Harnessing and Evaluating the Intrinsic Extrapolation Ability of Large Language Models for Vehicle Trajectory Prediction

Jiawei Liu[1,3], Yanjiao Liu[1,3], Xun Gong[1,3,*], Tingting Wang[2], Hong Chen[4], Yunfeng Hu[2]
[1]School of Artificial Intelligence, Jilin University
[2]Department of Control Science and Engineering, Jilin University
[3]Engineering Research Center of
Knowledge-Driven Human-Machine Intelligence, Jilin University
[4]College of Electronics and Information Engineering, Tongji University
{jiaweil23, yanjiaol24}@mails.jlu.edu.cn, {gongxun, wangtingting}@jlu.edu.cn,
chenhong2019@tongji.edu.cn, huyf@jlu.edu.cn

Abstract

Emergent abilities of large language models (LLMs) have significantly advanced their application in autonomous vehicle (AV) research. Safe integration of LLMs into vehicles, however, necessitates their thorough understanding of dynamic traffic environments. Towards this end, this study introduces a framework leveraging LLMs' built-in extrapolation capabilities for vehicle trajectory prediction, thereby evaluating their comprehension of the evolution of traffic agents' behaviors and interactions over time. The framework employs a traffic encoder to extract spatial-level scene features from agents' observed trajectories to facilitate efficient scene representation. To focus on LLM's innate capabilities, scene features are then converted into LLM-compatible tokens through a reprogramming adapter and finally decoded into predicted trajectories with a linear decoder. Experimental results quantitatively demonstrate the framework's efficacy in enabling *off-the-shelf, frozen* LLMs to achieve competitive trajectory prediction performance, with qualitative analyses revealing their enhanced understanding of complex, multi-agent traffic scenarios. Code and trained model checkpoints are available at *here*.

1 Introduction

Recent studies suggest that when the model parameter number and the training data volume surpass critical thresholds, LLMs can exhibit sophisticated, high-level "emergent abilities". These include in-context learning (Brown, 2020), instruction following (Wang et al., 2024b) and zero-shot *extrapolation* (Kojima et al., 2022; Gruver et al., 2024). The advanced extrapolation ability enables LLMs to create and predict coherent content beyond their initial input, underscoring their potential for creative problem-solving and indicating preliminary manifestations of Artificial General Intelligence (AGI) (Bubeck et al., 2023).

Simultaneously, LLMs are drawing increasing interest in the AV domain, with some researchers anticipating a "GPT moment" that could revolutionize this field (Zhou et al., 2023b). Off-the-shelf and general-purpose LLMs such as GPT are being leveraged to process textual descriptions of dynamic and complex traffic scenes, aiming for high-level scene understanding and transparent decision-making (Wen et al., 2024; Mao et al., 2023). Nevertheless, the rapid expansion of LLM applications in safety-critical and dynamic AV contexts introduces several critical concerns: **Q1**: *Should time series data, such as traffic agents' trajectories, be encoded and processed in a textual format by LLMs?* **Q2**: *Can LLMs accurately model and comprehend the temporal evolution of traffic agents' behaviors and interactions?*

To answer the above questions, this study explores trajectory encoding strategies to enhance LLMs' comprehension of dynamic traffic scenes and examine the potential to harness LLMs' built-in extrapolation capabilities for vehicle trajectory prediction. In advanced AV systems, prediction models analyze the observed trajectories of interested vehicles and their surrounding agents to forecast future waypoints, thus preventing collisions and enabling safe, efficient driving decisions. In this work, assessing LLMs' applicability to trajectory prediction offers an intuitive and quantitative measure of their understanding of sophisticated, dynamic traffic scenarios.

However, applying LLM for trajectory forecasting entails the following challenges:

- **A) Limited Reliance on Built-in Extrapolation**: Figure 1-(A) presents two intuitive frameworks that, despite not fully leveraging the LLM's innate capabilities, still exhibit considerable performance. The cascading frame-

*Corresponding author

Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 4379–4391
April 29 - May 4, 2025

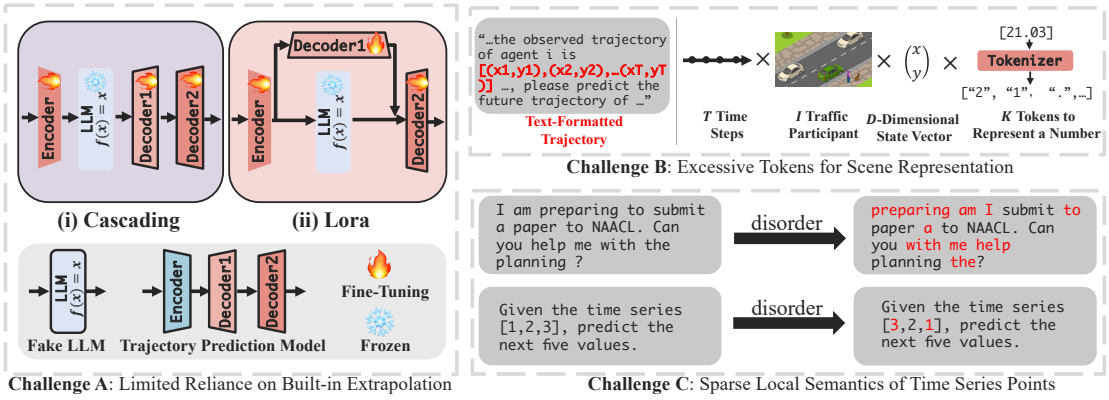


Figure 1: The three challenges for adapting LLMs to trajectory prediction task.

work feeds encoded features into the LLM, which then decodes them into predicted trajectories. Interestingly, even a "fake" LLM that merely performs identity mapping can still achieve notable performance within this framework, provided the encoder-decoder pair has already been optimized as a purpose-built prediction model. Similarly, in the LoRA-based framework (Wang et al., 2024a), the fake LLM can functions as a residual connection between the encoder and decoder, potentially enhancing performance through improved information flow. Thus, devising a valid framework to focus on LLMs' built-in extrapolation capabilities for prediction is a non-trivial task.

- **B) Excessive Tokens for Scene Representation**: To directly adapt LLMs for trajectory prediction, observed trajectories can be converted into text-formatted inputs compatible with LLM processing (Gruver et al., 2024). Nevertheless, as traffic scene complexity escalates, the number of tokens required for scene representation also escalates (i.e., $TIKD$ tokens for multiple observed trajectories in Figure 1-(B)). This rise in token count not only heightens the computational burden but also impairs LLMs' extrapolation abilities (Liu et al., 2024c). Therefore, an efficient scheme for representing and encoding multi-agent trajectories is essential for managing this complexity.

