TransLLM: A Unified Multi-Task Large Language Model for Urban Transportation via Learnable Prompting

Jiaming Leng[1], Yunying Bi[2], Chuan Qin[3,4], Zhenya Huang[1,5], Bing Yin[6], Haojie Ren[1], Yanyong Zhang[2,7,*], Chao Wang [2,5,*],

[1]School of Computer Science and Technology, University of Science and Technology of China

[2]School of Artificial Intelligence and Data Science, University of Science and Technology of China

[3]Computer Network Information Center, Chinese Academy of Sciences

[4]University of Chinese Academy of Sciences [5]State Key Laboratory of Cognitive Intelligence

[6]iFlytek Co. Ltd [7]Institute of Artificial Intelligence, Hefei Comprehensive National Science Center

{lengjm, biyunying, rhj}@mail.ustc.edu.cn, chuanqin0426@gmail.com, bingyin@iflytek.com

{huangzhy, yanyongz, wangchaoai}@ustc.edu.cn

Abstract

Urban transportation systems require precise modeling of dynamic spatiotemporal patterns across diverse tasks, such as traffic forecasting, electric vehicle (EV) charging demand prediction, and taxi dispatch. Existing approaches suffer from two key limitations: traditional deep learning models are task-specific and lack generalization capabilities, whereas Large Language Models (LLMs) struggle with structured spatiotemporal data and numerical reasoning. To bridge this gap, we propose TransLLM, a unified multi-task framework that synergizes spatiotemporal encoding with LLM reasoning through learnable prompt composition. To enable LLMs to perceive complex graph dependencies, we design a noise-augmented spatiotemporal encoder that projects structured signals into the LLM's embedding space. Furthermore, to overcome the rigidity of fixed prompt templates in heterogeneous traffic scenarios, we introduce an instance-level prompt routing mechanism trained via reinforcement learning. The framework operates by encoding spatiotemporal patterns into contextual representations, dynamically composing personalized prompts to guide LLM reasoning, and projecting the resulting representations through specialized output layers to generate task-specific predictions. Experiments on seven datasets and three tasks demonstrate that TransLLM outperforms many baselines, showing superior adaptability in both supervised and zero-shot settings with excellent generalization and robustness. Our code and data are available at https://github.com/lengjiaming/TransLLM.

1 Introduction

Urban transportation systems are the lifelines of modern cities, yet they are increasingly strained by growing travel demand, dynamic spatiotemporal patterns, and the pressing need for efficient resource management. Core tasks include traffic flow forecasting for congestion mitigation (Li et al., 2018), EV charging demand prediction for infrastructure planning (Yi et al., 2022), and taxi dispatch optimization for balancing supply and demand across regions (Yao et al., 2018). Although these tasks differ in objectives and outputs, they are coupled by shared mobility patterns and spatiotemporal dependencies (Zhang et al., 2017). This coupling motivates a unified framework that can handle heterogeneous spatiotemporal inputs and reuse shared knowledge across transportation domains.

Traditional urban transportation studies were predominantly approached through handcrafted heuristics and classical statistical models (Chen et al., 2011; Kumar, 2017; Smith et al., 2002). With the advent of deep learning over the past decade, spatiotemporal neural architectures—particularly graph-based models—have emerged as the dominant paradigm for urban mobility modeling (Guo et al., 2019; Fang et al., 2021). For instance, PDG2Seq (Fan et al., 2025) characterizes periodic traffic dynamics via a dynamic graph-to-sequence formulation, explicitly aligning temporal cycles with multi-step forecasting. Despite their strong performance within specific benchmarks, these models are typically engineered for a single objective under a predefined input/target specification, which limits knowledge transfer across heterogeneous transportation tasks and hinders the effective utilization of auxiliary context (Yao et al., 2019).

Recently, large language models (LLMs) such as GPT (OpenAI, 2025) and LLaMA (Team, 2024) provide a complementary capability: they can represent, integrate, and condition on heteroge-

*Corresponding Author

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 26028–26042

neous contextual information (e.g., temporal descriptors, metadata, and domain text) through a unified natural-language interface. However, off-the-shelf LLMs are not designed for continuous, structured spatiotemporal signals (Dziri et al., 2023). Converting real-valued traffic variables into text requires discretization/tokenization, creating a tokenization mismatch that reduces numerical fidelity and weakens modeling of fine-grained spatiotemporal dependencies. Recent hybrid paradigms have been proposed to bridge this mismatch. For example, UrbanGPT (Li et al., 2024) augments LLMs with a dedicated temporal encoder, while LLMLight (Lai et al., 2025) relies on task-specific prompt templates to steer decision-making. Nevertheless, existing designs are often constrained by fixed, task-wise prompting schemes and primarily target relatively simple prediction or classification settings, leaving open the need for a unified framework that (i) aligns structured spatiotemporal representations with LLMs and (ii) enables instance-adaptive prompting for diverse transportation tasks, including planning problems such as taxi dispatch (Mao et al., 2025).

Despite recent progress, building a unified multi-task traffic modeling framework remains challenging. First, multi-task transportation scenarios involve heterogeneous and structured spatiotemporal inputs (Wang et al., 2020) (e.g., traffic flow and geographic adjacency), which are difficult to express faithfully through a natural-language interface. Moreover, real-world sensing signals are frequently noisy and partially unreliable (e.g., detection errors), which can corrupt learned dependencies and degrade robustness, motivating noise-robust spatiotemporal representation learning. Second, even within a single task, input instances often exhibit substantial intra-task variability in temporal and spatial dynamics. As a result, fixed, task-wise prompting strategies are too coarse-grained to accommodate sample-level differences induced by regions, time periods, or contextual shifts. Third, a unified framework must support diverse task objectives and output formats while mitigating negative transfer and task interference, which complicates both generalization and robustness.

To align structured spatiotemporal signals with LLMs, we propose TransLLM, a unified multi-task framework that couples a noise-augmented spatiotemporal dependency encoder with LLM-based prompting. The encoder maps traffic observations into compact embeddings that are injected into prompts as contextual tokens. To handle instance-level variability, TransLLM uses a learnable prompt-routing mechanism to select and compose prompts per instance, avoiding fixed task-wise templates. For heterogeneous task objectives and output formats, we do not rely on the LLM to generate predictions directly; instead, task-specific output heads project LLM-conditioned representations to task outputs. Overall, TransLLM follows an effective pipeline: encoding spatiotemporal context, routing instance-adaptive prompts, and decoding via multi-task heads. Our key contributions are:

- We propose TransLLM, a unified multi-task framework that integrates spatiotemporal encoding with LLM-based prompting for transportation forecasting and optimization.
- We develop a lightweight noise-augmented spatiotemporal encoder with dilated temporal convolutions and dual-adjacency graph attention for robust dependency modeling.
- We design a prompt-routing method trained with reinforcement learning to adapt prompts to sample characteristics.
- TransLLM provides a unified modeling approach across task types, from regression to planning, and outperforms many baselines on seven datasets over three tasks.

2 Related Work

GNN-based Spatiotemporal Prediction Models. GNN-based spatiotemporal prediction networks typically combine temporal and spatial modeling components. Early works like STGCN (Yu et al., 2018) and ASTGCN (Guo et al., 2019) use a "sandwich" architecture to capture spatiotemporal dependencies. More recent approaches, such as SHARE (Zhang et al., 2020), incorporate dynamic node correlations via contextual graph convolutions, while DyHSL (Zhao et al., 2023) models non-pairwise dependencies with hyperedges. On the temporal side, PDG2seq (Fan et al., 2025) decomposes time into daily and weekly components using GRU, and STGODE (Fang et al., 2021) employs dilated convolutions for long-range temporal modeling. Despite these advancements, most models are scenario-specific and heavily reliant on labeled data, limiting their generalizability and scalability.

