Commun Transp Res

Research Article

Crash Risk Prediction Guided by Large Language Models Using Pre-Crash Trajectories

Kequan Chen[1,✉], Yuxuan Wang[2], Zhibin Li[3], Pan Liu[3,✉]

Cite this article: https://doi.org/10.26599/COMMTR.2026.9640051

ABSTRACT: This study develops a crash risk prediction model at the individual vehicle level by leveraging the reasoning capability of large language models (LLMs). Instead of using the LLM itself for online prediction, the proposed framework converts LLM reasoning into structured supervision and distills it into a lightweight temporal graph network, which is then calibrated with real crash data. The framework is evaluated using pre-crash vehicle trajectories from 109 real-world crashes captured in multi-year drone videos at freeway merging and weaving segments in Nanjing, China. Results show that the proposed model outperforms representative benchmarks, including direct LLM inference, direct supervised training on real crash data, and traditional surrogate safety measures. Compared with direct supervised training using the same student architecture, it increases AUC by 7.8% and reduces MAE by 28.0%. Moreover, the proposed model is particularly effective when real crash data are limited. When only 10% of the fine-tuning data is used, it achieves the best performance among the compared methods, reducing MAE by 19.6% relative to the LLM baseline and by 38.0% relative to direct supervised training. These findings suggest that the proposed framework can improve prediction performance and data efficiency when real crash data are limited.

KEYWORDS: crash risk prediction; large language model; pseudo-labeling; knowledge distillation; temporal graph network; real-world crash data

1. Introduction

Crash risk prediction at the individual vehicle level aims to determine whether an ongoing interaction among vehicles is likely to develop into a crash. With the growing use of proactive traffic control and safety intervention systems, fast and accurate crash risk prediction has become increasingly important. It determines how early a hazardous situation can be identified and how effectively control actions can be implemented. Thus, this topic has received increasing attention in traffic safety research.

Early studies on crash risk prediction for individual vehicles mainly relied on surrogate safety measures (SSMs) derived from vehicle trajectories. Typical indicators, such as time to collision (Hayward, 1972), post-encroachment time (Peesapati et al., 2018), and deceleration-based measures (Tak et al., 2015), assess crash risk by characterizing traffic conflicts in terms of the temporal or spatial proximity of vehicle interactions. These studies implicitly assume that traffic conflicts become more severe as vehicle interactions move closer to a crash, so a more severe conflict corresponds to a higher likelihood of crash occurrence (Ali et al., 2023). However, a single SSM usually captures only one aspect of a hazardous interaction and is therefore insufficient to represent the coupled and dynamic nature of multi-vehicle interactions in complex traffic situations. To address this limitation, later studies combined multiple SSMs within multivariate or composite frameworks (Arun et al., 2022), introduced richer conflict-based indicators that jointly consider proximity and collision severity (Tang et al., 2024), or developed safety-field-based risk representations (Wang et al., 2015). However, the validity of these SSMs-based approaches remains uncertain because they have rarely been validated against trajectories preceding actual crashes (Ali et al., 2023). This limitation mainly arises from the rarity and randomness of real-world crashes, which make detailed pre-crash vehicle trajectories difficult to obtain. As a result, it remains unclear whether these measures truly capture crash formation or merely identify severe interactions that do not end in crashes (Chen et al., 2025b). This weakens the interpretability of the resulting risk estimates and limits their reliability for safety intervention.

With the increasing feasibility of drone-based traffic observation, some recent studies have begun to obtain trajectories preceding real-world crashes or other safety-critical events through long-term data collection (Berghaus et al., 2024). Although real-world crashes are rare and random events, sustained observation over extended periods can still yield a limited but useful set of pre-crash trajectories. For example, Shi et al. (2018) validated pre-crash risk indicators using two reconstructed real-world crash cases from surveillance videos. Based on one year of drone recordings, Wang et al. (2024) obtained 20 rear-end crashes to develop and evaluate surrogate safety measures for traffic oscillations, while Chen et al. (2025a) collected 30 lane-changing crashes to develop an individual vehicle crash

[1] School of Civil and Environmental Engineering, Nanyang Technological University, Singapore 639798, Singapore. [2] College of Civil Aviation, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China. [3] School of Transportation, Southeast University, Nanjing 211189, China.

✉ Corresponding authors. E-mail: K. Chen, kequan.chen@ntu.edu.sg; P. Liu, liupan@seu.edu.cn

Received: March 30, 2026; Revised: June 29, 2026; Accepted: September 2, 2026

prediction model. Several studies have used reconstructed per-crash trajectories or in-depth crash data to extract typical scenarios or generate high-risk scenarios (Huang et al., 2024; Zhou et al., 2025). This shift is important because real crash trajectories preserve the actual interaction process before a crash and therefore provide a much stronger empirical basis for linking hazardous interactions to crash outcomes. At the same time, these studies also show how costly and time-consuming such data collection remains in practice. Even with long-term observation, usable crash trajectories still account for only a small proportion of the overall collected data. Previous studies have also noted that limited crash data remain a major obstacle to model development in crash risk prediction (Shew et al., 2013). This suggests that further progress in crash risk prediction cannot rely solely on collecting more real crash data. It also requires more effective ways to learn from the limited real crash data that can realistically be obtained (Jiao et al., 2026).

Large language models (LLMs) have recently attracted increasing attention in the field of transportation because of their strong capabilities in understanding, generation, and reasoning (Chen et al., 2026b; Kuang et al., 2025; Yang et al., 2026; Zhang et al., 2026). These capabilities enable LLMs to interpret complex traffic contexts and infer plausible developments. This is particularly relevant to traffic safety research, where safety-related judgments often depend on multiple interacting factors and rich contextual cues. Motivated by this potential, recent studies have begun to examine the use of LLMs across different traffic safety tasks. One line of work applies LLMs to text-based crash data, such as crash reports and crash narratives, for tasks including crash severity inference (Zhen and Yang, 2025), crash outcome interpretation (Zhao et al., 2025), and crash factor extraction (Arteaga and Park, 2025; Kuang et al., 2024). The basic idea is to use the broad knowledge and reasoning ability of LLMs to extract safety-relevant information and generate structured judgments from complex crash descriptions. More recently, LLM-based research has also extended to crash risk analysis at the individual vehicle level. In this setting, trajectory information can be organized into structured scene descriptions, allowing LLMs to reason about vehicle roles, motion states, relative positions, and recent interaction trends. Related studies have shown that LLMs can extract safety relevant information and generate interpretable judgments from structured traffic safety descriptions. For example, Lan et al. (2024) proposed Traj-LLM, which uses observed trajectories and scene semantics to support future trajectory prediction with pretrained LLMs. (Zhong et al., 2025) further developed an LLM-based system with causal inference and Chain of Thought reasoning for traffic scene risk assessment. Although these studies suggest that LLMs hold considerable promise for crash risk analysis at the individual vehicle level, several issues remain unresolved. Specifically, most existing studies use the reasoning outputs of the LLM itself as direct judgments of crash risk. This makes real-time deployment difficult because large models are computationally expensive and slow to respond (Zhong et al., 2025). In addition, the reasoning results of LLMs may vary across prompts or repeated runs even for the same event, which introduces extra uncertainty into crash risk prediction (Karim et al., 2025). More fundamentally, although LLMs possess broad knowledge, their understanding of traffic safety is learned mainly from general text data. It therefore remains unclear whether such safety understanding can faithfully reflect the actual evolution of risk before real crashes.

To address these gaps, this study develops and evaluates an LLM-guided framework with high data efficiency for crash risk prediction at the level of individual vehicles. The motivation for using the LLM is not to replace prediction models based on vehicle trajectories, but to introduce structured prior knowledge when real crash data are limited. Although the LLM is not trained specifically on the crash trajectories used in this study, it has learned broad knowledge about traffic safety, driving behavior, and crash-related interactions from large-scale pretraining. This knowledge can be used to generate initial risk assessments from structured scene descriptions, which provide useful supervision for training a compact prediction model. Specifically, the proposed framework first converts pre-crash vehicle interaction trajectories into scene descriptions and then uses an LLM to infer crash probability and remaining time to crash. These LLM-derived outputs are distilled into a lightweight temporal graph network and further calibrated with real crash data through fine-tuning. In this way, the framework uses the LLM as an offline teacher to improve data efficiency, while the final deployed model remains a lightweight predictor based on vehicle trajectories. The proposed framework is evaluated using pre-crash trajectories extracted from 109 real-world crash events.

The remainder of this paper is organized as follows. Section 2 describes the data source and the extraction of pre-crash vehicle trajectories. Section 3 introduces the proposed LDF framework, including LLM-guided crash risk inference, distillation into a lightweight temporal graph network, and fine-tuning with real crash labels. Section 4 reports the main experimental results and benchmark comparisons. Section 5 provides extended validation, including sensitivity analyses for different LLM teachers and student models, and evaluates deployment efficiency. Section 6 concludes the paper.

2. Data

The data used in this study are obtained from drone videos collected over multiple years at two busy expressway segments in Nanjing, China. The locations and geometric layouts of the two study sites are shown in Fig. 1(a). The first site is a merging segment, where videos were recorded from October 15, 2021 to March 3, 2023. The second site is a weaving segment, where videos were recorded from September 24, 2023 to August 1, 2024. At both sites, videos were collected on sunny weekdays during the morning peak period from 7:30 AM to 9:30 AM. These two sites and this time window were selected because they showed similar congested traffic conditions during the morning peak and had also been identified in traffic police records as crash-prone locations. This increased the likelihood of capturing video footage before real crash events. Environmental conditions were controlled during data collection. All videos were recorded on sunny days with good visibility, limiting the potential influence of adverse weather, poor lighting, and restricted visibility on vehicle behavior and crash formation. The drones were operated at an altitude of

approximately 300 m and covered 400 m of roadway. They were equipped with 4K cameras and recorded at 30 frames per second. This setup provides sufficient spatial and temporal detail to capture vehicle movements and interactions. In total, 109 crash events were obtained, including 76 at the merging site and 33 at the weaving site.

The recorded videos are processed using a high-precision vehicle trajectory extraction framework. In this framework, the raw footage is first stabilized to remove camera motion caused by wind and manual operation. Vehicles are then detected using a customized detection module with an accuracy exceeding 99%, and the mean localization error between the detected vehicle centers and ground truth is approximately 0.04 m. Based on these detections, a kinematics-driven tracking algorithm is used to reconstruct continuous trajectories for vehicles in the observation area. Finally, a smoothing procedure is applied to remove high-frequency noise, resulting in stable and physically consistent motion profiles suitable for subsequent analysis. Detailed descriptions and validations of this trajectory extraction framework can be found in our previous studies (Chen et al., 2021, 2024, 2025a; Wang et al., 2024). Using this framework, we obtained vehicle trajectories from each drone recording, including the trajectories of vehicles involved in crashes. The extracted trajectory data record the longitudinal position, lateral position, lane boundary location, speed, acceleration, heading angle, vehicle length, and vehicle width of each vehicle. From these trajectories, we further extracted the pre-crash trajectories of the vehicles involved in each crash. Fig. 1(b) shows examples of the extracted pre-crash trajectories, together with image snapshots that illustrate the crash development process. The traffic flow characteristics and vehicle composition were then summarized from the extracted trajectories to clarify the scope of the data used in this study. The average traffic volume was approximately 4,025 vehicles per hour at the merging site and 5,267 vehicles per hour at the weaving site. The corresponding average speeds were 4.1 m/s and 4.7 m/s, respectively. These values show that the analyzed crash events occurred under congested, low speed traffic conditions. We further examined the vehicle lengths during the analysis period. The observed vehicle lengths ranged from 3.9 m to 5.7 m, with a mean of 4.8 m and a standard deviation of 0.4 m. This range indicates that the analyzed vehicles were passenger cars, with no trucks or buses observed.