- **C) Sparse Local Semantics of Time Series Points**: Despite integrating positional encoding (Kazemnejad et al., 2024), the permutation-invariant attention mechanism in LLMs inherently lacks sensitivity to sequence order, resulting in an inevitable loss of temporal information (Zeng et al., 2023). As shown in Figure 1-(C), this anti-sequential bias is less problematic in semantically-rich domains like NLP, where individual tokens (or words) in sentence possess substantial local semantic meaning, thereby diminishing the reliance on exact positional cues. Conversely, numerical data in time series inherently lack this semantic richness, rendering sequence order critical for accurate temporal interpretation. Consequently, augmenting local semantic of trajectory encodings is crucial for effectively utilizing LLMs in trajectory prediction tasks (Nie et al., 2023).

To address the identified challenges, this work introduces a framework that exploits built-in extrapolation capabilities of general-purpose, off-the-shelf LLMs for vehicle trajectory prediction. Initially, instead of directly inputting text-formatted trajectories, the framework begins with a traffic scene encoder, which extracts spatial-level features from observed trajectories, capturing traffic agents' spatial layout and social interactions at each moment. Secondly, these scene features are mapped into the text embedding space via a reprogramming adapter (Jin et al., 2024), where each feature is reprogrammed as a combination of multiple text tokens from LLM vocabulary. This practice enables LLMs to comprehend the encoded features without re-training while mitigating catastrophic forgetting risk of their pre-existing capabilities. Furthermore, the scene tokens—reprogrammed scene features—serve as a compact representation of multi-agent trajectories (form $TIDK$ tokens to T tokens), enhancing local semantic richness by encoding environmental context. Finally, the LLM processes the scene

tokens alongside task-descriptive text tokens, with its output decoded into predicted trajectories using a simple linear decoder.

The main contributions in this work can be summarized as follows:

- An LLM-based trajectory prediction framework is proposed, which integrates a reprogramming adapter and a linear decoder to leverage the built-in extrapolation capabilities of off-the-shelf, frozen LLMs for accurate vehicle trajectory prediction.
- A scene encoder, combined with the reprogramming adapter, converts observed trajectories into spatial-level scene tokens, reducing the number of input tokens while enriching their local semantics.
- The proposed framework is comprehensively evaluated across various LLM architectures, achieving substantial performance improvements over comparison methods by efficiently leveraging environmental context.

Notably, this study is limited to evaluating the efficacy of LLMs in analyzing trajectory time series data, with predictions based on high-definition (HD) maps being reserved for future research.

2 Related Works

Large Language Model: The rising interest in LLMs stems from their sophisticated language capabilities, showcasing significant cognitive capability, cross-task transferability and zero-shot extrapolation (Kojima et al., 2022; Gruver et al., 2024). Versatile LLMs like GPT-4o exhibit exceptional proficiency in text-based tasks without requiring domain-specific fine-tuning, demonstrating potential in fields like math (Frieder et al., 2024) and programming (Liu et al., 2024b). Remarkably, some studies also indicate that LLMs trained exclusively on textual data can even comprehend visual concepts and create images through programming languages (Bubeck et al., 2023).

Recent research investigates the mechanisms underpinning the versatile and adaptive intelligence of LLMs. Studies by (Zhou et al., 2023a; Mirchandani et al., 2023) propose that LLMs function essentially as advanced pattern recognition systems, with the self-attention mechanism resembling principal component analysis. Research by (Jin et al., 2024) suggests that LLMs can leverage task-relevant language cues for time series forecasting. Another prominent hypothesis is that extensive and diverse datasets compel LLMs to cultivate generalized, functional "neural circuits" (Liu et al., 2022; Olsson et al., 2022). Their vast parameter space ensures adequate redundancy and variability, enabling these circuits to specialize and fine-tune for specific tasks (Bubeck et al., 2023).

Time Series Forecasting: Time series forecasting is pivotal in application such as trajectory prediction (Yuan et al., 2021; Salzmann et al., 2020), energy usage management and financial investments. Historically, forecasting methods have progressed from traditional statistical models (Ospina et al., 2023) to advanced deep learning-based methods. Presently, transformer-based architectures, exemplified by Informer (Zhou et al., 2021), which utilize self-attention mechanisms for efficient parallel processing, represent the forefront of sequence modeling advancements.

However, recent research (Zeng et al., 2023), challenges the efficacy of Transformers in time series forecasting, demonstrating that fully connected networks (FCN) can significantly outperform previous Transformer models. Subsequent research, termed PatchTST (Nie et al., 2023), reinforces this by demonstrating that Transformers exhibit limitations in time series analysis due to insufficient semantic granularity of individual time series data points. To mitigate this, PatchTST segments time series into overlapping subseries-level patches before encoding, thereby actively enriching the local semantic of time series embeddings.

Meanwhile, recent researches underscore the potential of leveraging LLMs for time series forecasting. A seminal work in this direction is GPT4TS (Zhou et al., 2023a), which integrates an LLM into the PatchTST framework while fine-tuning its normalization and output layers. Expanding on this breakthrough, Time-LLM (Jin et al., 2024) introduces a learnable reprogramming adapter to map patch embeddings into text embedding space while keeping the LLM parameters frozen. Additionally, Time-LLM offers an innovative perspective, indicating that time series forecasting can be reframed as a NLP task solvable by off-the-shelf, text-only LLMs.

3 Methodology

Framework Overview: An overview of the proposed framework is illustrated in Figure 2. Initially,

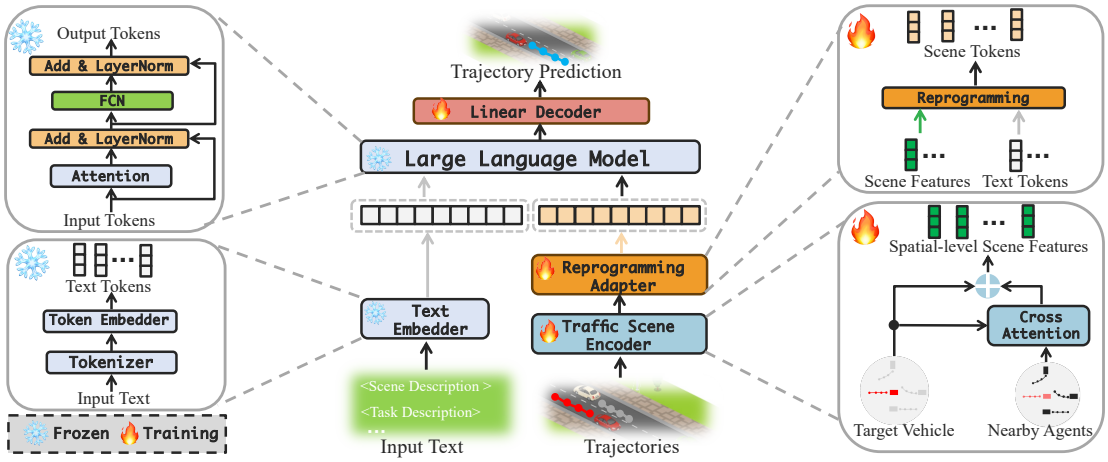


Figure 2: The proposed LLM-based framework for vehicle trajectory prediction.

a traffic scene encoder extracts spatial-level features from observed trajectories of traffic agents, capturing their spatial layouts and interactions at each timestamp. Next, a reprogramming adapter maps these scene features into the text embedding space, translating them into combinations of LLM vocabulary tokens. These reprogrammed scene features, referred to as scene tokens, are concatenated with text tokens that describe the prediction task and then fed into a pre-trained, forzen LLM backbone. Finally, a linear decoder, which performs a matrix transformation, converts the LLM's output into predicted trajectories.

