Supplementary Material for DriveCombo: Benchmarking Compositional Traffic Rule Reasoning in Autonomous Driving

Enhui Ma[1,2,6*†], Jiahuan Zhang[2,6*†], Guantian Zheng[2‡], Tao Tang[3†], Shengbo Eben Li[4], Yuhang Lu[5], Xia Zhou[3], Xueyang Zhang[3], Yifei Zhan[3], Kun Zhan[3], Zhihui Hao[3], Xianpeng Lang[3], Kaicheng Yu[2,6§]
[1]Zhejiang University [2]Autolab, Westlake University [3]Li Auto Inc
[4]Tsinghua University [5]The University of Hong Kong [6]Auwomo
{maenhui, kyu}@westlake.edu.cn

In this supplemental material, we provide additional details of our proposed DriveCTR benchmark. The content is organized as follows:

- **Section 1** provides additional details on the construction of the DriveCTR dataset.
- **Section 2** explains further experimental settings for evaluating MLLMs on DriveCTR.
- **Section 3** presents more analyses of the evaluation results, covering multi–rule compositional reasoning capabilities, and the potential of DriveCTR for closed-loop planning evaluation.

1. More Details about datasets

1.1. Candidate Rule Pairs Generation for Speed-Limit Rules

Speed-limit regulations exhibit numerical constraints that require a specialized pairing mechanism beyond the general normative-relation formulation described in the main paper. To properly model semantic and normative relationships among speed-related rules, we introduce a rule-pair generation strategy based on *speed-range intersection*.

Action-Type Filtering. We first isolate all atomic rules whose action type corresponds to speed regulation. $P = \{p_i = \{r_j, r_k\} \mid a_j = a_k = \text{SpeedLimit},\ r_j, r_k \in R\}_{i=1}^{|P|}$.

Speed-Range Modeling. Each speed-limit rule is mapped to a feasible speed interval: $sr_j = [l_j,\, u_j]$ and $sr_k = [l_k,\, u_k]$ where l_* and u_* denote the lower and upper bounds extracted from the rule's semantic content (e.g., "max 70 km/h," "min 110 km/h on expressways," or "max 30 km/h when visibility $<$ 50 m").

Derived Labels. For each candidate pair $p_i = \{r_j, r_k\}$, we assign three derived labels following the structure of the hierarchical rule system.

Each pair p_i is assigned three derived labels: the perceptual combination type b'_i, the normative relation type of speed ns'_i, and the hierarchical level l_i:

$$b'_i = \begin{cases} \text{Double Static}, & b_j = b_k = \text{static}, \\ \text{Double Dynamic}, & b_j = b_k = \text{dynamic}, \\ \text{Hybrid}, & \text{otherwise}. \end{cases}$$

$$ns'_i = \begin{cases} \text{Norm Conflict}, & sr_j \cap sr_k = \emptyset, \\ \text{Norm Harmony}, & sr_j \cap sr_k \neq \emptyset. \end{cases}$$

$$l_i = \begin{cases} 2, & b'_i = \text{Double Static}, n'_i = \text{Norm Harmony}, \\ 3, & b'_i = \text{Double Dynamic}, n'_i = \text{Norm Harmony}, \\ 4, & b'_i = \text{Hybrid}, n'_i = \text{Norm Harmony}, \\ 5, & n'_i = \text{Norm Conflict}. \end{cases}$$

Notably, all conflicting speed-limit rule pairs ($sr_j \cap sr_k = \emptyset$) are directly assigned to Level 5, as numerical incompatibility reflects a genuine priority-arbitration scenario in real driving.

1.2. Construction of Multi-level Questions

We create multi-level multiple-choice questions (L1–L5). Each includes a scenario, a standardized stem (e.g., "What should the driver do in this situation?"), and four options.

Correct Action Determination. The correct action a_i is generated according to the hierarchical rule level to which it belongs. For single-rule questions (L1), the correct action is directly specified by the atomic rule r_i. For compatible rule levels (L2–L4), a_i represents a behavior that simultaneously satisfies the semantic and normative constraints of the rule pair. When rule pairs at the conflict level (L5) yield contradictory instructions, a_i is determined according to the

*Co-first authors.
†Work done during an internship at Li Auto Inc.
‡Work done during their visiting at Autolab, Westlake University.
§Corresponding Author.

following priority principle [10, 11]. This principle ensures consistent and rational driving behavior in multi-signal conflicts, aligning the model's decisions with ethical and legal standards when normative cues compete. The behavioral priority order is summarized as follows:

> *Pedestrian safety > emergency vehicle avoidance > on-site command > traffic lights > traffic signs > road markings > interactive right-of-way > defensive driving > emergency exceptions.*

Option Design. Each question includes four options $o_i = \{o_i^*, o_{i1}, o_{i2}, o_{i3}\}$, where o_i^* is the correct, norm-compliant option derived from the correct action a_i. The other three distractors are generated by the LLM and manually filtered to ensure semantic plausibility while introducing normative deviations, including priority confusion, minor violations, or incorrect yielding. All options are jointly verified by the LLM and human experts

Finally, the final question set $T = \{(q_i, i_i, o_i)\}_{i=1}^{|S|}$ is ready, where q_i denotes the question stem, i_i the sequence of visual input frames, o_i the option set.

