched spatio-temporal input, and (iii) spatio-temporal memory retrieval augmentation, which retrieves most useful prompts from shared memories to further enhance prediction performance.

A. Spatio-Temporal Patching

While the temporal dimension of flow data naturally exhibits a sequential structure between consecutive timestamps, the spatial organization does not inherently follow a sequential structure. Considering the intertwined nature of spatio-temporal dynamics, relying solely on temporal correlations limits the ability of transformers to fully capture spatio-temporal correlations. Therefore, implementing spatio-temporal transformers is necessary.

To transform flow data into a sequential structure, we design separate lightweight patching modules to patch flow data for

both grid-based and graph-based types. Notably, for both types, we adopt a channel-independence strategy following existing practices [50]. This approach treats the flow data denoted as $T \times N \times C$ as C individual $T \times N$ data sequences. Thus, our goal is to transform the input with shape $T \times N$ into a sequence with length of L.

1) Spatio-Temporal Grid Patching: Grid-based flow data can be represented as $X \in \mathbb{R}^{T \times H \times W}$ with $H \times W = N$ spatial regions. To convert this data into a sequential structure, we first use a three-dimensional convolutional neural network (3D-CNN) with kernel and stride sizes equal to the patch size (p_t, p_s, p_s). In this way, we divide the data into L patches, i.e., $L = \frac{T}{p_t} \times \frac{H}{p_s} \times \frac{W}{p_s}$. We then arrange these smaller patches in a scan-line order, treating them as a sequence. The process can be formulated as follows:

$$X' = \text{CONV3D}(X, \text{KERNEL} = (p_t, p_s, p_s), \text{STRIDE} = (p_t, p_s, p_s)),$$
$$S = \text{FLATTEN}(X').$$

Thus, $S \in \mathbb{R}^{L \times D}$ is a sequence of patches that can be fed into the transformer model for further processing. D is the output dimension of the CNN model.

2) Spatio-Temporal Graph Patching: Graph-based flow data is represented as $X \in \mathbb{R}^{T \times N}$, together with a graph topology $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of nodes and $\mathcal{E}$ is the set of edges. Unlike the grid-based data, graph topology lacks a regular structure, making CNN-based patching approaches inapplicable. To tackle this issue, we use separate patching modules to patch graph-based data. For the temporal dimension, we still use a 1D-CNN to extract features from the time series data. To patch the spatial graph, we divide the overall graph into multiple subgraphs, each containing a set of nodes and edges. We adopt the METIS algorithm [51] to partition the graph into L subgraphs. The primary goal of METIS is to divide a graph into smaller parts while minimizing the number of edges that are cut and balancing the sizes of the parts.

After dividing the graph into multiple subgraphs, we treat each subgraph as a patch. We obtain the patching results by averaging the pooled temporal features of different nodes within each subgraph. The graph patching process is formulated as follows:

$$X' = \text{CONV1D}(X),$$
$$(g_1, g_2, \ldots, g_L) = \text{METIS}(G) \quad \text{where} \quad g_i = (V_i, E_i),$$
$$s_i = \text{AVERAGEPOOLING}(X'_{V_i}) \quad \text{for each} \quad g_i,$$
$$S = \text{FLATTEN}([s_1, s_2, \ldots, s_L]).$$

Thus, S is a sequence of patches that can be fed into the transformer model for further processing.

B. Sequential Modeling with Transformers

After converting different flow data into the sequential structure, we leverage a transformer model to capture the spatio-temporal dependencies. The transformer model consists of a stack of N identical layers, each containing a multi-head self-attention mechanism and a feed-forward neural network.

The multi-head self-attention mechanism allows the model to focus on fine-grained spatio-temporal correlations and dynamics within the input sequence.

Implementation details: For spatio-temporal flow prediction, we adopt an encoder-decoder architecture for the transformer model. The overall flow data S includes both historical records S_h and prediction horizons. The encoder masks the future data and processes only the historical data S_h. After obtaining the encoder output Z, we concatenate the embedding Z with the mask token embedding E_m to construct the complete embedding map. The decoder then performs self-attention over the entire sequence of embeddings. The process is formulated as follows:

$$S_h = \text{MASK}(S), \tag{2}$$
$$\text{Encoder:} \quad Z_e = \text{ENCODER}(S_h),$$
$$Z_d = \text{CONCAT}(Z_e, E_m), \tag{3}$$
$$\text{Decoder:} \quad Y = \text{DECODER}(Z_d),$$

where Z is the output of the encoder, which is then used as input to the decoder. Y is the output of the decoder, which represents the predicted results.

C. Spatio-Temporal Memory Retrieval Augmentation

We design Spatio-Temporal Memory Retrieval Augmentation (ST-MRA) to memorize and utilize useful spatio-temporal patterns. ST-MRA aims to enhance prediction accuracy by enabling cross-learning across different types of flow data. It does so by retrieving similar prompts from shared memories, thereby informing and improving current predictions. This approach addresses the challenge of integrating diverse spatio-temporal patterns, ultimately leading to more robust and informed predictions. We demonstrate the workflow of ST-MRA in Figure 3. The main steps of ST-MRA are as follows:

1) Formulate a spatio-temporal query based on the input data;
2) Retrieve relevant memories to generate useful prompts;
3) Utilize the retrieved prompts to augment the input data.

In the following subsections, we elaborate on the details of our proposed ST-MRA.

1) Spatio-Temporal Query Formulation: We extract spatio-temporal patterns from the input data to create queries, considering both temporal and spatial perspectives. Specifically, we consider two perspectives to obtain queries to represent patterns of the input flow data: temporal and spatial perspectives. Temporal queries are designed to capture patterns and trends over time, ensuring that the model understands temporal dynamics. In contrast, spatial queries focus on depicting spatial correlations, allowing the model to recognize and leverage the relationships between different locations. This dual approach ensures a comprehensive representation of spatio-temporal flow data.

Temporal queries. To effectively understand temporal dynamics, we analyze data through two key characteristics: the time domain and the frequency domain. This enables a comprehensive understanding of both short-term and long-term patterns.
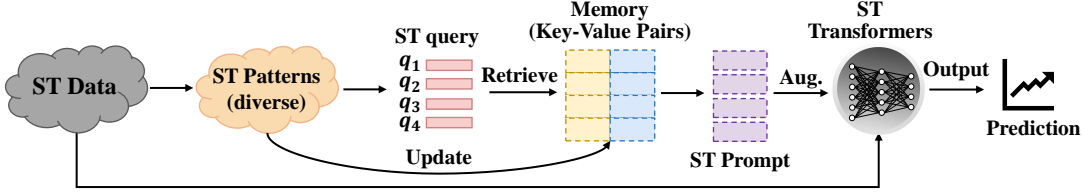
Fig. 3: ST-MRA workflow in UniFlow.

- *Time domains.* This refers to the direct representation of data over time, capturing the sequence of changes. Urban phenomena often depend on consecutive time steps, such as traffic flow being influenced by the previous minutes or hours.
- *Frequency domains.* This domain uses Fourier analysis to transform temporal data into frequency components, revealing periodic and cyclical behaviors that are not visible in the time domain. Spatio-temporal dynamics often show periodic patterns, such as daily and weekly traffic flows.

Each of these domains offers unique perspectives and advantages that are essential for comprehensively modeling temporal patterns. Specifically, we use self-attention to obtain time-domain patterns and Fast Fourier Transform (FFT) to obtain frequency-domain patterns. The process can be formulated as follows:

$$E_t = \text{SELFATTENTION}(S_h),$$
$$E_f = \text{FFT}(S_h),$$

where E_t and E_f are the extracted time-domain and frequency-domain patterns, respectively.