Traffic Scene Encoder: Effective and efficient traffic scene encoding is fundamental to trajectory forecasting task, as it endows models with a comprehensive understanding of both social interactions (agent-to-agent interactions) and temporal dependencies (evolution of agent movements over time) (Huang et al., 2022). For instance, Trajectron++ (Salzmann et al., 2020) models traffic scenes as directed graphs, separately encoding social interactions and temporal correlations. HiVT (Zhou et al., 2022), on the other hand, employs a hierarchical framework, initially capturing social interactions and then using a temporal model for trajectory prediction. In contrast, AgentFormer (Yuan et al., 2021) argues that separate or sequential modeling of social and temporal correlations can lead to information loss, advocating for joint modeling to improve accuracy.

While advanced joint encoding schemes can potentially improve prediction performance, the proposed framework prioritizes the utilization of an encoder that only captures spatial layouts and social interactions among traffic agents. This design ensures that temporal correlations are managed by LLMs, thereby allowing a focused evaluation of their extrapolation capabilities for trajectory prediction.

Specifically, the framework first vectorizes the spatial states of the target vehicle 0 and its neighboring agents $i \in \{1, 2, ..., I\}$ for each timestamp $t \in \{1, 2, ..., T\}$ as follows:

$$\vec{x}_t^i = [x_t^i - x_{t-1}^i;\ x_t^i - x_t^0], \tag{1}$$

where $x_t^i \in \mathbb{R}^2$ represents the spatial coordinate, I and T denote the number of neighboring agents and observed timestamps, respectively. The vectorized state $\vec{x}_t^i \in \mathbb{R}^4$ integrates the displacement vector of agent i and its relative position to the target vehicle. Furthermore, all vectorized state vectors are rotated to align the target vehicle's orientation, ensuring that the trajectory encoding is invariant to both translation and rotation.

Next, a cross-attention layer is used to encode the interactions between target vehicle and neighboring agents into the vector $h_t' \in \mathbb{R}^{d_{scene}}$ though:

$$h_t' = \sum_{i=1}^{I} \text{Softmax}(q_t^0 k_t^i / \sqrt{d_{scene}}) v_t^i. \tag{2}$$

Here, the query vector $q_t^0 \in \mathbb{R}^{d_{scene}}$ derived from the target vehicle 0, the key vector $k_t^i \in \mathbb{R}^{d_{scene}}$ and value vectors $v_t^i \in \mathbb{R}^{d_{scene}}$ derived from the neighboring agent i are calculated as follows:

$$q_t^0 = W_Q \Phi(\vec{x}_t^0),\ k_t^i = W_K \Phi(\vec{x}_t^i),\ v_t^i = W_V \Phi(\vec{x}_t^i), \tag{3}$$

where $\Phi : \mathbb{R}^4 \to \mathbb{R}^{d_{scene}}$ is a multi-layer FCN, and W_Q, W_K, $W_V \in \mathbb{R}^{d_{scene} \times d_{scene}}$ are learnable query, key and value weight matrices, respectively.

Finally, a learnable fusion scheme is utilized to integrate the neighboring agents interactions and the target vehicle's spatial state according to:

$$h_t = \text{Sigmoid}(\alpha) \circ h'_t + \text{Sigmoid}(1-\alpha) \circ h^0_t, \quad (4)$$

where $h^0_t = \Phi(\vec{x}^0_t)$, $\alpha \in \mathbb{R}^{d_{scene}}$ is a learnable vector and $\circ$ denotes the Hadamard product.

Reprogramming Adapter: Recent researches indicates the emergent capabilities of LLMs position them as general-purpose artificial assistants (Bubeck et al., 2023). Consequently, there is growing interest in extending LLMs to additional modalities, such as images, to broaden their functional scope and applicability. Specifically, BLIP-2 (Li et al., 2023) utilizes image features extracted from a ResNet backbone as key and value vectors, which are then integrated with learnable query vectors and processed through a cross-attention layer to generate visual tokens comprehensible to LLMs. Similarly, LLaVA (Liu et al., 2024a) derives visual tokens by directly mapping the image features, produced by the CLIP encoder (Radford et al., 2021), into the text embedding space with a trainable FCN.

In contrast, research on adapting LLMs for trajectory data is limited due to several factors: 1) Data Interpretability: Unlike image pixel values, structured trajectory data is inherently more interpretable due to its organized and sequential nature. 2) Input Compatibility: The sequential structure of trajectory data aligns well with LLMs' input format. 3) Data Prevalence: Time series data, prevalent in domains such as finance, meteorology and transportation, is extensively included in LLMs' training data. Consequently, many researchers have opted to convert vehicle dynamics into textual format for direct input into LLMs (Wen et al., 2024; Gruver et al., 2024). Nevertheless, the resultant increase in token count can constrain the LLM's ability to effectively interpret intricate traffic scenarios (Liu et al., 2024c).

Towards this end, the proposed framework adopts the trainable reprogramming adapter (Jin et al., 2024), $\text{Reprogram} : \mathbb{R}^{d_{scene}} \rightarrow \mathbb{R}^{d_{llm}}$, to projects the extracted scene feature h_t from Equation (4) into the LLMs' text embedding space, formulated as:

$$b^{scene}_t = \text{Reprogram}(h_t). \quad (5)$$

In essence, the scene feature h_t is transformed into a combination of multiple LLM vocabulary tokens, $b^{scene}_t \in \mathbb{R}^{d_{llm}}$, via the learned mapping function Reprogram. This transformation yields a compact textual representation of h_t, enhancing input tokens' local semantic richness and facilitating LLMs' efficient comprehending of scene information. Compared with BLIP-2 and LLaVA, this adaption scheme allows for a more targeted assessment of LLMs' extrapolation ability using the representations they have learned from natural language data. Refer to Appendix A for further details of this section.

Linear Trajectory Decoder: Architectures such as RNN and LSTM are frequently employed as trajectory decoders due to their proficiency in capturing temporal dependencies. These autoregressive models function similarly to vehicle motion models (Salzmann et al., 2020), generating future waypoints iteratively based on preceding predictions to maintain both temporal and spatial coherence in the decoding process.

Nonetheless, utilizing autoregressive architectures to decode LLM outputs can inadvertently transfer the prediction challenge to the sophisticated decoder, potentially obscuring the evaluation of LLMs' innate capabilities. For example, if the decoder mirrors a constant velocity motion model, an LLM with limited extrapolation ability might still appear to perform well by simply outputting constant speeds for constant-velocity trajectories. This decoupling can mask the LLM's true predictive capabilities, as it relies on the decoder model to handle extrapolation tasks.

