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 System 1 and System 2. 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]:

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

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 novel 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 for constructing TGAT input features. Node features are represented as three-dimensional bounding boxes for traffic actors and comprise their geometric and semantic attributes. Edge features describe the physical relationships between different actors. The spatial information of all agents is unified within CARLA's world coordinate system. Each traffic actor is associated with its position coordinates p_l heading angle γ_l and velocity vector v_l.

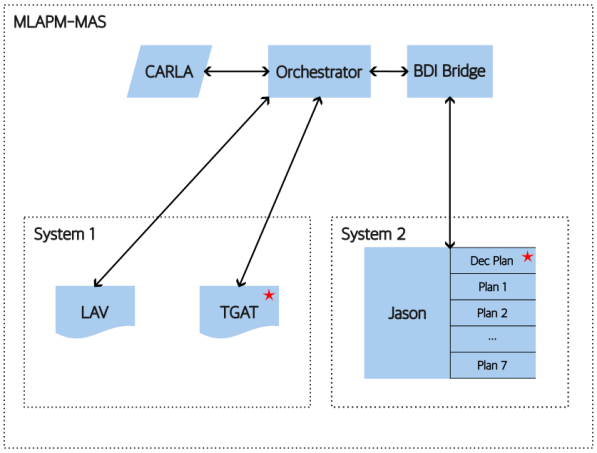

Fig. 1: MLAPM-MAS overview, with the ⋆ representing the novel extensions. Dec Plan are the novel deceleration plans.

The information is expressed in each agent's local LiDAR coordinate system and cannot be used directly until it is transformed into a common world coordinate frame. DeepAccident provides the corresponding coordinate transformation parameters for this purpose. The rotation matrix R_{le} and the corresponding translation vector t_{le} are used to transform coordinates from the LiDAR coordinate frame to the ego vehicle[2] coordinate frame. The rotation matrix R_{ew} and its corresponding translation vector t_{ew} are used to transform coordinates from the ego vehicle coordinate frame to the world coordinate frame.

All object states are expressed in the CARLA world frame using dataset-provided rigid transforms:

$$p^w = T_{ew}T_{le}p^l, \qquad v^w = R_{ew}R_{le}v^l, \qquad \gamma^w = \mathrm{yaw}(R_{ew}R_{le}, \gamma^l), \tag{2}$$

where T_{le}, T_{ew} are SE(3) transforms (with rotations R_{le}, R_{ew}).

After transforming all required information for each object into the CARLA world coordinate system, the spatial distances and relative orientations between vehicles can be compared at any time instant. We abstract each vehicle and pedestrian as a rectangular bounding box and compute geometric relationships within a unified world coordinate frame for accident prediction.

Whether two vehicles exhibit spatial overlap can be approximated by examining the separation between their projections along the line connecting their centres. We adopt the minimum projection distance as the edge feature in the graph structure. This feature jointly captures both the relative spatial distance

[2] An ego vehicle is the main vehicle being modelled, controlled, or observed in a driving scenario.

and relative orientation between vehicles, and serves as an important physical measure for assessing potential collision risk.

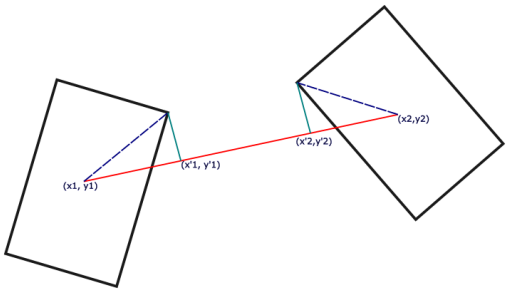

Fig. 2: Geometric illustration of the minimum projected distance between two vehicles.

Each vehicle is approximated as an oriented 2D rectangle (length l, width w) in the world frame. Let $d = \|p_2 - p_1\|_2$ and let ϕ be the angle between the line-of-centres and the vehicle yaw. The maximum half-extent of an oriented rectangle along direction ϕ is:

$$s(\phi) = \frac{l}{2}|\cos\phi| + \frac{w}{2}|\sin\phi|. \tag{3}$$

For a vehicle pair $(1, 2)$, define $s_1 = s(\phi_1)$ and $s_2 = s(\phi_2)$ with respect to the shared line-of-centres. The minimum projected separation (edge feature) is then:

$$d_s = d - (s_1 + s_2). \tag{4}$$

A negative d_s indicates overlap. A smaller d_s indicates a higher collision risk.

The corresponding graph structure is constructed once the node features and edge features have been defined. This graph structure represents the relationships between individual nodes.

3.2 TGAT Model for Accident Prediction

The training dataset must be updated before training the accident prediction model. The node features and edge features have already been defined in the preceding section. However, these features describe static relationships. To capture the temporal interactions among vehicles, the data needs to be organised into a temporal graph. Each interaction is represented as a timestamp edge (u, i, ts, r). u and i denote the node identifiers of the two interacting entities, ts denotes the timestamp at which the interaction occurs, and r represents the edge feature computed from the physical relationship between them. The timestamp ts of

the ego vehicle node n in the current event is first determined during temporal neighbour aggregation in TGAT. Neighbour timestamps ts_i satisfying $ts_i < ts$ are then selected from the historical timeline. The time differences Δt_i are encoded using (1) and concatenated with node and edge features prior to temporal attention aggregation.