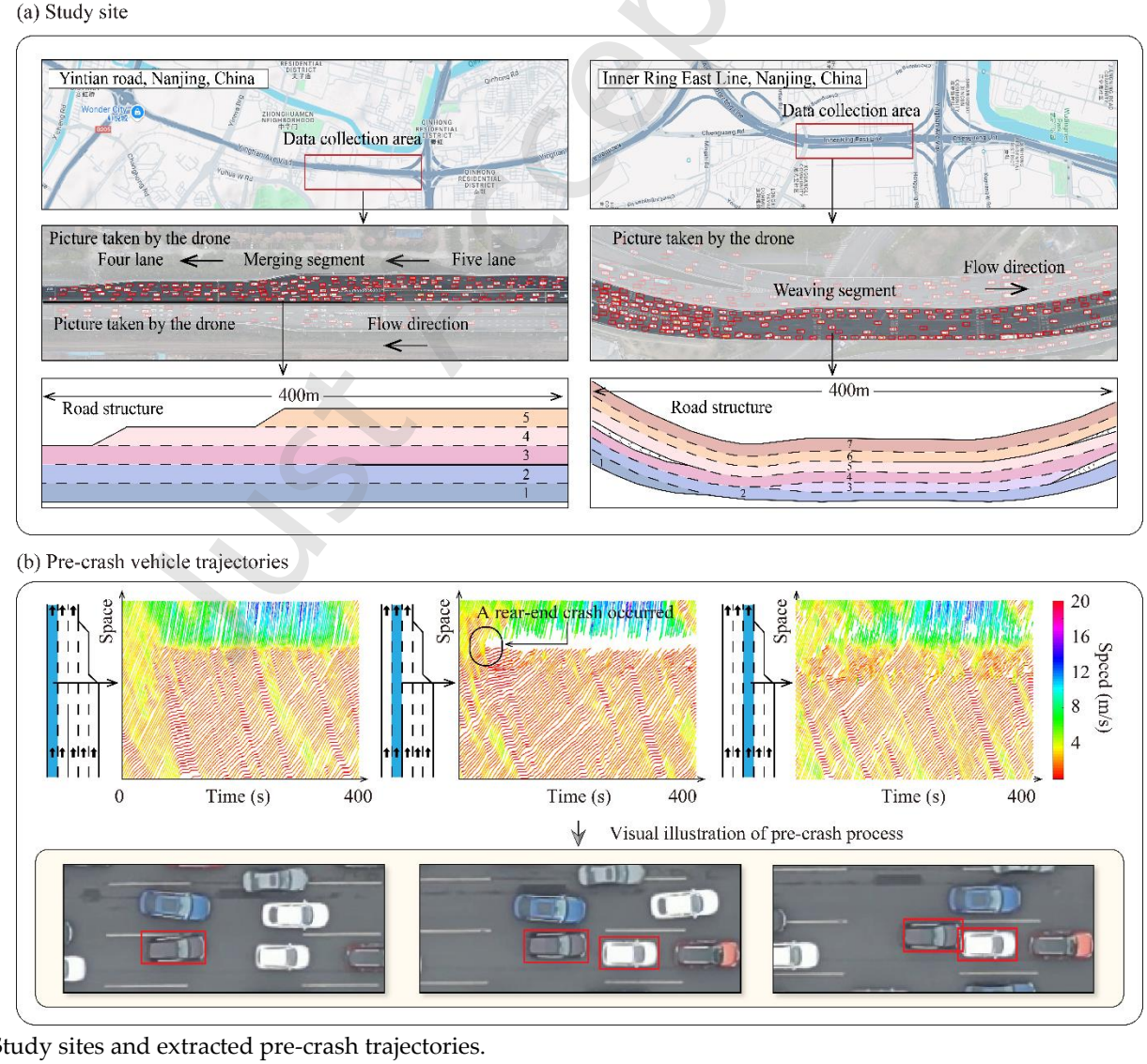

Fig. 1. Study sites and extracted pre-crash trajectories.

3. Methodology

3.1. LLM-Guided Crash Risk Inference

We introduce an LLM-based reasoning module to assess the crash risk level of a multi-vehicle interaction state and generate pseudo labels for offline supervision. An open-source LLM, Qwen3-235B-A22B, is employed as the reasoning engine. The overall workflow of the LLM-based crash risk inference module is illustrated in Fig. 2.

Since the LLM is not well suited to directly process raw numerical trajectories, the vehicle trajectories are first converted into structured textual scene descriptions. Specifically, each pre-crash trajectory is divided into a sequence of decision observations at 0.1 s intervals. Each decision observation is anchored at one prediction time before the crash and is then converted into one textual scene description. For a given decision observation, the input includes the target vehicle and up to six surrounding vehicles, including the leading and following vehicles in the current lane, the left lane, and the right lane. The input describes both the current interaction state at the prediction time and the recent trajectory history over the preceding 1.0 s observation window. The current interaction state includes speed, acceleration, longitudinal position, lateral position, heading angle, vehicle length, and vehicle width for each vehicle. The historical part summarizes the temporal evolution of these variables over the observation window. This design allows the LLM to assess not only the instantaneous spatial configuration, but also how the interaction is evolving toward or away from a crash.

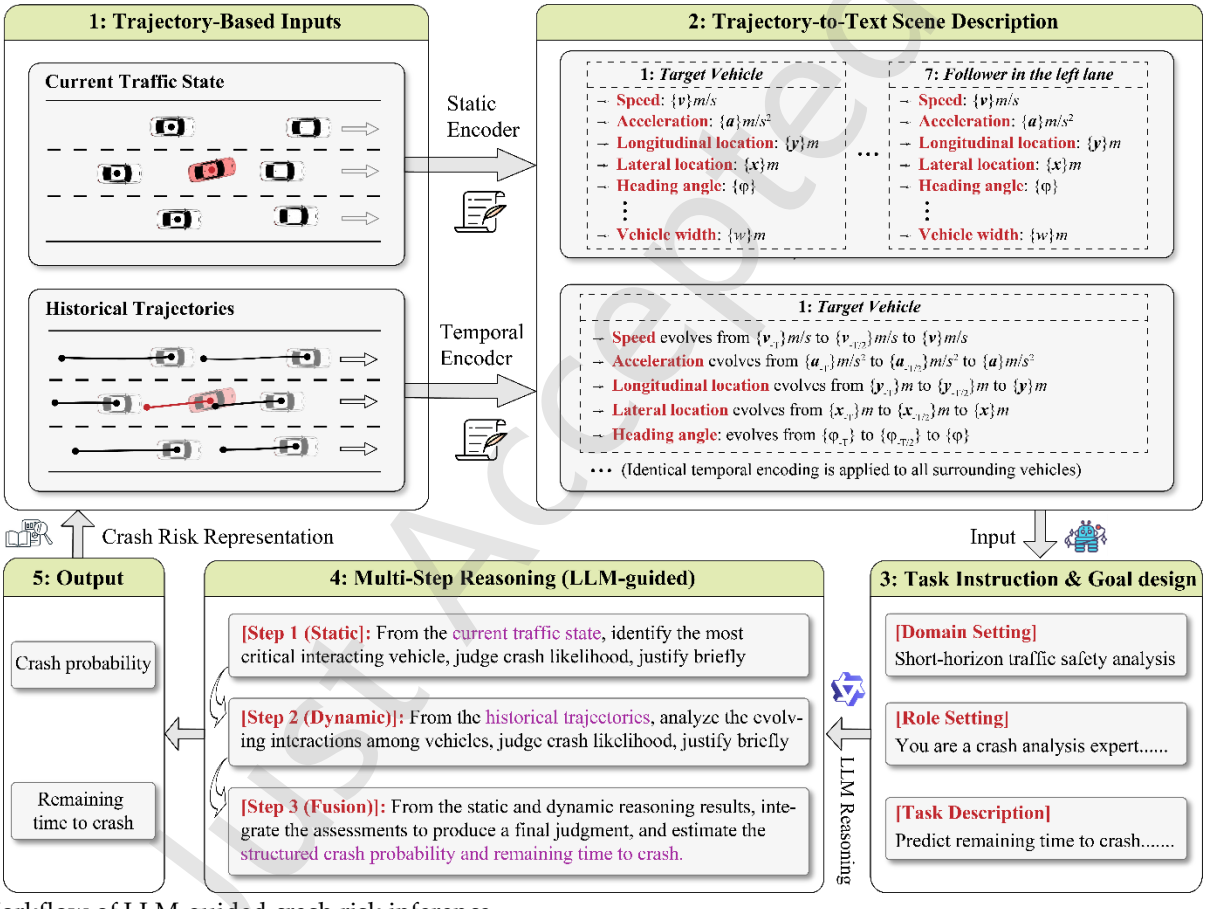

Fig. 2. Workflow of LLM-guided crash risk inference.

The prompt further specifies the role and task of the LLM. Specifically, the LLM is instructed to act as a traffic safety analyst and to assess whether the current interaction is likely to develop into a crash. If so, it then estimates the remaining time to crash. The LLM performs three consecutive reasoning steps. First, it analyzes the current interaction state to determine whether the spatial configuration and motion pattern indicate a dangerous situation. Second, it examines the recent trajectory history to determine whether the interaction trend is becoming more critical. Third, it combines the current state and recent evolution to make the final judgment. To ensure stable parsing and consistent supervision, the final LLM judgment is constrained to a predefined structured format rather than unconstrained free text. Specifically, the output contains the estimated crash probability and the estimated remaining time to crash. This staged reasoning process improves interpretability because the final judgment is grounded in explicit intermediate analyses rather than direct one-step generation. For decision observation n, let S_n denote the textual prompt constructed from the corresponding interaction scene at the corresponding decision instant. The LLM output is defined as

$$(\tilde{p}_n^{LLM}, \tilde{\tau}_n^{LLM}) = f_{LLM}(S_n) \quad (1)$$

where $\tilde{p}_n^{LLM} \in [0,1]$ denotes the crash probability estimated by the LLM, and $\tilde{\tau}_n^{LLM}$ denotes the remaining time to crash estimated by the LLM.