To avoid distributing the prediction task to sophisticated decoders, the proposed framework adopts a simple linear model to decode the LLM's output through:

$$\hat{\tau}^0_{1:N} = \text{LinearDecoder}(\{e^{scene}_t\}^T_{t=1}). \quad (6)$$

where $e^{scene}_t \in \mathbb{R}^{d_{llm}}$ is the processed scene token b^{scene}_t by the frozen LLM backbone. The decoder LinearDecoder first flattens the $\{e^{scene}_t\}^T_{t=1}$ into a vector and then project it into the predicted trajectory $\hat{\tau}^0_{1:N} \in \mathbb{R}^{N\times 2}$ with a learnable matrix $W_{decode} \in \mathbb{R}^{Td_{llm}\times N2}$, where N denotes the number of timestamps in the prediction time horizon.

Overall, despite employing the cascading architecture depicted in Figure 1, the proposed framework ensures that temporal dependencies are managed solely by LLMs through a meticulously designed sequence of encoder, adapter and decoder modules, thereby facilitating a targeted assessment of LLMs' extrapolation capabilities in trajectory prediction.

Table 1: Quantitative comparison of trajectory prediction results.

Method	Env. Info. †	Evaluation Metric				
		ADE±STD (2s) ↓ *	ADE±STD (4s) ↓	ADE±STD (6s) ↓	FDE±STD (6s) ↓	MR ↓
ZS-LLM	×	2.900 ± 3.704	3.951 ± 5.225	5.401 ± 6.697	9.767 ± 9.963	0.236%
Time-LLM ‡	×	0.861 ± 1.145	1.951 ± 2.768	3.404 ± 4.946	7.750 ± 8.014	**0.000%**
CoT-LLM	×	2.357 ± 2.519	3.437 ± 3.994	4.936 ± 6.154	9.450 ± 9.608	0.005%
IC-LLM	×	1.370 ± 1.902	2.607 ± 3.623	4.210 ± 5.937	8.975 ± 9.381	**0.000%**
IC-LLM	✓	8.030 ± 9.282	11.075 ± 12.976	14.265 ± 16.918	23.187 ± 23.669%	16.461%
Ours	×	0.897 ± 1.123	1.924 ± 2.533	3.725 ± 4.983	7.945 ± 8.531	**0.000%**
Ours	✓	**0.714 ± 0.812**	**1.539 ± 2.031**	**2.656 ± 3.750**	**6.062 ± 6.187**	**0.000%**

† Env. Info. refers to the environmental information, specifically the trajectories of neighboring agents.
* ADE±STD (2s) denotes the average distance error (ADE) and its standard deviation (STD) in 2 seconds (2s).
‡ Time-LLM can not leverage Env. Info. due to its channel-independent nature.

4 Experiment

4.1 Experimental Setting

This section details our experimental setting. All the experiments were conducted on a computer with an Intel Xeon Gold 5220 CPU clocked at 2.20GHz and 4 NVIDIA A40 GPU with 48GB memory.

Dataset: The proposed method is evaluated using the nuScenes dataset (Caesar et al., 2020), a comprehensive dataset for autonomous driving featuring 1000 scenes from Boston and Singapore. Each scene, lasting 20 seconds (s) and annotated at a frequency of 2 Hz, includes up to 23 semantic object classes.

Comparison Methods: A total of four comparison methods is considered in this work: 1) ZS-LLM (Gruver et al., 2024), which utilizes LLM in a zero-shot paradigm by embedding text-formatted trajectories into the template fine-tuned for chat interactions. 2) IC-LLM (Brown, 2020) incorporates task-specific exemplars into input text and extends ZC-LLM by utilizing LLMs' in-context learning capabilities. 3) CoT-LLM (Wei et al., 2022), extending ZC-LLM, utilizes LLMs' chain-of-thought capabilities to infer future trajectories. 4) Time-LLM (Jin et al., 2024) represents a seminal approach that leverages off-the-shelf LLMs for time series forecasting, thus is determined as the baseline method in this work.

Evaluation Metric: Trajectory prediction performance is mainly evaluated through the metrics of Average Displacement Error (ADE) and Final Displacement Error (FDE) (Salzmann et al., 2020). ADE measures the average Euclidean distance between the predicted waypoints and the ground-truth waypoints in a specified prediction horizon. FDE quantifies the Euclidean distance between the final predicted waypoint and the final ground-truth waypoint. Additionally, the Missing Rate (MR) metric is introduced to quantify the proportion of prediction failures, where a prediction failure occurs when the number of waypoints generated by LLMs deviates from the expected count of predicted timestamps.

Implementation Detail: The observation window in this work comprises 4 timestamps in 2 seconds (2s), while the prediction window encompasses 12 timestamps in 6s. The framework's training objective is to minimize the L2 loss between the predicted and ground-truth trajectories, using the Adam optimizer with a cosine annealing learning rate scheduler initialized at 1×10^{-4}. To ensure reproducibility, implementation details and the trained model checkpoints are available at *here*.

See Appendix B for more details on the comparison methods, evaluation metrics and **considered LLMs**.

4.2 Quantitative Evaluation

Table 1 provides a quantitative assessment of the proposed framework, benchmarked against comparison methods across 18,680 test trajectories, utilizing the LlaMa3-8B model fine-tuned for chat as the pre-trained foundation LLM. Initially, analyzing the experimental results of ZS-LLM, CoT-LLM, and IC-LLM reveals that, in cases focused solely on the target vehicle's trajectory, in-context learning capabilities of LLMs can markedly enhance prediction accuracy by incorporating task-specific exemplars into the input text, while chain-of-thought prompting only offers comparatively marginal improvements. However, as traffic complexity escalates with increased surrounding agent density, IC-LLM experiences significant performance degradation and a pronounced decline in task completion

rate. This observation exposes a critical limitation in LLMs' ability to process excessive text tokens for multiple-agent trajectory representation. Conversely, the proposed framework demonstrates a significant performance enhancement by integrating observed trajectories from neighboring agents. This highlights the necessity for advanced trajectory data encoding schemes to equip LLMs with the capability to effectively comprehend intricate and dynamic traffic environments.

4.3 Qualitative Evaluation

Figure 3 visualizes the trajectory prediction results of the proposed framework alongside IC-LLM, offering qualitative evidence that underscores the framework's superior efficiency in exploiting contextual environmental cues. Initially, compared to simpler linear trajectories, the performance disparity is especially pronounced in complex scenarios involving target vehicle turning maneuvers and interactions with multiple neighboring agents. Specifically, as shown in Figure 3-3, the proposed framework, despite lacking local HD maps, accurately forecasts the target vehicle's turning behavior by utilizing contextual cues from trajectories in both the target and adjacent lanes. Likewise, Figure 3-5 showcases the framework's effectiveness in predicting the target vehicle's evasive maneuvers for pedestrians.