LLM and LLM-based Models for Traffic Tasks. Large language models have made significant

progress in recent years. Foundational models like GPT (OpenAI, 2025) and LLaMA (Team, 2024) provide a solid base for developing intelligent agents. Initially designed for text understanding and generation, LLMs have been applied to specialized domains such as code generation (Rozière et al., 2023), robotic control (Kim et al., 2024), recommendation systems (Wang et al., 2025) and biomedical information extraction (Wu et al., 2025). In the transportation domain, LEAF (Zhao et al., 2025) uses an LLM to guide model selection, UrbanGPT (Li et al., 2024) enhances temporal modeling with a time-series encoder, and LLMLight (Lai et al., 2025) applies imitation learning for traffic signal selection. However, these approaches use fixed templates and fail to adapt to instance-level variations. They are also limited to simple regression or classification tasks, fail to generalize to more sophisticated planning scenarios.

3 Preliminaries

We consider two types of urban mobility tasks: (1) spatiotemporal forecasting, which includes both traffic flow and charging demand prediction, and (2) taxi dispatch optimization. For each task type, we describe the construction of the spatial graph, the representation of spatiotemporal data, and the task formulation.

3.1 Spatiotemporal Forecasting

Spatial Structure. The spatial domain is represented as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A})$, where each node $v_i \in \mathcal{V}$ corresponds to an urban monitoring site. Edges $\mathcal{E}$ capture spatial relationships through road connectivity or geographical proximity, and the adjacency matrix $\mathcal{A} \in \mathbb{R}^{N \times N}$ encodes them.

Spatiotemporal Data. Historical observations are encoded as a tensor $\mathbf{X} \in \mathbb{R}^{K \times N \times F}$, where K is the number of historical time steps, N is the number of nodes, and F denotes the feature dimension, such as traffic volume, charging demand, or other auxiliary signals.

Task Definition. Given past observations and spatial structure, the goal is to forecast future values over the next T steps for all nodes. Formally:

$$[\hat{\mathbf{X}}^{t+1}, \ldots, \hat{\mathbf{X}}^{t+T}] = \mathcal{F}\left(\mathbf{X}^{[t-K+1:t]}, \mathcal{A}\right), \quad (1)$$

where $\mathcal{F}$ denotes the spatiotemporal forecasting model to be learned.

3.2 Taxi Dispatch Optimization

Spatial Structure. For the taxi dispatch optimization task, the urban area is partitioned into grids with a side length of 3 km, ensuring that vehicles can reach neighboring grids within five minutes. The grids are also represented as nodes in the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A})$. Edges $\mathcal{E}$ connect spatially adjacent grids, and the adjacency matrix $\mathcal{A} \in \mathbb{R}^{N \times N}$ captures the neighborhood relationships.

Spatiotemporal Data. At each decision step t, we model three key variables for region v_c and its eight neighbors, forming a 3×3 neighborhood $\mathcal{N}_9(v_c)$: (1) the number of vacant taxis $\mathbf{X}^t_{v_c,v} \in \mathbb{R}^{1\times 9}$, representing the available vehicles in v_c and its neighbors ; (2) the predicted passenger demand $\mathbf{X}^t_{v_c,d} \in \mathbb{R}^{1\times 9}$, indicating expected ride requests; (3) the predicted competing taxis $\mathbf{X}^t_{v_c,c} \in \mathbb{R}^{1\times 9}$, denoting available competing vehicles. We use K-step historical sequences $\mathbf{X}^{[t-K:t-1]}_{v_c,d}$ and $\mathbf{X}^{[t-K:t-1]}_{v_c,c}$ as inputs to a spatiotemporal encoder, which generates future predictions for demand and competing taxis.

Task Definition. We model vehicle dispatching as a localized resource optimization task, where each decision reallocates vacant taxis from the central region v_c to its 3×3 neighborhood $\mathcal{N}_9(v_c)$. The dispatching decision for region v_c is defined as:

$$\mathbf{D}_c = \mathcal{F}\left(\mathbf{X}^t_{v_c,v}, \mathbf{X}^{[t-K:t-1]}_{v_c,d}, \mathbf{X}^{[t-K:t-1]}_{v_c,c}; \mathcal{A}\right), \quad (2)$$

where $\mathbf{D}_c \in \mathbb{R}^{1\times 9}$ denotes the dispatching proportions from v_c to each region in $\mathcal{N}_9(v_c)$.

4 Methodology

In this section, we detail the components of TransLLM, including the architecture of the spatiotemporal encoder (ST-Encoder), the reinforcement learning (RL)-based prompt personalization process, the multi-task output layers, and the overall training mechanism. The overall architecture of TransLLM is illustrated in Fig. 1.

4.1 Noise-Augmented Spatiotemporal Dependency Modeling

To capture complex spatiotemporal dependencies, we design a noise-augmented spatiotemporal encoder consisting of multiple spatiotemporal blocks. Each block follows a "sandwich" structure, defined as:

$$\mathbf{H} = \mathrm{TCN}\left(\mathrm{GAT}\left(\mathrm{TCN}(\mathbf{X}) + \phi(\mathbf{V}), \mathcal{A}\right)\right), \quad (3)$$

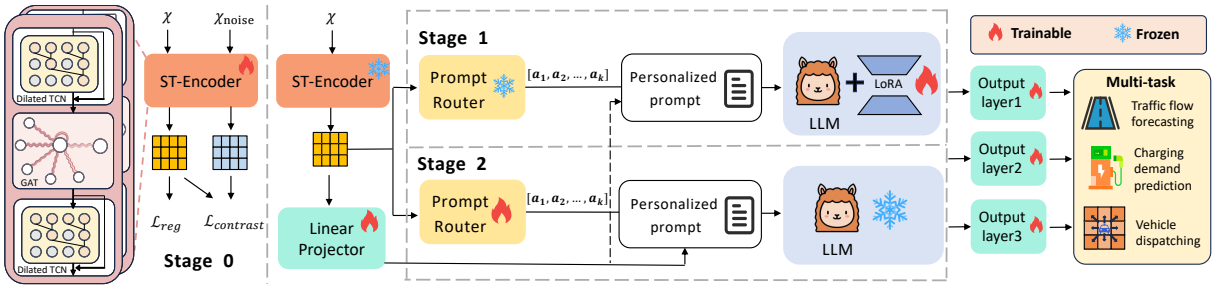

Figure 1: The overall architecture of TransLLM.

where TCN denotes a dilated Temporal Convolutional Network (Bai et al., 2018), GAT refers to a Graph Attention Network (Veličković et al., 2017), and $\phi(\cdot)$ encodes node-level meta attributes $\mathbf{V}$. We also include time-of-day and day-of-week encodings to capture periodic temporal patterns, helping the ST-Encoder model both spatial and temporal dependencies. To enhance robustness, we incorporate contrastive learning by adding Gaussian noise to the traffic flow features. Both the clean and noisy inputs are processed through the same encoder, producing two distinct embeddings that are compared using a contrastive loss, which maximizes their similarity. The ST-Encoder leverages two adjacency matrices: a spatial adjacency matrix $\mathcal{A}_{sp}$, based on road connectivity or proximity, and a semantic matrix $\mathcal{A}_{se}$, derived from Dynamic Time Warping (Berndt and Clifford, 1994) to capture functional similarity. The encoder processes these matrices to produce two representations, $\mathbf{H}_{sp}$ and $\mathbf{H}_{se} \in \mathbb{R}^{T \times N \times D}$, which are concatenated to form the final representation $\mathbf{H_f}$ for downstream tasks.

4.2 RL-based Prompt Personalization

To dynamically select the most suitable prompt for each instance, we introduce a prompt routing mechanism built upon an Actor–Critic reinforcement learning framework (Fan et al., 2019). We begin by outlining the construction of diverse prompt pools, followed by a description of how instance-specific routing actions are generated. Additionally, we detail the textual-spatiotemporal alignment strategy and the routing update mechanism.

