A Dual-System Neuro-Symbolic Framework with Accident Prediction for Autonomous Driving

Summary of changes to the EMAS Post-Proceedings version

Significant changes throughout the paper are highlighted in blue.
The following is a summary of the changes:

- Removed references to interpretability and traceability, as these are inherent advantages of BDI but not discussed in this paper.
- Added explanations of the abbreviations ML-MAS and MLAPM-MAS.
- Rewrote subsection 3.1 to clarify how temporal interaction graphs are constructed from traffic-scene observations, covering node feature encoding, edge feature definition, and graph representation at each time step. Added a new figure illustrating the edge-feature definition and the graph at time t.
- Rewrote subsection 3.2 on the TGAT-based accident prediction model to describe the temporal dynamic graph formulation, temporal neighbour sampling, recursive TGAT aggregation, contrastive sample construction from collision labels, training loss, and inference-stage risk scoring. Added a new figure illustrating temporal neighbour sampling and TGAT convolution.
- Split the subsection Online Inference and BDI Integration into two. The first, subsection 3.3 presents the CARLA online inference procedure for the TGAT-based accident prediction model, including ego-centred neighbour selection, historical interaction tracking, query-time temporal aggregation, risk-score-based warning generation, and integration with the ML-MAS Orchestrator.
- The second, subsection 3.4 discusses the BDI integration with accident prediction, including a formal definition of MLAPM-MAS as a dual decision-making system, the warning-to-belief transformation, belief update, Jason plan selection, and action arbitration between BDI and baseline LAV actions. Added a new figure illustrating the MLAPM-MAS decision process.
- Added an analysis of the evaluation results shown in Figure 6 to discuss the model's ability to learn effective collision-risk features and generalise across seen and unseen scenario types.
- Added a description of the system-level comparison setup, including the four evaluated configurations, baseline definitions, evaluation routes, and CARLA Leaderboard metrics in section 5.1. Reduced repetition in the subsequent result discussion.
- The related work section was completely rewritten. It now includes recent LLM-based and neuro-symbolic autonomous driving studies and repositions MLAPM-MAS around system design. The revised discussion clarifies that MLAPM-MAS uses learned accident-risk warnings as symbolic beliefs to support BDI plan selection, while keeping LAV as the baseline driving policy and the BDI agent as the executable decision-making layer.

A Dual-System Neuro-Symbolic Framework with Accident Prediction for Autonomous Driving

Jianming Wang[0009−0006−2655−0512], Leonardo Amado[0000−0001−6119−4601], and Rafael C. Cardoso[0000−0001−6666−6954]

University of Aberdeen, Aberdeen, United Kingdom
{j.wang4.25,leonardo.amado,rafael.cardoso}@abdn.ac.uk

Abstract. Machine learning models demonstrate strong perception and pattern recognition capabilities in complex traffic scenarios, while symbolic decision-making mechanisms retain significant advantages in interpretability and rule constraints. This paper proposes MLAPM-MAS, a dual system Neuro-Symbolic accident prediction framework based on the ML-MAS architecture, designed to enable collaboration between machine learning accident prediction models and BDI reasoning. System 1 models dynamic vehicle interactions using a Temporal Graph Attention Network constructed from interpretable interaction features, enabling continuous estimation of collision risk in complex traffic scenarios. The resulting accident prediction assessments are mapped onto symbolic beliefs and decision constraints, which are then supplied to System 2, a BDI agent, to guide subsequent planning and execution. Experimental results on the CARLA simulation benchmark show that incorporating accident-prediction constraints improves safety-related metrics and reduces certain traffic violations.

Keywords: Autonomous Driving · Neuro-Symbolic AI · BDI Agent · Temporal Graph Attention Network.

1 Introduction

The key challenge in autonomous driving is enabling vehicles to make autonomous decisions in complex traffic environments. In recent years, rapid advances in Machine Learning (ML), especially Deep Learning, have provided strong technical support for autonomous driving [24]. Autonomous driving systems consist of multiple functional modules, each with its own problem formulation and modelling approach rather than uniformly adopting a single end-to-end learning model [2,7]. Deep learning can transform sensor data into semantically meaningful representations of the environment, providing a foundation for decision-making and control in autonomous driving systems. Vehicle trajectory planning and vehicle control are typically achieved through rule-based or optimisation-based models.

Although data-driven ML methods have achieved significant performance gains, their end-to-end training paradigm lacks explicit structural and semantic constraints on the internal decision-making process and is therefore often

regarded as a black-box model. This characteristic makes such models difficult to explain [16] and often results in limited generalisation when they encounter complex scenarios outside the training distribution [14].

Symbolic Artificial Intelligence (AI) approaches to reasoning use symbolic representations and logical inference mechanisms, making the reasoning process relatively interpretable and supporting formal analysis and verification [15]. Hybrid AI approaches that integrate neural networks with symbolic reasoning have attracted increasing attention in recent years. The dual-system theory in cognitive science suggests that intelligent behaviour is supported by two systems [10]. System 1 is primarily responsible for fast, perception-driven processing, while System 2 is associated with slower deliberative reasoning. In AI, the theory is often used to characterise the functional division between data-driven learning methods and symbolic reasoning approaches, and to provide a conceptual framework for their integration [3]. A key challenge within this framework lies in abstracting environmental perception from neural networks into condition representations that can be processed by symbolic reasoning systems, thereby enabling decision-making based on logical rules [11].

The Machine Learning Multi-Agent System (ML-MAS) framework [1] integrates ML models with Jason BDI agents [4] and applies them to autonomous driving tasks in the CARLA simulation environment [8]. Neural network–based System 1 and the symbolic reasoning–based System 2 operate as relatively independent functional components in ML-MAS.

This paper proposes the Machine Learning-based Accident Prediction Model Multi-Agent System (MLAPM-MAS) as an extension of ML-MAS, introducing a graph neural network for accident prediction that is integrated into the ML-MAS framework as another subsystem of System 1. The model's output is used to constrain the triggering conditions for the driving plans. The goal is to effectively reduce the search space of candidate actions during the reasoning stage without increasing the risk of collisions. The vehicle's final control decisions are still generated by an explicit BDI reasoning mechanism, which allows the decision-making process to be understood and analysed in terms of rules and state transitions. This study adopts the CARLA "Longest6 benchmark" [6] and conducts comparative experiments on the first three routes of the *Town01* map. ML-MAS serves as the baseline, and all experimental results are evaluated using the standard CARLA Leaderboard metrics. The results demonstrate that the proposed extension effectively reduces collision rates while maintaining strong overall driving performance.