Spatial queries. Spatio-temporal data differ from traditional time series data due to their inherent spatial correlations, making effective modeling of spatial patterns crucial for accurate predictions. Both grid-based and graph-based data have unique spatial organizations but share the common concept that their spatial organizations both represent relationships between different parts of a city. Thus, we design an adaptive approach to extract spatial patterns for both types of data. Based on the learned patterns in both the time and frequency domains, we adaptively determine the spatial topology between different locations. This adaptive learning process is formulated as follows:

$$A_t = softmax(\text{RELU}(E_t \cdot E_t^T)),$$
$$A_f = softmax(\text{RELU}(E_f \cdot E_f^T)),$$

where A_t and A_f are the learned spatial topology based on time-domain patterns and frequency-domain patterns, respectively.

Using the learned graph topology, we employ a Graph Convolutional Network (GCN) to learn the spatial representations between different locations:

$$E_{st} = \text{GCN}(E_t, A_t), \quad E_{sf} = \text{GCN}(E_f, A_f), \quad (4)$$

where E_{st} and E_{sf} are extracted spatial patterns, i.e., spatial queries. It is worth noting that these spatial queries are not purely from a spatial perspective but incorporate entangled spatio-temporal relationships.

2) Memory Retrieval Augmentation: As shown in Figure 3, we design shared spatio-temporal memories to store and recall spatio-temporal patterns. These memories act as powerful repositories for retrieving relevant pattern, enabling the model to access and leverage already-learned patterns effectively.

Spatio-temporal memories. We maintain external memories for spatio-temporal retrieval augmentation. Based on well-established spatio-temporal knowledge [1], we categorized these memories into four groups:

- *Time-domain memory:* Stores time-domain patterns extracted from historical data, capturing sequential dependencies.
- *Frequency-domain memory:* Stores frequency-domain patterns, capturing periodic and cyclical behaviors.
- *Time-spatial memory:* Stores spatial patterns, capturing spatial correlations between different locations based on time-domain patterns.
- *Frequency-spatial memory:* Stores spatial patterns, capturing spatial correlations and relationships between different locations based on frequency-domain patterns.

Memory structure. Each of these memories is structured as key-value pairs (K, V), where both keys and values are $N \times D$ learnable embeddings. Here, N represents the number of memory units in the base, which is a hyper-parameter, and D is the embedding dimension, which equals the output size of the encoder.

Prompt retrieval. These memories store knowledge from the time domain, frequency domain, and adaptive spatial topology derived from both domains. The retrieval process involves querying these memories to obtain useful prompts. We use the learned temporal and spatial patterns as queries to guide this process. For each query, we calculate the similarity between the query and each key unit in the memory, then extract the corresponding value based on these similarity scores. The process is formulated as follows:

$$Q_i \in \{E_t, E_f, E_{st}, E_{sf}\}, \ K_i \in \{K_t, K_f, K_{st}, K_{sf}\},$$
$$\alpha_i = softmax(Q_i \cdot K_i^T),$$
$$P_i = \sum_j \alpha_{ij} V_i, \ \text{where } V_i \in \{V_t, V_f, V_{st}, V_{sf}\},$$

where Q_i represents the input queries, K_i represents the keys in the memory, and V_i represents the corresponding values.

TABLE II: Results of short-term prediction on six datasets in terms of MAE and RMSE. We use the average prediction errors over all prediction steps. Bold denotes the best results and underline denotes the second-best results.

| | Grid Data | | | | | | | | Graph Data | | | |
| | TaxiBJ | | FlowSH | | CrowdNJ | | TaxiNYC | | TrafficBJ | | TrafficSH | |
Model	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
HA	91.55	53.036	38.920	13.435	0.932	0.487	77.10	26.49	2.205	1.457	2.381	1.554
ARIMA	291	57.5	26.7031	9.152	0.989	0.4435	99.220	23.911	2.608	1.642	1.7496	0.620
STResNet	37.96	26.55	59.82	45.63	0.718	0.511	26.88	14.81	-	-	-	-
ACFM	30.95	19.87	46.92	24.95	0.468	0.284	20.82	9.85	-	-	-	-
STNorm	31.21	19.00	28.46	11.88	0.384	0.231	26.94	10.43	-	-	-	-
STGSP	27.31	17.54	38.77	17.54	0.410	0.263	25.94	10.52	-	-	-	-
MC-STL	38.50	28.51	46.06	33.83	0.727	0.504	36.75	26.01	-	-	-	-
MAU	71.07	46.37	45.04	21.38	0.648	0.402	49.15	21.79	-	-	-	-
MIM	68.18	42.40	47.29	22.49	0.715	0.399	24.53	9.151	-	-	-	-
SimVP	35.58	21.67	28.59	15.87	0.282	0.191	19.69	9.08	-	-	-	-
TAU	26.43	15.86	26.04	15.22	0.326	0.219	19.46	9.08	-	-	-	-
PromptST	27.42	16.12	23.01	9.37	0.306	0.161	22.82	8.24	-	-	-	-
UniST	<u>23.67</u>	<u>14.04</u>	<u>19.95</u>	<u>9.10</u>	<u>0.191</u>	<u>0.119</u>	<u>17.55</u>	<u>5.85</u>	-	-	-	-
STGCN	-	-	-	-	-	-	-	-	2.53	1.94	2.20	1.66
DCRNN	-	-	-	-	-	-	-	-	2.05	1.47	2.09	1.49
GWN	-	-	-	-	-	-	-	-	2.40	1.82	2.15	1.56
MTGNN	-	-	-	-	-	-	-	-	1.76	1.23	<u>1.85</u>	<u>1.27</u>
AGCRN	-	-	-	-	-	-	-	-	2.37	1.78	2.55	1.91
GTS	-	-	-	-	-	-	-	-	2.44	1.89	2.84	2.21
STEP	-	-	-	-	-	-	-	-	2.11	1.56	2.15	1.57
STID	25.55	16.36	21.19	12.92	0.234	0.160	18.49	8.32	<u>1.75</u>	<u>1.16</u>	2.06	1.39
PatchTST	53.36	30.55	28.17	10.69	0.465	0.223	50.45	17.03	2.06	1.368	2.24	1.47
PatchTST-v2	60.55	33.62	31.79	12.16	0.811	0.403	58.61	21.27	2.53	1.67	2.78	1.82
iTransformer	42.17	24.05	25.91	10.19	0.466	0.216	45.19	45.19	2.04	1.35	2.29	1.52
Time-LLM	51.20	29.55	28.19	10.57	0.405	0.210	52.94	17.65	2.07	1.37	2.23	1.47
UniFlow	**20.35**	**12.14**	**15.18**	**6.08**	**0.180**	**0.107**	**16.02**	**5.81**	**1.72**	**1.12**	**1.84**	**1.24**

Augmentation. The augmentation process is straightforward. After obtaining the retrieved prompt P_i, we add it to the input embedding of the decoder for augmentation and feed it into the transformer model for further processing. This augmented input is then passed through the decoder to generate the final predictions.

V. PERFORMANCE EVALUATIONS

A. Experimental Settings

Datasets: We evaluate UniFlow using nine spatio-temporal datasets[1], encompassing both grid-based and graph-based structures. The grid-based datasets include TaxiBJ, TaxiNYC, CrowdNJ, CrowdBJ, FlowSH, and PopSH. The graph-based datasets consist of TrafficBJ, TrafficSH, and TrafficNJ, where the road networks within the cities form the graph topology. It is noteworthy that the graph datasets we utilize are large-scale, containing over 10,000 nodes, in contrast to the smaller spatio-temporal graph datasets that were commonly used in the past [42], [52]. Detailed descriptions and characteristics of each dataset are provided in Appendix Table VI and Table VII. For each dataset, we partition the data into training, validation, and testing sets with a ratio of 6:2:2.