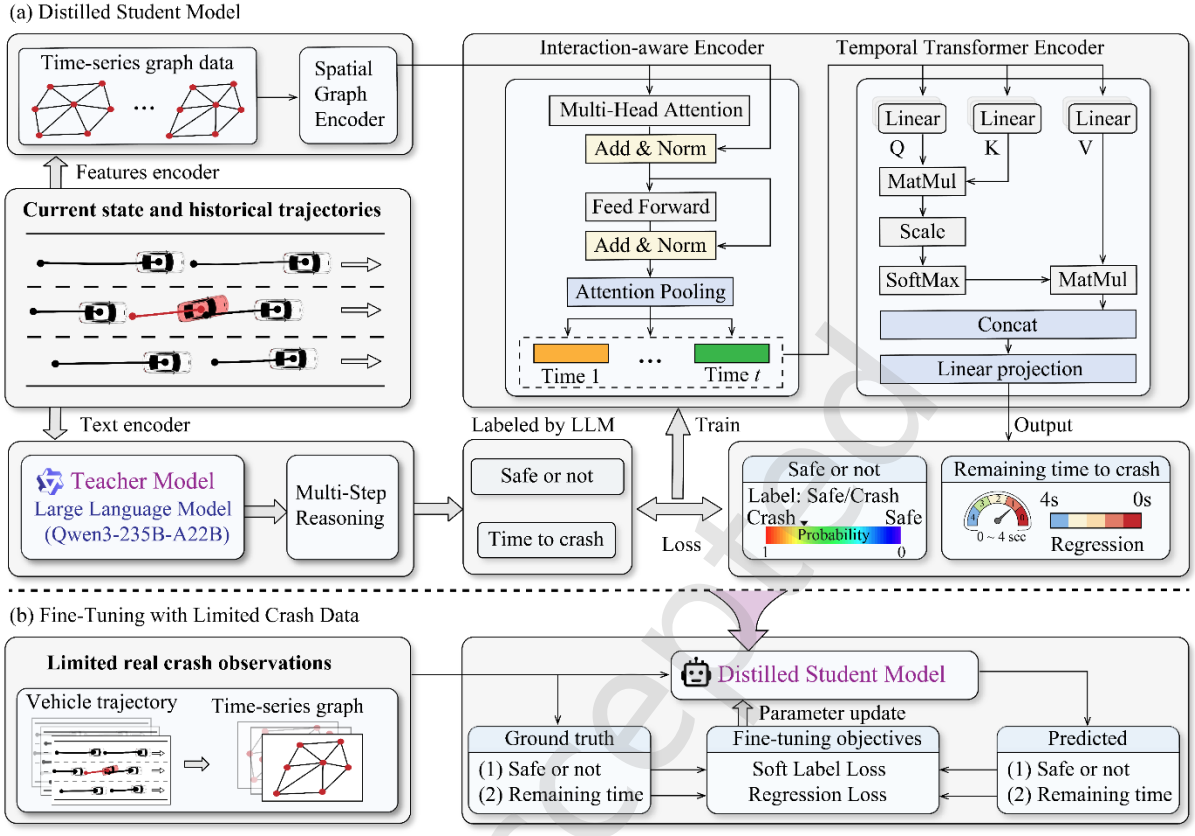

Fig. 3. Distillation and fine-tuning framework of the student model.

3.2. Distilling a Lightweight Temporal Graph Network

Building on the structured outputs generated by the LLM in Section 3.1, we next distill the LLM knowledge into a lightweight temporal graph network for efficient online inference. Here, the LLM serves as the teacher model, and the lightweight temporal graph network serves as the student model. We use a graph network because multi-vehicle interactions can be naturally represented as a graph, in which each vehicle and its interactions with nearby vehicles are explicitly modeled. Recent studies have also shown that graph-based methods are effective for modeling traffic interactions and vehicle behavior (Chen et al., 2026a). Unlike the LLM, which relies on prompts and step-by-step language reasoning, the student model operates directly on vehicle trajectory data while producing the same outputs, namely crash probability and remaining time to crash. This design substantially reduces inference cost and latency. The workflow of the lightweight temporal graph network is shown in Fig. 3(a).

At observation time t, the preceding T observations are represented as an ego-centered temporal interaction graph sequence $G_{(t-T+1:T)} = \{G_{(t-T+1)}, \dots, G_t\}$. For each time step k, $G_k = (V_k, E_k, X_k, R_k)$, where V_k and E_k denote the node set and edge set, respectively, $X_k \in \mathbb{R}^{|V_k| \times d_x}$ is the node feature matrix, and $R_k \in \mathbb{R}^{|E_k| \times d_r}$ is the edge feature matrix. In this study, T is set to 10, corresponding to a 1.0 s interaction history sampled at 0.1 s intervals. Each graph contains the target vehicle and up to six surrounding vehicles in fixed relative positions around the target vehicle. For vehicle node $i \in V_t$, the node feature vector is defined as

$$x_{i,t} = [v_{i,t}, a_{i,t}, \psi_{i,t}] \tag{2}$$

where $v_{i,t}$, $a_{i,t}$, and $\psi_{i,t}$ denote the vehicle speed, acceleration, and heading angle, respectively.

For each directed interaction edge $(0,j) \in E_t$, where 0 denotes the target vehicle and j denotes one of its surrounding vehicles, the edge feature vector is defined as

$$r_{0j,t} = [d_{0j,t}, \Delta v_{0j,t}, \Delta x_{0j,t}, \Delta y_{0j,t}] \tag{3}$$

where $d_{0j,t}$ is the remaining distance between the target vehicle and vehicle j, $\Delta v_{0j,t}$ is the relative speed, and $\Delta x_{0j,t}$ and $\Delta y_{0j,t}$ are the relative longitudinal and lateral offsets, respectively. If a surrounding vehicle is absent at a predefined position, its corresponding node and edge features are zero-padded and accompanied by binary masks so that the graph layout remains consistent across observations.

As shown in Fig. 3(a), the student model consists of a spatial graph encoder, an interaction-aware encoder, and a temporal attention encoder. At each time step t, the spatial graph encoder first projects the raw node and edge features into a latent space for subsequent interaction modeling:

$$H_t^0 = X_t W_x + b_x, \quad M_t^0 = R_t W_r + b_r \tag{4}$$

where H_t^0 and M_t^0 denote the initial node and edge embeddings, respectively, W_x and W_r are learnable weight matrices, and b_x and b_r are learnable bias vectors. Let $h_{i,t}$ denote the embedding of node i extracted from H_t^0, and let $m_{oj,t}$ denote the embedding of edge $(0,j)$ extracted from M_t^0. To summarize the overall interaction information at time step t, the edge embeddings are transformed by a shared multilayer perceptron and then aggregated as

$$f_{r,t} = \text{MaxPool}\left(\text{MLP}\left(M_t^0\right)\right) \tag{5}$$

where $f_{r,t}$ is a global interaction descriptor. To explicitly incorporate pairwise interactions into the subsequent encoder, an edge-aware interaction token is constructed for each target-neighbor pair:

$$s_{0j,t}^0 = [W_0 h_{0,t} || W_n h_{j,t} || W_m m_{0j,t} || f_{r,t}] \tag{6}$$

where $||$ denotes concatenation, W_0, W_n, and W_m are learnable projection matrices, $j \in \mathcal{N}_0$, and $\mathcal{N}_0$ denotes the set of available neighboring vehicles around the target vehicle.

In this way, the edge features are explicitly incorporated through $m_{0j,t}$ into the interaction tokens processed by the interaction-aware encoder. Let $S_t^0 = \{s_{0j,t}^0\}_{j \in \mathcal{N}_0}$ denote the set of interaction tokens at time step t. The interaction-aware encoder refines these tokens as

$$S_t^{(l)} = \text{TransformerBlock}\left(S_t^{(l-1)}\right) \tag{7}$$

where $l = 1, \ldots, L$, and TransformerBlock$(\cdot)$ denotes a standard encoder block consisting of multi-head self-attention, residual connections, layer normalization, and a feed-forward network. After the final layer L, attention pooling is applied to obtain the scene embedding g_t

$$\beta_{0j,t} = \frac{\exp\left(q^\top \tanh\left(W_p s_{0j,t}^L\right)\right)}{\sum_{k \in \boldsymbol{\mathcal{N}}_0} \exp\left(q^\top \tanh\left(W_p s_{0k,t}^L\right)\right)} \tag{8}$$

$$g_t = \sum\nolimits_{j \in \boldsymbol{\mathcal{N}}_0} \beta_{0j,t}\, s_{0j,t}^L \tag{9}$$

where W_p and q are learnable parameters. This design allows the model to adaptively emphasize the most safety-critical interactions at each time step.

The sequence of scene embeddings $\{g_{(t-T+1)}, \ldots, g_t\}$ is then fed into the temporal attention encoder to capture the evolution of the interaction patterns over the historical observation window. Let $G^{seq} = \{g_{(t-T+1)}, \ldots, g_t\}^\top$. The temporal attention encoder applies multi-head self-attention over the temporal dimension. For each attention head, the query, key, and value matrices are obtained by linear projection:

$$Q = G^{seq} W_Q, \quad K = G^{seq} W_K, \quad V = G^{seq} W_V \tag{10}$$

The attention output is computed as

$$\text{Attn}(Q, K, V) = \text{SoftMax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)\text{V} \tag{11}$$

where d_k is the key dimension. The outputs of all attention heads are concatenated and linearly projected to obtain the final temporal representation:

$$h = \text{Linear}\left(\text{Concat}(A_1, A_2, \ldots, A_m)\right) \tag{12}$$

where A_1,…, A_m are the outputs of the m attention heads. Based on h, two task-specific output heads are used to predict crash probability and remaining time to crash, respectively:

$$p^S = \sigma\left(f_{\text{cls}}(h)\right), \quad \hat{\tau}^S = T_{\max} \cdot \sigma\left(f_{\text{reg}}(h)\right) \tag{13}$$

where $p^S \in (0,1)$ denotes the predicted crash probability, $\hat{\tau}^S$ denotes the estimated remaining time to crash, and $\sigma(\cdot)$ denotes the sigmoid function. Here, $T_{\max}$ is the upper bound imposed on the regression output so that $\hat{\tau}^S$ is restricted to the range $[0, T_{\max}]$.

Using the LLM outputs defined in Section 3.1 as supervision, the student model is trained to match the structured assessments produced by the LLM. The probability distillation loss is defined as

$$L_{\text{cls}}^{\text{dist}} = \left(\frac{1}{N}\right)\sum_{n=1}^{N} \text{BCE}(p_n^S, \tilde{p}_n^{LLM}) \tag{14}$$

where N is the mini-batch size, and BCE$(\cdot)$ denotes the binary cross entropy loss used to measure the discrepancy between the student-predicted crash probability and the LLM-estimated crash probability.