TGAT aggregates temporally valid neighbours $N(u, ts) = \{(i, ts_i, r_i) \mid ts_i < ts\}$ using multi-head attention over concatenated inputs $[n_i \| r_i \| \Phi_m(ts - ts_i)]$. The resulting temporal embedding is:

$$e_u(ts) = \sum_{(i,ts_i,r_i)\in N(u,ts)} \alpha_{ui}(ts)\, g(n_i, r_i, \Phi_m(ts - ts_i))\,, \tag{5}$$

where $\alpha_{ui}(ts)$ are attention weights and $g(\cdot)$ is a learned projection (standard TGAT, we omit the details for simplicity).

Labels in DeepAccident are assigned to the two vehicles involved in a collision. All edge events corresponding to collisions are selected as positive samples (u_c, i_c, ts, r_c). TGAT performs attention aggregation for the ego vehicles u_c and the colliding vehicle i_c at time ts. It recursively aggregates information from their respective historical interaction sequences. Two temporal embeddings $e_{u_c}(ts)$ and $e_{i_c}(ts)$ are thus obtained for the two vehicles. Negative samples are constructed by randomly selecting a vehicle i_n that did not collide with u_c at the same time step ts. This is used to construct a negative edge sample (u_c, i_n, ts, r_n) and to obtain the corresponding negative embedding $e_{i_n}(ts)$. A similarity function f_θ is defined to compute similarity scores for positive and negative sample pairs, where θ is the learnable parameter of TGAT. s_p and s_n are scalar scores obtained by applying the similarity function to the temporal node embeddings, and they serve as logits in contrastive learning. Their values is in the range $s_p, s_n \in (-\infty, +\infty)$.

Training follows standard TGAT and DeepAccident practices: collision edges are treated as positive samples, and non-collision edges as negative samples. A similarity score $s = f_\theta(e_u(ts), e_i(ts))$ is mapped to a risk probability via a sigmoid and optimised with binary cross-entropy.

$$p = \sigma(s), \qquad \mathcal{L}(\theta) = \mathrm{BCE}(p, y). \tag{6}$$

TGAT uses the similarity function to evaluate the association between a vehicle pair and the occurrence of a collision event at a given timestamp in the contrastive learning framework. This association is determined by historical interaction patterns and their temporal dependencies. The current interaction continues to influence the propensity for future collision events, and this influence is reflected in the similarity function's output.

3.3 Online Inference and BDI Integration

TGAT is deployed in the CARLA to perform online evaluation of interaction relationships between vehicles in dynamic traffic scenarios. First, the three-dimensional bounding-box information and semantic labels for each traffic actor

are obtained from CARLA, and the relevant attributes are extracted and encoded into node feature vectors. Subsequently, the global positions and orientations of other vehicles are extracted to construct edge features in later stages.

The ego vehicle in Fig. 3 is taken as the reference vehicle, and a single frame is selected in which the minimum projected distance to all traffic actors in the scene is computed. The traffic actors are ranked in ascending order according to the projected distance, and the n actors with the smallest distance are selected as fixed interaction neighbours. Let $I_u = \{i_1, \ldots, i_n\}$ denote the set of fixed neighbours of the ego vehicle node u. Corresponding interaction event (u, i_k, ts_1, r_1) is constructed at the reference frame timestamp ts_1 for the ego vehicle node u and any neighbour node $i_k \in I_u$.

The selected n fixed neighbours are continuously tracked, and their edge feature data are collected up to the current time t. A historical interaction time series of length t, denoted as $\{(u, i_k, ts_j, r_j)\}_{j=1}^{t}$, is constructed for each node pair (u, i_k). ts_j is the historical timestamp of timestamp j. r_j means the edge feature computed from the minimum projected distance at the timestamp j.

We introduce virtual time indices after the observed frames as temporal anchors for subsequent embedding computation. The virtual time indices $t+1$ retain the static node features of the corresponding traffic actors while setting the edge feature to 0, thereby constructing timestamps that do not correspond to real physical states. The node features n_u of the ego vehicle and n_i of the neighbouring vehicle remain unchanged within the selected time window $[0, t+1]$. The node features and the edge features r_j at each historical timestamp j are projected into attention space via a linear mapping. This projection generates an attention embedding for the historical interaction events.

TGAT aggregates attention embeddings from historical interactions before the virtual time index, producing temporal embeddings for the ego vehicle and neighbouring vehicles at that timestamp (Fig. 4). $T1$ and $T2$ denote the starting points of different historical time windows, t denotes the length of the historical interaction sequence, and $t+1$ is an introduced virtual time index that serves as the temporal anchor for accident inference. Assuming the initial time is set to zero, the similarity function computes a similarity score s_{t+1} for the temporal embeddings at virtual time index $t+1$, which is then mapped through a sigmoid function to obtain the risk score $\sigma(s_{t+1})$.