Baselines: We compare UniFlow with a broad collection of state-of-the-art spatio-temporal prediction models, which can be categorized into four main groups[2]:

- *Classical methods.* These include methods such as HA and ARIMA, which use predefined rules or simple mathematical models to make predictions without requiring extensive training.
- *Deep spatio-temporal prediction models (grid):* These include STResNet [3], ACFM [15], STNorm [53], STGSP [54], MC-STL [55], MAU [56], MIM [57], SimVP [58], TAU [59], STUD [52], PromptST [46], UniST [40]. These deep learning-based models are designed to handle grid-based flow data, leveraging techniques that can capture spatial dependencies in a grid layout.
- *Deep spatio-temporal prediction models (graph):* These include STGCN [2], GWN [17], MTGNN [60], GTS [61], DCRNN [9], STEP [62], AGCRN [10]. These methods focus on graph-based flow data, utilizing GNN-based techniques to model the complex relationships and interactions within graph structures.
- *Time series prediction models:* These include PatchTST [50], iTransformer [63], Time-LLM [38]. These approaches are proposed for multivariate time series prediction. We also train a one-for-all PatchTST, denoted as PatchTST-v2, as the baseline.

---

[1] We select cities with both grid-based and graph-based datasets.

[2] We provide detailed description of selected baselines in Appendix B.

TABLE III: Results of long-term prediction on three datasets in terms of MAE and RMSE.

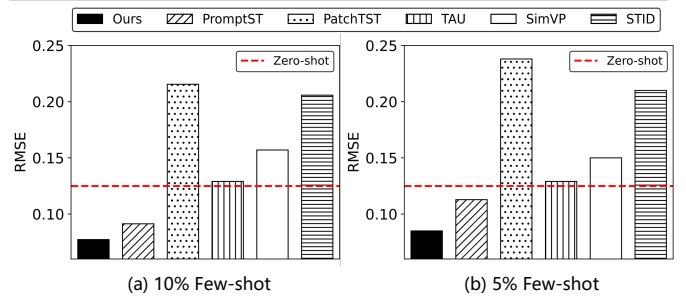| | Grid Data | | | | Graph Data | |
| | TaxiBJ | | FlowSH | | TrafficBJ | |
Model	RMSE	MAE	RMSE	MAE	RMSE	MAE
HA	74.16	43.55	41.84	18.02	2.20	1.50
ARIMA	101	55.69	53.01	21.73	2.75	1.87
STResNet	43.98	29.49	72.87	50.88	-	-
ACFM	59.85	40.50	45.92	27.45	-	-
STNorm	41.69	25.26	39.59	21.60	-	-
STGSP	26.92	17.09	33.86	19.37	-	-
MC-STL	48.06	35.25	50.61	36.62	-	-
MAU	37.32	21.59	38.86	17.98	-	-
MIM	36.52	20.92	41.73	22.92	-	-
SimVP	33.78	20.13	31.04	16.74	-	-
TAU	31.28	18.59	34.68	20.58	-	-
PromptST	38.83	22.52	34.83	13.80	-	-
UniST	25.21	15.66	34.18	14.30	-	-
STGCN	-	-	-	-	3.15	2.57
DCRNN	-	-	-	-	2.79	2.22
GWN	-	-	-	-	3.12	2.52
MTGNN	-	-	-	-	2.93	2.30
AGCRN	-	-	-	-	2.23	1.64
GTS	-	-	-	-	3.25	2.66
STEP	-	-	-	-	2.44	1.81
STID	29.69	19.46	29.51	17.99	1.89	1.28
PatchTST	34.43	18.56	26.41	9.50	1.91	1.22
iTransformer	36.62	19.76	25.54	10.35	1.93	1.27
Time-LLM	33.85	18.64	44.69	16.52	1.92	1.25
UniFlow	**23.63**	**13.75**	**19.10**	**7.65**	**1.81**	**1.19**

Fig. 4: Performance comparison of few-shot predictions between UniFlow and baseline models on the CrowdSH dataset. The red dashed line represents zero-shot performance.

requiring specialized designs for flow data. For graph-based data, MTGNN exhibits relatively good performance. These results differ from previous works [52], [62], likely due to the very large graph topology in our dataset, with more than 10,000 nodes.

Multivariate time series approaches also fail to perform well. The one-for-all PatchTST performed even worse than separate PatchTST models, indicating its inability to learn mutual benefits across different datasets. Time-LLM performed poorly in urban spatio-temporal flow prediction, indicating the necessity of training a foundation model specifically for urban spatio-temporal flow data.

UniFlow achieves the best performance across all datasets, including both grid-based and graph-based data. Compared to the best baseline, UniFlow with single model shows a relative improvement of 9.1%. The superiority of UniFlow highlights the potential to unify different flow data for enhancing emergent capabilities.

Experimental Configuration: The model is configured with 4 encoder layers and 4 decoder layers. The hidden size is set to 256. The maximum number of training epochs is 200, with early stopping employed to prevent overfitting. All baseline models are also trained with a maximum of 200 epochs for a fair comparison. The batch size is adjusted for each dataset to ensure relatively similar training iterations across different datasets. Each memory contains 512 embeddings. The learning rate is initially set to 5×10^{-4} and decays to 5×10^{-5} after 200 epochs.

B. Short-Term Prediction

Setups: For short-term prediction, we define both the input step and prediction horizon as 12, denoted as $12 \rightarrow 12$. For the baseline models, we train a specific model tailored to each dataset, while UniFlow is evaluated across all datasets. It is worth noting that the step size of 12 corresponds to different temporal scales for different datasets. For instance, in the Traffic dataset, 12 represents three hours, whereas in the TaxiBJ dataset, 12 corresponds to six hours.

Results: Table II illustrates the results of short-term prediction.[3] As shown, video prediction models, such as MAU, MIM and SimVP, did not perform very well. In contrast, urban spatio-temporal baselines, such as STID, STResNet, ACFM, PromptST, and UniST, achieve better performance than video models. This indicates that the spatio-temporal dynamics in urban flow data are quite different from those in video data,

[3]The remaining results can be found in Appendix D.

C. Long-Term Prediction

Setups: For long-term prediction, we extend our evaluation to capture the model's ability to predict over extended periods. In this setup, both the input step and prediction horizon are set to 64, denoted as $64 \rightarrow 64$. This evaluation aims to test the model's capability to maintain accuracy over longer prediction horizons.

Results: Table III presents the results of long-term prediction. As the prediction horizon increases, we observe that video prediction models generally show improvement, while urban spatio-temporal models, including both grid-based and graph-based approaches, exhibit a decline in performance. This suggests that video models are better suited to handling long-term scenarios. Consistent with its superior performance in short-term prediction, UniFlow also achieves the best performance in the long-term prediction setting. Compared with the best baseline, it achieves a relative improvement of 11.9% in terms of RMSE.