For the prediction of remaining time to crash, the regression head is updated only when both the LLM and the student predict that a crash will occur. Let $\hat{y}_n^{LLM}$ and $\hat{y}_n^S$ denote the corresponding binary crash decisions obtained from $\hat{p}_n^{LLM}$ and p_n^S, respectively, using a predefined threshold of 0.5. We then define

$$\delta_n = \begin{cases} 1, if\ \hat{y}_n^{LLM} = 1\ and\ \hat{y}_n^S = 1 \\ 0, otherwise \end{cases} \tag{15}$$

The remaining time to crash regression loss can be computed as

$$L_{\text{reg}}^{\text{dist}} = \frac{1}{\sum_{n=1}^{N} \delta_n} \sum_{n=1}^{N} \delta_n\, \text{Huber}(\hat{\tau}_n^{LLM}, \hat{\tau}_n^S) \tag{16}$$

where Huber$(\cdot)$ denotes the Huber loss. The overall

distillation objective is

$$L^{\text{dist}} = \lambda_{\text{cls}} L_{\text{cls}}^{\text{dist}} + \lambda_{\text{reg}} L_{\text{reg}}^{\text{dist}} \tag{17}$$

where λ_{cls} and λ_{reg} are weighting coefficients.

3.3. Fine-Tuning with Real Crash Data

In this section, we describe how the student model is further fine-tuned using real crash data with ground-truth annotations. To construct fine-tuning labels from real crash trajectories, we set a critical warning horizon τ_c. For a crash event, an observation is labeled as 1 if its remaining time to crash is no greater than τ_c, and 0 otherwise. Let τ_n^{gt} denote the ground truth remaining time to crash for observation n. The corresponding binary label is defined as

$$y_n^{gt} = \begin{cases} 1, \tau_n^{gt} \le \tau_c \\ 0, \tau_n^{gt} > \tau_c \end{cases} \tag{18}$$

The choice of τ_c should balance practical usefulness and prediction difficulty. If τ_c is set too short, such as immediately before the crash, the prediction task becomes relatively easy because the interacting vehicles are already in an extreme conflict state. However, such a setting provides insufficient time for a control system or driver to respond, thereby limiting the practical value of proactive safety intervention. In contrast, an excessively long horizon may introduce weakly informative early-stage observations that are less directly related to the imminent crash. In this study, the largest warning horizon considered is 4 s. Accordingly, $T_{\max}$ in Eq. (13) is set to 4 s so that the regression output covers the full range of candidate warning horizons.

After distillation, the student model is initialized with the distilled parameters and further trained end-to-end using the real crash data. All parameters of the student model are initialized from the distilled model and are further updated during supervised fine-tuning. Full fine-tuning is adopted because the student model is lightweight and the fine-tuning task remains closely aligned with the preceding distillation task, except that the supervision is now provided by real crash data rather than LLM outputs (Jiao et al., 2020). The classification fine-tuning loss is defined as

$$L_{\text{cls}}^{\text{ft}} = \left(\frac{1}{N}\right) \sum_{n=1}^{N} BCE\left(p_n^S, y_n^{gt}\right) \tag{19}$$

Consistent with the gating strategy in Eq. (15), the remaining time to crash regression loss is activated only when the ground truth indicates that a crash will occur within the predefined horizon and the student also predicts that a crash will occur. We therefore define

$$\gamma_n = \begin{cases} 1, if\ y_n^{gt} = 1\ and\ \hat{y}_n^S = 1 \\ 0, otherwise \end{cases} \tag{20}$$

Based on this gating variable, the remaining time to crash regression loss is given by

$$L_{\text{reg}}^{\text{ft}} = \frac{1}{\sum_{n=1}^{N} \gamma_n} \sum_{n=1}^{N} \gamma_n \,\text{Huber}\left(\hat{\tau}_n^S, \tau_n^{gt}\right) \tag{21}$$

The overall fine-tuning objective combines the classification and remaining time to crash regression losses:

$$L^{\text{ft}} = \mu_{\text{cls}} L_{\text{cls}}^{\text{ft}} + \mu_{\text{reg}} L_{\text{reg}}^{\text{ft}} \tag{22}$$

where μ_{cls} and μ_{reg} are weighting coefficients.

For brevity, the overall framework consisting of **L**LM-based pseudo-labeling, **d**istillation, and **f**ine-tuning with real crash data is hereafter referred to as **LDF**.

3.4. Evaluation Metrics

We evaluate the performance of LDF using two groups of metrics, one for warning classification and the other for estimating the remaining time to crash. For warning classification, we first construct a confusion matrix following standard binary-classification evaluation practice (Fawcett, 2006). Observations within the warning horizon are treated as positive observations, while observations outside the warning horizon are treated as negative observations. Accordingly, true positives (TP) denote correctly issued warnings within the warning horizon, false negatives (FN) denote missed warnings within the warning horizon, false positives (FP) denote false warnings outside the warning horizon, and true negatives (TN) denote correctly rejected non-warning observations. Based on the confusion matrix, Recall and the false alarm rate (FAR) are computed as Recall = TP / (TP + FN) and FAR = FP / (FP + TN), respectively. Recall measures the ability of the model to detect observations within the warning horizon. A higher Recall means that fewer true warning cases are missed. FAR measures the proportion of observations outside the warning horizon that are incorrectly identified as warnings. A lower FAR means that fewer false alarms are generated. To provide a single summary measure of warning performance, we further report the F1 score:

$$\text{F1} = 2\text{TP}/(2\text{TP} + \text{FP} + \text{FN}) \tag{23}$$

The F1 score jointly considers missed warnings and false warnings. A larger F1 score indicates better overall warning performance, with a maximum value of 1. In computing Recall, FAR, and F1, the predicted warning probabilities are converted into binary warning labels using a decision threshold of 0.5. In addition to these metrics, we use the area under the receiver operating characteristic curve (AUC) to evaluate the ranking ability of the model. Unlike Recall, FAR, and F1, which require binary warning labels, AUC is computed from the continuous warning scores. It measures whether the model assigns higher warning scores to observations within the warning horizon than to observations outside the warning horizon. Let N_+, and N_- denote the numbers of observations within and outside the warning horizon in the evaluation set, respectively. The

AUC is computed as

$$AUC = \frac{1}{N_+ N_-} \sum_{i:y_i^{gt}=1} \sum_{j:y_j^{gt}=0} \left[\boldsymbol{I}(p_i^S > p_j^S) + \frac{1}{2} \boldsymbol{I}(p_i^S = p_j^S) \right] \quad (24)$$

where p_i^S and p_j^S are the predicted probabilities assigned by the student to observations within and outside the warning horizon, respectively, and $\boldsymbol{I}(\cdot)$ denotes the indicator function. A larger AUC indicates better discrimination between observations within and outside the warning horizon.

For remaining time to crash estimation, we report the MAE only for observations for which the ground truth indicates that the crash will occur within the warning horizon, and the student also predicts that the observation falls within the warning horizon. Then, we define

$$\delta_n^{eval} = \begin{cases} 1, if\, y_n^{gt} = 1 \; and \; \hat{y}_n^S = 1 \\ 0, otherwise \end{cases} \quad (25)$$

Based on this indicator, the MAE is computed as

$$MAE = \frac{1}{\sum_{n=1}^{N} \delta_n^{eval}} \sum_{n=1}^{N} \delta_n^{eval} \left| \hat{\tau}_n^S - \tau_n^{gt} \right| \quad (26)$$

4. Results

4.1. LLM Reasoning and Distillation Results

Before presenting the distillation results, we first use a representative case to illustrate how the LLM assesses a risky interaction state in Fig. 4. In this case, the observation time point is 0.5 s before the actual crash. For visual clarity, the upper panel of Fig. 4 shows several representative historical interaction states from 1.5 s to 0.5 s at 0.2 s intervals, rather than the full input sequence used by the model. The target vehicle is highlighted in yellow. During this period, a vehicle in the adjacent left lane accelerates and moves laterally toward the space ahead of the target vehicle. As the interaction develops, the situation becomes progressively more critical. By the current frame, the scene has evolved into a near-crash state with a very limited safety margin.

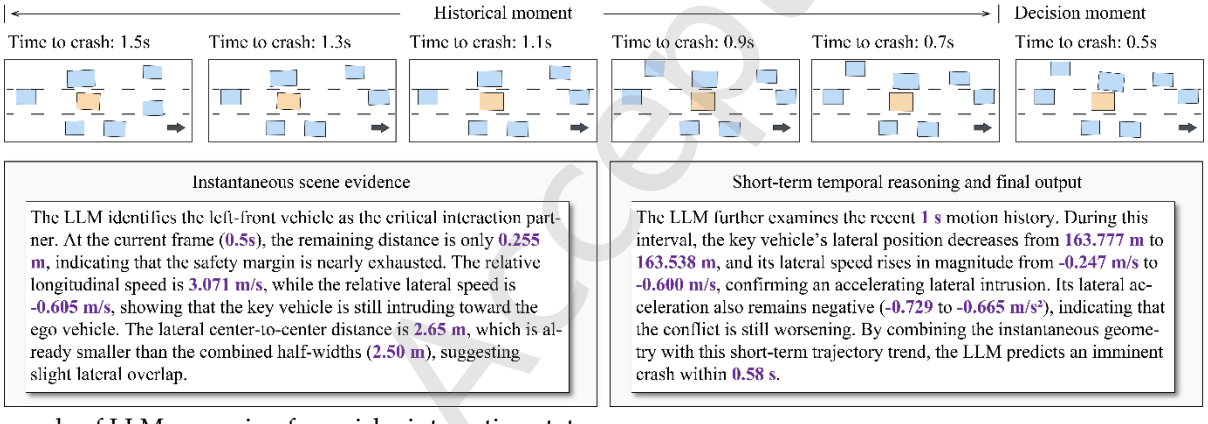

Fig. 4. Example of LLM reasoning for a risky interaction state.

The lower part of Fig. 4 summarizes the LLM reasoning process. Based on the current interaction state, the LLM identifies the vehicle in the left lane as the critical interaction partner. It detects a very small remaining distance of 0.255 m and a relative longitudinal speed of 3.071 m/s. It also finds that the lateral center-to-center distance is 2.65 m, which is already close to the combined half-widths of the two vehicles (2.50 m). This indicates that the lateral clearance is extremely limited at the observation time point. The LLM then examines the trajectory history and finds that the interacting vehicle continues to intrude laterally toward the target vehicle. Its lateral position decreases from 163.777 m to 163.538 m, while the magnitude of its lateral speed increases from -0.247 m/s to -0.600 m/s. The lateral acceleration also remains negative during this period. By combining the current interaction state with the recent trajectory history, the LLM concludes that this scene is likely to result in a crash within the prediction horizon. It therefore judges that a crash will occur, and estimates the remaining time to crash as 0.58 s.