Specific Generation Process. For each rule or rule pair, we first prompt Claude Sonnet 4.5 [1] to generate a full MCQ sample in a fixed JSON format containing a scenario description, question stem, four options (A–D), the design logic, the correct answer, and an explanation. The prompt is shown in Table 4, 5 and 6. Next, to ensure correctness and rule consistency, each generated question is independently validated by Gemini 2.5 Pro [14], GPT 5 Pro [12], and Claude Sonnet 4.5 [1], which assess whether the answer is logically valid, whether the scenario faithfully reflects the input rules, and whether the options are well-structured. The prompt is shown in Table 7. Each model outputs a binary decision (`1` for valid, `0` for invalid), and a question is accepted only if all three models return `1`; otherwise, it is regenerated. After automated filtering, 5% of the accepted questions are randomly sampled for human expert verification to ensure overall quality and alignment with the intended cognitive levels.

1.3. Multi-Country Rule and Jurisdiction Modeling

To account for cross-country rule differences (e.g., driving side) and evaluate MLLM adaptability across regulatory systems, DriveCombo avoids mixing traffic rules across countries within the same evaluation instance. Instead, separate rule sets are constructed for each country, and modeling and evaluation are conducted independently. All countries' rules are built using the same data processing pipeline to ensure consistent rule modeling. We further report per-country performance in Sec. 3.2. In addition, the jurisdiction identity is explicitly specified in each question prompt, ensuring that MLLMs reason under the correct country-specific traffic rules.

1.4. Prompt Details for Scene Generation

This section provides detailed descriptions of the prompt designs used in the Rule2Scene pipeline, including semantic structuring, coexistence validation, scenario transcription, DSL translation with self-consistency checking, and LLM-based quality scoring.

Semantic Structuring Prompt. In the semantic structuring stage, the LLM is instructed to transform a natural-language traffic rule into a normalized atomic representation containing rule content, perceptual types, norm types, action types, and numeric constraints. This step ensures that every rule is converted into a consistent machine-interpretable structure before rule pairing or scene generation. The prompt is shown in Table 8.

Coexistence Validation Prompt. To determine whether the given atomic rules can coexist in a physically and semantically coherent driving scenario, the LLM receives a paired-rule input and evaluates spatial compatibility, temporal feasibility, weather or road-type requirements, and agent-role consistency. The model outputs a binary indicator representing feasibility. Only rule pairs that are jointly executable proceed to scenario generation. This filtering step prevents contradictory or impossible combinations from entering higher-level compositional reasoning tasks. The prompt is shown in Table 9.

Scenario Transcription Prompt. For each validated rule or rule pair, the LLM is prompted to convert the structured rule representation into a high-level natural-language scene description. This transcription step bridges abstract normative semantics with concrete driving contexts and provides the narrative foundation for subsequent DSL conversion and visual rendering. The prompt instructs the model to produce detailed but concise scene descriptions that correctly embed all rule constraints. This process is embedded within the prompt used for problem generation.

DSL Translation Prompt. After producing the scene description, the LLM is prompted to translate the textual scenario into a structured semantic DSL representation, including entities, relations, positions, trajectories, and environmental settings. The prompt is shown in Table 10.

LLM-Based Quality Scoring. To ensure the reliability of the generated scenarios, we incorporate an LLM-based quality assessment mechanism across all four stages of the pipeline: *Semantic Structuring*, *Coexistence Validation*, *Scenario Transcription*, and *DSL Translation*. At each stage, three advanced language models, Gemini 2.5 Pro [14], GPT 5 Pro [12], and Claude Sonnet 4.5 [1], independently evaluate the generated output and assign a quality score within the range $[0, 1]$. The final quality score is computed as the arithmetic mean of the three model predictions. If the average score falls below 0.6, the system automatically flags the sample and requests human intervention to ensure correctness and consistency. The prompt is shown in Table 11, 12, 13 and 14.

1.5. Action Type Set

To comprehensively represent the behavioral space of real-world driving, DriveCombo defines a structured action type set $\mathcal{A}$ composed of three major categories: *Driving Maneuvers*, *Lighting & Signaling*, and *Parking & Yielding*. These categories cover the full range of decision-making primitives required for traffic-rule reasoning and scenario generation.