2 Background

In this section, we discuss: how the ML-MAS framework operationalises the dual-system concept as an executable multi-agent architecture; the CARLA autonomous driving simulation platform as our testing environment for the coordinated operation of ML models and BDI reasoning mechanisms; and the concepts of Temporal Graph Attention Network (TGAT), which we use to train an ML

accident prediction model within the ML-MAS framework to characterise temporal interaction relationships among traffic actors.

2.1 The ML-MAS Framework

The dual-system theory is regarded as an important approach for integrating Neuro-Symbolic AI [3,11,9]. The ML-MAS framework [1] embodies a dual-system perspective by integrating ML models, in particular LAV [5], corresponding to System 1, and the Jason BDI reasoning corresponding to System 2, thereby constructing an autonomous agent framework that reflects dual-system principles. Pre-trained ML models handle primary control decisions in routine scenarios, whereas in exceptional situations, the system switches to a BDI agent for logical reasoning and planning.

Prior work has identified ten core research questions centred on dual-system architectures [3]. One of these focuses on the mechanisms of cooperation and coordination between the two systems. The ML-MAS framework introduces a centralised coordination module, termed the *Orchestrator*, to achieve unified control between the two systems. The BDI bridge within the framework provides communication via socket-based mechanisms and JSON messages. The Orchestrator is responsible for managing and invoking the ML components within the system, as well as collecting and preprocessing multi-source sensor data from the simulation environment. The processed information is then synchronously distributed to System 1 and System 2. Through this unified coordination and information-sharing mechanism, the ML-MAS framework enables the collaborative operation of data-driven learning and symbolic rational reasoning within CARLA autonomous driving simulation scenarios.

2.2 CARLA and the DeepAccident Dataset

CARLA is an open source urban autonomous driving simulation platform that incorporates a high-fidelity simulation engine to model complex urban road driving environments [8]. CARLA support flexible sensor configurations, including multi-view RGB cameras, LiDAR point clouds, and multi-class semantic segmentation outputs. The platform can output driving behaviour and safety evaluation metrics, thus providing comprehensive support for the development and testing of autonomous driving systems. CARLA provides dedicated testing and evaluation tools. *ScenarioRunner* is a testing framework for defining and executing complex traffic scenarios and for systematically verifying and evaluating autonomous driving systems.

CARLA Leaderboard[1] provides a unified benchmarking platform for comparing the overall driving performance of different autonomous driving approaches under diverse scenario conditions. The benchmark quantitatively evaluates system behaviour in simulation using driving performance metrics, offering a standardised basis for comparison.

[1] http://leaderboard.carla.org/ (Accessed: 21/02/2026)

DeepAccident is a large-scale autonomous driving dataset constructed using the CARLA simulation platform [18]. The dataset comprises 285000 annotated samples and 57000 frames of multi-agent cooperative perception data captured from multiple viewpoints, sampled at 10Hz. The simulated environment is configured as urban intersection scenarios, covering 12 representative types of traffic accidents. These scenarios are defined based on real-world crash cases described in pre-crash reports issued by the U.S. National Highway Traffic Safety Administration (NHTSA).

Each accident scenario is constructed with four vehicles and a single infrastructure node. All vehicles and infrastructure units are equipped with multi-view RGB cameras and LiDAR sensors to generate image data and point cloud data from different viewpoints. The collected data integrates both spatial and motion information of traffic actors, providing a unified data foundation for a range of perception and prediction tasks in autonomous driving. This dataset also includes collision-free normal traffic samples, enhancing scenario diversity for motion modelling tasks.

Table 1: DeepAccident data types.

Field Name	**Description**
Object Type	The category of the detected object
h,w,l	Shape of object
x,y,z	The position information of the object
yaw	Yaw angle (rotated around the Z-axis) units
vx,vy	The velocity of the object in the x/y direction
Object ID	The unique ID of the object
LiDAR Points	The number of laser points in the 3D box of the object
Visible	Can the target be seen by the camera

As shown in Table 1, DeepAccident provides attribute descriptions of the spatial structure and motion states of vehicles and pedestrians. However, traffic accidents often arise from interactions among multiple traffic actors that evolve over time, and static attributes alone are insufficient to capture temporal dependencies.

2.3 Temporal Graph Attention Network

Temporal Graph Attention Network (TGAT) models interactions as timestamp events and injects continuous time encodings into a self-attention mechanism, enabling dynamic modelling of the influence of historical interactions through multi-hop message passing under temporal causality constraints [20]. TGAT can represent the mutual influences and evolving relationships among multiple agents across different temporal scales. Continuous time is represented through a learnable functional mapping, such that the inner product between encodings at different time points depends solely on their temporal difference. This method

leverages Bochner's theorem to approximate a translation-invariant temporal kernel using a random feature expansion. It represents continuous time in a finite dimensional functional form $\Phi_m(t)$ parameterised by a set of learnable frequency parameters $\{\omega_1\}_{i=1}^{d}$. Temporal information can be incorporated into the self-attention computation in a differentiable manner and optimised through end-to-end learning.

We use TGAT with a standard sinusoidal/learned time encoding [20]:

$$\Phi_m(\Delta t) = \frac{1}{\sqrt{m}}\left[\cos(\omega_k \Delta t), \sin(\omega_k \Delta t)\right]_{k=1}^{m} \tag{1}$$

where $\{\omega_k\}$ are learned frequencies. The encoding is concatenated with node and edge features and consumed by a temporal attention aggregator (standard TGAT, we omit the details for simplicity).

The Temporal Graph Attention Layer in TGAT aggregates neighbourhoods that satisfy temporal constraints at a given time point, and feeds node features, edge features, and time encodings into a self-attention mechanism. The attention mechanism learns the influence weights of different historical neighbours on the target node representation. The model leverages stacked Temporal Graph Attention Layers to propagate multi-hop temporal information. TGAT can therefore characterise time-evolving interactions between an individual target and multiple actors in complex traffic scenarios, providing temporal modelling capabilities for accident prediction analysis.

3 Neuro-Symbolic Accident Prediction Framework

This section introduces our ML-MAS extension, *MLAPM-MAS*. Fig. 1 shows the extended overview of MLAPM-MAS. In System 1, we employ a TGAT model to capture dynamic interactions among vehicles and to predict potential collision risk. The output of System 1 is further mapped into symbolic risk beliefs and used as input conditions for the BDI agent (System 2), providing constraints for subsequent symbolic decision-making and action planning through new deceleration plans. The remaining elements remain the same as in the original ML-MAS [1] (see Section 2.1 for details).