The TGAT training objective encourages temporal embeddings associated with interaction patterns similar to collision samples to receive higher similarity scores in the representation space. Embedding corresponding to non-collision interaction patterns tends to receive lower scores. On this basis, a decision threshold can be defined in the risk-score space to distinguish between different risk levels. When the risk scores $\sigma(s_{t+1})$ fall below the threshold, the corresponding interactions are regarded as low risk. Risk scores exceeding the threshold indicate a higher risk of collision and trigger a collision warning. The ML-MAS Orchestrator converts this warning into a symbolic alert belief and updates the BDI belief base accordingly. This alert belief is maintained online at each control step and removed when no longer valid. The BDI agent then uses this belief during

Fig. 3: ⋆ represents the ego vehicle.

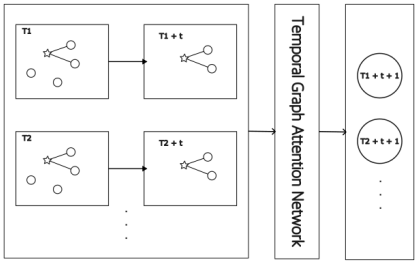

Fig. 4: Accident inference via TGA.

context checking and plan selection, enabling the corresponding risk-avoidance plan when necessary.

This decision-making strategy maps TGAT's risk assessment outputs into symbolic beliefs, enabling effective integration between the data-driven perception model and the rule-based cognitive decision mechanism. The ML model abstracts latent risk patterns from vehicle interaction data. The symbolic decision layer uses the threshold risk beliefs to activate goals and select intentions. The ML model is combined with interpretable symbolic logic rules, and both collectively participate in the control decision process. Although the ML model itself is difficult to interpret, the symbolic rule layer provides explicit logical constraints and traceable reasoning paths for decision making.

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 $t + 1$ in the training scenario. 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.

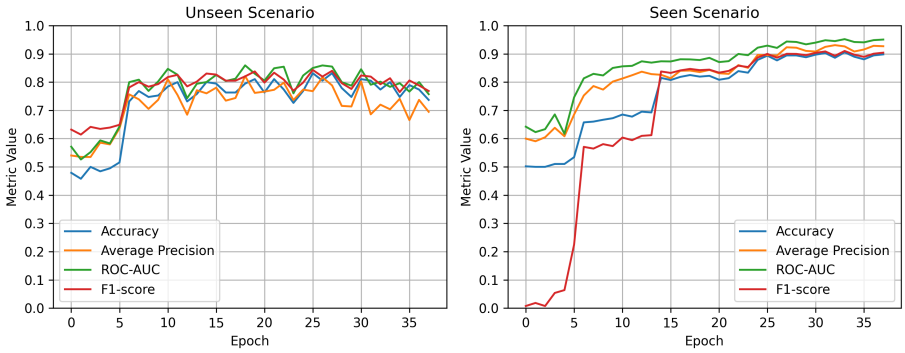

Fig. 5: Training results on unseen and seen scenarios.

As shown in Fig. 5, 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 leads to reduced 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 novel 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 novel 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 lower the vehicle speed.

We run experiments on the first three routes of the Town01 map in the CARLA Longest6 benchmark. The original crossing-detection plan in ML-MAS serves as the baseline scheme. LAV-only is introduced as an additional comparative baseline. All results are evaluated using the standard CARLA Leaderboard metrics for collision rate and overall driving score, based on the average of five runs per approach. Table 2 shows that introducing a BDI control layer within the Neuro-Symbolic architecture leads to an improvement in overall driving performance compared with the configuration using only the neural network LAV, with the Driving Score increasing from 43 to 56. Introducing the deceleration plan reduces the collision-rate metric to 0.00. Adding the Accident Prediction Model further improves the overall driving score to almost perfect, except for a slight penalty for lane deviation.

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 minimum projected distance 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. 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 $N = tf$.

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 is exceeding the speed limit, and executes the 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

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

$HighSpeed \leftarrow (Sp > Sp_{\text{thr}})$
if $mlAlert \wedge HighSpeed$ **then**
    $u \leftarrow$ `SlowDown`($duration = 1$ frame)
return u

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 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**

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[3] and selected the results from the best run (higher driving score) for each. MLAPM-MAS shows improved performance in pedestrian and vehicle collision detection. 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 the majority of potential pedestrian accident scenarios

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

and has the capability to 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 are reduced. 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 [21,17]. In contrast, MLAPM-MAS leverages a graph neural network to capture relational dependencies and temporal interaction patterns among multiple agents, supporting continuous risk assessment rather than static state reasoning. The use of a BDI framework provides an explicit, executable decision-making structure in which beliefs, goals, and plans are clearly defined and updated at runtime. This allows learned risk information to be directly incorporated into the reasoning process as actionable constraints.

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 interpretability 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)
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, D.H., Chaudhari, T.D., Ramesh, J.V.N., Kranthi, A.S., Muniyandy, E.: 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. da Xu, chuanwei ruan, evren korpeoglu, sushant kumar, kannan achan: 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