Using the same reasoning procedure, we then apply the LLM to a large pool of non-crash observations to construct pseudo labels for student distillation. These observations are extracted from non-crash traffic segments recorded at the same study sites and are not part of the 109 crash events used for fine-tuning and testing. For each non-crash observation, the LLM first determines whether the current interaction is safe or dangerous. For observations assigned to the danger class, the LLM further provides an estimated remaining time to crash. Based on these structured outputs, we select 10,000 safe observations and 10,000 danger observations. To avoid bias in the pseudo-labeled training set, the danger observations are further stratified according to the remaining time to crash estimated by the LLM. Specifically, the interval from 0 s to 4 s is divided into 0.5 s bins, and the same number of danger observations is drawn from each bin so that the final pseudo-labeled set more evenly covers different levels of short-horizon risk. These pseudo-labeled non-crash observations are then used to distill the lightweight temporal graph network in the next stage.

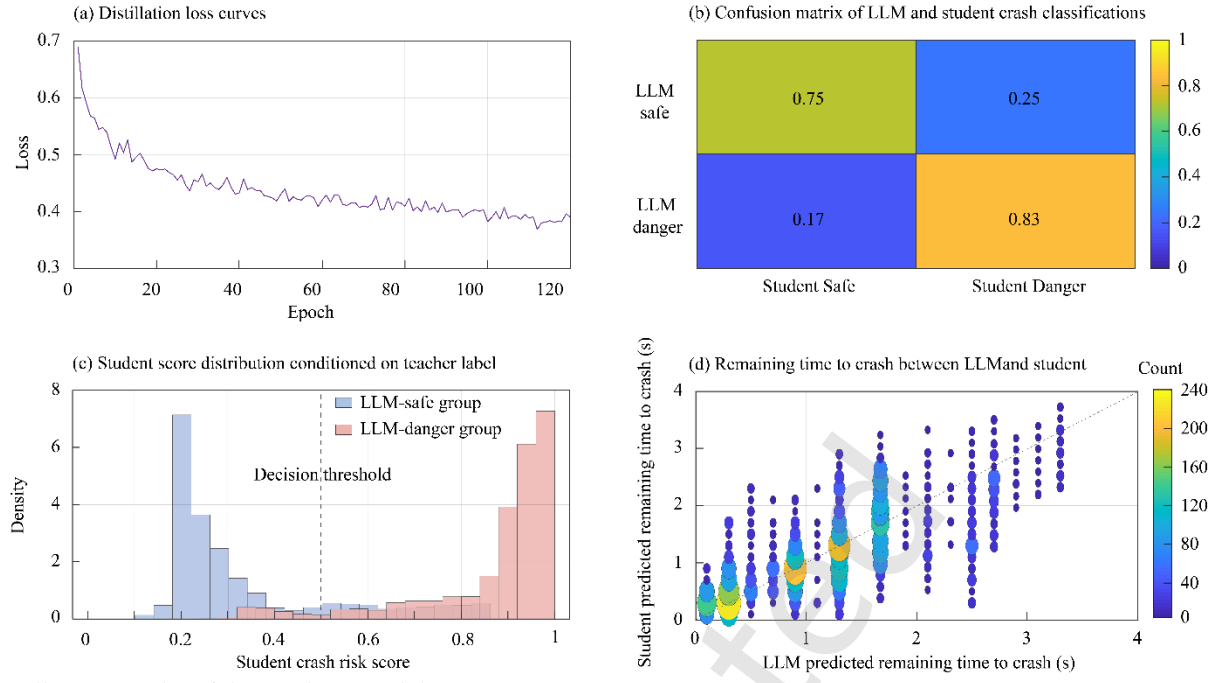

Fig. 5. Distillation results of the student model.

Fig. 5 presents the distillation results of the student model. As shown in Fig. 5(a), the distillation loss decreases steadily and gradually stabilizes as training proceeds, indicating stable convergence of the student model under LLM supervision. This knowledge transfer is also reflected in the student's score distribution in Fig. 5(b). In this panel, the blue histogram represents observations labeled as safe by the LLM, whereas the red histogram represents observations labeled as dangerous by the LLM. The dashed vertical line indicates the decision threshold. Most red bars are concentrated in the high score region, whereas most blue bars are concentrated in the low score region, with only limited overlap near the threshold. This separation indicates that the student model assigns higher danger scores to dangerous observations than to safe observations, suggesting that the student model has learned the risk ordering provided by the LLM.

Fig. 5(c) further supports this result by comparing the classification outputs of the LLM and the student model. Specifically, 75% of the observations classified as safe by the LLM are also classified as safe by the student model, while 83% of the observations classified as dangerous by the LLM are also classified as dangerous by the student model. By contrast, 25% of the observations classified as safe by the LLM are classified as dangerous by the student model, and 17% of the observations classified as dangerous by the LLM are classified as safe by the student model. These results indicate that the student model largely preserves the classification pattern of the LLM. This agreement is not limited to classification. As shown in Fig. 5(d), the remaining time to crash predicted by the student model also agrees well with that predicted by the LLM for observations identified as dangerous. Most points are distributed around the diagonal, and the MAE between the remaining times to crash estimated by the LLM and the student model is 0.41 s. This result suggests that the student model has learned not only the LLM's classification judgment, but also its assessment of remaining time to crash. Taken together, the results in Fig. 5 show that the student model captures the key crash risk information provided by the LLM, which provides a meaningful initialization for the subsequent fine-tuning stage using real crash data.

4.2. Effect of Warning Horizon

For all fine-tuning and subsequent quantitative evaluations, the 109 crash events are divided into 70% for training and 30% for testing. Following Section 3.3, we examine warning horizons from 1 s to 4 s at 1 s intervals, with 4 s as the maximum candidate horizon. The warning horizon determines how early a potential crash risk is identified before the actual crash, and therefore reflects the lead time available for possible safety intervention. For a given horizon, observations within that horizon before the crash are treated as positive observations, whereas earlier pre-crash observations are treated as negative observations. All models in this subsection are initialized from the distilled student and fine-tuned under the same settings, including a learning rate of 1e-4, a batch size of 128, 120 training epochs, a hidden dimension of 32, a dropout rate of 0.5, and a weight decay of 1e-4. The experiments are conducted on an NVIDIA GeForce RTX 5060 Ti GPU.

To evaluate performance at different pre-crash stages, we divided the pre-crash period into 1 s bins, namely 0-1 s, 1-2 s, 2-3 s, and 3-4 s. This bin-wise evaluation allows us to assess whether predictive quality remains stable as the warning horizon is extended. For each target bin, we computed the AUC and the MAE of remaining time to crash prediction. The results are summarized in Fig. 6. Fig. 6(a) shows that performance remains relatively stable under the 1 s, 2 s, and 3 s warning horizons, but drops noticeably under the 4 s warning horizon. Specifically, for the 0-1 s bin, the AUC is 0.84, 0.86, and 0.85 under the 1 s, 2 s, and 3 s settings, respectively, but falls to 0.78 when the warning horizon is extended to 4 s. A similar pattern appears in the 1-2 s bin. The AUC remains high at 0.82 and 0.83 under the 2 s and 3 s settings, but decreases to 0.75 under the 4 s setting. The same

trend continues in the 2-3 s bin, where the 3 s setting still achieves an AUC of 0.78, whereas the 4 s setting drops to 0.68. Moreover, the additional 3-4 s bin introduced only under the 4 s setting performs worst, with an AUC of only 0.60. These results suggest that extending the warning horizon to 4 s introduces earlier pre-crash states with weaker crash-related signals. As a result, the distinction between observations within and outside the warning horizon becomes less clear, which may reduce classification performance not only in the added 3-4 s bin but also in the earlier bins.

Fig. 6(b) shows a similar pattern for remaining time to crash prediction error. For the 0-1 s bin, the MAE remains low at 0.15, 0.16, and 0.15 under the 1 s, 2 s, and 3 s warning horizons, but increases to 0.22 under the 4 s warning horizon. For the 1-2 s bin, the MAE is 0.22 at 2 s and 0.22 at 3 s, but rises to 0.32 at 4 s. In the 2-3 s bin, the MAE further increases from 0.27 under the 3 s setting to 0.36 under the 4 s setting, while the 3-4 s bin reaches the largest error of 0.42. These results indicate that the 2 s and 3 s warning horizons achieve similar remaining time to crash prediction accuracy in the shared pre-crash intervals, but the 3 s setting covers a longer pre-crash period. By contrast, although the 4 s warning horizon extends the prediction range further, it also leads to a clear increase in prediction error. Considering both the AUC and the MAE, the 3 s warning horizon provides the best overall trade-off between predictive performance and warning horizon. From an application perspective, this horizon is also more practical than the shorter and longer alternatives considered here. A 1 s horizon may be too close to the crash to provide sufficient time for proactive warning or control, whereas a 4 s horizon introduces earlier pre-crash states with weaker crash-related signals and lower prediction reliability. The 3 s horizon therefore provides a useful balance between early risk identification and dependable prediction for short-horizon crash risk warning, and is used in the subsequent analyses.

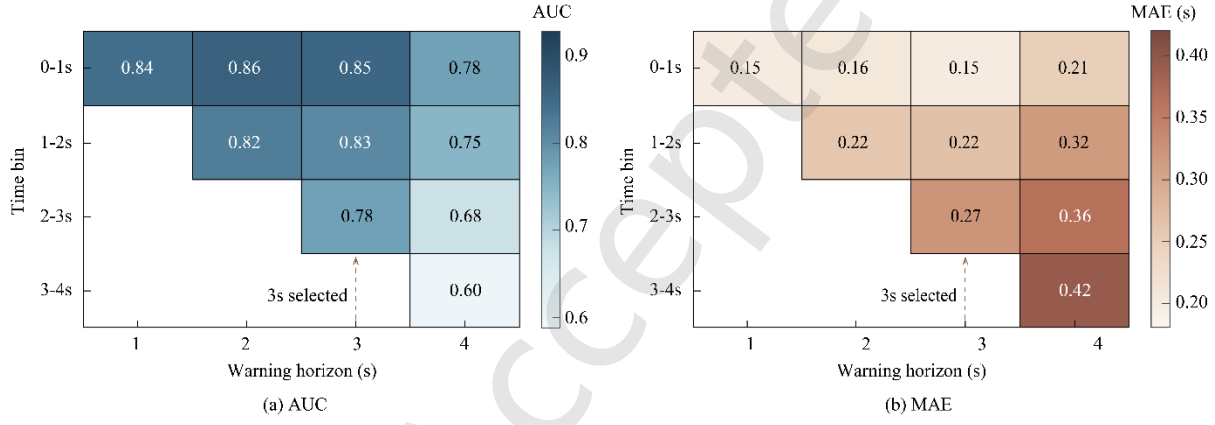

Fig. 6. Performance across warning horizons.

4.3. Benchmark Comparison

In this section, we compare the performance of the proposed model with a set of expanded benchmark methods. The comparison is conducted under the selected 3 s warning horizon using the same 70%/30% train-test split. These benchmarks include direct LLM inference, direct supervised training without distillation, commonly used learning-based models trained with crash data, and surrogate safety measures (SSMs).