3.1 Encoding of Traffic Vehicle Node Features and Edge Features

The DeepAccident dataset provides raw LiDAR data and three-dimensional bounding box annotations of traffic participants, which can be used to construct the input temporal graph of TGAT. Before constructing the temporal interaction graph, this paper first uses the LiDAR-to-ego and ego-to-world rigid transformations provided by the dataset to uniformly transform the original object states from the local LiDAR coordinate system to the CARLA world coordinate system. This preprocessing step ensures that the subsequent node features, edge features, and interaction relationships are all computed under a consistent spatial reference frame.

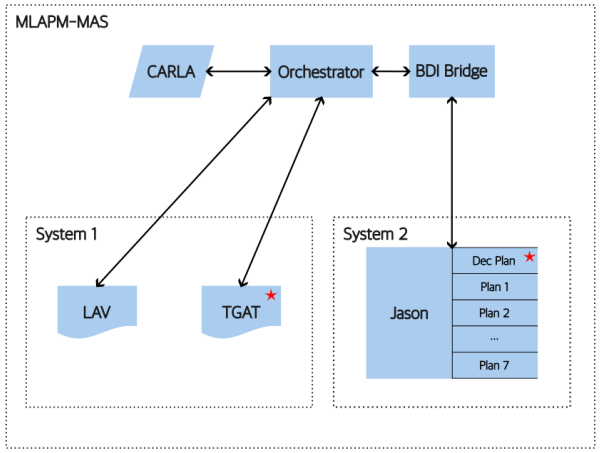

Fig. 1: MLAPM-MAS overview, with the ⋆ representing the new extensions. Dec Plan are the new deceleration plans.

After unifying the coordinate system, each traffic participant is modelled as a node in the temporal graph. For node i at time step t, its node feature consists of position, yaw angle, velocity, three-dimensional bounding box attributes, and semantic category, where the semantic category is used to distinguish different types of traffic participants, such as vehicles and pedestrians.

For two traffic participants i and j that are within the interaction range at time step t, an edge $(i, j, t, e_{i,j}^t)$ is established between them. This edge represents the physical interaction relationship between the two traffic participants in the current time frame. The edge feature $e_{i,j}^t$ is defined as the relative spacing between two traffic participants based on bounding-box projection. It is com-

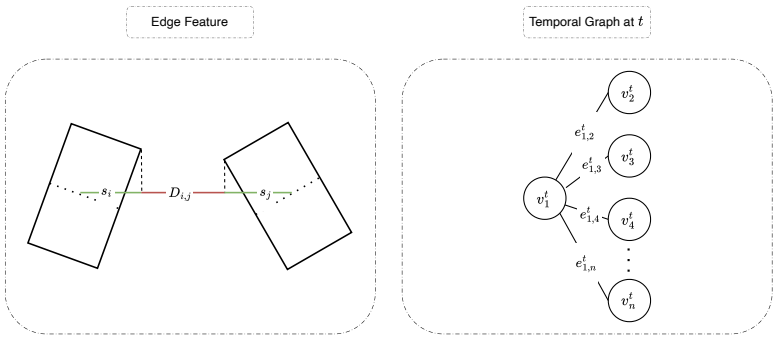

Fig. 2: Illustration of the edge feature and the graph at time t.

puted as $D^t_{i,j} = g^t_{i,j} - (s^t_i + s^t_j)$. $g^t_{i,j}$ denotes the distance between the centre points of the two traffic participants. The terms s^t_i and s^t_j denote the projected half-lengths of their bounding boxes along the line connecting the two centres. In the left of Fig. 2, the $D_{i,j}$ (red line) segment represents $D^t_{i,j}$ at a particular time step, while the s_i and s_j (green line) segments represent s^t_i and s^t_j. When $D^t_{i,j} < 0$, it indicates that the projected bounding boxes of the two traffic participants overlap. When $D^t_{i,j}$ is a small positive value, it indicates that the two participants are spatially close to a potential contact state. This edge feature can serve as a physical proximity indicator in accident risk prediction.

Based on the node and edge features defined above, each time frame can be represented as a graph $G_t = (V_t, E_t)$, as shown in the right panel of Fig. 2. V_t denotes the set of traffic participant nodes observed at time step t, and E_t denotes the set of physical interaction edges between these nodes.

3.2 TGAT Model for Accident Prediction

Following the graph construction described above, the entire traffic scene sequence can be represented as a temporal dynamic graph $G = (V, E, T)$. $V = \{n_1, \ldots, n_m\}$ denotes the set of all observed traffic participant nodes in the scene, and T denotes the set of discrete time steps. Each node $n_i \in V$ corresponds to one traffic participant. This paper models each traffic participant as a node in the temporal dynamic graph. Each node n_i is represented by an input feature $x_{n_i} = [b_i, c_i]$, which combines the bounding-box attributes and semantic category of the corresponding traffic participant. The edges in the temporal dynamic graph are defined as $E = \{(u, i, t, e^t_{u,i}) \mid u, i \in V,\ t \in T\}$, where (u, i, t) indicates a temporal interaction edge between nodes u and i at time step t, and edge feature $e^t_{u,i}$ denotes the corresponding edge feature, namely the relative spacing $D^t_{u,i}$.

In the left of Fig. 3, the model searches the existing temporal dynamic graph for the interaction records of node n_i before time t, and obtains the historical neighbour set $\mathcal{N}_i(t) = \left\{(n_p, e^{t_p}_{i,p}, t_p) \mid t_p < t\right\}$. n_p is a neighbour node that interacted with node n_i before time t, and $e^{t_p}_{i,p}$ is their interaction edge feature at the historical time t_p. The condition $t_p < t$ indicates that all historical interactions included in $\mathcal{N}_i(t)$ occurred before the current query time t.