4.2.1 Diverse Prompt Pool Construction

The effectiveness of the Prompt Router relies on a well-designed and diverse prompt candidate pool. As shown in Fig. 2, we construct this pool by partitioning the prompt into K distinct functional slots, each containing N_c candidate sentences with varying focus and context. For illustration, in the context of a spatiotemporal forecasting task, we define the following diverse slots:

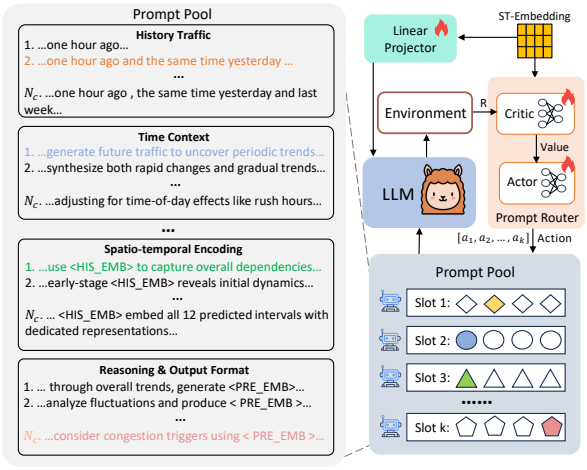

Figure 2: Instance-wise prompt generation.

Historical spatiotemporal information: This slot captures varying temporal influences by providing historical traffic data at different granularities. For instance, it could include "traffic data from one hour ago" for short-term patterns or "data from the same time on the previous day and week" to account for daily and weekly trends.

Time Context: Given the strong periodicity in traffic flow, such as daily peaks or holiday effects, this slot defines the temporal context, including the time of day, day of the week, and prediction horizon. To enhance adaptability, we offer candidate sentences emphasizing different temporal characteristics, like "uncovering periodic trends" or "adjusting for rush hours," helping the model focus on the most relevant temporal signals.

Spatiotemporal Encoding: To enable the language model to interpret spatiotemporal features, this slot uses placeholder tokens like ⟨HIS_EMB⟩, which are replaced with embeddings from a spatiotemporal encoder. Candidate sentences vary in focus—some describe the full prediction dependen-

cies, while others highlight the interpretability of intermediate representations to clarify the model's decision-making process.

Reasoning & Output Format: This slot defines how the LLM should generate predictions. All candidate sentences guide the model to produce a forecast embedding using the ⟨PRE_EMB⟩ placeholder token, but they differ in their reasoning approach. Some instruct the model to reason through temporal and spatial patterns step by step, improving interpretability, while others leverage domain knowledge to guide the forecast heuristically.

4.2.2 Routing Action Generation

To select appropriate sentences for each slot, we use the spatiotemporal representation $\mathbf{H_f}$ generated by the ST-Encoder as input. The actor and critic networks are implemented as two separate MLPs. For each slot k, the actor network maps the spatiotemporal representation $\mathbf{H_f}$ to a probability distribution π_k over the candidate sentence options, while the critic network computes the expected reward V_k to guide the actor's training. The actor and critic functions for slot k are defined as:

$$\pi_k = \text{Softmax}(\text{MLP}(\mathbf{H_f})), \quad V_k = \text{MLP}(\mathbf{H_f}). \tag{4}$$

An action a_k is sampled from the distribution π_k, and actions from all K slots are concatenated into a composite vector $\mathbf{a}_t = [a_1, a_2, \ldots, a_K]$, which determines how the final prompt is constructed.

4.2.3 Textual-Spatiotemporal Alignment

To align spatiotemporal signals with natural language input, we embed the encoded representations directly into the LLM's input space via a token-based interface. The token sequence ⟨st_start⟩, ⟨st_patch⟩$^{\times T}$, ⟨st_end⟩ replaces ⟨HIS_EMB⟩ and ⟨PRE_EMB⟩, where T is the number of prediction steps and each ⟨st_patch⟩ corresponds to a future time step. The spatiotemporal embeddings $\mathbf{E} \in \mathbb{R}^{T \times 2D}$ are then projected to the LLM's hidden dimension d_{L} via a linear layer, producing $\mathbf{E}' \in \mathbb{R}^{T \times d_{\text{L}}}$, which are inserted into the prompt by replacing the ⟨st_patch⟩ tokens.

4.2.4 Reinforcement-Guided Routing Update

To support diverse downstream tasks, the reward signal $\hat{R}$ for training the prompt router is defined as either the prediction loss or task-specific metrics such as taxi dispatch rewards, directly linking prompt quality to task performance. For each router, we apply a shared reward signal to evaluate all slots, enabling each slot to independently learn its optimal prompt routing policy through dedicated actor-critic networks. The optimization objectives for the actor and critic networks of slot k are defined as follows:

$$\mathcal{L}_a^{(k)} = -\log \pi_k \cdot \left(\hat{R}_t - V_k\right), \quad \mathcal{L}_c^{(k)} = \left(\hat{R}_t - V_k\right)^2. \tag{5}$$

4.3 Multi-task Output Layers

To improve accuracy in handling continuous values and prevent precision loss from token-level discretization, we avoid using the LLM to directly generate outputs, opting instead for task-specific output layers. These customized layers better align with the objective formats and evaluation metrics of different tasks, reducing interference from mismatched output spaces.

Spatiotemporal Forecasting: The predicted hidden state from the LLM, denoted as $\mathbf{H_f}'$, and the spatiotemporal encoder, denoted as $\mathbf{H_f}$, are each processed through separate MLPs. The resulting embeddings are then concatenated, and a final linear layer is applied to produce the final prediction.

Taxi Dispatch Optimization: For the dispatching task, the goal is to generate a probability distribution over the nine candidate regions in the 3×3 neighborhood surrounding the current location. To achieve this, the hidden state from the ⟨st_patch⟩ token is first passed through a linear layer. Then, a softmax function is applied to normalize the output, producing the dispatching probability distribution. The resulting output represents the proportion of vacant vehicles to be reallocated from the central grid to its surrounding regions.

4.4 Training Mechanism

The entire training process is divided into three stages. In stage 0, we train the encoder independently to generate high-quality embeddings. Then, we adopt a two-stage iterative training strategy: In stage 1, we fine-tune the LLM using LoRA (Hu et al., 2022) while keeping the Prompt Router frozen; in stage 2, we train the Prompt Router while freezing the LLM. In practice, a single alternation is sufficient to achieve optimal performance.

4.5 Loss Function

During stage 0, the loss function is defined as the weighted sum of the regression loss and the contrastive learning loss. In stages 1 and 2, we design a composite loss function formulated as:

$$\mathcal{L} = \mathcal{L}_{LLM} + \lambda_t \cdot \mathcal{L}_t, \tag{6}$$

where $\mathcal{L}_{LLM}$ denotes the standard cross-entropy loss for LLM, $\mathcal{L}_t$ represents the task loss, and λ_t is a weighting coefficient. For the spatiotemporal forecasting task, we use the L1 loss as the regression loss. For taxi dispatching task, we propose a multi-objective reinforcement learning loss function:

$$\begin{aligned} \mathcal{L}_t &= \mathcal{L}_{rf} + \lambda_w \mathcal{L}_w + \lambda_e \mathcal{H}(\hat{P}), \quad \mathcal{L}_{rf} = -\sum_{g=1}^{9} R_g, \\ \mathcal{L}_w &= \sum_{g=1}^{9} \left| \mathcal{C}(\hat{P})_g - \mathcal{C}(P)_g \right|, \quad R_g = \beta M_g - \gamma D_g, \end{aligned} \tag{7}$$

where $\mathcal{L}_{rf}$ denotes the reinforcement learning loss computed as the negative reward, $\mathcal{L}_w$ measures the Wasserstein distance between predicted and ground-truth distributions using cumulative distribution functions $\mathcal{C}$, and $\mathcal{H}(\hat{P})$ is an entropy regularizer. The reward term R_g balances service efficiency and dispatch cost, where M_g represents the matching rate in grid g and D_g is a predefined distance penalty. λ_w, λ_e, β, and γ control the contribution of each component.

5 Experiment

In this section, we comprehensively validate the effectiveness of TransLLM against three categories of baselines. Furthermore, zero-shot experiments are conducted to evaluate its generalizability across unseen scenarios. Additionally, we perform ablation studies and hyperparameter sensitivity analyses to verify the contributions of core components. Finally, we examine the behavior of the Prompt Router, which underscores the efficacy of instance-level prompting.