In contrast, IC-LLM's prediction accuracy is hindered by limited environmental data, rendering it incapable of reliably forecasting vehicle turning behavior in most scenarios. Notably, even with ample environmental data (w. Env.), IC-LLM can still struggle with scene understanding due to the high volume of text tokens required to encapsulate multiple agent trajectories. As shown in Figure 3-(a), this constraint can even hinder accurate predictions of straightforward linear movements, highlighting the necessity for advanced numerical trajectory encoding schemes.

4.4 Generalizability Assessment

This section assesses the adaptability of the proposed framework across various 7B-LLMs. As detailed in Table 2, although trajectory prediction performance varies among the LLMs, they all outperform both the IC-LLM and the baseline Time-LLM in Table 1. Notably, the Qwen LLM, which features a 120,000-token vocabulary, achieves the highest prediction accuracy, underscoring the framework's versatility and effective adaptation to diverse off-the-shelf LLMs.

Table 2: Trajectory prediction accuracies of various LLMs.

LLM	Evaluation Metric			
	ADE (2s) ↓	ADE (4s) ↓	ADE(6s)	FDE(6s) ↓
LlaMa2	0.793	1.680	2.938	6.668
WizardLM	0.801	1.750	3.017	6.687
QWen	**0.754**	**1.508**	**2.640**	**6.125**
Vicuna	0.832	1.763	3.014	6.781
Mistral	0.860	1.695	2.806	6.656

4.5 Semantic Richness Assessment

As highlighted in this work's introduction, individual words within a sentence encapsulate substantial local semantics, thereby reducing dependence on exact positional information [1]. Consequently, this section investigates the effects of disordering spatial-level scene tokens prior to their input into LLMs, with the goal of assessing the semantic richness of these tokens. Ideally, if the scene tokens possess semantic richness comparable to text tokens, shuffling should only result in slight performance degradation.

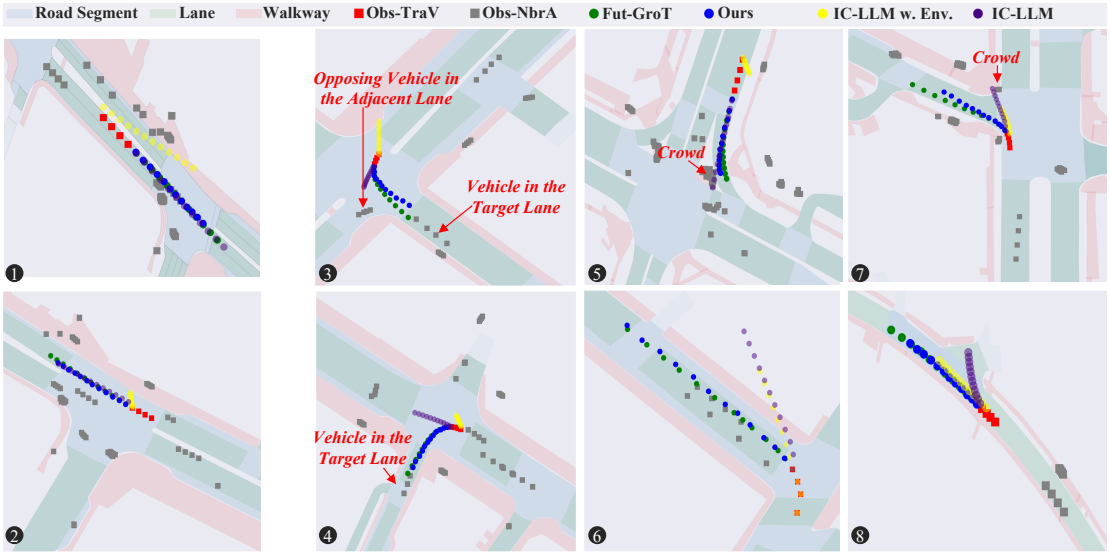
Table 3 demonstrates that, compared to the baseline Time-LLM, the proposed framework exhibits a less significant performance drop when shuffling input token order. This observation suggests that the framework's scene tokens encapsulate richer semantic information. The enhancement arises from the fact that the proposed framework incorporates environmental context, including spatial layouts and inter-agent dynamics, during token generation. In contrast, Time-LLM is limited to token encoding based solely on subseries-level patches. See Appendix C for more detailed results.

Besides, Appendix D examines the composition of encoded scene tokens, highlighting a subtle relationship between NLP tasks and trajectory forecasting within the proposed framework.

4.6 Comparison with Specialized Models

To better understand the limits of LLMs' capabilities, this section compares the proposed LLM-based framework with specialized trajectory prediction models. As shown in Table 4, the framework delivers prediction performance comparable to specialized models, though certain gaps persist.

[1]To illustrate, interested readers can input the disordered text about "NAACL planning" depicted in Figure 1 into GPT to evaluate its comprehension capabilities.



(a) Vehicle Moving Straight (b) Vehicle Making a Turn

Figure 3: Visualization of trajectory prediction results. Obs-TraV: Observed trajectory of target vehicle. Obs-NbrA: Observed trajectory of nearby agents. Fut-GroT: Ground-truth trajectory within prediction horizon. Ours: Predicted trajectory using the proposed framwork. IC-LLM w. Env: Predicted trajectory of IC-LLM with environmental information.

Table 3: Performance Drop Rate.

Token Order	Model	Degradation Rate for Evaluation Metrics			
		ADE ± STD (2s) ↓	ADE ± STD (4s) ↓	ADE ± STD (6s) ↓	FDE ± STD (6s) ↓
[2,1,3]	Time-LLM	39.024% ± 25.764%	19.067% ± 8.706%	11.398% ± 4.266%	5.690% ± 3.312%
	Ours	5.456% ± 2.832	3.508% ± 2.062%	2.937% ± 3.828%	2.573% ± 1.519%
[3,2,1]	Time-LLM	120.441% ± 102.445%	91.850% ± 72.326%	71.151% ± 54.407%	52.232% ± 43.274%
	Ours	36.694% ± 36.576%	33.983% ± 30.773%	28.802% ± 16.150%	22.171% ± 14.611%
[1,3,2]	Time-LLM	21.719% ± 27.424%	27.166% ± 31.900%	28.172% ± 30.186%	28.077% ± 28.812%
	Ours	12.605% ± 13.916%	13.190% ± 12.309%	11.746% ± 10.000%	9.798% ± 8.582%

† The degradation rate for all the evaluation metrics is 0.000% ± 0.000% when the token order is [1,2,3].

Table 4: Performance Comparison with Specialized Models.

Model	ADE (6s) ↓	FDE (6s) ↓
Trajectron++ (Salzmann et al., 2020)	**2.10**	5.00
AgentFormer (Yuan et al., 2021)	2.20	**4.82**
Ours	2.65	6.06

Table 5: Real-Time Performance Assessment.