The TGAT Convolution part of Fig. 3 illustrates TGAT multi-layer convolution as a recursive temporal neighbourhood aggregation process. The model recursively expands through historical temporal neighbours until it reaches the initial node representation $h^{(0)}_n = x_n$. Historical neighbour information is then aggregated layer by layer from the bottom upwards. Through this process, the representation of node n_i can simultaneously encode its own feature x_{n_i}, the historical neighbour node feature $h^{(l-1)}_{n_p}(t_p)$, the historical interaction edge feature $e^{t_p}_{i,p}$, and the historical interaction time information introduced by the time

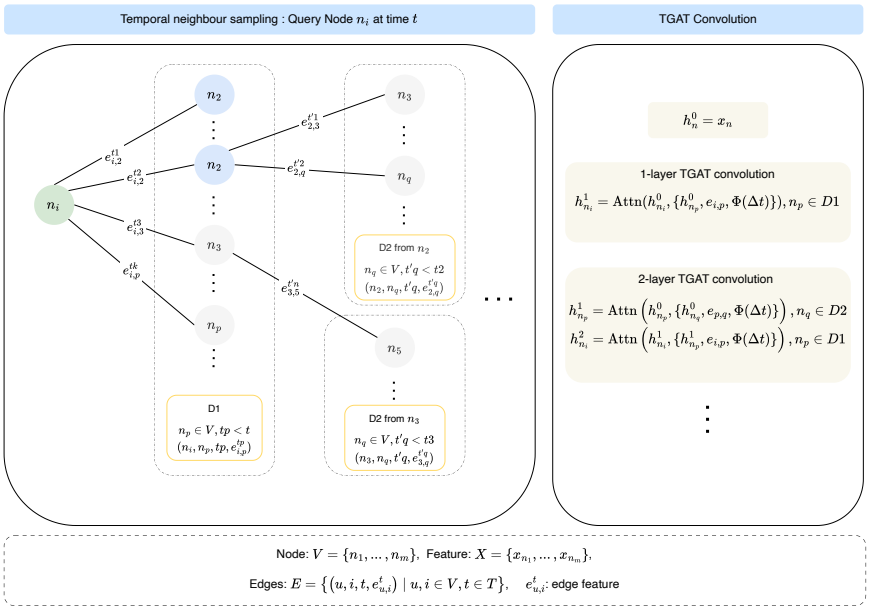

Fig. 3: Temporal neighbour sampling and TGAT convolution.

encoding $\Phi(t - t_p)$.

$$
\begin{aligned}
h_{n_i}^{(l)}(t) &= \mathrm{TGAT}^{(l)}\left(h_{n_i}^{(l-1)}(t), \left\{h_{n_p}^{(l-1)}(t_p) \parallel e_{i,p}^{t_p} \parallel \Phi(t - t_p)\right\}_{n_p \in \mathcal{N}_i(t)}\right) \\
E_{n_i}(t) &= h_{n_i}^{(L)}(t)
\end{aligned}
\tag{2}
$$

The representations $E_u(t)$ and $E_i(t)$ of traffic participants u and i at time t can encode their temporal interaction patterns before the current time, including the distance information in historical interactions.

The existing collision labels in the dataset are used to construct contrastive learning samples for training. Traffic participant pairs that collide are used as positive samples, denoted as $(E_u(t), E_p(t))$. Traffic participant pairs that do not collide are used as negative samples, denoted by $(E_u(t), E_n(t))$. A learnable scoring function f_θ is used to compute the risk score between two node representations. The training objective is to make the risk score s_p of the positive sample pair higher and the risk score s_n of the negative sample pair lower. By minimising the loss function $\mathcal{L} = \mathrm{BCE}(s_p, 1) + \mathrm{BCE}(s_n, 0)$, the model learns time-dependent node representations that can distinguish high-risk interactions from low-risk interactions.

In the inference stage, the representation of a node at the future query time $t + 1$ is also computed from the historical interactions before that time. For any pair of traffic participants u and i, the scoring function can be used to compute

the risk score $s_{u,i}^{t+1}$ according to their time-dependent node representations $E_u(t+1)$ and $E_i(t+1)$. Based on the risk score, it can be determined whether there is a collision risk between them at time $t+1$.

3.3 Online Accident-Risk Inference Using TGAT

TGAT is deployed in the CARLA simulation environment to perform online risk assessment of vehicle interactions in dynamic traffic scenarios. In each observation frame, the system obtains the three-dimensional bounding boxes, semantic categories, global positions, and orientations of traffic participants from CARLA, and encodes the three-dimensional bounding-box attributes and semantic categories as node features. Based on the relative positions of vehicles, interaction edge features are computed to construct an online temporal interaction sequence.

With the ego vehicle u as the centre, namely the star-marked vehicle in Fig. 4, the system calculates the relative distances between the ego vehicle and surrounding traffic participants in the reference frame ts_1. The traffic participants are then sorted in ascending order according to their relative distances, and the first n traffic participants are selected as fixed interaction neighbours. Let $I_u = \{i_1, \ldots, i_n\}$ denote the fixed neighbour set of the ego vehicle. For any neighbour $i_k \in I_u$, the system continuously tracks this vehicle during subsequent observations and records the historical interaction event sequence between the ego vehicle and this neighbour $\{(u, i_k, ts_j, e_j)\}_{j=1}^{t}$. Where ts_j denotes the j-th historical time index, and e_j denotes the edge feature calculated from the relative distance at that time.

Fig. 4: Vehicle interaction illustration. In a reference frame, a fixed number of interacting neighbour are selected for the ego vehicle (star) based on the minimum projected distance to surrounding traffic participants, which are used for subsequent interaction modelling.

To perform online risk reasoning, this paper introduces a virtual time index $t+1$ after the real observation sequence as a query-time anchor. This time index does not correspond to a new real physical observation, but is used to trigger TGAT to perform temporal neighbourhood aggregation over the existing historical interaction sequence. At this query time, TGAT aggregates all historical interaction events satisfying $ts_j < t+1$ and obtains the time-dependent node representations of the ego vehicle and its neighbouring vehicles, $E_u(t+1), E_{i_k}(t+1)$.

The scoring function f_θ calculates the risk score s_{u,i_k}^{t+1} based on the temporal embeddings of the ego vehicle and the neighbouring vehicle, and obtains the risk rating $\sigma(s_{u,i_k}^{t+1})$ through a sigmoid function. The risk rating is then compared with a predefined warning threshold τ. If the risk rating satisfies the high-risk interaction condition $\sigma(s_{u,i_k}^{t+1}) \geq \tau$, a collision warning is generated. Otherwise, the interaction is regarded as low risk. The generated collision warning is then passed to the ML-MAS Orchestrator for subsequent symbolisation and BDI decision-making.

3.4 BDI Integration with Accident Prediction

MLAPM-MAS system is defined as a dual decision-making system (as shown in Fig. 5), $S_{MLAPM-MAS} = (E, M, O, A_{BDI}, \rho, \alpha)$. E is the CARLA simulation environment. M is the Accident Prediction Model, which generates accident risk warnings. O is the Orchestrator component in ML-MAS. A_{BDI} is the Jason BDI agent. ρ is a function that transforms risk warnings into beliefs that can be processed by the BDI agent. α is the action orchestration function.