5.1 Experimental Setup

Datasets. To evaluate the effectiveness of TransLLM across diverse transportation tasks, we conduct experiments on five public datasets covering traffic forecasting, charging demand prediction, and taxi dispatching. Specifically, LargeST-SD (Liu et al., 2023) and PEMS08 are used for traffic forecasting, ST-EVCDP (Qu et al., 2024) and UrbanEV (Li et al., 2025) for charging demand prediction, and Taxi-SH for taxi dispatching. Among these, UrbanEV has a temporal resolution of 1 hour, while the others have a resolution of 5 minutes. More details are provided in Appendix A.

Evaluation Metrics. For spatiotemporal forecasting tasks, we employ the Mean Absolute Error (MAE) and the Root Mean Squared Error (RMSE) to quantify prediction accuracy.

$$\begin{aligned} \text{MAE} &= \frac{1}{N}\sum_{i=1}^{N} |y_i - \hat{y}_i|, \\ \text{RMSE} &= \sqrt{\frac{1}{N}\sum_{i=1}^{N} (y_i - \hat{y}_i)^2}. \end{aligned} \tag{8}$$

For taxi dispatching, we introduce two metrics: Mean Matching Rate (MMR) for vehicle matching and Mean Driving Distance (MDD) for dispatching costs. The metrics are formulated as:

$$\begin{aligned} \text{MMR} &= \frac{1}{N}\sum_{i=1}^{N} \left(\frac{\sum_{g=1}^{9} \hat{M}_{i,g}}{V_i} \right), \\ \text{MDD} &= \frac{1}{N}\sum_{i=1}^{N} \left(\sum_{g=1}^{9} \hat{p}_{i,g} \cdot C_{i,g} \right), \end{aligned} \tag{9}$$

Here, $\hat{M}_{i,g}$ is the predicted matched vehicles in region g for instance i; V_i represents the empty vehicles in the central region; $\hat{p}_{i,g}$ is the predicted dispatch probability; and $C_{i,g}$ denotes the predefined distance cost.

Baselines. We conducted a thorough comparison with fifteen baseline models to ensure a comprehensive evaluation of our proposed approach. These baselines can be broadly categorized into three groups: (1) GNN-based Deep Learning Models, such as STGCN (Yu et al., 2018), DGCRN (Li et al., 2023), ASTGCN (Guo et al., 2019), D^{2}STGNN (Shao et al., 2022), GWNET (Wu et al., 2019), STGODE (Fang et al., 2021) and PDG2seq (Fan et al., 2025); (2) Generalist Large Language Models, including Deepseek-v3 (DeepSeek-AI, 2024), GPT-5.2 (OpenAI, 2025) and Gemini-3 (Google DeepMind, 2025); and (3) Spatiotemporal Foundation Models, which include prompt-enhanced spatiotemporal universal models like UrbanDiT (Yuan et al., 2024b), FactoST (Zhong et al.), PromptST (Zhang et al., 2023) and UniST (Yuan et al., 2024a), as well as graph-enhanced LLMs such as ST-LLM+ (Liu et al., 2025) and UrbanGPT (Li et al., 2024). For detailed descriptions of these baselines, please see Appendix B.

Implementation Details. In our experimental setup, the key hyperparameters are configured as follows. The ST-Encoder is built by stacking 2 ST-Blocks, with a final output dimension D of 64. The Prompt Router is configured with 4 slots and 4 sentences. For prediction tasks, we predict 12 future

Models	Traffic Forecasting				Charging Demand Prediction				Taxi Dispatching	
	LargeST-SD		PEMS08		ST-EVCDP		UrbanEV		Taxi-SH	
	MAE ↓	RMSE ↓	MAE ↓	RMSE ↓	MAE ↓	RMSE ↓	MAE ↓	RMSE ↓	MMR(%) ↑	MDD ↓
GNN-based Deep Learning Models										
STGCN	13.93	26.10	10.30	14.77	2.11	3.64	3.02	5.30	19.10	5.80
DGCRN	11.46	24.63	10.45	14.47	1.71	4.46	3.04	5.47	18.89	3.60
ASTGCN	12.34	25.09	10.17	14.48	1.85	3.45	3.32	5.81	19.91	3.96
D^2STGNN	11.72	25.18	8.89	12.72	1.39	2.56	2.65	4.59	18.66	3.50
GWNET	13.43	26.42	9.26	12.90	1.45	2.65	2.57	4.27	19.29	3.49
STGODE	11.83	24.63	9.18	13.16	1.50	2.73	2.66	4.53	19.26	3.47
PDG2seq	12.16	25.02	9.54	13.53	1.46	2.93	3.33	7.38	18.92	3.49
Generalist Large Language Models										
Deepseek-v3	39.25	52.28	23.41	30.26	2.13	4.73	4.60	9.02	16.10	4.09
GPT-5.2	31.12	42.78	18.72	25.51	2.28	5.05	4.51	8.93	19.03	**3.12**
Gemini-3	10.92	22.97	11.25	16.02	2.22	4.28	5.38	10.51	17.64	3.27
Spatiotemporal Foundation Models										
UrbanDiT	15.02	28.27	12.45	17.37	1.51	3.06	3.26	5.65	19.88	3.44
FactoST	14.66	28.12	10.17	15.06	1.31	2.60	2.29	3.98	20.52	3.46
PromptST	13.42	26.52	9.08	12.55	1.63	3.31	2.74	4.65	18.66	3.52
UniST	12.64	25.28	9.52	12.86	2.58	3.95	3.37	5.36	-	-
ST-LLM+	12.27	25.24	9.53	13.35	1.74	3.33	3.00	4.84	19.25	3.51
UrbanGPT	11.28	23.17	10.23	13.53	2.09	5.86	2.65	6.12	19.46	3.21
TransLLM(vicuna)	10.98	21.42	7.88	11.33	1.39	2.73	2.13	5.25	24.46	3.15
TransLLM(llama3)	**9.41**	**16.78**	**7.26**	**10.68**	**1.26**	**2.16**	**1.78**	**3.87**	**24.78**	3.25

Table 1: Performance comparison across baseline models. **Bold**: Best, underline: Second best, Double underline: Third best.

time steps based on a 12-step historical sequence. Both the history length K and prediction length T are set to 12. For the taxi dispatch task, we predict the dispatch probabilities over nine grids. The model is trained with a learning rate of 1×10^{-4}, and the composite loss function balances objectives with weights $\lambda_t = 1.0$, $\lambda_w = 0.01$, $\lambda_e = 0.008$, and reward coefficients $\beta = 2.0$ and $\gamma = 0.05$. All experiments are conducted over multiple independent runs, and the mean performance is reported.

5.2 Overall Performance

Tab. 1 summarizes the overall results. Note that UniST pretraining relies on masking the same feature, making it unsuitable for Taxi-SH with 3 input features and 9 outputs features. According to the table, we make the following observations: First, TransLLM consistently outperforms all baselines across the five datasets, demonstrating the effectiveness of the framework. Moreover, the Llama3-based variant performs better than the Vicuna-based one, indicating the benefit of using a stronger foundation model. Second, the performance gap between model categories varies across datasets. For example, the gain of TransLLM on ST-EVCDP is smaller than that on traffic flow datasets like PEMS08. Our analysis indicates that ST-EVCDP has weaker periodicity and higher volatility, as detailed in Appendix C, making it more challenging and narrowing the performance differences across methods. Third, GNN-based models perform well in supervised regression tasks but struggle with taxi dispatching tasks due to the complex nature of real-time decision-making. Additionally, Gemini-3 outperforms other generalist LLMs in traffic forecasting, likely due to its native multimodal architecture and enhanced pre-training on spatiotemporal data. GPT-5.2 achieves the lowest MDD but at the cost of matching performance. It is important to note that, in the taxi dispatching task, MMR is the most critical metric, as it reflects the probability of successful passenger–driver matching, whereas MDD measures the dispatching cost. Proactive dispatching inevitably incurs higher costs, while conservative strategies tend to keep many vehicles within their original grids, thereby reducing MMR. We find that the relatively low MDD of GPT-5.2 stems from its limited ability to perceive and respond to real-time demand and vehicle availability. By focusing primarily on cost constraints, it adopts a conservative dispatching policy. Although this reduces travel distance, it comes at the expense