D. Few-Shot and Zero-Shot Capabilities

Setups: Foundation models have demonstrated remarkable generalization abilities, performing well even with limited data or no data at all. To verify whether our proposed model inherits
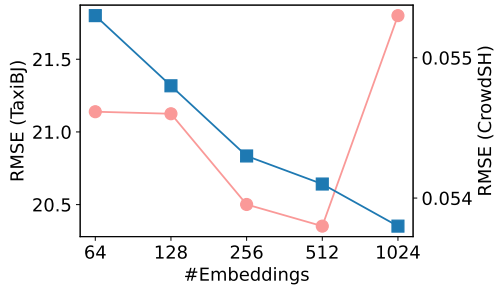
Fig. 5: Results of ablation studies of UniFlow in terms of the number of units in each memory.
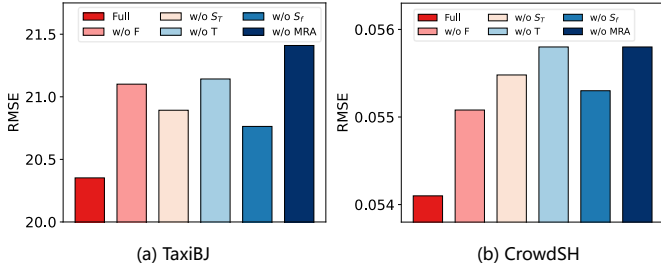

Fig. 6: Results of ablation studies of UniFlow in terms of the type of memory.

this capability, we design experiments for both few-shot and zero-shot predictions. We divide the datasets into two groups: source datasets and target datasets, with the CrowdSH dataset designated as the target and the remaining datasets as the source. The model is trained solely on the source datasets.

For few-shot prediction, we fine-tune the model with a limited amount of data from the target datasets and evaluate its performance on the same target datasets. This setup challenges the model to learn effectively from sparse data and aims to assess its ability to generalize from limited information. We select small portions of the target dataset for fine-tuning, specifically 5% and 10%.

For zero-shot prediction, we do not provide any training data from the target domain. Instead, we directly evaluate the model's performance on the target datasets without any fine-tuning.

Results: Figure 4 shows the comparison results of UniFlow against selected baselines with relatively good performance in short-term prediction. As we can observe, with limited training data, UniFlow exhibits the best few-shot learning capabilities, nearly achieving results comparable to normal prediction with complete training data. However, baseline approaches show significant performance degradation, indicating their limitations in sparse data scenarios. The red dashed line denotes the zero-shot performance of UniFlow, which surprisingly surpasses most baseline approaches that have some training data, demonstrating its superior generalization ability.

E. Ablation Study

We perform ablation studies to examine how the different designs in ST-MRA contribute to the final performance. Our

TABLE IV: Short-term prediction performance with different noise levels on TrafficBJ (graph) in terms of RMSE and MAE. We use the average prediction errors over all prediction steps.

Model	1%		5%		10%	
	RMSE	MAE	RMSE	MAE	RMSE	MAE
STGCN	2.497	1.972	3.292	2.662	4.254	2.965
DCRNN	2.119	1.542	2.947	2.277	3.212	2.519
GWN	2.448	1.867	2.935	2.307	3.149	2.497
MTGNN	1.776	1.253	2.214	1.661	2.701	2.070
AGCRN	2.408	1.825	2.942	2.300	3.326	2.616
GTS	2.447	1.901	2.624	2.066	3.003	2.380
STEP	2.119	1.570	2.401	1.834	2.927	2.267
PatchTST	2.09	1.38	2.32	1.63	2.79	2.11
iTransformer	2.05	1.35	2.13	1.47	2.37	1.73
Time-LLM	2.08	1.37	2.08	1.37	2.10	1.39
UniFlow	**1.75**	**1.15**	**1.76**	**1.15**	**1.76**	**1.16**

ablation studies focus on two perspectives. The first is the number of memory units (embeddings) in each memory. The second is the type of memories.

Number of Memory Units: Figure 5 shows the results with different numbers of memory units. As we can observe, increasing the number of units initially reduces the prediction RMSE, indicating that memories with higher representation capabilities benefit the prompt retrieval process. However, beyond a certain point, further increasing the number of units leads to an increase in RMSE. This can be attributed to the fact that the memory stores prototypes representing typical spatio-temporal patterns. When the number of prototypes becomes too large, it becomes difficult to learn and categorize these typical prototypes. As a result, the prompt retrieval process becomes less effective.

Type of ST-MRA: Figure 6 shows the results of UniFlow and its degraded versions by removing certain parts of the ST-MRA. We use the notations T, F, S_T, and S_F to represent time-domain, frequency-domain, time-derived spatial, and frequency-derived spatial prompts, respectively. Additionally, we include an ablation study with all memory units removed, indicated as "w/o MRA". The results show that removing any memory significantly degrades the prediction performance. This indicates that they all contribute uniquely and critically to the overall model's effectiveness.

F. Model Robustness

To examine the robustness of our proposed model, we evaluate its performance under noise perturbations. Specifically, we introduce Gaussian noise into the spatio-temporal data, with a mean value of 0. We test three levels of noise standard deviation: 1%, 5%, and 10% of the mean value of the ST data. By varying the noise levels, we can assess the model's robustness and its ability to handle different degrees of data corruption effectively. Table IV and Table V illustrate the performance of our model and baseline methods under noise perturbations.

As we can observe from Table VII, models such as STGCN, DCRNN, MTGNN, AGCRN, GTS, and STEP experienced significant performance degradation under noise perturbations.

TABLE V: Short-term prediction performance with different noise levels on TaxiBJ (grid) in terms of RMSE and MAE. We use the average prediction errors over all prediction steps.

Model	1%		5%		10%	
	RMSE	MAE	RMSE	MA	RMSE	MAE
STResNet	39.38	27.78	71.70	48.85	139.8	92.1
ACFM	30.90	19.89	31.388	20.28	32.66	21.24
STNorm	31.36	19.16	34.48	21.99	42.18	28.09
STGCP	27.31	17.54	27.31	17.54	27.31	17.54
MC-STL	38.53	28.54	38.58	28.57	38.71	28.64
MAU	71.07	46.38	71.05	46.37	71.02	46.35
MIM	76.29	48.01	76.31	48.02	76.38	48.07
SimVP	35.61	21.70	36.87	23.12	40.51	26.79
TAU	26.59	15.96	27.80	17.06	31.27	19.91
PromptST	27.42	16.12	27.43	16.14	27.45	16.16
UniST	23.87	14.59	23.89	14.60	23.90	14.61
STID	25.59	16.41	26.48	17.48	29.09	20.22
PatchTST	55.37	33.20	86.38	61.98	137.5	105.7
iTransformer	44.33	26.59	60.69	45.39	102.0	80.6
Time-LLM	33.79	18.79	33.99	20.42	35.25	23.58
UniFlow	20.83	12.56	22.76	14.44	22.97	14.52

Similarly, multivariate time series prediction approaches, including PatchTST, iTransformer, and Time-LLM, also exhibited substantial performance reductions. In contrast, our proposed model, UniFlow, maintained nearly steady performance under noise perturbations, demonstrating its robustness. Table VI corroborates these findings, showing that UniFlow maintains good performance even under noise perturbations, while the baseline models experience significant degradation. Additionally, UniST also achieves robust performance under noise perturbations, highlighting the advantage of robustness that a universal model can provide.

To further elaborate on the advantages of foundation models, it is important to highlight their capability to achieve outstanding performance and robustness through extensive training on vast amounts of data. Our proposed foundation model, UniFLow, leverages diverse and large-scale datasets to learn comprehensive spatio-temporal patterns. This allows UniST to generalize well across various tasks and handle diverse types of data perturbations effectively.

G. Case Study

To gain a deeper understanding of our designed ST-KRA process, we conducted case studies to investigate how different input spatio-temporal data interact with the memory. Specifically, for various spatio-temporal inputs, we calculated how each input retrieves information from the memory by recording the aggregation weights for each memory unit. We then compared the similarity between these weights using cosine similarity. This approach allowed us to investigate which scenarios exhibit similar behavior during the prompt retrieval process.