At each time step t, the Orchestrator obtains an observation from the environment, denoted as $o_t = (L_t, v_t, TL_t, X_t)$, where L_t is the LiDAR input, v_t is the current speed of the ego vehicle, TL_t is the traffic light state, and X_t is the state of surrounding obstacles. Based on this observation, LAV generates a baseline control action: $act_t^{LAV} = LAV(o_t)$. Meanwhile, the Accident Prediction Model generates an accident risk indicator according to the current traffic observation $m_t = M(o_t)$, where $m_t \in \{0, 1\}$.

When $m_t = 1$, the warning state is kept active for a short duration determined by the current speed and simulation frame rate. In this paper, ω_t indicates whether the warning is active in the current frame. The function ρ then transforms the warning state into a set of symbolic beliefs $\{mlAlert(F)\}$ that can be processed by the BDI agent:

$$\omega_t = \begin{cases} 1, & m_t = 1 \vee q_t > 0, \\ 0, & \text{otherwise}, \end{cases} \qquad \rho(\omega_t) = \begin{cases} \{mlAlert(F)\}, & \omega_t = 1, \\ \emptyset, & \omega_t = 0. \end{cases} \tag{3}$$

q_t is the remaining active duration of the warning state, and F denotes the current frame number.

The state of the BDI agent at time step t is defined as $A_t = (B_t, \Pi)$. Here, B_t is the current belief base, and Π denotes the Jason plan library. The belief update process is defined as: $B_{t+1} = Update(B_t, \rho(\omega_t))$. Based on the updated

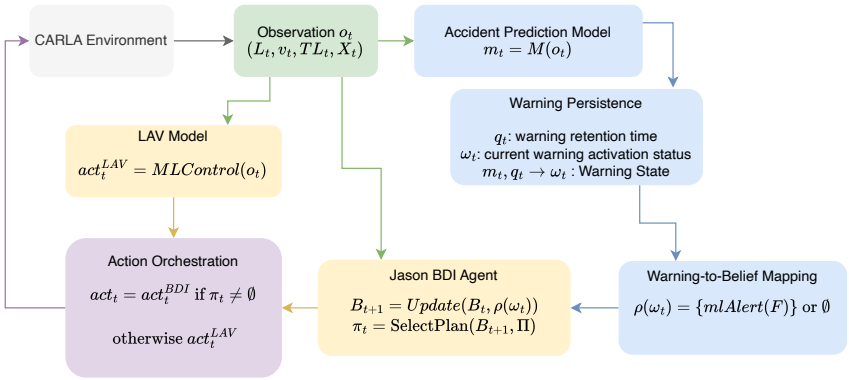

Fig. 5: Decision process of MLAPM-MAS.

belief base B_{t+1}, the Jason agent selects an applicable plan from the plan library Π: $\pi_t = SelectPlan(B_{t+1}, \Pi)$.

Belief-level consistency between the APM component and the BDI agent is checked through the Jason plan context. The APM output is transformed into the symbolic belief $mlAlert(F)$. If this belief conflicts with other current beliefs in B_{t+1} or does not satisfy the context of a candidate Jason plan, that plan is regarded as inapplicable. Only when no applicable BDI plan is selected does the system use the baseline LAV action. The final control action is then selected as follows:

$$act_t = \begin{cases} act_t^{BDI}, & \pi_t \neq \emptyset, \\ act_t^{LAV}, & \pi_t = \emptyset. \end{cases} \tag{4}$$

The selected action act_t is then sent to CARLA as the executable control command for the ego vehicle.

4 Accident Prediction Model Training and Evaluation

During training, highly similar edge features across consecutive frames can introduce redundant observations and reduce training efficiency. Based on these considerations, for each collision event (u, i_k, ts_t, r_t), vehicle interaction data from the 14 consecutive frames preceding the collision are selected as the historical observation window. This window is divided into $t = 8$ time segments to reduce redundancy between adjacent observations. The first six time segments are used for model training. The eighth time segment is used for prediction. These values were selected to balance predictive performance with computational efficiency.

Adam optimiser is used during the model training phase to optimise all learnable parameters of TGAT, with the learning rate set to 4×10^{-4}. The training objective is formulated as a binary classification problem and optimised using

the binary cross-entropy loss with logits. The training process uses mini-batch stochastic gradient descent with a batch size of 8. The training sample indices are randomly shuffled in each epoch, and the samples are divided into multiple batches, which are trained sequentially. This design prevents the model from repeatedly encountering interaction samples with highly similar temporal structures across successive iterations.

The validation dataset is split into two subsets. The first validation set is drawn from the same data collection as the training set but is strictly separated from the training samples along the temporal dimension. The model learns from interaction events within the historical time window $[1, t]$ during training. The model uses a historical observation window of length t as input to evaluate its ability to predict the interaction outcome at $t + 1$ during training. The second validation set consists of data never used during training. Samples are ranked and classified by predicted score to evaluate the model's ability to identify collision-related interaction samples in this set.

TGAT assigns a risk score to each pair of the ego vehicle and a neighbouring vehicle to estimate the likelihood of a future collision. This work evaluates model performance from multiple perspectives using metrics such as Average Precision (AP), Area Under the ROC Curve (AUC), Accuracy, and F1 score. AP on the validation set is used as the primary early-stopping criterion during model training. When the AP achieved on the validation set in the current epoch exceeds the historical best, the corresponding model parameters are saved as the current best model to prevent overfitting in ranking performance.

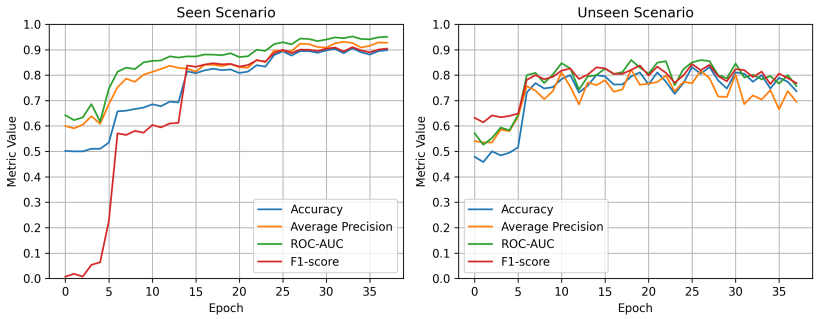

Fig. 6: Training results on unseen and seen scenarios.

As shown in Fig. 6, the results show that the model learns effective representations, as both seen and unseen test scenarios achieve relatively high metric values. However, the weaker improvement and greater fluctuations in unseen scenarios indicate that the learned representation transfers less reliably to novel scenario distributions.