The first benchmark is the LLM itself, which directly produces crash judgments and remaining time to crash estimates from the trajectory-to-text prompts. This comparison examines whether the proposed student model can retain the interaction knowledge provided by the LLM and further improve it through distillation and fine-tuning. Then, we use the same lightweight graph architecture as the second benchmark. It is directly trained with real crash data and does not include the LLM distillation stage. For simplicity, we refer to this benchmark as the Direct model. This comparison is introduced to isolate the contribution of the distillation stage and to test whether the LLM knowledge can improve the student beyond direct supervised training alone. In addition, we include LSTM, Transformer (Vaswani et al., 2017), GCN (Kipf and Welling, 2016), and GAT (Velickovic et al., 2018) as learning-based benchmarks. They are trained using the same real crash data as the Direct model, but with different model architectures. LSTM and Transformer are included as commonly used sequence-learning models, while GCN and GAT are included as commonly used graph-learning models. This comparison is used to examine whether common learning-based models can also be trained effectively from the available crash data. Finally, two commonly used SSM-based methods are considered as benchmarks: Time-to-Collision (TTC) and Modified Time-to-Collision (MTTC). These methods are selected as they are widely used in crash risk assessment and also measure conflict severity in terms of time to collision. Under the selected 3 s warning horizon, TTC classifies the current scene as dangerous when the minimum instantaneous TTC between the target vehicle and its surrounding vehicles is smaller than the warning threshold. For the regression task, this minimum TTC is used as the corresponding remaining time to crash estimate. MTTC follows a similar logic but further considers relative acceleration when estimating the potential collision time. Unlike the other benchmarks, these SSM-based methods rely only on the current interaction state and do not use learned representations from crash data.

Following the evaluation metrics defined in Section 3.4, the comparison results are reported in Table 1. The SSM-based methods show limited performance in predicting crash risk for an individual vehicle. This result is understandable because SSMs are mainly designed to

describe the severity of traffic conflicts. Their common use is to analyze how conflicts form and then support safety improvement by reducing the frequency or severity of such conflicts. Previous studies have also shown that directly applying conflict measures to real-time pre-crash prediction for individual vehicles remains challenging (Chen et al., 2025a). In addition, these methods depend on simplified and manually specified assumptions. For example, TTC assumes that two vehicles maintain their current speed and moving direction over a short future period. MTTC further considers relative acceleration, but it still relies on a simplified motion description. Such assumptions are difficult to fully satisfy in complex real traffic interactions.

Table 1. Performance comparison between the proposed LDF model and benchmark methods

Methods	Recall	FAR	F1 score	AUC	MAE
TTC	0.63	0.15	0.71	0.67	0.43
MTTC	0.85	0.41	0.75	0.63	0.35
LSTM	0.71	0.21	0.74	0.70	0.26
Transformer	0.69	0.17	0.74	0.69	0.28
GCN	0.81	0.21	0.80	0.74	0.25
GAT	0.83	0.30	0.78	0.72	0.24
Direct	0.85	0.19	0.83	0.77	0.25
LLM	0.73	0.28	0.73	0.74	0.27
LDF (our proposed model)	0.89	0.14	0.88	0.83	0.18

By contrast, learning-based models trained with real crash data achieve better performance. However, their effectiveness also depends on how the vehicle interaction process is represented. LSTM and Transformer use temporal trajectory sequences as inputs, and both improve the AUC and reduce the MAE compared with the SSM-based methods. This suggests that learning temporal patterns from crash data provides more useful information than relying only on instantaneous conflict indicators. However, crash formation also depends strongly on the interactions between the target vehicle and its surrounding vehicles. Since these interactions can be naturally represented as a graph, graph-based models provide a more suitable structure for describing risk evolution involving multiple vehicles. This is also consistent with previous studies showing the value of graph representations for modeling vehicle interactions. When graph-based architectures are used, GCN, GAT, and the Direct model further improve the overall performance compared with LSTM and Transformer.

Although the LLM is not trained on the real crash data in this study, it performs better than the SSM-based methods and is comparable to several learning-based baselines. This indicates that the LLM has some ability to interpret crash risk from the trajectory-based scene descriptions. A possible reason is that the trajectory to text transformation presents vehicle positions, speeds, relative distances, and recent motion trends in a structured semantic form. This allows the LLM to use its pretrained reasoning ability and general traffic knowledge to identify potentially hazardous interaction patterns. However, direct LLM inference alone is still not sufficient to achieve the best warning performance. Further improvement is achieved when LLM-based distillation is followed by fine-tuning on real crash data. Under this setting, the proposed LDF model achieves the best overall performance among all compared methods, with the highest Recall, F1 score, and AUC, the lowest FAR, and the lowest MAE. These results suggest that the LLM provides useful structured prior knowledge, while the fine-tuning stage aligns this knowledge with real crash data. Their combination therefore yields better performance than direct LLM inference, direct supervised training, conventional learning-based models, and SSM-based methods.

4.4. Data Efficiency under Different Amounts of Training Data

The main motivation for introducing the LLM is to use its structured prior knowledge to reduce the dependence of the final model on large amounts of real crash data. To examine this point, we further evaluate model performance under different amounts of fine-tuning data. Specifically, starting from the same 70% training split used above, we divide the training data into ten subsets, from 10% to 100%, while keeping the 30% test set fixed in all experiments. Fig. 7 shows the corresponding results under the selected 3 s warning horizon. Because the LLM and TTC do not require fine-tuning on real crash data, their results remain unchanged and serve as fixed reference baselines. By contrast, the performance of the Direct model and the proposed model vary with the amount of training data.

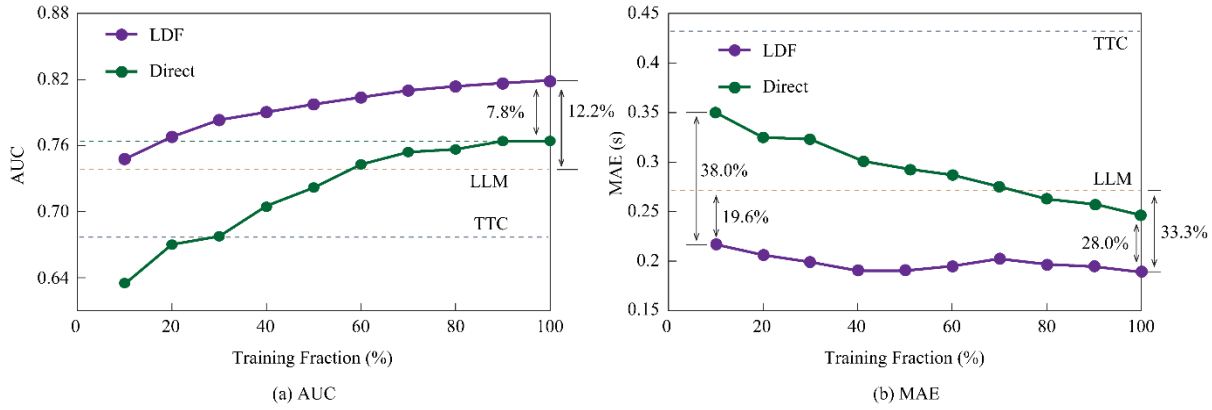

Fig. 7. Data efficiency with limited fine-tuning data.

As shown in Fig. 7, the two trainable models exhibit clearly different patterns as the amount of training data changes. When only 10% of the training data is used, the Direct model performs poorly and even remains below the TTC baseline. As the amount of real crash data increases, its performance improves gradually. However, it begins to outperform TTC only when the training fraction exceeds about 30%, and it does not outperform the LLM baseline until the training fraction exceeds about 70%. This pattern is reflected in both metrics, with AUC increasing and MAE decreasing gradually as more real crash data become available.

By contrast, the proposed model already outperforms all comparison methods when only 10% of the training data is used. This advantage can be observed in both AUC and MAE, and it is especially pronounced for MAE. Specifically, at the 10% setting, the MAE of the proposed model is 19.6% lower than that of the LLM baseline and 38.0% lower than that of the Direct model. With the full fine-tuning data, the proposed model further improves AUC by 7.8% and reduces MAE by 28.0% relative to the Direct model. These results indicate that the LLM-generated supervision provides an effective initialization for the student model, allowing the subsequent fine-tuning stage to achieve strong performance even when real crash labels are scarce. In other words, the main benefit of the proposed framework is not only higher final accuracy, but also substantially better data efficiency when real crash labels are limited.

5. Extended Validation and Deployment Analysis

5.1. Sensitivity to Different LLM Teachers

In this section, we examine whether the proposed framework is robust to the choice of LLM teacher. In addition to the default Qwen3 teacher used in the main experiments, three alternative LLM teachers are considered, namely DeepSeek-V4-Flash, GPT-5.4-mini, and GLM-4-Plus. These models represent different model families and deployment providers. The purpose of this comparison is not to rank the LLMs themselves, but to test whether the proposed LLM-guided distillation framework remains effective when the teacher model is changed. For each LLM teacher, we use the same prompt, the same trajectory to text representation, and the same structured output schema. The original Qwen3 teacher is replaced by each alternative teacher to regenerate the LLM outputs for all training and test observations. The resulting pseudo labels are then used to train the student model following the distillation procedure in Section 3.2. After distillation, each student model is further fine-tuned with the same real crash observations following Section 3.3. The final crash risk prediction performance is evaluated using the metrics defined in Section 3.4.

Table 2. Sensitivity of the proposed framework to different LLM teachers under the 3 s warning horizon.

LLM teacher	Distilling model		Fine-tuned model				
	AUC_d	MAE_d	Recall	FAR	F1	AUC	MAE
Qwen3	0.86	0.41	0.89	0.14	0.88	0.83	0.18
DeepSeek-V4-Flash	0.89	0.81	0.79	0.17	0.82	0.79	0.24
GPT-5.4-mini	0.83	0.36	0.87	0.19	0.84	0.82	0.21
GLM-4-Plus	0.84	0.33	0.92	0.23	0.87	0.83	0.19

Consistent with the distillation evaluation in Section 4.1, we summarize the matching quality between the teacher and student using two distillation metrics, denoted as AUC_d and MAE_d. Here, AUC_d measures whether the student assigns higher risk scores to observations labeled as dangerous by the corresponding teacher than to observations labeled as safe by that teacher. MAE_d measures the absolute difference between the remaining time to crash estimated by the student and that estimated by the teacher. These two metrics describe how well the student learns the structured supervision provided by each LLM teacher. To distinguish them from the final prediction metrics, the subscript d is used to indicate that they are computed against the LLM teacher outputs rather than the real crash labels. For the final fine-tuned model, Recall, FAR, F1 score, AUC, and MAE are computed against real crash labels on the test set, following Section 3.4.