Model	Total Param.	Trainable Param.	Freq.↓
ZS-LLM	6.608B	0B	0.299Hz
Time-LLM	6.641B	0.033B	**24.402Hz**
Ours	6.652B	0.044B	21.681Hz

Thus, enhancing LLMs' trajectory prediction for dynamic traffic scenarios continues to be a promising research direction.

It is crucial to note that the primary goal of this work is evaluation, not achieving SOTA performance. Trajectory prediction is used here to assess the LLM's understanding of traffic dynamics, rather than as the ultimate objective. To this end, the framework is deliberately designed with constraints to isolate and evaluate the LLM's inherent capabilities.

4.7 Real-Time Performance Assessment

This section evaluates the real-time performance of the proposed framework. As shown in Table 5, the framework achieves a high processing frequency of 21Hz, which explains the growing interest in end-to-end LLM-based methods (Shao et al., 2024).

4.8 Ablation Study

This section presents an ablation study on the proposed framework. The first row of Table 6 shows the results obtained by directly replacing the LLM with an identity mapping. This practice causes a

Table 6: Ablation Study.

Model	LLM	Restraining	ADE ± STD(6s)↓	FDE ± STD(6s)↓
Ours	×	×	12.570 ± 15.375	23.375 ± 22.500
	×	✓	3.177 ± 4.061	7.158 ± 7.686
	✓	✓	**2.656** ± **3.750**	**6.602** ± **6.187**

significant decline in prediction performance, addressing the concern raised in Challenge A (Figure 1) and demonstrating LLMs' contribution to the prediction task.

The second row shows the results when the LLM is replaced with an identity mapping and the framework is re-trained. This modification not only leads to inferior performance, but also disrupts the connection between the NLP and trajectory prediction tasks in the original framework. Specifically, it can cause the vocabulary token weights to become overly concentrated, skewing the probabilities towards a smaller set of meaningless tokens.

5 Conclusion

To assess the proficiency of off-the-shelf, text-only LLMs in understanding temporal dynamics of traffic agents, this study presents a framework leveraging the built-in extrapolation capabilities of LLMs for vehicle trajectory forecasting. Key findings from our comprehensive evaluation include: 1) Given text-formatted observed trajectories, LLMs can, to some extent, predict simple linear trajectory of target vehicle by leveraging in-context, task-specific exemplars. 2) However, as the density of neighboring traffic agents increases, the predictive performance of LLMs deteriorates significantly, underscoring their limitation in handling the excessive tokens required for multi-agent trajectory representation. 3) To mitigate this limitation, this work propose to treat trajectory data as a distinct modality from text. Specifically, the framework represents multiple trajectories as semantically enriched, spatial-level scene tokens. This practice significantly enhances the prediction performance and enables vehicle turning maneuver predictions without HD maps.

6 Acknowledgements

This work was supported by Development and Reform Commission Foundation of Jilin Province (No.2023C034-3, No.2024C003), Doctoral Student Research Innovation Capacity Enhancement Program of the Education Department of Jilin Province (No.2024KC126) and China Postdoctoral Science Foundation (No.2023M731283).

7 Limitations

While the trajectory encoding scheme can augment LLMs' comprehension of dynamic traffic scenarios, their trajectory prediction performance remains inferior to that of SOTA specialized models. This disparity underscores the limitations of LLMs in acquiring a nuanced comprehension of traffic agent dynamics, prompting us to reflect on the security and reliability of recent LLM-based AV applications. Consequently, reconciling "coarse-grained" high-level language capabilities of LLMs with the "fine-grained" demands of safety-critical AV applications will still be a pivotal research focus in the future.

References

Jinze Bai and et al. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Tom B Brown. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. 2023. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*.

Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. 2020. nuscenes: A multimodal dataset for autonomous driving. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Simon Frieder, Luca Pinchetti, Ryan-Rhys Griffiths, Tommaso Salvatori, Thomas Lukasiewicz, Philipp Petersen, and Julius Berner. 2024. Mathematical capabilities of chatgpt. *Advances in neural information processing systems*.

Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew G Wilson. 2024. Large language models are zero-shot time series forecasters. *Advances in Neural Information Processing Systems*.

Yanjun Huang, Jiatong Du, Ziru Yang, Zewei Zhou, Lin Zhang, and Hong Chen. 2022. A survey on trajectory-prediction methods for autonomous driving. *IEEE Transactions on Intelligent Vehicles*.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, et al. 2024. Time-llm: Time series forecasting by reprogramming large language models. *The Twelfth International Conference on Learning Representations*.

Amirhossein Kazemnejad, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Payel Das, and Siva Reddy. 2024. The impact of positional encoding on length generalization in transformers. *Advances in Neural Information Processing Systems*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *International conference on machine learning*.

Bingbin Liu, Jordan T Ash, Surbhi Goel, Akshay Krishnamurthy, and Cyril Zhang. 2022. Transformers learn shortcuts to automata. *arXiv preprint arXiv:2210.10749*.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2024a. Visual instruction tuning. *Advances in neural information processing systems*.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. 2024b. Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation. *Advances in Neural Information Processing Systems*.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024c. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*.

Jiageng Mao, Yuxi Qian, Hang Zhao, and Yue Wang. 2023. Gpt-driver: Learning to drive with gpt. *arXiv preprint arXiv:2310.01415*.

Suvir Mirchandani, Fei Xia, Pete Florence, Brian Ichter, Danny Driess, Montserrat Gonzalez Arenas, Kanishka Rao, Dorsa Sadigh, and Andy Zeng. 2023. Large language models as general pattern machines. *The Seventh Annual Conference on Robot Learning*.

Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. 2023. A time series is worth 64 words: Long-term forecasting with transformers. *The Eleventh International Conference on Learning Representations*.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, et al. 2022. In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*.

Raydonal Ospina, João AM Gondim, Víctor Leiva, and Cecilia Castro. 2023. An overview of forecast analysis with arima models during the covid-19 pandemic: Methodology and case study in brazil. *Mathematics*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. *International conference on machine learning*.

Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. 2020. Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. *European Conference on Computer Vision*.

Hao Shao, Yuxuan Hu, Letian Wang, Guanglu Song, Steven L Waslander, Yu Liu, and Hongsheng Li. 2024. Lmdrive: Closed-loop end-to-end driving with large language models. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Haixin Wang, Xinlong Yang, Jianlong Chang, Dian Jin, Jinan Sun, Shikun Zhang, Xiao Luo, and Qi Tian. 2024a. Parameter-efficient tuning of large-scale multimodal foundation model. *Advances in Neural Information Processing Systems*.

Yuanhao Wang, Qinghua Liu, and Chi Jin. 2024b. Is rlhf more difficult than standard rl? a theoretical perspective. *Advances in Neural Information Processing Systems*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*.