In the "Seen Scenario" the model exhibits a rapid improvement across all performance metrics during the early stages of training. Within the epoch range

of 10 to 15, all performance metrics exceed 0.8 and remain stable. The model converges stably after epoch 15.

The model in the Unseen scenario likewise produces usable discriminative results during the early stages of training. All four evaluation metrics achieve higher performance from epoch 10 onwards, with values exceeding 0.70. Accuracy and the F1 score are both computed based on a fixed decision threshold. When the model's risk scores vary slightly around the threshold, some samples may switch between positive and negative classes even if the overall ranking structure remains unchanged. This switching leads to concurrent changes in the numbers of true positives, false positives, and false negatives, thereby amplifying fluctuations in Accuracy and the F1 score. These results indicate that the model learns a stable structure for ranking collision risk, with good generalisation capability in both scenarios. However, using a fixed decision threshold reduces the sensitivity of classification performance across scenarios.

5 MLAPM-MAS Evaluation

Possible collision risks are identified in the original ML-MAS framework using obstacle detection based on spatial region and lateral displacement. The system records two-dimensional position (x, y) for each perceived target in the vehicle coordinate frame at each frame. x represents the relative distance of the target along the vehicle's forward direction. y is the lateral offset of the target relative to the vehicle's longitudinal axis. The crossing risk assessment region is defined by target positions that satisfy $\{(x, y) \mid x < x_l \wedge |y| < y_l\}$. x_l and y_l are the threshold values for the forward distance and lateral offset. When the lateral positions y and y_2 of the same target in two consecutive frames satisfy $y - y_2 \neq 0$, the target is considered to display lateral displacement over time. The above detection strategy has limitations in a high-speed driving scenario. The effective reaction distance defined by the predefined detection region may be insufficient to support adequate vehicle deceleration at high speeds. The limited reaction time available to the system once a target enters the detection region increases the risk of collision.

We introduce a new deceleration plan for high-speed scenarios to mitigate the insufficient braking response of the existing obstacle detection mechanism. The output of the accident prediction model is incorporated as an additional constraint on this deceleration plan to suppress unnecessary plan activation in low-risk scenarios.

5.1 BDI Deceleration Strategy

Our new deceleration plan within the ML-MAS framework imposes additional safety constraints for high-speed driving scenarios. When the vehicle speed exceeds a predefined threshold, a single-frame deceleration command is applied to reduce it.

To evaluate the contribution of each component, we conduct a system-level comparison among four configurations: LAV-only, the original ML-MAS crossing-detection plan, ML-MAS with the proposed deceleration plan, and the complete MLAPM-MAS with the Accident Prediction Model. LAV-only serves as the neural driving baseline, while the original ML-MAS serves as the BDI-based baseline. The experiments are conducted on the first three routes of the Town01 map in the CARLA Longest6 benchmark. All results are evaluated using the standard CARLA Leaderboard metrics. As shown in Table 2, introducing the BDI control layer improves the Driving Score from 43 to 56 compared with the LAV-only baseline. Adding the proposed deceleration plan reduces both pedestrian and vehicle collision metrics to 0.00. Incorporating the Accident Prediction Model produces the best overall Driving Score of 99, with only a slight lane-deviation penalty. These results indicate that the BDI control layer, the deceleration strategy, and the Accident Prediction Model each contribute to driving safety and overall performance.

Table 2: Performance comparison in front-crossing scenarios. "Dec Plan" denotes ML-MAS equipped with the proposed deceleration plan, while MLAPM-MAS integrates the accident prediction model on top of the same deceleration plan. Bold font identifies best values.

Metric	LAV	ML-MAS	Dec Plan	MLAPM-MAS
Collision (Pedestrian)	0.020	0.018	**0.000**	**0.000**
Collision (Vehicle)	0.018	0.008	**0.000**	**0.000**
Red Light Violations	0.008	0.009	0.026	**0.000**
Lane Deviation	**0.000**	**0.000**	**0.000**	0.008
Driving Score	43	56.000	78.100	**99**

5.2 BDI Deceleration Plan Triggered by Accident Prediction Model

To ensure safety while reducing unnecessary activations of the deceleration plan and thereby lowering decision-making overhead, we introduce the accident prediction model as a triggering constraint. This accident prediction model in MLAPM-MAS is implemented by TGAT. The raw input data of the inference phase are obtained from the CARLA simulation environment. The Orchestrator module in the ML-MAS framework is used to obtain global traffic-state information from the current simulation scenario. The relative spacing between vehicles is computed from this information and used as the edge-feature input during model inference. We compute and compare the accident prediction scores for the two vehicles closest to the ego vehicle, and select the highest score as the assessment result at the current time step. The historical observation sequence is divided into 6 time segments, each sampled at 2 fps, to construct the temporal input for the ego vehicle and its interacting neighbours. An additional 8th time segment is introduced after the observed frames and used as a target timestamp.

Algorithm 1 Single-frame slowdown triggered by neural risk belief.

Input:
Risk alert belief from Neural Network: $mlAlert$
Ego state at frame F_1: speed Sp, steering st
Scene information at frame F_1: $info(F_1, Sp, FS, DS)$
Speed threshold for high-speed handling: Sp_{thr}
Output:
Control command u

```
HighSpeed ← (Sp > Sp_thr)
if mlAlert ∧ HighSpeed then
    u ← SlowDown(duration = 1 frame)
return u
```

The edge features for this segment are set to zero, so it serves solely as the target time point for risk assessment. The risk decision threshold is set to 0.6 in the CARLA online inference-and-control scenario. When the risk score exceeds this threshold, the risk assessment model returns a collision warning signal to the Orchestrator module.

If the collision warning signal is activated, the BDI agent takes over vehicle control to initiate deceleration. Given the current vehicle speed v_0 and a predefined deceleration rate a, the time required to complete deceleration can be estimated as $t\ \frac{v_0}{a}$. Based on the simulation control frequency f, this deceleration time is then mapped to the corresponding number of control frames $N = tf$.

The Orchestrator continuously provides the collision-warning signal as a Belief to the BDI agent for N consecutive control frames. The BDI agent checks whether obstacles are ahead and whether the vehicle exceeds the speed limit, and executes a deceleration action when these conditions are satisfied, as shown in Algorithm 1.