Driving Maneuvers. This category captures fundamental vehicle motion behaviors and lane-level interactions.

- *Overtake*: Move ahead of a slower vehicle by passing it safely.
- *Left Turn*: Steer the vehicle to enter a road or lane on the left side.
- *Right Turn*: Steer the vehicle to enter a road or lane on the right side.
- *U-turn*: Rotate the vehicle 180 degrees to reverse its direction of travel.
- *Lane Change*: Move laterally from one lane to another while maintaining direction.
- *Merge Main Road*: Enter a primary roadway from a branch or side lane.
- *Enter Ramp*: Access a highway or exit using an on-ramp or off-ramp.
- *Acceleration*: Increase the vehicle's speed to match traffic flow or complete maneuvers.
- *Deceleration*: Reduce speed to adapt to road conditions or prepare for stopping.
- *Reverse*: Move the vehicle backward using rearward motion.
- *Emergency Lane Usage*: Drive or stop on the emergency lane under special conditions.

Lighting and Signaling. This category includes communication and visibility-related actions that convey driver intent.

- *Left Turn Signal*: Activate the left indicator to show intention to turn or change lanes leftward.
- *Right Turn Signal*: Activate the right indicator to signal rightward turning or lane change.
- *Low Beam*: Use dipped headlights for normal nighttime or low-light driving.
- *High Beam*: Use strong headlights to extend visibility when no oncoming traffic is present.
- *Flashing Headlights*: Briefly flash headlights to warn or signal other road users.
- *Double Flashers*: Activate hazard lights to indicate emergencies or temporary stopping.
- *Fog Lights*: Use specialized lights designed for low-visibility foggy conditions.
- *Position Lights*: Turn on minimal lighting to indicate the vehicle's presence when stationary or parked.
- *Honk Horn*: Use the horn to warn pedestrians or communicate urgency to nearby vehicles.

Parking and Yielding. This category covers controlled stop-and-yield maneuvers essential for compliance with road regulations.

- *Temporary Parking*: Stop the vehicle briefly at a roadside location without turning off the engine.
- *Pull Over*: Move the vehicle to the roadside for inspection, hazard avoidance, or instructions.
- *Yield*: Give way to pedestrians or other vehicles according to right-of-way rules.

Together, these action types define the complete action space $\mathcal{A}$ used throughout DriveCombo for atomic rule extraction, rule pairing, and scene generation. The coverage of lane operations, signaling behaviors, and yielding-related actions ensures that the benchmark faithfully reflects the breadth of real-world driving rules and constraints.

1.6. Scene Mapping in Simulator

The pipeline takes structured semantics d_i^* as input and converts the abstract semantics to a standardized OpenSCENARIO [2] dynamic scene w_i through layered parsing, instantiation, and trajectory generation. To make the pipeline clearer, we illustrate the three intermediate outputs of Scene Weaver in our proposed Rule2Scene Agent in Figure 8. It is worth noting that our pipeline is built on TARGET [3] and further extends it from synthesizing scenes that support only a single atomic traffic rule to synthesizing complex scenes that support multiple traffic rules. We also expand the capability from inserting only one actor to inserting multiple actors, and introduce a new ability for joint inference of dynamic trajectories for multiple agents.

Construction of Static Initial Scene. This stage transforms the semantic description into an initial scene state with explicit spatial and environmental attributes. This stage first maps the abstract categories to the assets that can be loaded in CARLA [4] according to the entity type configuration in the structured semantics d_i^*. It then obtains precise geometric information through CARLA road topology queries, lane centerline extraction, and coordinate transformation interfaces. This information is used to compute the initial position, orientation, and speed of each traffic participant, while also supporting relative constraint-based generation and region-specified sampling. In addition, this stage parses the environmental conditions based on the weather field in d_i^*. It applies rule-based mappings to convert weather semantics in d_i^* into concrete weather parameters required by CARLA to ensure executability during simulation. To ensure overall scene usability, the process continuously performs positional legality checks, collision detection, orientation validation, and boundary condition verification, and resamples when necessary. The static scene produced at this stage contains complete initial entity states and environmental settings, providing the static initial scene for subsequent trajectory generation.