Licheng Wen, Daocheng Fu, Xin Li, Xinyu Cai, MA Tao, Pinlong Cai, Min Dou, Botian Shi, Liang He, and Yu Qiao. 2024. Dilu: A knowledge-driven approach to autonomous driving with large language models. *The Twelfth International Conference on Learning Representations*.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, Qingwei Lin, and Daxin Jiang. 2024. Wizardlm: Empowering large pre-trained language models to follow complex instructions. *The Twelfth International Conference on Learning Representations*.

Ye Yuan, Xinshuo Weng, Yanglan Ou, and Kris M Kitani. 2021. Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting. *Proceedings of the IEEE/CVF International Conference on Computer Vision*.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. 2023. Are transformers effective for time series forecasting? *Proceedings of the AAAI conference on artificial intelligence*.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. 2021. Informer: Beyond efficient transformer for long sequence time-series forecasting. *Proceedings of the AAAI conference on artificial intelligence*.

Tian Zhou, Peisong Niu, Liang Sun, Rong Jin, et al. 2023a. One fits all: Power general time series analysis by pretrained lm. *Advances in neural information processing systems*.

Xingcheng Zhou, Mingyu Liu, Bare Luka Zagar, Ekim Yurtsever, and Alois C Knoll. 2023b. Vision language models in autonomous driving and intelligent transportation systems. *arXiv preprint arXiv:2310.14414*.

Zikang Zhou, Luyao Ye, Jianping Wang, Kui Wu, and Kejie Lu. 2022. Hivt: Hierarchical vector transformer for multi-agent motion prediction. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

A Reprogramming Adapter

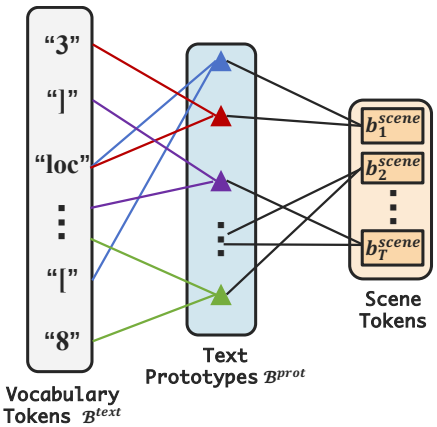


Figure 4: Scene reprogramming progress.

The proposed framework employs a trainable reprogramming adapter (Jin et al., 2024), denoted as $\text{Reprogram} : \mathbb{R}^{d_{scene}} \rightarrow \mathbb{R}^{d_{llm}}$, to transform the extracted spatial-level scene feature $h_t \in \mathbb{R}^{d_{scene}}$ into combinations of multiple vocabulary text tokens through:

$$b_t^{scene} = \text{Reprogram}(h_t). \tag{7}$$

Specifically, this process can be further decomposed as two steps as shown in Figure 4.

First, to mitigate the complexity of a large and potentially dense reprogramming space, the reprogramming adapter compresses and transforms the vocabulary tokens into a set of text prototypes through

$$\mathcal{B}^{prot} = W^{prot}\mathcal{B}^{text}. \tag{8}$$

Here, $\mathcal{B}^{text} \in \mathbb{R}^{J \times d_{llm}}$ denotes J text tokens $\left\{b_j^{text}\right\}_{j=1}^{J}$ ($b_j^{text} \in \mathbb{R}^{d_{llm}}$) in the LLM vocabulary, $W^{prot} \in \mathbb{R}^{M \times J}$ is a trainable matrix, and $\mathcal{B}^{prot} \in \mathbb{R}^{M \times d_{llm}}$ represents a collection of M text prototypes $b_m^{prot} \in \mathbb{R}^{d_{llm}}$. Each text prototypes can be seen as linear combination of the vocabulary tokens.

Second, a cross-attention layer is applied to reprogram the scene feature $h_t \in \mathbb{R}^{d_{scene}}$ into the scene token $b_t^{scene} \in \mathbb{R}^{d_{llm}}$, formulated as:

$$b_t^{scene} = \sum_{m=1}^{M} \text{Sotmax}(q_t^{rep} k_m^{rep} / \sqrt{d_{llm}}) v_m^{rep}. \tag{9}$$

Here, $q_t^{rep}, k_m^{rep}, v_m^{rep} \in \mathbb{R}^{d_{llm}}$ are query, key and value vectors obtained with:

$$\begin{aligned} &q_t^{rep} = W_Q^{rep}\Phi^{rep}(h_t), k_m^{rep} = W_K^{rep} b_m^{prot}, \\ &v_m^{rep} = W_V^{rep} b_m^{prot}, \end{aligned} \tag{10}$$

where W_Q^{rep}, W_K^{rep}, $W_V^{rep} \in \mathbb{R}^{d_{llm} \times d_{llm}}$ are learnable weight matrices for query, key and value transformations, respectively. The function $\Phi^{rep} : \mathbb{R}^{d_{scene}} \rightarrow \mathbb{R}^{d_{lmm}}$ denotes a fully connected network (FCN). In practical applications, the reprogramming process is implemented using a multi-head cross-attention layer. For the sake of notational simplicity, Equation (9) illustrates a single-head version.

B Experimental Setting

Comparison Methods: A total of four comparison methods are considered in this work:

1) ZS-LLM (Gruver et al., 2024), which utilizes LLM in a zero-shot paradigm by embedding text-formatted trajectories into the template fine-tuned for chat interactions.

2) IC-LLM (Brown, 2020) integrates task-specific exemplars within the LLM input text and extends ZC-LLM by utilizing LLMs' in-context learning capabilities.

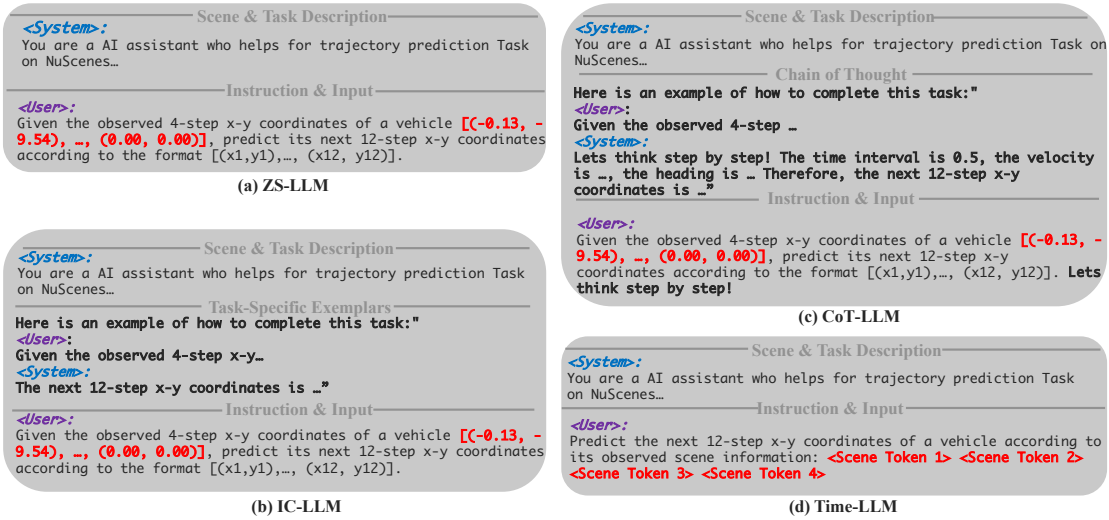


Figure 5: Input templates of the comparison methods.