As shown in Table 2, the "Deceleration Plan (MLAPM-MAS)" indicates that introducing collision warnings as additional constraints does not increase the collision rate compared to using the deceleration plan alone. Table 3 shows that although the total number of frames increases across all routes, the proportion of BDI interventions presents a consistent decreasing trend. These results demonstrate that the neural network's risk beliefs act as a filter within the symbolic decision layer, reducing unnecessary plan activations and control takeovers in low-risk scenarios. This mechanism maintains stable overall driving performance while reducing the frequency of BDI decision interventions and the associated symbolic reasoning overhead.

Table 4 compares the driving performance metrics of ML-MAS and MLAPM-MAS across all of the "Longest6 benchmark" routes. The execution is dynamic, and different runs can yield different outcomes. We ran each framework three times[2] and selected the results from the best run (higher driving score) for each. MLAPM-MAS shows improved performance in detecting pedestrian and vehicle

[2] Each run of the entire benchmark can take several days, even on a powerful machine.

Table 3: Comparison of BDI intervention ratios with and without accident prediction model. "Dec Plan" denotes ML-MAS equipped with the proposed deceleration plan, while MLAPM-MAS integrates the accident prediction model into the same deceleration plan. Bold font identifies best values.

Metric	Dec Plan	MLAPM-MAS
Route 1 Frames	6947	8897
Route 1 Ratio	0.071	**0.033**
Route 2 Frames	11715	11422
Route 2 Ratio	0.035	**0.028**
Route 3 Frames	6456	7759
Route 3 Ratio	0.051	**0.019**

collisions. The pedestrian collision rate is reduced from 0.024 to 0, while the vehicle collision rate decreases from 0.191 to 0.132. These results indicate that the MLAPM-MAS can cover most potential pedestrian accident scenarios and can identify and respond to high-risk interactions between vehicles.

The additional deceleration actions triggered by accident prediction also positively affect metrics related to traffic rules. Red-light violations and lane deviations have decreased. In general, lower driving speeds improve a car's controllability. However, the deceleration plan also introduces certain side effects. Collisions with layouts and traffic-jam events are increasing. This partially explains the reduction in the Route Completion Score from 91.43 to 87.79.

Table 4: Driving performance metrics on CARLA Longest6 Benchmark. Bold font identifies best values. Lower values are better in all metrics except the last three, where higher values are better. The penalty score is a coefficient that starts at 1 and decreases with each traffic violation.

Driving Performance Metrics	ML-MAS	MLAPM-MAS
Collision (Layout)	**0.063**	0.083
Collision (Pedestrian)	0.024	**0.000**
Collision (Vehicle)	0.191	**0.132**
Lane Deviation	0.103	**0.089**
Red Light Violations	0.201	**0.161**
Route Deviation	**0.041**	0.045
Route Timeout	0.070	**0.048**
Stop Infractions	0.140	**0.139**
Traffic Jam Incidents	**0.050**	0.084
Penalty Score	0.476	**0.580**
Route Completion Score	**91.430**	87.790
Driving Score	42.600	**48.47**

6 Related Work

Recent works have explored the use of large language models to enhance decision-making in autonomous driving systems. Instead of relying solely on direct sensor-to-action mappings, these approaches introduce hierarchical architectures in which LLMs serve as high-level reasoning modules, transforming perceptual inputs into semantic representations and guiding decisions through structured inference. A common design choice in these methods is to distinguish between reactive decision-making and deliberative reasoning, where immediate responses are complemented by higher-level reasoning processes [13,23,22].

The reactive component ensures real-time responsiveness through data-driven or end-to-end control, while the reasoning component leverages large language models to perform language-guided decision-making via structured prompting and semantic reasoning. Specifically, perceptual inputs are transformed into linguistic or semantic representations, which are then processed through mechanisms such as chain-of-thought reasoning, rule extraction, or reflective prompting to generate high-level decisions [19,13,23,22]. These works highlight the growing role of language-based reasoning over semantic representations for improving decision-making in complex driving scenarios.

Another line of work focuses on integrating neural perception with explicit symbolic reasoning for decision-making in autonomous driving. Instead of relying on implicit representations learned through end-to-end training, these approaches introduce intermediate symbolic abstractions, such as knowledge graphs, logical rules, or domain-specific programs, which serve as the foundation for decision-making [21,12,17]. A key characteristic of these methods is the formulation of driving decisions as a structured reasoning process over symbolic representations [21,17]. Neural components are responsible for perception and feature extraction, while symbolic modules enforce constraints, encode prior knowledge, and guide decision logic [12,17]. Although these approaches differ in their strategies, including reinforcement learning, program synthesis, and imitation learning, they share a common objective of learning or constructing policies that operate on structured representations rather than raw observations. Overall, recent LLM-based and neuro-symbolic approaches show that autonomous driving decisions can benefit from structured intermediate representations. In these methods, the neural component is often part of the decision-making mechanism itself.

MLAPM-MAS addresses a different design challenge. In this system, LAV provides the baseline driving policy, while the BDI agent reasons over its own belief base and plan library rather than being determined by LAV. The Accident Prediction Model adds accident-risk warnings to the BDI belief base, so the BDI layer does not need to manually define all complex risk conditions as plan contexts. This design separates learned risk inference from executable BDI decision-making while allowing the inferred risk to serve as a belief that guides BDI plan selection.

7 Conclusion

This work builds upon the dual-system neuro-symbolic architecture provided by ML-MAS, mapping accident-risk predictions from a TGAT model into symbolic beliefs within a BDI agent and using them to constrain the activation of BDI plans. Results demonstrate that while maintaining stable overall driving performance and safety metrics, introducing accident-prediction beliefs reduces unnecessary symbolic plan interventions in low-risk scenarios, thereby lowering the execution frequency of the symbolic decision layer. The neural model is responsible for continuous risk estimation, while the symbolic reasoning layer executes rule-based decisions, with a clear separation of roles maintained at the decision stage. The neural network model influences only the input conditions of symbolic decision-making, rather than replacing rule-based reasoning or plan execution. This preserves the explicit structure of the symbolic decision process.

Future work will explore how symbolic information generated during the BDI reasoning process can be fed back into neural network inference or training. This would enable bidirectional interaction between System 1 and System 2.