Models	PEMS03		PEMS04	
	MAE ↓	RMSE ↓	MAE ↓	RMSE ↓
Small-scale GNN-based Deep Learning Models				
STGCN	57.43	64.74	39.08	49.93
DGCRN	58.05	61.86	29.60	39.30
ASTGCN	80.23	88.66	45.02	54.65
D^2STGNN	63.17	66.65	30.92	39.23
GWNET	55.51	59.21	27.06	33.46
STGODE	73.54	81.98	64.33	77.18
PDG2seq	88.07	101.10	55.88	69.15
Generalist LLMs				
Deepseek-v3	25.15	38.28	37.79	55.83
GPT-5.2	25.49	38.25	36.58	53.86
Gemini-3	23.62	36.14	27.75	43.71
Spatiotemporal Foundation Models				
UrbanDiT	48.58	64.28	26.76	32.14
FactoS	56.41	61.95	**25.43**	**31.75**
PromptST	40.15	54.45	59.03	65.36
UniST	43.12	53.27	57.13	70.60
ST-LLM+	50.31	54.58	30.55	37.48
Urbangpt	23.27	37.58	41.52	69.76
TransLLM	**18.92**	**30.27**	28.69	47.27

Table 2: Zero-shot performance on PEMS03 and PEMS04.

of matching performance. In contrast, TransLLM employs a more proactive dispatching strategy, improving MMR by over 5% with only a marginal increase of 0.1 km in travel distance.

5.3 Zero-shot Scenarios Performance

To evaluate TransLLM's performance in a zero-shot scenario, we train on PEMS08 and evaluate on PEMS03 and PEMS04, which were not seen during training. We observe that language-based methods yield noticeably larger gains on PEMS03, whereas on PEMS04 their performance becomes comparable to other models. This may be due to the difference in distribution similarity: PEMS04 resembles the training dataset PEMS08 more closely, with average traffic volumes of 226.13 and 230.68 and peak values of 896.0 and 1147.0 respectively. In contrast, PEMS03 has a lower mean flow of 149.52 but a much sharper peak at 1852.0, reflecting a stronger distribution shift. TransLLM demonstrates strong zero-shot performance on the PEMS03 dataset, while it does not achieve the best results on the PEMS04 dataset. One possible reason is that the LLM backbone is fine-tuned with a relatively limited amount of data—approximately one-tenth of that used by other spatiotemporal foundation models without LLM components. Such a data constraint reduces the relative potential gains

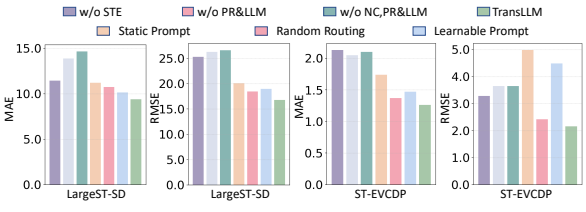

Figure 3: Ablation study results on LargeST-SD and ST-EVCDP datasets.

on datasets like PEMS04, which are more similar to the fine-tuning dataset PEMS08.

5.4 Ablation Study

To investigate the impact of individual modules on the overall performance, we evaluated TransLLM variants by removing or replacing key components: the ST-Encoder, LLM, and Prompt Router modules. Fig. 3 presents the ablation results on the LargeST-SD and ST-EVCDP datasets.

Effect of ST-Encoder. As shown in Fig. 3, removing the ST-Encoder (w/o STE) causes a significant drop in performance, demonstrating that the ST-Encoder effectively processes complex spatiotemporal patterns, providing the LLM with reliable non-textual spatiotemporal information.

Effect of the Large Language Model and noise augmentation mechanism. Note that once the LLM is ablated, the Prompt Router becomes non-trainable. Therefore, the variant without both the Prompt Router and the LLM (w/o PR&LLM) retains only the ST-Encoder. Furthermore, the w/o NC,PR&LLM variant additionally removes the noise augmentation and contrastive learning modules, leading to a further performance decline. These modifications result in a substantial drop in performance, highlighting both the effectiveness of the noise-enhanced contrastive learning design and the importance of the LLM's capacity in traffic-related tasks.

Effect of Prompter Router. We tested three alternatives for the Prompt Router module: static prompts, random routing, and learnable prompts. Static prompts use a fixed prompt for each instance, random routing selects a prompt randomly from the pool, and learnable prompts prepend a differentiable embedding to the fixed prompt. All three alternatives performed worse than the Prompt Router, demonstrating that dynamic, instance-specific prompts are crucial for unlocking the LLM's full potential.

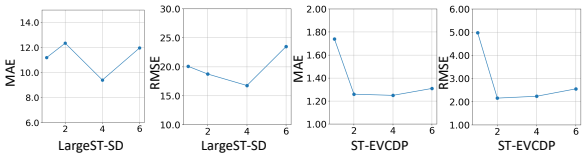

Figure 4: Effect of number of candidate sentences per slot N_c on LargeST-SD and ST-EVCDP datasets.

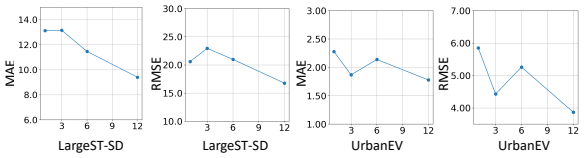

Figure 5: Effect of number of ⟨st_patch⟩ tokens N_p on LargeST-SD and UrbanEV datasets.

5.5 Parameter Sensitivity

To investigate how hyperparameters affect TransLLM's performance, we vary two key settings: the number of candidate sentences per slot N_c and the number of spatiotemporal feature tokens N_p. As shown in Fig. 4, increasing N_c does not yield monotonic gains—excessive candidates introduce redundancy, while too few limit diversity, with the optimal configuration being four candidates on LargeST-SD and two on ST-EVCDP. Meanwhile, Fig. 5 shows that performance generally improves as N_p increases, with the best results obtained when the token count matches the prediction horizon. Shorter token sequences fail to capture sufficient context and degrade prediction accuracy. We present the complete results of the parameter study in Appendix D.

5.6 Prompt Routing Behavior Analysis

To further examine how the Prompt Router behaves in practice, we visualize the selection frequency of candidate sentences in each slot during inference on the ST-EVCDP and PEMS08 datasets, as shown in Fig. 6. It is observed that within the same slot, every candidate sentence is selected at least once, indicating that the router does not rely on a fixed template but adaptively selects prompts based on input features. Meanwhile, the selection frequency distribution is highly imbalanced: certain sentences are chosen much more frequently, as they capture the dominant spatiotemporal patterns of most locations, whereas others are selected less often. However, a lower selection frequency does not imply inferior quality; instead, these sentences tend to capture rare but critical anomalies, such as temporary road blockages caused by traffic accidents.

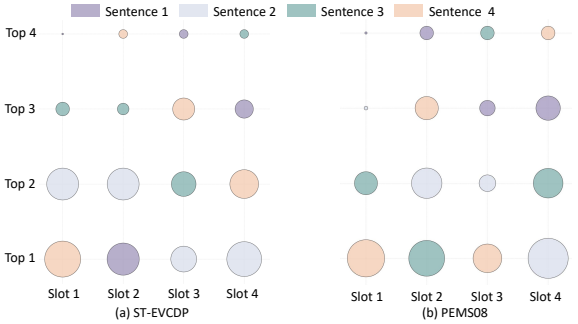

Figure 6: Sentence selection frequency per slot. The size of the circles represents the selection frequency of each sentence.