The results are reported in Table 2. Although the numerical results vary across teachers, all fine-tuned student models achieve strong crash risk prediction performance. Specifically, the fine-tuned models obtain AUC values from 0.79 to 0.83 and MAE values from 0.18 s to 0.24 s. This suggests that the structured outputs generated by different LLM teachers can all provide useful prior knowledge for student model training. The differences among LLM teachers mainly appear in the tradeoff between missed warnings, false alarms, and remaining time to crash estimation. Some teachers produce more sensitive warnings with higher Recall but also higher FAR, whereas others produce more conservative warnings with lower FAR but lower Recall. These differences are expected because different LLMs may have different risk preferences when interpreting the same trajectory-based scene descriptions. Among the compared teachers, Qwen3 provides the most balanced result in this experiment, with the highest F1 score of 0.88, the lowest FAR of 0.14, and the lowest MAE of 0.18 s. Therefore, Qwen3 is retained as the default teacher in the main experiments. More importantly, changing the LLM teacher does not lead to a major change in the effectiveness of the proposed framework. This supports the generality of using LLM-generated structured supervision for student model training.

5.2. Sensitivity to Different Student Models

Here, we examine how the choice of student model affects the proposed LDF pipeline. The LLM teacher is fixed as Qwen3, and the same pseudo labels, real crash data, and evaluation metrics are used for all student models. Six student models are compared, including MLP, LSTM, Transformer, GCN, GAT, and the proposed lightweight temporal graph network. These models differ mainly in how they encode the input information described in Section 3.2. When MLP is used as the student model, the trajectory features of the target vehicle and surrounding vehicles over the 1.0 s observation window are flattened into a single feature vector as input. LSTM and Transformer use the same 1.0 s historical trajectory information as sequential input. LSTM models the temporal evolution of the trajectory through recurrent hidden states, while Transformer uses self-attention to learn temporal dependencies across the observation window. GCN, GAT, and the proposed model represent the target vehicle and its surrounding vehicles as a graph. In this graph, vehicles are treated as nodes and their pairwise relations are represented through the graph structure. GCN is included as a representative graph neural network that aggregates information from neighboring nodes. GAT extends this idea by assigning attention weights to neighboring vehicles during aggregation.

All experiments follow the same training procedure as in Section 5.1. Each student model is first trained by distillation using the Qwen3 teacher outputs, following Section 3.2. The distillation performance is evaluated using AUC_d and MAE_d, which are computed against the Qwen3 teacher outputs. Each distilled student model is then fine-tuned using the same real crash data, following Section 3.3. The final crash risk prediction performance is evaluated using Recall, FAR, F1 score, AUC, and MAE, as defined in Section 3.4. The results are reported in Table 3.

From Table 3, the student model structure has a clear effect on the final prediction performance. When MLP, LSTM, and Transformer are used as student models, their performance is improved compared with their directly trained counterparts in Table 1, where the same model types are trained only with real crash labels. This suggests that supervision generated by Qwen3 provides useful prior knowledge for these models. However, their final performance remains lower than that of the proposed LDF model. A likely reason is that these models use flattened or sequence-based trajectory inputs. Although these inputs contain the historical motion information of the target vehicle and surrounding vehicles, they do not directly represent the pairwise interaction structure among vehicles. This structure is important for crash risk prediction since crash formation depends not only on the motion of each vehicle, but also on relative distance, relative speed, lateral offset, and how these interaction states evolve over time. Therefore, supervision generated by the LLM can provide useful prior knowledge, but the student model still needs an appropriate representation of multi-vehicle interactions.

Table 3. Sensitivity of the proposed framework to different student model structures under the 3 s warning horizon.

Student model	Distilling model		Fine-tuned model				
	AUC_d	MAE_d	Recall	FAR	F1	AUC	MAE
MLP	0.71	0.58	0.62	0.19	0.69	0.71	0.45
LSTM	0.72	0.57	0.67	0.18	0.72	0.72	0.35
Transformer	0.74	0.48	0.72	0.65	0.61	0.73	0.37
GCN	0.79	0.53	0.87	0.45	0.75	0.79	0.22
GAT	0.78	0.50	0.83	0.40	0.74	0.77	0.20
Proposed model	0.86	0.41	0.89	0.14	0.88	0.83	0.18

The results obtained with GCN and GAT further show the value of graph-based interaction modeling. Compared with MLP, LSTM, and Transformer, these graph-based student models achieve higher final AUC values and lower MAE values. However, they still underperform the proposed lightweight temporal graph network. This difference is mainly related to how pairwise interactions are represented. As described in Section 3.2, the proposed model explicitly incorporates edge features, including remaining distance, relative speed, and relative longitudinal and lateral offsets. In contrast, standard GCN and GAT mainly aggregate node information through the graph structure and do not use these edge features in the same explicit way. Therefore, they may lose pairwise interaction information that is directly relevant to crash risk.

5.3. Deployment Efficiency

In this section, we compare the online inference efficiency of direct LLM prediction and the proposed framework. The comparison includes the four LLMs considered in Section 5.1. For direct online LLM prediction, each LLM is directly applied to the test observations to generate crash risk predictions, and the token usage and inference latency are recorded. For the proposed framework, the deployed model is the distilled student. Since the student model performs prediction directly from trajectory features, it does not require online LLM calls or token consumption. Therefore, we compare the two deployment routes in terms of token usage, model size, and inference latency. The results are reported in Table 4.

Table 4. Inference efficiency of direct LLM prediction and the proposed framework.

Route	LLM call	Tokens	Model size	Latency (per observation)	Relative latency (per observation)
Qwen3	Yes	2,021	235B-A22B	13,686 ms	8,584.3 ×
DeepSeek-V4-Flash	Yes	2,163	671B-A37B	5,715 ms	3,584.6 ×
GPT-5.4-mini	Yes	2,287	N.A.	6,081 ms	3,814.2 ×
GLM-4-Plus	Yes	2,030	N.A.	14,836 ms	9,305.6 ×
LDF	No	0	0.167 MB	1.59 ms	1.0 ×

Note: For API-based LLMs, model size refers to the public parameter scale when available. For LDF, model size refers to the deployed student model size. N.A. indicates that the public parameter scale was not available.

As shown in Table 4, direct online LLM prediction requires a large number of input and output tokens for each observation. For the API-based LLMs with recorded token usage, the total token usage ranges from 2,021 to 2,287 tokens per observation. This leads to second-level inference latency. The mean latency is 5,715.0 ms for DeepSeek-V4-Flash, 6,081.0 ms for GPT-5.4-mini, 13,686.0 ms for Qwen3, and 14,836.0 ms for GLM-4-Plus. Such latency is too high for real-time crash warning, where the warning horizon is only a few seconds.

Although the LLM is used in the offline teacher stage, the proposed framework does not require any LLM call during online inference. The online model is the distilled student, which occupies 0.167 MB. Its mean inference latency is 1.59 ms per observation. Compared with direct online LLM prediction, the proposed framework is about 3,584.6 to 9,305.6 times faster. This reduction is achieved by moving the expensive language reasoning process to the offline teacher stage, so that the online stage only performs a lightweight neural network forward pass. This low latency is important for the intended safety application of the model. In high risk expressway interaction areas, the deployed student can be applied to each target vehicle whenever a new trajectory frame is received. At each observation time, it uses the recent trajectories of the target vehicle and surrounding vehicles to predict whether the target vehicle is likely to be involved in a crash within the warning horizon. It also estimates the remaining time to crash. In this way, the model can provide real-time and continuously updated risk information for short-horizon safety applications. These applications include connected vehicle warnings, driver assistance, and traffic management support in lane changing and car following crash scenarios. In this sense, the framework retains the benefit of LLM-guided supervision while avoiding the token cost, API dependence, and latency of online LLM calls.

6. Conclusion

This study developed a crash risk prediction framework that uses LLM reasoning as structured offline supervision rather than as an online prediction engine. Using multi-year drone videos collected at freeway merging and weaving segments in Nanjing, China, we extracted pre-crash vehicle trajectories from 109 real-world crashes. The proposed LDF framework converts trajectory interactions into structured scene descriptions, uses an LLM to infer crash probability and remaining time to crash, distills these outputs into a lightweight temporal graph network, and then calibrates the student model with real crash labels. This design allows the framework to use the contextual reasoning capability of LLMs while retaining the speed and deployability of a

compact trajectory-based predictor.

The empirical results provide three main findings. First, LLM-generated structured supervision improves crash risk prediction when it is distilled into the student model and further fine-tuned with real crash data. Under the 3 s warning horizon, the proposed model achieves the best overall performance among the compared methods, increasing AUC by 7.8% and reducing MAE by 28.0% relative to direct supervised training with the same student architecture. Its advantage is especially clear when only a small fraction of the fine-tuning data is available. With 10% of the training data, the proposed model reduces MAE by 19.6% relative to direct LLM inference and by 38.0% relative to direct supervised training. Second, the extended validation indicates that the framework is not tied to a single teacher model or a generic student architecture. Different LLM teachers all provide useful supervision, while the proposed lightweight temporal graph network outperforms MLP, LSTM, Transformer, GCN, and GAT student models because it explicitly represents pairwise edge features among interacting vehicles. Third, the deployment analysis shows that the final student model removes the need for online LLM calls, reduces inference latency to 1.59 ms per observation, and is 3,584.6 to 9,305.6 times faster than direct online LLM prediction. These results suggest that LLM-guided distillation can improve both data efficiency and deployment feasibility for short-horizon crash warning.

Several limitations remain. First, the empirical data cover a specific operating domain, namely low speed, high demand interactions among passenger cars at expressway merging and weaving segments under sunny weather. The transferability of the framework to other roadway types, traffic states, vehicle classes, and weather conditions should be examined using additional real crash trajectory data. Second, the current model uses extracted vehicle trajectories as input. This is appropriate for studying crash formation from motion and interaction patterns, but practical deployment in connected and automated vehicle environments would require these trajectories to be obtained in real-time from cameras, LiDAR, radar, roadside sensors, or vehicle-to-infrastructure communication. Future work should integrate the proposed framework with perception modules and examine whether the same LLM-guided distillation strategy can be extended from inputs based on trajectories to multimodal sensing representations. Finally, although the teacher sensitivity analysis shows consistent benefits across several LLMs, future studies should further examine prompt robustness, uncertainty calibration, and failure cases in more diverse safety-critical traffic scenarios.

Replication and Data Sharing

The data and codes used in this study are available at https://ets-data.sciopen.com/ETSD.2026.9190078.

Author contributions

Kequan Chen: Writing – original draft, Validation, Software, Methodology, Formal analysis. Yuxuan Wang: Data curation, Conceptualization, Methodology, Writing – review & editing. Zhibin Li: Software, Data curation. Pan Liu: Writing – review & editing, Formal analysis, Methodology.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (52232012, 52525204). The authors thank the anonymous reviewers for their time to review our article and their constructive comments.

Declaration of competing interest

The authors have no competing interests to declare that are relevant to the content of this article.

Declaration of generative AI and AI-assisted technologies in the writing process

During preparation of this work, the authors used OpenAI Codex for document formatting and language/consistency checks. The authors reviewed and edited the output and take full responsibility for the content of the publication.