Figure 7 illustrates two pairs of regions in Beijing and Shanghai, China. The numerical values above the dashed red and blue lines represent the similarity scores between the retrieval weights. The results are highly interpretable. Region A in Beijing encompasses central areas such as Changan Street and Qianmen Road, while Region A in Shanghai also
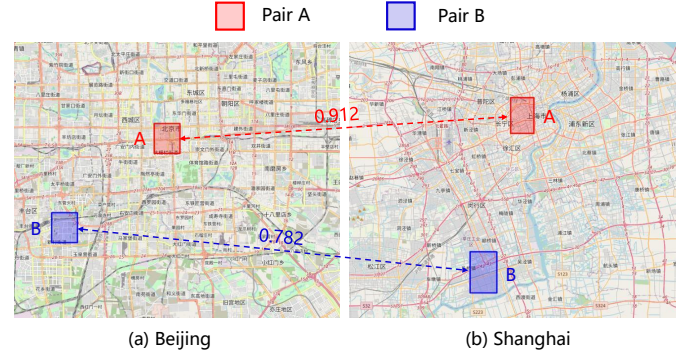

Fig. 7: Case studies of the spatio-temporal memory retrieval augmentation. We select two pairs in Beijing and Shanghai that have similar retrieved prompts.

represents the city's central part. Both regions exhibit high similarity scores, indicating similar spatio-temporal dynamics typically observed in urban centers. Region B in Beijing and Region B in Shanghai are both relatively remote residential areas. The similarity in prompt retrieval for these regions suggests that they share comparable spatio-temporal characteristics, likely due to similar patterns of residential activity and lower traffic density compared to the central regions.

VI. CONCLUSION

In this paper, we develop a foundation model for unified urban spatio-temporal flow prediction. By utilizing the transformer architecture, we achieved uniform sequential modeling of both grid-based and graph-based flow data. We also design spatio-temporal memory retrieval augmentation to generate useful prompts, which facilitate effective cross-learning of different data. Experimentally, the proposed one-for-all UniFlow model achieves state-of-the-art performance across various datasets. Further analysis reveals the efficacy of the memory retrieval augmentation. UniFlow demonstrates the potential to unify diverse types of spatio-temporal data to build a foundation model. We believe it inspires the development of more one-for-all foundation models in urban contexts. In the future, we will explore additional enhancements, such as expanding the model's applicability to more complex spatio-temporal scenarios, such as meteorology and earth sciences.

REFERENCES

[1] Y. Zheng, L. Capra, O. Wolfson, and H. Yang, "Urban computing: concepts, methodologies, and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 5, no. 3, pp. 1–55, 2014.

[2] 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.

[3] J. Zhang, Y. Zheng, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 31, no. 1, 2017.

[4] Z. Zhou, J. Ding, Y. Liu, D. Jin, and Y. Li, "Towards generative modeling of urban flow through knowledge-enhanced denoising diffusion," in *Proceedings of the 31st ACM International Conference on Advances in Geographic Information Systems*, 2023, pp. 1–12.

[5] Y. Li, Y. Yuan, J. Ding, and D. Jin, "Learning the complexity of urban mobility with deep generative collaboration network," 2023.

[6] S. Wang, J. Cao, and S. Y. Philip, "Deep learning for spatio-temporal data mining: A survey," *IEEE transactions on knowledge and data engineering*, vol. 34, no. 8, pp. 3681–3700, 2020.

[7] G. Jin, Y. Liang, Y. Fang, Z. Shao, J. Huang, J. Zhang, and Y. Zheng, "Spatio-temporal graph neural networks for predictive learning in urban computing: A survey," *IEEE Transactions on Knowledge and Data Engineering*, 2023.

[8] M. Jin, Q. Wen, Y. Liang, C. Zhang, S. Xue, X. Wang, J. Zhang, Y. Wang, H. Chen, X. Li *et al.*, "Large models for time series and spatio-temporal data: A survey and outlook," *arXiv preprint arXiv:2310.10196*, 2023.

[9] 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.

[10] L. Bai, L. Yao, C. Li, X. Wang, and C. Wang, "Adaptive graph convolutional recurrent network for traffic forecasting," *Advances in neural information processing systems*, vol. 33, pp. 17 804–17 815, 2020.

[11] X. Geng, Y. Li, L. Wang, L. Zhang, Q. Yang, J. Ye, and Y. Liu, "Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, no. 01, 2019, pp. 3656–3663.

[12] C. Yu, X. Ma, J. Ren, H. Zhao, and S. Yi, "Spatio-temporal graph transformer networks for pedestrian trajectory prediction," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XII 16*. Springer, 2020, pp. 507–523.

[13] W. Chen, F. Wang, and H. Sun, "S2tnet: Spatio-temporal transformer networks for trajectory prediction in autonomous driving," in *Asian Conference on Machine Learning*. PMLR, 2021, pp. 454–469.

[14] Y. Liang, K. Ouyang, L. Jing, S. Ruan, Y. Liu, J. Zhang, D. S. Rosenblum, and Y. Zheng, "Urbanfm: Inferring fine-grained urban flows," in *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2019, pp. 3132–3142.

[15] L. Liu, R. Zhang, J. Peng, G. Li, B. Du, and L. Lin, "Attentive crowd flow machines," in *Proceedings of the 26th ACM international conference on Multimedia*, 2018, pp. 1553–1561.

[16] Z. Pan, Y. Liang, W. Wang, Y. Yu, Y. Zheng, and J. Zhang, "Urban traffic prediction from spatio-temporal data using deep meta learning," in *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2019, pp. 1720–1730.

[17] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, "Graph wavenet for deep spatial-temporal graph modeling," in *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, 2019, pp. 1907–1913.

[18] Y. Liang, S. Ke, J. Zhang, X. Yi, and Y. Zheng, "Geoman: Multi-level attention networks for geo-sensory time series prediction." in *IJCAI*, vol. 2018, 2018, pp. 3428–3434.

[19] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel *et al.*, "Retrieval-augmented generation for knowledge-intensive nlp tasks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 9459–9474, 2020.

[20] Y. Yuan, C. Shao, J. Ding, D. Jin, and Y. Li, "Spatio-temporal few-shot learning via diffusive neural network generation," in *The Twelfth International Conference on Learning Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=QyFm3D3Tzi

[21] Y. Yuan, J. Ding, C. Shao, D. Jin, and Y. Li, "Spatio-temporal diffusion point processes," in *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2023, pp. 3173–3184.

[22] L. Zhao, Y. Song, C. Zhang, Y. Liu, P. Wang, T. Lin, M. Deng, and H. Li, "T-gcn: A temporal graph convolutional network for traffic prediction," *IEEE transactions on intelligent transportation systems*, vol. 21, no. 9, pp. 3848–3858, 2019.

[23] Y. Jin, K. Chen, and Q. Yang, "Transferable graph structure learning for graph-based traffic forecasting across cities," in *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2023, pp. 1032–1043.

[24] Y. Wang, M. Long, J. Wang, Z. Gao, and P. S. Yu, "Predrnn: Recurrent neural networks for predictive learning using spatiotemporal lstms," *Advances in neural information processing systems*, vol. 30, 2017.

[25] Y. Wang, Z. Gao, M. Long, J. Wang, and S. Y. Philip, "Predrnn++: Towards a resolution of the deep-in-time dilemma in spatiotemporal predictive learning," in *International Conference on Machine Learning*. PMLR, 2018, pp. 5123–5132.

[26] Z. Lin, M. Li, Z. Zheng, Y. Cheng, and C. Yuan, "Self-attention convlstm for spatiotemporal prediction," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 07, 2020, pp. 11 531–11 538.

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

[28] J. Jiang, C. Han, W. X. Zhao, and J. Wang, "Pdformer: Propagation delay-aware dynamic long-range transformer for traffic flow prediction," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 37, no. 4, 2023, pp. 4365–4373.