6 Conclusion

In this work, we present TransLLM, a unified framework that integrates spatiotemporal modeling with large language models for urban transportation tasks. The key contributions include a novel design that connects noise-augmented spatiotemporal encoders with LLMs through structured embeddings, as well as a learnable prompt composition mechanism with instance-level routing for dynamic personalization. Extensive experiments across traffic forecasting, electric vehicle charging demand prediction, and taxi dispatching tasks on seven datasets demonstrate strong performance in both supervised and zero-shot settings, highlighting excellent generalization and cross-task adaptability. Overall, TransLLM represents a promising step toward building foundation models tailored for intelligent transportation applications.

7 Acknowledgments

This work was supported by Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China (No. JYB2025XDXM113), the National Natural Science Foundation of China (No. 62332016), the Natural Science Foundation of Anhui Province (No. 2508085QF211), the National Natural Science Foundation of China (No. 62506348), New Generation Artificial Intelligence-National Science and Technology Major Project (No. 2025ZD0122601), and the Opening Foundation of State Key Laboratory of Cognitive Intelligence, iFLYTEK (COGOS-2025HE02).

Limitations

Due to computational resource constraints, we evaluate TransLLM on two base LLMs (Llama-8B and Vicuna-7B). While these models provide a representative testbed, we have not yet conducted a systematic study with larger-scale LLMs.

Ethical Considerations

This study is based entirely on open-access datasets curated for research applications. It does not involve human subject interaction, nor does it process any personal or sensitive information. All data handling respects their respective licenses and ACL code of ethics.

References

Shaojie Bai, J Zico Kolter, and Vladlen Koltun. 2018. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*.

Donald J Berndt and James Clifford. 1994. Using dynamic time warping to find patterns in time series. In *Proceedings of the 3rd international conference on knowledge discovery and data mining*, pages 359–370.

Chenyi Chen, Jianming Hu, Qiang Meng, and Yi Zhang. 2011. Short-time traffic flow prediction with arima-garch model. In *2011 IEEE Intelligent Vehicles Symposium (IV)*, pages 607–612. IEEE.

DeepSeek-AI. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jiang, Bill Yuchen Lin, Sean Welleck, Peter West, Chandra Bhagavatula, Ronan Le Bras, Jena D. Hwang, Soumya Sanyal, Xiang Ren, Allyson Ettinger, Zaïd Harchaoui, and Yejin Choi. 2023. Faith and fate: Limits of transformers on compositionality. *Advances in neural information processing systems*, 36:70293–70332.

Jin Fan, Wenchao Weng, Qikai Chen, Huifeng Wu, and Jia Wu. 2025. Pdg2seq: Periodic dynamic graph to sequence model for traffic flow prediction. *Neural Networks*, 183:106941.

Zhou Fan, Rui Su, Weinan Zhang, and Yong Yu. 2019. Hybrid actor-critic reinforcement learning in parameterized action space. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 2279–2285.

Zheng Fang, Qingqing Long, Guojie Song, and Kunqing Xie. 2021. Spatial-temporal graph ode networks for traffic flow forecasting. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pages 364–373.

Google DeepMind. 2025. Gemini 3.0. https://blog.google/technology/safety-alignment/gemini-3-collection/.

Shengnan Guo, Youfang Lin, Ning Feng, Chao Song, and Huaiyu Wan. 2019. Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 922–929.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Paul Foster, Pannag R. Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. 2024. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*.

Selvaraj Vasantha Kumar. 2017. Traffic flow prediction using kalman filtering technique. *Procedia Engineering*, 187:582–587.

Siqi Lai, Zhao Xu, Weijia Zhang, Hao Liu, and Hui Xiong. 2025. Llmlight: Large language models as traffic signal control agents. In *Proceedings of the 31th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

Fuxian Li, Jie Feng, Huan Yan, Guangyin Jin, Fan Yang, Funing Sun, Depeng Jin, and Yong Li. 2023. Dynamic graph convolutional recurrent network for traffic prediction: Benchmark and solution. *ACM Transactions on Knowledge Discovery from Data*, 17(1):1–21.

Han Li, Haohao Qu, Xiaojun Tan, Linlin You, Rui Zhu, and Wenqi Fan. 2025. Urbanev: An open benchmark dataset for urban electric vehicle charging demand prediction. *Scientific Data*, 12(1):523.

Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. 2018. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *International Conference on Learning Representations*.

Zhonghang Li, Lianghao Xia, Jiabin Tang, Yong Xu, Lei Shi, Long Xia, Dawei Yin, and Chao Huang. 2024. Urbangpt: Spatio-temporal large language models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 5351–5362.

Chenxi Liu, Kethmi Hirushini Hettige, Qianxiong Xu, Cheng Long, Shili Xiang, Gao Cong, Ziyue Li, and Rui Zhao. 2025. St-llm+: Graph enhanced spatio-temporal large language models for traffic prediction. *IEEE Transactions on Knowledge and Data Engineering*.

Xu Liu, Yutong Xia, Yuxuan Liang, Junfeng Hu, Yiwei Wang, Lei Bai, Chao Huang, Zhenguang Liu, Bryan Hooi, and Roger Zimmermann. 2023. Largest: A benchmark dataset for large-scale traffic forecasting. *Advances in Neural Information Processing Systems*, 36:75354–75371.

Wenyu Mao, Jiancan Wu, Weijian Chen, Chongming Gao, Xiang Wang, and Xiangnan He. 2025. Reinforced prompt personalization for recommendation with large language models. *ACM Transactions on Information Systems*, 43(3):1–27.

OpenAI. 2025. Gpt-5.2 system card. Available at https://cdn.openai.com/pdf/3a4153c8-c748-4b71-8e31-aecbde944f8d/oai_5_2_system-card.pdf.

Haohao Qu, Haoxuan Kuang, Qiuxuan Wang, Jun Li, and Linlin You. 2024. A physics-informed and attention-based graph learning approach for regional electric vehicle charging demand prediction. *IEEE Transactions on Intelligent Transportation Systems*, 25(10):14284–14297.

Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, Ivan Evtimov, Joanna Bitton, Manish Bhatt, Cristian Canton-Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre Défossez, Jade Copet, and 6 others. 2023. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.

Zezhi Shao, Zhao Zhang, Wei Wei, Fei Wang, Yongjun Xu, Xin Cao, and Christian S Jensen. 2022. Decoupled dynamic spatial-temporal graph neural network for traffic forecasting. *Proceedings of the VLDB Endowment*, 15(11):2733–2746.

Brian L Smith, Billy M Williams, and R Keith Oswald. 2002. Comparison of parametric and nonparametric models for traffic flow forecasting. *Transportation Research Part C: Emerging Technologies*, 10(4):303–321.

Llama Team. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Chao Wang, Yixin Song, Jinhui Ye, Chuan Qin, Dazhong Shen, Lingfeng Liu, Xiang Wang, and Yanyong Zhang. 2025. Face: A general framework for mapping collaborative filtering embeddings into llm tokens. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Senzhang Wang, Jiannong Cao, and S Yu Philip. 2020. Deep learning for spatio-temporal data mining: A survey. *IEEE transactions on knowledge and data engineering*, 34(8):3681–3700.

Wei Wu, Chao Wang, Liyi Chen, Mingze Yin, Yiheng Zhu, Kun Fu, Jieping Ye, Hui Xiong, and Zheng Wang. 2025. Structure-enhanced protein instruction tuning: Towards general-purpose protein understanding with llms. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 3216–3227.

Z Wu, S Pan, G Long, J Jiang, and C Zhang. 2019. Graph wavenet for deep spatial-temporal graph modeling. In *The 28th International Joint Conference on Artificial Intelligence*.

Huaxiu Yao, Xianfeng Tang, Hua Wei, Guanjie Zheng, and Zhenhui Li. 2019. Revisiting spatial-temporal similarity: A deep learning framework for traffic prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 5668–5675.