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

References

1. Al Shukairi, H., Cardoso, R.C.: ML-MAS: A hybrid AI framework for self-driving vehicles. In: AAMAS'23: Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS) (2023). https://doi.org/10.65109/ETRN2911
2. Bachute, M.R., Subhedar, J.M.: Autonomous Driving Architectures: Insights of Machine Learning and Deep Learning Algorithms. Machine Learning with Applications **6**, 100164 (Dec 2021). https://doi.org/10.1016/j.mlwa.2021.100164
3. Booch, G., Fabiano, F., Horesh, L., Kate, K., Lenchner, J., Linck, N., Loreggia, A., Murugesan, K., Mattei, N., Rossi, F., Srivastava, B.: Thinking fast and slow in AI. In: Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021. pp. 15042–15046. AAAI Press (2021). https://doi.org/10.1609/AAAI.V35I17.17765
4. Bordini, R.H., Hübner, J.F., Wooldridge, M.: Programming multi-agent systems in AgentSpeak using Jason. John Wiley & Sons (2007)
5. Chen, D., Krahenbuhl, P.: Learning from All Vehicles. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 17201–17210. IEEE, New Orleans, LA, USA (Jun 2022). https://doi.org/10.1109/CVPR52688.2022.01671
6. Chitta, K., Prakash, A., Jaeger, B., Yu, Z., Renz, K., Geiger, A.: Transfuser: Imitation with transformer-based sensor fusion for autonomous driving. IEEE Transactions on Pattern Analysis and Machine Intelligence pp. 1–18 (2022). https://doi.org/10.1109/TPAMI.2022.3200245

7. Darapaneni, N., R, P.R., Reddy Paduri, A., Anand, E., Rajarathinam, K., Eapen, P.T., K, S., Krishnamurthy, S.: Autonomous car driving using deep learning. In: 2021 2nd International Conference on Secure Cyber Computing and Communications (ICSCCC). pp. 29–33 (May 2021). https://doi.org/10.1109/ICSCCC51823.2021.9478090
8. Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., Koltun, V.: CARLA: An Open Urban Driving Simulator. In: Proceedings of the 1st Annual Conference on Robot Learning. pp. 1–16. PMLR (Oct 2017), https://proceedings.mlr.press/v78/dosovitskiy17a.html, iSSN: 2640-3498
9. Fabiano, F., Ganapini, M.B., Loreggia, A., Mattei, N., Murugesan, K., Pallagani, V., Rossi, F., Srivastava, B., Venable, K.B.: Thinking Fast and Slow in Human and Machine Intelligence. Communications of the ACM **68**(8), 72–79 (Aug 2025). https://doi.org/10.1145/3715709
10. Kahneman, D.: Thinking, fast and slow. Farrar, Straus and Giroux, New York (2011)
11. Kautz, H.: The third AI summer: AAAI Robert S. Engelmore memorial lecture. AI Magazine **43**(1), 105–125 (Mar 2022). https://doi.org/10.1002/aaai.12036
12. Khan, H., Chaudhari, T.D., Ramesh, J.V.N., Kranthi, A.S., Muniyandy, E., El-Ebiary, Y.A., Devadhas, D.N.P.: Neuro-Symbolic Reinforcement Learning for Context-Aware Decision Making in Safe Autonomous Vehicles. International Journal of Advanced Computer Science and Applications **16**(5) (2025). https://doi.org/10.14569/IJACSA.2025.0160558
13. Ma, Y., Wei, T., Zhong, N., Mei, J., Hu, T., Wen, L., Yang, X., Shi, B., Liu, Y.: Leapvad: A leap in autonomous driving via cognitive perception and dual-process thinking. IEEE Transactions on Neural Networks and Learning Systems **37**(4), 1963–1977 (Apr 2026). https://doi.org/10.1109/TNNLS.2025.3626711
14. Ovadia, Y., Fertig, E., Ren, J., Nado, Z., Sculley, D., Nowozin, S., Dillon, J., Lakshminarayanan, B., Snoek, J.: Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 32. Curran Associates, Inc. (2019), https://proceedings.neurips.cc/paper_files/paper/2019/file/8558cb408c1d76621371888657d2eb1d-Paper.pdf
15. Rawat, D.B.: Towards Neuro-Symbolic AI for Assured and Trustworthy Human-Autonomy Teaming. In: 2023 5th IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA). pp. 177–179 (Nov 2023). https://doi.org/10.1109/TPS-ISA58951.2023.00030
16. Rudin, C.: Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nature Machine Intelligence **1**, 206–215 (05 2019). https://doi.org/10.1038/s42256-019-0048-x
17. Sun, J., Sun, H., Han, T., Zhou, B.: Neuro-Symbolic Program Search for Autonomous Driving Decision Module Design. In: Proceedings of the 2020 Conference on Robot Learning. pp. 21–30. PMLR (Oct 2021), https://proceedings.mlr.press/v155/sun21a.html, iSSN: 2640-3498
18. Wang, T., Kim, S., Wenxuan, J., Xie, E., Ge, C., Chen, J., Li, Z., Luo, P.: DeepAccident: a motion and accident prediction benchmark for V2X autonomous driving. In: Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence. AAAI'24/IAAI'24/EAAI'24, AAAI Press (2024). https://doi.org/10.1609/aaai.v38i6.28370

19. Wu, M., Yu, F.R., Liu, P.X., He, Y.: Facilitating autonomous driving tasks with large language models. IEEE Intelligent Systems **40**(1), 45–52 (Jan 2025). https://doi.org/10.1109/MIS.2024.3466518
20. Xu, D., Ruan, C., Korpeoglu, E., Kumar, S., Achan, K.: Inductive representation learning on temporal graphs. In: International Conference on Learning Representations (2020), https://openreview.net/forum?id=rJeW1yHYwH
21. Yang, S., Zhang, M., Feng, X., Hua, Y., Cao, Y.: Deep reinforcement learning-based knowledge graph reasoning for autonomous driving systems. IEEE Transactions on Industrial Informatics **22**(4), 3341–3351 (Apr 2026). https://doi.org/10.1109/TII.2025.3649292
22. Zhang, X., Wang, K., Hu, T., Ma, H.: Enhancing autonomous driving through dual-process learning with behavior and reflection integration. In: ICASSP 2025 - 2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 1–5. IEEE, Hyderabad, India (Apr 2025). https://doi.org/10.1109/ICASSP49660.2025.10889717
23. Zhang, Y., Liu, J., Xu, C., Hang, P., Sun, J.: Lead: The llm enhanced planning system converged with end-to-end autonomous driving. In: 2025 IEEE 28th International Conference on Intelligent Transportation Systems (ITSC). pp. 1028–1035. IEEE, Gold Coast, Australia (Nov 2025). https://doi.org/10.1109/ITSC60802.2025.11423844
24. Zhao, J., Wu, Y., Deng, R., Xu, S., Gao, J., Burke, A.: A survey of autonomous driving from a deep learning perspective. ACM Comput. Surv. **57**(10), 263:1–263:60 (May 2025). https://doi.org/10.1145/3729420