[29] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale *et al.*, "Llama 2: Open foundation and fine-tuned chat models," *arXiv preprint arXiv:2307.09288*, 2023.

[30] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[31] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 10 684–10 695.

[32] Y. Bai, X. Geng, K. Mangalam, A. Bar, A. Yuille, T. Darrell, J. Malik, and A. A. Efros, "Sequential modeling enables scalable learning for large vision models," *arXiv preprint arXiv:2312.00785*, 2023.

[33] J. Feng, Y. Du, T. Liu, S. Guo, Y. Lin, and Y. Li, "Citygpt: Empowering urban spatial cognition of large language models," *arXiv preprint arXiv:2406.13948*, 2024.

[34] F. Xu, J. Zhang, C. Gao, J. Feng, and Y. Li, "Urban generative intelligence (ugi): A foundational platform for agents in embodied city environment," *arXiv preprint arXiv:2312.11813*, 2023.

[35] J. Feng, J. Zhang, J. Yan, X. Zhang, T. Ouyang, T. Liu, Y. Du, S. Guo, and Y. Li, "Citybench: Evaluating the capabilities of large language model as world model," *arXiv preprint arXiv:2406.13945*, 2024.

[36] Z. Li, L. Xia, J. Tang, Y. Xu, L. Shi, L. Xia, D. Yin, and C. Huang, "Urbangpt: Spatio-temporal large language models," 2024.

[37] D. Cao, F. Jia, S. O. Arik, T. Pfister, Y. Zheng, W. Ye, and Y. Liu, "Tempo: Prompt-based generative pre-trained transformer for time series forecasting," *arXiv preprint arXiv:2310.04948*, 2023.

[38] M. Jin, S. Wang, L. Ma, Z. Chu, J. Y. Zhang, X. Shi, P.-Y. Chen, Y. Liang, Y.-F. Li, S. Pan *et al.*, "Time-llm: Time series forecasting by reprogramming large language models," in *The Twelfth International Conference on Learning Representations*.

[39] T. Zhou, P. Niu, X. Wang, L. Sun, and R. Jin, "One fits all: Power general time series analysis by pretrained LM," in *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. [Online]. Available: https://openreview.net/forum?id=gMS6FVZvmF

[40] Y. Yuan, J. Ding, J. Feng, D. Jin, and Y. Li, "Unist: a prompt-empowered universal model for urban spatio-temporal prediction," in *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2024, pp. 4095–4106.

[41] K. Jin, J. Wi, E. Lee, S. Kang, S. Kim, and Y. Kim, "Trafficbert: Pre-trained model with large-scale data for long-range traffic flow forecasting," *Expert Systems with Applications*, vol. 186, p. 115738, 2021.

[42] Z. Li, L. Xia, Y. Xu, and C. Huang, "Generative pre-training of spatio-temporal graph neural networks," in *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. [Online]. Available: https://openreview.net/forum?id=nMH5cUaSj8

[43] J. Ji, J. Wang, C. Huang, J. Wu, B. Xu, Z. Wu, J. Junbo, and Y. Zheng, "Spatio-temporal self-supervised learning for traffic flow prediction," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 4, pp. 4356–4364, 2023.

[44] Z. Shao, Z. Zhang, F. Wang, and Y. Xu, "Pre-training enhanced spatial-temporal graph neural network for multivariate time series forecasting," in *KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 14 - 18, 2022*. ACM, 2022, pp. 1567–1577.

[45] L. Liu, S. Yu, R. Wang, Z. Ma, and Y. Shen, "How can large language models understand spatial-temporal data?" 2024.

[46] Z. Zhang, X. Zhao, Q. Liu, C. Zhang, Q. Ma, W. Wang, H. Zhao, Y. Wang, and Z. Liu, "Promptst: Prompt-enhanced spatio-temporal multi-attribute prediction," in *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, 2023, pp. 3195–3205.

[47] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, and H. Wang, "Retrieval-augmented generation for large language models: A survey," *arXiv preprint arXiv:2312.10997*, 2023.

[48] X. Cheng, D. Luo, X. Chen, L. Liu, D. Zhao, and R. Yan, "Lift yourself up: Retrieval-augmented text generation with self-memory," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[49] J. Lin, R. Shan, C. Zhu, K. Du, B. Chen, S. Quan, R. Tang, Y. Yu, and W. Zhang, "Rella: Retrieval-enhanced large language models for lifelong sequential behavior comprehension in recommendation," in *Proceedings of the ACM on Web Conference 2024*, 2024, pp. 3497–3508.

[50] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam, "A time series is worth 64 words: Long-term forecasting with transformers," *arXiv preprint arXiv:2211.14730*, 2022.

[51] G. Karypis and V. Kumar, "Metis: A software package for partitioning unstructured graphs, partitioning meshes, and computing fill-reducing orderings of sparse matrices," 1997.

[52] Z. Shao, Z. Zhang, F. Wang, W. Wei, and Y. Xu, "Spatial-temporal identity: A simple yet effective baseline for multivariate time series forecasting," in *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, 2022, pp. 4454–4458.

[53] J. Deng, X. Chen, R. Jiang, X. Song, and I. W. Tsang, "St-norm: Spatial and temporal normalization for multi-variate time series forecasting," in *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, 2021, pp. 269–278.

[54] L. Zhao, M. Gao, and Z. Wang, "St-gsp: Spatial-temporal global semantic representation learning for urban flow prediction," in *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, 2022, pp. 1443–1451.

[55] X. Zhang, Y. Gong, X. Zhang, X. Wu, C. Zhang, and X. Dong, "Mask-and contrast-enhanced spatio-temporal learning for urban flow prediction," in *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, 2023, pp. 3298–3307.

[56] Z. Chang, X. Zhang, S. Wang, S. Ma, Y. Ye, X. Xinguang, and W. Gao, "Mau: A motion-aware unit for video prediction and beyond," *Advances in Neural Information Processing Systems*, vol. 34, pp. 26 950–26 962, 2021.

[57] Y. Wang, J. Zhang, H. Zhu, M. Long, J. Wang, and P. S. Yu, "Memory in memory: A predictive neural network for learning higher-order non-stationarity from spatiotemporal dynamics," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 9154–9162.

[58] Z. Gao, C. Tan, L. Wu, and S. Z. Li, "Simvp: Simpler yet better video prediction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 3170–3180.

[59] C. Tan, Z. Gao, L. Wu, Y. Xu, J. Xia, S. Li, and S. Z. Li, "Temporal attention unit: Towards efficient spatiotemporal predictive learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 18 770–18 782.

[60] Z. Wu, S. Pan, G. Long, J. Jiang, X. Chang, and C. Zhang, "Connecting the dots: Multivariate time series forecasting with graph neural networks," in *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, 2020, pp. 753–763.

[61] C. Shang and J. Chen, "Discrete graph structure learning for forecasting multiple time series," in *Proceedings of International Conference on Learning Representations*, 2021.

[62] Z. Shao, Z. Zhang, F. Wang, and Y. Xu, "Pre-training enhanced spatial-temporal graph neural network for multivariate time series forecasting," in *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, 2022, pp. 1567–1577.

[63] Y. Liu, T. Hu, H. Zhang, H. Wu, S. Wang, L. Ma, and M. Long, "itransformer: Inverted transformers are effective for time series forecasting," *arXiv preprint arXiv:2310.06625*, 2023.

TABLE VI: The basic information of grid-based spatio-temporal data.