Huaxiu Yao, Fei Wu, Jintao Ke, Xianfeng Tang, Yitian Jia, Siyu Lu, Pinghua Gong, Jieping Ye, and Zhenhui Li. 2018. Deep multi-view spatial-temporal network for taxi demand prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Zhiyan Yi, Xiaoyue Cathy Liu, Ran Wei, Xi Chen, and Jiangpeng Dai. 2022. Electric vehicle charging demand forecasting using deep learning model. *Journal of Intelligent Transportation Systems*, 26(6):690–703.

Bing Yu, Haoteng Yin, and Zhanxing Zhu. 2018. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, page 3634–3640.

Yuan Yuan, Jingtao Ding, Jie Feng, Depeng Jin, and Yong Li. 2024a. 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*, pages 4095–4106.

Yuan Yuan, Chonghua Han, Jingtao Ding, Guozhen Zhang, Depeng Jin, and Yong Li. 2024b. Urbandit: A foundation model for open-world urban spatio-temporal learning.

Junbo Zhang, Yu Zheng, and Dekang Qi. 2017. Deep spatio-temporal residual networks for citywide crowd flows prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31.

Weijia Zhang, Hao Liu, Yanchi Liu, Jingbo Zhou, and Hui Xiong. 2020. Semi-supervised hierarchical recurrent graph neural network for city-wide parking availability prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 1186–1193.

Zijian Zhang, Xiangyu Zhao, Qidong Liu, Chunxu Zhang, Qian Ma, Wanyu Wang, Hongwei Zhao, Yiqi Wang, and Zitao Liu. 2023. Promptst: Prompt-enhanced spatio-temporal multi-attribute prediction.

In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 3195–3205.

Yusheng Zhao, Xiao Luo, Wei Ju, Chong Chen, Xian-Sheng Hua, and Ming Zhang. 2023. Dynamic hypergraph structure learning for traffic flow forecasting. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*, pages 2303–2316. IEEE.

Yusheng Zhao, Xiao Luo, Haomin Wen, Zhiping Xiao, Wei Ju, and Ming Zhang. 2025. Embracing large language models in traffic flow forecasting. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 8108–8123.

Siru Zhong, Junjie Qiu, Yangyu Wu, Xingchen Zou, Zhongwen Rao, Bin Yang, Chenjuan Guo, Hao Xu, and Yuxuan Liang. Learning to factorize spatio-temporal foundation models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

A Dataset Details

We train both TransLLM and the baseline models on five datasets, covering three types of tasks: traffic forecasting, charging demand prediction, and vehicle dispatching. LargeST-SD is a subset of the LargeST dataset (Liu et al., 2023), containing traffic flow records from 716 loop detectors on highways in San Diego County, spanning from January 1 to December 31, 2021. Pems08 includes traffic data from 170 sensors in California, collected between July 1 and August 31, 2016. ST-EVCDP (Qu et al., 2024) is a charging demand dataset collected from 247 areas in Shenzhen between June 19 and July 18, 2022, while UrbanEV (Li et al., 2025) captures large-scale charging behavior across urban regions from September 1, 2022 to February 28, 2023. For the vehicle dispatching task, we use the Taxi-SH dataset, which contains GPS trajectories of taxis in Shanghai from April 13 to April 19, 2015. The city is divided into 3 km × 3 km grids, and we aggregate the number of vacant taxis, passenger demand, and competing vehicles at 5-minute intervals. In addition, we evaluate the models under a zero-shot setting on two previously unseen datasets, PEMS03 and PEMS04, to assess their generalization capabilities. Among these, UrbanEV has a temporal resolution of one hour, while the others have a resolution of five minutes. All datasets used are available under their respective open-source licenses. We confirm that all artifacts used in this work are for research purposes, consistent with their intended use.

We perform temporal splits for each dataset. For all small-scale baseline models, the first 30% of each dataset is used for training. Due to the higher computational cost of training TransLLM, we employ a reduced training subset. For instance, only 4 days of data from Large-SD and 24 days from UrbanEV are used to train TransLLM. For evaluation, we extract a test set of $N \times 12$ samples from each dataset, where N is the number of nodes, ensuring that all samples are drawn from unseen time periods. To assess generalization under zero-shot settings, we evaluate on two previously unseen datasets, PEMS03 and PEMS04, by extracting 2 hours of data from 170 nodes in each. This node count is aligned with PEMS08 to ensure architectural compatibility, as several baseline models hard-code the number of nodes into their design, making them unsuitable for varying node configurations.

B Baselines Details

We conducted a thorough comparison with ten baseline models to ensure a comprehensive evaluation of our proposed approach. These baselines can be broadly categorized into three groups:

1. GNN-based Deep Learning Models. These models are developed specifically for spatiotemporal forecasting and are trained in an end-to-end manner on individual datasets. They typically rely on carefully designed architectures that integrate graph-based spatial encoders with temporal modeling components. Representative examples includes:

- **STGCN** (Yu et al., 2018): It combines graph and gated temporal convolutions to capture spatial-temporal patterns in traffic data.
- **DGCRN** (Li et al., 2023): This framework uses hyper-networks to generate dynamic graphs at each time step and integrates them with static topology to model time-varying traffic correlations.
- **ASTGCN** (Guo et al., 2019): It employs spatial and temporal attention mechanisms to focus on salient dependencies across time and space.
- **D^{2}STGNN** (Shao et al., 2022): This framework decouples diffusion and inherent traffic signals and learns dynamic graphs to represent evolving traffic structures more accurately.
- **GWNET** (Wu et al., 2019): This model learns an adaptive adjacency matrix and applies di-

lated convolutions to uncover hidden spatial relations and long-range temporal trends.

- **STGODE** (Fang et al., 2021): It leverages neural ordinary differential equations and semantic adjacency to jointly capture spatial-temporal dynamics.
- **PDG2Seq** (Fan et al., 2025): This method models periodic traffic patterns using a dynamic graph-to-sequence framework that aligns temporal cycles with future trend prediction.

2. Generalist Large Language Models (LLMs). These models, including Deepseek-v3, GPT-5.2 and Gemini-3 are not specifically trained for spatiotemporal or transportation tasks. We evaluate them in a zero-shot setting without any task-specific tuning.

3. Spatiotemporal Foundation Models. This category leverages large-scale heterogeneous spatiotemporal data and pre-training to enable broad generalization across diverse scenarios. We also include LLM-based architectures that incorporate structured spatiotemporal priors.

- **UrbanDiT** (Yuan et al., 2024b): It scales diffusion transformers to urban computing by unifying heterogeneous data into sequential formats and employing an adaptive prompt framework to support multi-task spatio-temporal learning.
- **FactoST** (Zhong et al.): It introduces a factorized foundation model that decouples universal temporal pretraining from spatio-temporal adaptation via a two-stage training strategy.
- **PromptST** (Zhang et al., 2023): It introduces a spatiotemporal transformer model for prediction by utilizing parameter-sharing training and spatiotemporal prompt tokens.
- **UniST** (Yuan et al., 2024a): It proposes a universal solution for urban spatiotemporal prediction by integrating four masking strategies for pre-training and prompt learning to handle diverse scenarios.
- **ST-LLM+** (Liu et al., 2025): It enhances traffic prediction with graph-based attention to capture spatiotemporal dependencies, using partially frozen graph attention and a LoRA-augmented training strategy for fine-tuning.
- **UrbanGPT** (Li et al., 2024): It enhances LLMs for urban forecasting by incorporating a temporal encoder that explicitly models periodic patterns and time dependencies in structured urban data.

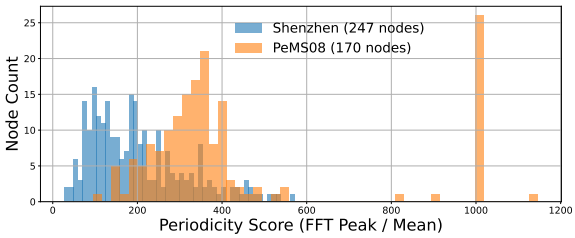

Figure 7: Periodicity Score Distribution of Charging and Traffic Datasets

C Comparison of PEMS08 and ST-EVCDP Dataset Characteristics

PEMS08 represents a typical road traffic flow dataset, whereas ST-EVCDP represents a charging demand dataset. These two types of spatiotemporal data exhibit distinct characteristics. Charging demand shows weaker periodicity and greater volatility than traffic flow data. We conduct a Fourier-based periodicity analysis by computing the ratio of the dominant FFT peak to the average signal amplitude (i.e. FFT Peak/Mean) for each node. As shown in Fig. 7, the periodicity scores for Shenzhen's 247 charging nodes are significantly lower and more dispersed than those of PeMS08's 170 traffic sensors. This indicates that while traffic exhibits strong and regular daily or weekly cycles, charging behavior is influenced by more irregular human and behavioral factors.