Graphical abstract and Highlights (mandatory)

References

Ali, Y., Haque, M.M., Mannering, F., 2023. Assessing traffic conflict/crash relationships with extreme value theory: Recent developments and future directions for connected and autonomous vehicle and highway safety research. Analytic Methods in Accident Research 39, 100276. https://doi.org/10.1016/j.amar.2023.100276

Arteaga, C., Park, J., 2025. A large language model framework to uncover underreporting in traffic crashes. Journal of Safety Research 92, 1–13. https://doi.org/10.1016/j.jsr.2024.11.009

Arun, A., Haque, Md.M., Washington, S., Sayed, T., Mannering, F., 2022. How many are enough?: Investigating the effectiveness of multiple conflict indicators for crash frequency-by-severity estimation by automated traffic conflict analysis. Transportation Research Part C: Emerging Technologies 138, 103653. https://doi.org/10.1016/j.trc.2022.103653

Berghaus, M., Lamberty, S., Ehlers, J., Kalló, E., Oeser, M., 2024. Vehicle trajectory dataset from drone videos including off-ramp and congested traffic – Analysis of data quality, traffic flow, and accident risk. Communications in Transportation Research 4, 100133. https://doi.org/10.1016/j.commtr.2024.100133

Chen, K., Li, Z., Liu, P., Xu, C., Wang, Y., 2025a. Real-time lane-changing crash prediction model at the individual vehicle level using real-world trajectories prior to crashes. Transportation Research Part C: Emerging Technologies 176, 105171. https://doi.org/10.1016/j.trc.2025.105171

Chen, K., Liu, P., Li, Z., Wang, Y., Lu, Y., 2021. Modeling anticipation and relaxation of lane changing behavior using deep learning. Transportation Research Record 2675, 186–200. https://doi.org/10.1177/03611981211028624

Chen, K., Wang, Y., Liu, P., Knoop, V.L., Wang, D.Z.W., Han, Y., 2026a. How vehicles change lanes after encountering crashes: empirical analysis and modeling. Transportation Research Part A: Policy and Practice 206, 104929. https://doi.org/10.1016/j.tra.2026.104929

Chen, K., Xu, C., Liu, P., Li, Z., Wang, Y., 2024. Evaluating the performance of traffic conflict measures in real-time crash risk prediction using pre-crash vehicle trajectories. Accident Analysis & Prevention 203, 107640. https://doi.org/10.1016/j.aap.2024.107640

Chen, K., Yu, H., Liu, P., Li, Z., Wang, Y., 2025b. Factors influencing lane-changing crashes and conflicts using vehicle trajectories before crashes. Journal of

Transportation Safety & Security 17, 1195–1220. https://doi.org/10.1080/19439962.2025.2509932
Chen, T., Shen, Z., Zhou, B., Liu, Y., Wang, S., Ke, J., 2026b. Addressing the online incremental transport mode choice prediction problem with an LLM-augmented class-incremental learning approach. Transportation Research Part C: Emerging Technologies 188, 105709. https://doi.org/10.1016/j.trc.2026.105709
Fawcett, T., 2006. An introduction to ROC analysis. Pattern Recognition Letters 27, 861–874. https://doi.org/10.1016/j.patrec.2005.10.010
Hayward, J.C., 1972. Near-miss determination through use of a scale of danger. Highway Research Record 384, 24–34.
Huang, H., Huang, X., Zhou, R., Zhou, H., Lee, J.J., Cen, X., 2024. Pre-crash scenarios for safety testing of autonomous vehicles: A clustering method for in-depth crash data. Accident Analysis & Prevention 203, 107616. https://doi.org/10.1016/j.aap.2024.107616
Jiao, X., Yin, Y., Shang, L., Jiang, X., Chen, X., Li, L., et al., 2020. TinyBERT: Distilling BERT for natural language understanding. In: Findings of the Association for Computational Linguistics: EMNLP 2020. Association for Computational Linguistics, Online, pp. 4163–4174. https://doi.org/10.18653/v1/2020.findings-emnlp.372
Jiao, Y., Calvert, S.C., Van Cranenburgh, S., Van Lint, H., 2026. Learning collision risk proactively from naturalistic driving data at scale. Nat Mach Intell 8, 337–350. https://doi.org/10.1038/s42256-026-01189-w
Karim, M.M., Shi, Y., Zhang, S., Wang, B., Nasri, M., Wang, Y., 2025. Large language models and their applications in roadway safety and mobility enhancement: A comprehensive review. Artificial Intelligence for Transportation 1, 100004. https://doi.org/10.1016/j.ait.2025.100004
Kipf, T.N., Welling, M., 2016. Semi-supervised classification with graph convolutional networks. https://doi.org/10.48550/arXiv.1609.02907
Kuang, S., Liu, Y., Qu, X., Wei, Y., 2025. Traffic-IT: Enhancing traffic scene understanding for multimodal large language models. Transportation Research Part C: Emerging Technologies 180, 105325. https://doi.org/10.1016/j.trc.2025.105325
Kuang, S., Liu, Y., Wang, X., Wu, X., Wei, Y., 2024. Harnessing multimodal large language models for traffic knowledge graph generation and decision-making. Communications in Transportation Research 4, 100146. https://doi.org/10.1016/j.commtr.2024.100146
Lan, Z., Li, H., Liu, L., Fan, B., Lv, Y., Ren, Y., et al., 2024. Traj-LLM: A new exploration for empowering trajectory prediction with pre-trained large language models. https://doi.org/10.48550/arXiv.2405.04909
Peesapati, L.N., Hunter, M.P., Rodgers, M.O., 2018. Can post encroachment time substitute intersection characteristics in crash prediction models? Journal of Safety Research 66, 205–211. https://doi.org/10.1016/j.jsr.2018.05.002
Shew, C., Pande, A., Nuworsoo, C., 2013. Transferability and robustness of real-time freeway crash risk assessment. Journal of Safety Research 46, 83–90. https://doi.org/10.1016/j.jsr.2013.04.005
Shi, X., Wong, Y.D., Li, M.Z.F., Chai, C., 2018. Key risk indicators for accident assessment conditioned on pre-crash vehicle trajectory. Accident Analysis & Prevention 117, 346–356. https://doi.org/10.1016/j.aap.2018.05.007
Tak, S., Kim, S., Yeo, H., 2015. Development of a deceleration-based surrogate safety measure for rear-end collision risk. IEEE Trans. Intell. Transport. Syst. 16, 2435–2445. https://doi.org/10.1109/TITS.2015.2409374
Tang, S., Lu, Y., Liao, Y., Cheng, K., Zou, Y., 2024. A new surrogate safety measure considering temporal–spatial proximity and severity of potential collisions. Applied Sciences 14, 2711. https://doi.org/10.3390/app14072711
Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., et al., 2017. Attention is all you need. https://doi.org/10.48550/arXiv.1706.03762
Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y., 2018. Graph attention networks. In: International Conference on Learning Representations. arXiv:1710.10903.
Wang, J., Wu, J., Li, Y., 2015. The driving safety field based on driver–vehicle–road interactions. IEEE Trans. Intell. Transport. Syst. 16, 2203–2214. https://doi.org/10.1109/TITS.2015.2401837
Wang, Y., Li, Z., Liu, P., Xu, C., Chen, K., 2024. Surrogate safety measures for traffic oscillations based on empirical vehicle trajectories prior to crashes. Transportation Research Part C: Emerging Technologies 161, 104543. https://doi.org/10.1016/j.trc.2024.104543
Yang, Y., Hu, Z., Zhang, C., Zhang, Z., Gao, K., Liu, Y., 2026. Large language models as decision-making agents for autonomous vehicles in dynamic urban scenarios. Transportation Research Part E: Logistics and Transportation Review 210, 104803. https://doi.org/10.1016/j.tre.2026.104803
Zhang, S., Lin, H., Wang, M., Wei, B., Liu, Y., Qu, X., 2026. A dynamic prompting and scenario generation method for autonomous driving perception via large-model optimization. Transportation Research Part C: Emerging Technologies, 188, 105672. https://doi.org/10.1016/j.trc.2026.105672
Zhao, Yang, Wang, P., Zhao, Yibo, Du, H., Yang, H.F., 2025. SafeTraffic Copilot: adapting large language models for trustworthy traffic safety assessments and decision interventions. Nat Commun 16, 8846. https://doi.org/10.1038/s41467-025-64574-w
Zhen, H., Yang, J.J., 2025. CrashSage: A large language model-centered framework for contextual and interpretable traffic crash analysis. Artificial Intelligence for Transportation 3–4, 100030. https://doi.org/10.1016/j.ait.2025.100030
Zhong, W., Huang, J., Wu, M., Luo, W., Yu, R., 2025. Large language model based system with causal inference and Chain-of-Thoughts reasoning for traffic scene risk assessment. Knowledge-Based Systems 319, 113630. https://doi.org/10.1016/j.knosys.2025.113630
Zhou, R., Gui, W., Huang, H., Liu, X., Wei, Z., Bian, J., 2025. DiffCrash: leveraging denoising diffusion probabilistic models to expand high-risk testing scenarios using in-depth crash data. Expert Systems with Applications 287, 128140. https://doi.org/10.1016/j.eswa.2025.128140

Author biography

Kequan Chen received the Ph.D. degree from Southeast University in 2024. From 2022 to 2023, he was a visiting Ph.D. student at the Faculty of Civil Engineering and Geosciences, Delft University of Technology, the Netherlands. He is currently a Research Fellow at Nanyang Technological University, Singapore. His research interests broadly include object detection, computer vision, traffic safety, and driving behaviors.

Yuxuan Wang received her B.S. degree from Central South University and her Ph.D. degree from Southeast University, China. She was a postdoctoral researcher at the State Key Laboratory of Internet of Things for Smart City, University of Macau, and is currently an Associate Professor at Nanjing University of Aeronautics and Astronautics. Her research focuses on traffic safety, driving behavior modeling, and safety assessment of autonomous driving.

Zhibin Li received the Ph.D. degree from the School of Transportation, Southeast University, Nanjing, China, in 2014. He is currently a Professor with Southeast University. From 2015 to 2017, he was a Postdoctoral Researcher with the University of Washington, Seattle, WA, USA, and The Hong Kong Polytechnic University, Hong Kong. From 2010 to 2012, he was a visiting student with the University of California at Berkeley, Berkeley, CA, USA. His research interests include intelligent transportation and traffic control. Prof. Li won the Best Doctoral Dissertation Award from China Intelligent Transportation Systems Association in 2015. He was selected as the World's Top 2% Scientists and as the Highly Cited Chinese Researchers by Elsevier in 2021 and 2022.

Pan Liu received the Ph.D. degree in Civil Engineering from the University of South Florida, Tampa, USA, in 2006. He is a Professor with the School of Transportation, Southeast University, Nanjing, China. His research interests include traffic operations and safety and intelligent transportation systems. He was a recipient of the Distinguished Young Scientist Foundation of NSFC in 2013 and the Outstanding Young Scientist Foundation of NSFC in 2019.