Dataset	City	Type	Temporal Period	Spatial partition	Interval	Mean	Std
TaxiBJ	Beijing	Taxi flow	2013/06/01 - 2013/10/30	32×32	Half an hour	97.543	122.174
TaxiNYC	New York City	Taxi flow	2015/01/01 - 2015/03/01	10×20	Half an hour	38.801	103.924
FlowSH	Shanghai	Mobility flow	2016/04/25 - 2016/05/01	20×20	15min	31.935	137.926
CrowdBJ	Beijing	Crowd flow	2021/10/25 - 2021/11/21	28×24	One hour	0.367	0.411
CrowdNJ	Nanjing	Crowd flow	2021/02/02 - 2021/03/01	20×28	One hour	0.872	1.345
PopSH	Shanghai	Dynamic population	2014/08/01 - 2014/08/28	32×28	One hour	0.175	0.212

TABLE VII: The basic information of Graph-based spatio-temporal data.

Dataset	City	Type	Temporal Period	Interval	#Nodes	#Edges	Mean	Std
TrafficBJ	Beijing	Traffic speed	2022/03/05 - 2022/04/05	15min	13675	24444	6.837	3.412
TrafficSH	Shanghai	Traffic speed	2022/01/27 - 2022/02/27	15min	21099	39065	7.815	4.044
TrafficNJ	Nanjing	Traffic speed	2022/03/05 - 2022/04/05	15min	13419	25100	6.699	4.253

APPENDIX

A. Datasets

1) Basic Information: We provide detailed information about the datasets used in our study, which fall into two categories: grid-based data and graph-based data. Table VI presents the basic information and statistics for the grid-based datasets, while Table VII offers similar details for the graph-based datasets. It is worth mentioning that we did not utilize popular spatio-temporal graph datasets such as PEMS-03, PEMS-07, and PEMS-Bay. The primary reason is the lack of corresponding grid-based spatio-temporal data for these cities. We have observed that graph-based data benefit more from grid-based data when both come from the same city.

2) Data Preprocessing: For each dataset, we divided it into three distinct segments: the initial 60% was allocated for training, the subsequent 20% for validation, and the remaining 20% for testing. All datasets were normalized to a scale of $[0, 1]$. The prediction results presented are the denormalized values.

B. Baselines

1) Basic information:

- **HA**: History Average uses the average of past data from corresponding periods to forecast future values.
- **ARIMA**: The Auto-Regressive Integrated Moving Average model is a widely-used statistical method for time series forecasting. It effectively analyzes and predicts time series data collected at regular intervals.
- **MIM** [57]: This model leverages differential information between adjacent recurrent states to handle non-stationary properties. Multiple stacked MIM blocks enable the modeling of higher-order non-stationarity.
- **MAU** [56]: The Motion-aware Unit broadens the temporal receptive fields of prediction units to capture inter-frame motion correlations. It includes an attention module and a fusion module for video prediction.
- **SimVP** [58]:A simple yet effective video prediction model built entirely on convolutional neural networks and using MSE loss, serving as a solid baseline in video prediction tasks.
- **TAU** [59]: The Temporal Attention Unit decomposes temporal attention into intra-frame and inter-frame components, introducing differential divergence regularization to account for inter-frame variations.
- **STResNet** [3]: It utilizes residual neural networks to capture temporal closeness, periodicity, and trends in the data.
- **ACFM** [15]: Attentive Crowd Flow Machine model predicts crowd dynamics by employing an attention mechanism to adaptively aggregate sequential and periodic patterns.
- **STGSP** [54]: This model emphasizes the importance of global and positional information in the temporal dimension for spatio-temporal prediction. It uses a semantic flow encoder for temporal positional signals and an attention mechanism for multi-scale temporal dependencies.
- **MC-STL** [55]: It employs mask-enhanced contrastive learning to effectively capture relationships in the spatio-temporal dimension.
- **STNorm** [53]: It proposes two types of normalization modules: spatial normalization and temporal normalization, which separately handle high-frequency and local components.
- **STID** [52]: This MLP-based spatio-temporal prediction model identifies indistinguishability in spatio-temporal dimensions, demonstrating efficiency and effectiveness in its design.
- **PromptST** [46]: A state-of-the-art pre-training and prompt-tuning approach designed specifically for spatio-temporal prediction.
- **UniST** [40]: A universal model urban spatio-temporal prediction with grid-based data. It leverages diverse spatio-temporal masking strategies for pre-training and spatio-temporal knowledge-based prompt fine-tuning.
- **STGCN** [2]: Spatio-Temporal Graph Convolutional Network is a deep learning framework for traffic forecasting that leverages both spatial and temporal dependencies. It combines graph convolutional layers with convolutional sequence learning layers to model multi-scale traffic networks.
- **GWN** [17]: Graph WaveNet is a CNN-based method designed to address the limitations of existing spatial-temporal graph modeling approaches. It proposed a self-adaptive adjacency matrix and stacked dilated causal convolutions to capture temporal dependencies effectively.
- **MTGNN** [60]: MTGNN is a framework specifically designed for multivariate time series data. It automatically

extracts uni-directed relations among variables through a graph learning module and integrates external knowledge like variable attributes.

- **GTS** [61]: GTS is a method that learns the structure of a graph simultaneously with a Graph Neural Network (GNN) for forecasting multiple time series. It parameterizes the graph structure through a neural network, enabling discrete graphs to be sampled differently, and optimizes the mean performance over the graph distribution.
- **DCRNN** [9]: Diffusion Convolutional Recurrent Neural Network is a deep learning model for spatiotemporal forecasting. It models traffic flow as a diffusion process on a directed graph, capturing spatial dependencies through bidirectional random walks and temporal dependencies using an encoder-decoder architecture with scheduled sampling.
- **STEP** [62]:Spatial-temporal Graph Neural Network Enhanced by Pre-training is a framework that incorporates a scalable time series pre-training model to improve multivariate time series forecasting.
- **AGCRN** [10]: AGCRN enhances Graph Convolutional Networks (GCNs) with two adaptive modules: Node Adaptive Parameter Learning (NAPL) and Data Adaptive Graph Generation (DAGG). It captures fine-grained spatial and temporal correlations in traffic series without relying on pre-defined graphs.
- **PatchTST** [50]: It introduces patching and self-supervised learning for multivariate time series forecasting, segmenting the time series into patches to capture long-term correlations and processing different channels independently with a shared network.
- **iTransformer** [63]: This state-of-the-art model for multivariate time series employs attention and feed-forward operations on an inverted dimension, focusing on multivariate correlations.
- **Time-LLM** [38]: TIME-LLM is the state-of-the-art model to apply large language models to time series forecasting. It utilizes a reprogramming framework designed to repurpose LLMs for general time series forecasting, all while keeping the backbone language models intact.

2) Baseline implementation: **Dealing with very large graph.** Assuming the Graph-based ST data is $X \in \mathbb{R}^{T \times N \times C}$, where N corresponds to the total node number, T represents the temporal period length and C is the number of variables. The overly large N (over 10,000) leads to an excessively high computational complexity for training models. To better train baseline models, we evenly split the spatial graph into multiple subgraphs. We convert X to $X' \in \mathbb{R}^{T \times n \times m \times C}$, where m is the number of subgraphs and n is the node number of each subgraph. When training the model, randomly select a subgraph from m subgraphs as the training sample. The input of the model $x_{in} \in \mathbb{R}^{B \times h \times n \times C}$, and the target $x_{tar} \in \mathbb{R}^{B \times t \times n \times C}$. B is batch size, h is historical input steps, and t is prediction horizons. The information of subgraphs for three datasets is shown in Table VIII.

One-for-all models. Among the selected baselines, PatchTST and UniST are channel-independent (variate-independent) models. To ensure a fair comparison, we train

TABLE VIII: Information of subgraphs for different datasets

Dataset	N	n	m
TrafficBJ	13675	549	25
TrafficSH	21099	541	39
TrafficNJ	13419	497	27

TABLE IX: Performance comparison of short-term prediction on the remaining datasets in terms of MAE and RMSE. We use the average prediction errors over all prediction steps.