Table 7: LLM Specifications

LLM	Parameter Num.	Vocab. Size	Training Tokens	Features	Max Token
LLaMa2 (Touvron et al., 2023)	7B	26519	2T	RLHF	4K
WizardLM (Xu et al., 2024)	7B	26497	2T	Evol-Instruct	4K
QWen (Bai and et al., 2023)	7B	150311	3T	RLHF	32K
Vicuna (Chiang et al., 2023)	7B	26519	2T	Gradient Checkpoint, Flash Attention	2K
Mistral (Jiang et al., 2023)	7B	26497	2T	Group-Query Attention, Sliding Window Attention	32K

3) CoT-LLM (Wei et al., 2022), extending ZC-LLM, utilizes LLMs' chain-of-thought capabilities to infer future trajectories.

4) Time-LLM (Jin et al., 2024) represents a seminal approach that leverages off-the-shelf LLMs for time series forecasting, thus is determined as the baseline method in this work.

The simplified input templates for the comparison methods are depicted in Figure 5.

Evaluation Metric: Average Displacement Error (ADE) measures the average Euclidean distance between the predicted trajectory $\hat{\tau}^0_{1:N} \in \mathbb{R}^{N\times 2}$ and the ground truth trajectory points $\tau^0_{1:N} \in \mathbb{R}^{N\times 2}$, calculated as:

$$\frac{1}{N} \times \sum_{n=1}^{N} ||\hat{\tau}^0_n - \tau^0_n||_2, \tag{11}$$

where $\hat{\tau}^0_n \in \mathbb{R}^2$ denote the x-y coordinates of the n-th predicted waypoint. Similarly, Final Displacement Error (FDE) quantifies the Euclidean distance between the final predicted and the ground-truth waypoints through:

$$||\hat{\tau}^0_N - \tau^0_N||_2, \tag{12}$$

Utilizing both ADE and FDE provides comprehensive insight into the model's performance. ADE captures the overall trajectory accuracy, while FDE focuses on the critical endpoint accuracy. This dual metric approach highlights models that may perform well on average but underperform at the endpoint, or vice versa, thus offering a nuanced evaluation of the model's strengths and weaknesses.

LLMs Used for Evaluation: A total of five LLMs are considered in this work. Their specifications and characteristics are summarized in Table 7.

C Semantic Richness Assessment

Individual words within a sentence encode significant local semantics, diminishing reliance on precise positional information. This section examines the impact of spatial-level scene token disordering before input into LLMs, aiming to evaluate the semantic richness of scene tokens. If scene tokens exhibit semantic richness akin to text tokens, shuffling should cause minimal performance degradation. Table 8 presents the prediction performance when the input token order is shuffled.

D Token Weight Analysis

The previous research (Jin et al., 2024) indicates that the reprogramming adapter converts time series embeddings into a combination of textual to-

Table 8: Impact of Input Token Order Shuffling on Trajectory Prediction Performance.

Token Order	Model	Evaluation Metrics			
		ADE ± STD (2s)	ADE ± STD (4s)	ADE ± STD (6s)	FDE ± STD (6s)
[1,2,3]	Time-LLM	0.861 ± 1.145	1.951 ± 2.768	3.404 ± 4.946	7.750 ± 8.014
	Ours	0.714 ± 0.812	1.539 ± 2.031	2.656 ± 3.750	6.062 ± 6.187
[1,3,2]	Time-LLM	1.048 ± 1.459	2.481 ± 3.651	4.363 ± 6.439	9.926 ± 10.323
	Ours	0.797 ± 0.910	1.710 ± 2.234	2.921 ± 4.062	6.531 ± 6.625
[2,1,3]	Time-LLM	1.197 ± 1.440	2.323 ± 3.009	3.792 ± 5.157	8.191 ± 8.625
	Ours	0.769 ± 0.882	1.648 ± 2.171	2.828 ± 3.968	6.406 ± 6.500
[2,3,1]	Time-LLM	2.052 ± 2.470	3.935 ± 4.971	6.007 ± 7.836	12.239 ± 11.972
	Ours	0.960 ± 1.140	2.071 ± 2.734	3.484 ± 4.781	7.625 ± 7.593
[3,1,2]	Time-LLM	1.166 ± 1.525	2.540 ± 3.683	4.393 ± 6.491	9.901 ± 10.463
	Ours	0.941 ± 1.140	2.015 ± 2.703	3.406 ± 4.750	7.500 ± 7.625
[3,2,1]	Time-LLM	1.898 ± 2.318	3.743 ± 4.770	5.826 ± 7.537	11.798 ± 11.482
	Ours	1.054 ± 1.296	2.250 ± 3.046	3.765 ± 5.250	8.187 ± 8.312

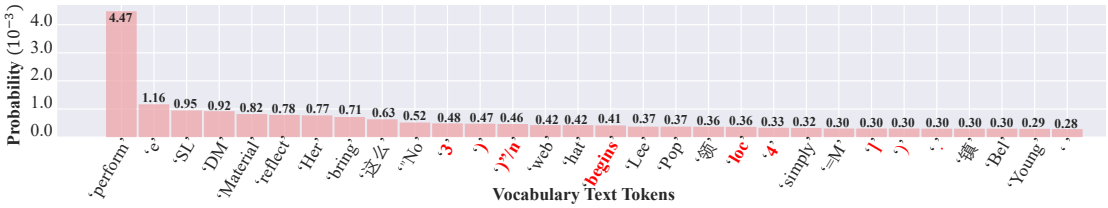


Figure 6: Normalized weights of vocabulary tokens in the reprogramming adapter.

kens such as "down", "up" and "seasonal" which describe the time series' characteristics. To verify this hypothesis, this work normalizes the weights of different text tokens from LLM vocabulary within the trained reprogramming adapter. Specifically, given the learnable matrix W^{prot} that maps the vocabulary tokens $\mathcal{B}^{text}$ to the text prototypes $\mathcal{B}^{prot}$, this work initially computes the sum of the absolute values of the weights associated with each vocabulary token in W^{prot}. Subsequently, these summed weights are divided by their total sum and normalized to form a probability distribution. Notably, a vocabulary token with a higher normalized weight indicates greater significance of its semantic information in representing traffic scenes.

Figure 6 illustrates the top-ranked vocabulary tokens by the normalized weights for the best-performing QWEN among the evaluated 7B-parameter LLMs. Notably, tokens associated with numerical concepts, decimal points and delimiters—essential for structured trajectory representation—exhibit significant weights, reflecting a nuanced relationship between NLP tasks and time series forecasting.