D Additional Parameter Sensitivity Analysis

To further validate the robustness of TransLLM, we conduct supplementary parameter sensitivity analyses on three additional datasets: PEMS08, UrbanEV (or ST-EVCDP), and Taxi-SH. As illustrated in Fig. 8, the performance across all three datasets exhibits a consistent U-shaped trend with respect to the number of candidate sentences per slot N_c. Specifically, the model achieves its optimal performance at $N_c = 4$ for all datasets. Regarding the number of ⟨st_patch⟩ tokens N_p shown in Fig. 9, we observe a steady improvement in prediction accuracy for PEMS08 and ST-EVCDP as N_p increases. And the model achieves its optimal performance when the number of tokens matches the prediction horizon, indicating that a perfectly

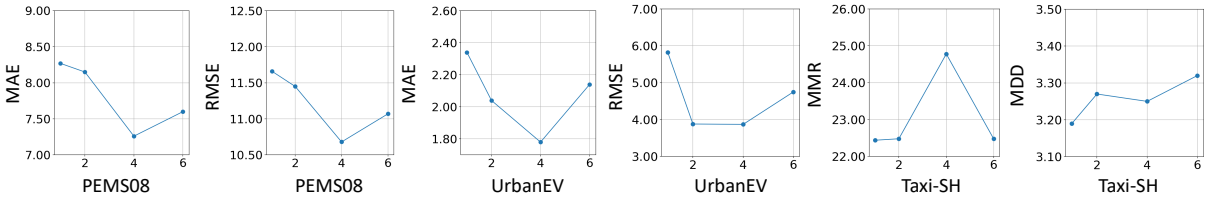

Figure 8: Effect of number of candidate sentences per slot N_c on PEMS08, UrbanEV and Taxi-SH datasets.

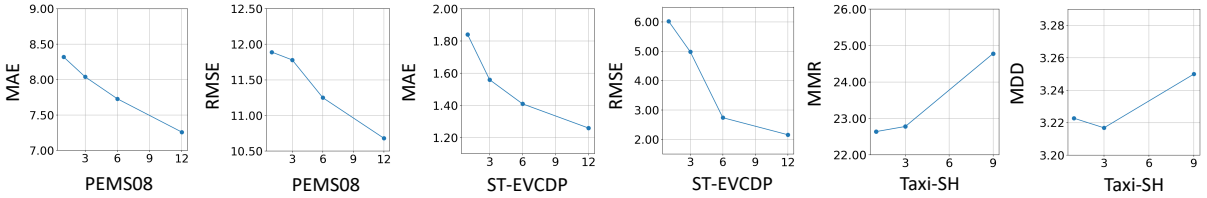

Figure 9: Effect of number of ⟨st_patch⟩ tokens N_p on PEMS08, ST-EVCDP and Taxi-SH datasets.

aligned token-to-step granularity is crucial for capturing temporal dependencies. Similarly, for the taxi dispatch task, a larger N_p contributes to a higher MMR; although this comes at the cost of a slight increase in MMD, it is considered an acceptable trade-off to prioritize dispatching efficiency.

E Training Costs and Real-Time Deployment Efficiency

We present the training and inference costs of TransLLM. Using the ST-EVCDP dataset as an example, which consists of 278k training samples, training takes approximately 24 hours on an NVIDIA A100 GPU with 32 GB of memory. The average inference time per sample is 1,170 ms, compared to 913 ms for UrbanGPT. The additional latency in TransLLM is due to the use of a Graph Attention Network for modeling spatial relationships, while UrbanGPT lacks this spatial modeling component. Each TransLLM request involves approximately 339 input tokens and 47 output tokens, making it suitable for real-time city-scale applications. The RL routing computation adds 36.77 ms per sample, with the overall inference time still remaining within practical limits for large-scale, real-time deployment.

F Case Study

In this section, we assess the effectiveness of different large models on a representative traffic flow forecasting instance. The detailed input information and the corresponding model responses are summarized in Table 3. Notably, the 'personalized prompt' is dynamically generated by our Prompt Router for this specific instance. Notably, generalist LLMs like Deepseek-v3 and GPT-5.2, while capable of parsing the instructions, demonstrate significant limitations. Deepseek-v3 failed to capture the correct temporal patterns, producing a forecast that deviates substantially from the ground truth. GPT-5.2 adopts a more conservative approach, producing an overly smooth forecast that misses the inherent volatility. Although UrbanGPT successfully captures the inherent volatility and the overall downward trend, its predictive accuracy is notably lower than that of TransLLM.

Table 3: Response Comparison of Different Models on a Specific Traffic Flow Forecasting Instance

Input Prompt Given the historical data for traffic flow over 12 time steps at a highway traffic monitoring point, the recorded traffic flow values are [62 67 60 83 81 55 50 56 52 64 41 37]. The recording time of the historical data is 'November 25, 2021, 23:00, Thursday to November 25, 2021, 23:55, Thursday, with data points recorded at 5-minute intervals'. To capture spatial and temporal dependencies, a spatiotemporal convolution model is utilized to encode the historical traffic data as embeddings <ST_EMB>. Now we want to predict the traffic flow for the next 12 time steps during the time period of 'November 26, 2021, 00:00, Friday to November 26, 2021, 00:55, Friday, with data points recorded at 5-minute intervals'. Please analyze the traffic patterns in this region, taking into account the provided historical data, time encoding, and spatiotemporal embeddings, and generate the predictive tokens for regression in the form "<ST_PRE>".
Personalized Prompt 'Using traffic flow recorded one hour ago [62, 67, 60, 83, 81, 55, 50, 56, 52, 64, 41, 37], we examine recent short-term variations to identify immediate trends. With this historical context in mind, the model will generate a 12-step forecast beginning at November 26, 2021, 00:00, Friday to November 26, 2021, 00:55, Friday, incorporating 5-minute resolution to synthesize both rapid changes and gradual trends. In order to better understand the early dynamics of traffic flow, we incorporate 12 spatiotemporal embeddings that correspond to future time intervals: <ST_EMB>. Please reason step-by-step through both temporal patterns and spatial influences. After that, generate the predicted traffic volume for the next 12 time intervals using the token <ST_PRE>.
Ground Truth Based on the given historical traffic flow data, time encoding, and spatiotemporal embeddings, the predictive tokens for the traffic flow in this region are:[45, 43, 41, 44, 36, 28, 29, 33, 28, 27, 35, 33].
Deepseek-v3 Responds: To analyze the traffic patterns and generate predictive tokens for the next 12 time steps, letś break down the problem step-by-step...Based on the above, the predicted traffic flow for the next 12 steps (in 5-minute intervals) is:[58, 62, 59, 78, 79, 54, 49, 54, 50, 62, 40, 36]...
GPT-5.2 Responds: Step 1. Analyze the historical traffic flow...Step 2. Incorporate temporal features... Predict next 12 time steps: [59, 61, 59, 60, 63, 66, 70, 75, 68, 62, 57, 54]...
UrbanGPT Responds: Based on the given historical traffic flow data, time encoding, and spatiotemporal embeddings, the predictive tokens for the traffic flow in this region are: [52, 50, 49, 47, 47, 44, 44, 41, 41, 38, 38, 37].
TransLLM Responds: Based on the given historical traffic flow data, time encoding, and spatiotemporal embeddings, the predictive tokens for the traffic flow in this region are [49, 47, 38, 42, 33, 32, 33, 29, 29, 28, 29, 31].