	Grid Data				Graph Data	
	CrowdBJ		PopSH		TrafficNJ	
Model	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**
HA	0.343	0.232	0.165	0.1003	3.10	2.04
ARIMA	0.404	0.236	0.197	0.112	4.83	1.96
STResNet	0.751	0.546	0.138	0.102	-	-
ACFM	0.200	0.141	0.078	0.0549	-	-
STNorm	0.198	0.132	0.065	0.042	-	-
STGSP	0.229	0.157	0.057	0.040	-	-
MC-STL	0.311	0.235	0.100	0.076	-	-
MAU	0.256	0.166	0.125	0.081	-	-
MIM	0.298	0.214	0.126	0.079	-	-
SimVP	0.213	0.148	0.069	0.048	-	-
TAU	0.196	0.135	0.063	0.044	-	-
PromptST	0.171	0.099	0.069	0.0431	-	-
UniST	0.172	0.106	0.055	0.036	-	-
STGCN	-	-	-	-	2.38	1.71
DCRNN	-	-	-	-	2.15	1.50
GWN	-	-	-	-	2.23	1.58
MTGNN	-	-	-	-	2.08	1.44
AGCRN	-	-	-	-	2.54	1.86
GTS	-	-	-	-	2.76	2.14
STEP	-	-	-	-	2.19	1.55
STID	0.262	0.2037	0.055	0.038	2.45	1.64
PatchTST	0.291	0.189	0.108	0.062	2.83	1.92
PatchTST(one-for-all)	0.279	0.176	0.148	0.089	3.45	2.302
iTransformer	0.249	0.154	0.0722	0.0445	2.70	1.79
Time-LLM	0.195	0.115	0.102	0.060	2.85	1.91
UniFlow	0.148	0.089	0.054	0.035	2.36	1.56

a one-for-all model for both.

C. Training and Evaluation

1) Training: We train the overall model in an end-to-end manner, aiming to minimize the mean squared error (MSE) loss between the predicted spatio-temporal data and the ground truth data. Specifically, we calculate the MSE only on the prediction horizon, even though the decoder outputs the complete spatio-temporal data. The loss function is formulated as follows:

$$\mathcal{L} = \frac{1}{N} \frac{1}{P} \sum_{i=1}^{N} \sum_{p=t_0+1}^{t_0+P} ||\hat{Y}_{i,p} - Y_{i,p}||^2$$

where $\hat{Y}_{i,p}$ is the predicted value, $Y_{i,p}$ is the ground truth, N is the number of samples in the dataset, and P is the prediction horizon. We use the Adam optimizer to minimize the loss function and update the model parameters.

We train the from scratch using all datasets simultaneously and directly evaluate its performance on the test sets of each dataset. This approach does not involve separate pre-training and fine-tuning stages. Instead, the model is trained end-to-end to perform the prediction task across all datasets. We

TABLE X: Performance comparison of long-term prediction on the remaining datasets in terms of MAE and RMSE. We use the average prediction errors over all prediction steps.

| | Grid Data | | Graph Data | | | |
| | TaxiNYC | | TrafficSH | | TrafficNJ | |
Model	RMSE	MAE	RMSE	MAE	RMSE	MAE
HA	60.73	21.14	2.43	1.64	3.19	2.22
ARIMA	86.61	27.68	2.05	3.89	4.11	2.83
STResNet	36.51	18.42	-	-	-	-
ACFM	30.24	13.85	-	-	-	-
STNorm	29.52	11.55	-	-	-	-
STGSP	25.59	11.06	-	-	-	-
MC-STL	37.85	24.45	-	-	-	-
MAU	30.55	10.56	-	-	-	-
MIM	56.47	57.01	-	-	-	-
SimVP	23.84	10.35	-	-	-	-
TAU	23.59	10.11	-	-	-	-
PromptST	29.11	10.02	-	-	-	-
UniST	22.69	7.85	-	-	-	-
STGCN	-	-	3.41	2.72	3.61	2.96
DCRNN	-	-	3.26	2.63	3.29	2.57
GWN	-	-	3.61	2.96	3.51	2.76
MTGNN	-	-	3.39	2.70	3.47	2.64
AGCRN	-	-	2.57	1.90	2.61	1.91
GTS	-	-	3.12	2.46	3.61	2.83
STEP	-	-	2.62	1.94	2.76	2.04
STID	22.04	9.59	2.24	1.56	2.52	1.72
PatchTST	31.79	10.43	2.19	1.38	2.56	1.69
iTransformer	32.83	11.09	2.18	1.47	2.56	1.75
Time-LLM	29.56	9.72	2.10	1.37	2.57	1.71
UniFlow	21.1	6.49	2.09	1.43	2.44	1.62

TABLE XI: The information of training time and inference time.

Model	train time all	inference time
STGCN	17min	2s
DCRNN	77min	8s
GWN	16min	1s
MTGNN	14min	0.8s
AGCRN	21min	2s
GTS	126min	17s
STEP	177min	27s
STResNet	5.7min	0.6s
ACTM	56min	0.9s
STNorm	46min	5s
STGCP	8min	4s
MC-STL	31min	7s
MAU	82min	13s
MIM	84min	14s
TAU	22min	6s
PromptST	45min	9s
UniST	5h	19s
STID	10min	4s
PatchTST	33min	5s
iTransformer	23min	6s
Time-LLM	382min	5min
UniFlow	6h (multiple datasets)	10s

the number of samples in the dataset, and P is the prediction horizon.

D. Additional Results

1) Prediction Performance: Table IX reports short-term prediction results of the remaining datasets, and Table X reports long-term prediction results of the remaining datasets.

2) Efficiency Evaluation: Table XI reports model efficiency in terms of overall training time and inference time. Although the training time of UniFlow is significantly longer than the baseline models due to its inclusion of all datasets, it is important to note that training separate models for each dataset and summing the total training time results in comparable times between UniFlow and the baseline methods. Moreover, UniFlow achieves the best performance across all datasets with a single, unified model. This demonstrates the efficiency and effectiveness of UniFlow in delivering superior performance without the need for multiple specialized models.

use multiple datasets during training, and for each batch, we randomly sample from one of the datasets, allowing the model to jointly train across different data types. We use an A100 GPU for our experiments. The input data dimensions vary across different datasets, and these dimensions are provided in Appendix Table VI and Table VII. To adjust the batch sizes for different datasets, we employ the following strategy: $B = \frac{N}{K}$, where K is the pre-determined number of iterations for each dataset and N is the total number of training samples. We first set the batch size B for the dataset with the largest spatial locations, ensuring optimal utilization of GPU memory. From this, we calculate K and subsequently adjust the batch sizes for the remaining datasets accordingly. This approach allows for similar training iterations per epoch across different datasets while preventing GPU memory overflow.

2) Evaluation Metrics: To comprehensively evaluate the performance of our model, we employ two widely-used metrics: Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE). The formulas for RMSE and MAE are as follows:

$$\text{RMSE} = \frac{1}{N}\frac{1}{P}\sum_{i=1}^{N}\sum_{p=t_0+1}^{t_0+P}(\hat{Y}_{i,p} - Y_{i,p})^2 \quad (5)$$

$$\text{MAE} = \frac{1}{N}\frac{1}{P}\sum_{i=1}^{N}\sum_{p=t_0+1}^{t_0+P}|\hat{Y}_{i,p} - Y_{i,p}| \quad (6)$$

where $\hat{Y}_{i,p}$ is the predicted value, $Y_{i,p}$ is the actual value, N is