Trajectory Generation and Dynamic Scene Construction. This stage generates complete temporal trajectories based on the static initial state described above, enabling continuous dynamic evolution of the scene in CARLA. The pipeline first generates feasible paths for each entity according to CARLA road topology and lane connectivity, while inferring driving strategies from the behavioral semantics in d_i^*. The strategies include driving along the centerline, lane changes, following, interactive approaching, and pedestrian navigation. After generating the paths, the pipeline samples the trajectories in the temporal dimension. These temporal trajectories are then encoded as OpenSCENARIO action structures and rewritten into motion commands that can be executed directly in CARLA through the trajectory element, as shown in Figure 8(c). This stage integrates spatial layout with traffic behaviors, producing motion that is physically plausible and consistent with the topology.

Through integration with CARLA map topology, the pipeline establishes a continuous and coherent transformation pipeline among semantic parsing, spatial instantiation, and temporal trajectory generation, thereby achieving a complete generation process from high-level intent to executable traffic simulation.

1.7. More task examples of DriveCombo

We provide additional examples of MCQs under the #Rules=2 setting in DriveCombo, with Figures 2 through 6 corresponding to tasks L1 through L5. In addition, we present examples involving combinations of multiple traffic rules, with Figure 7 showing MCQs examples under the #Rules=3, #Rules=4, and #Rules=5 settings.

2. More Details about Experiment Setup

2.1. General Evaluation Setup

To ensure reproducibility and eliminate variance introduced by stochastic decoding, all models are evaluated using a deterministic greedy decoding strategy. Specifically, we set temperature to 0 and fix both top-p and top-k to 1 for all inference runs.

For the DriveCombo benchmark, the evaluation input consists of four visual frames extracted from each generated scenario, serving as the multimodal observation for the model. For the DriveCombo-Text variant, we replace visual inputs with the corresponding textual scene descriptions while keeping all other components identical. The prompt is shown in Table 15 and 16.

2.2. Enhancement Evaluation Setup

To further investigate whether conventional reasoning-enhancement strategies can mitigate the performance gaps observed in DriveCombo, we evaluate three representative approaches: Chain-of-Thought (CoT) [17], Retrieval-Augmented Generation (RAG) [9], and Supervised Fine-Tuning (SFT). The following details the implementation of each method.

Chain-of-Thought (CoT). For CoT prompting, we apply standard step-by-step reasoning instructions without additional supervision or fine-tuning. During inference, each model receives a fixed CoT template encouraging explicit logical decomposition ("think step by step"), followed by answering the final multiple-choice question. No rule-specific optimization or task-dependent heuristics are applied, ensuring a training-free enhancement protocol.

Retrieval-Augmented Generation (RAG). For RAG, we construct a retrieval corpus consisting exclusively of the *original traffic rule books* used in building our atomic rule set R. Each model retrieves top-k relevant passages (with $k = 5$) using a dense retriever and conditions its answer on the concatenation of the query, retrieved rule excerpts, and the scenario description. This design ensures that the retrieved content reflects authentic legal constraints and avoids contamination from model-generated text. No scenario frames or DriveCombo samples are included in the retrieval database.

Supervised Fine-Tuning (SFT). For supervised adaptation, we employ the LLaMA-Factory framework [20] to fine-tune a set of representative open-source models: Gemma 3 4B [15], Gemma 3 12B, Qwen3-VL 2B [18], Qwen3-VL 8B, and Llama 3.2 11B [19]. Our goal is to enhance the models' capability for compositional traffic-rule reasoning through direct exposure to DriveCombo training samples. All experiments are conducted on a cluster equipped with 8 NVIDIA H800 GPUs (80GB per GPU). We adopt LoRA (Low-Rank Adaptation) [5] as the core optimization strategy. By leveraging low-rank matrix decomposition, LoRA enables efficient parameter adjustment while avoiding the substantial computational cost associated with full fine-tuning. Specifically, we set the LoRA rank to 8, an empirically determined configuration that balances model expressiveness, the volume of parameter updates, and memory consumption. To ensure the efficiency and stability of distributed training, we employ the DeepSpeed ZeRO-3 optimization strategy. This approach partitions model parameters, gradients, and optimizer states, thereby enabling highly efficient memory management and computational scheduling. As a result, it significantly improves training speed and parallelism in multi-GPU or multi-node environments. Regarding training configuration, we set the per-device batch size to 4, which helps avoid out-of-memory errors under limited GPU memory while maintaining a reasonable level of diversity. We further set the gradient accumulation steps to 4, allowing us to simulate a larger effective batch size despite the small actual batch size. This improves the accuracy of gradient estimation without increasing the computational load per step. To facilitate stable convergence, we use a learning rate of 1e-4. Finally, we train the model for three epochs, striking a balance between computational efficiency and adequate model refinement under resource constraints.

3. More Analysis of Results

3.1. Potential of Closed-loop Planning Evaluation

In the main paper, we show that the DriveCombo dataset can evaluate the reasoning of traffic rules in MLLMs through a question and answer format. We further note that the dataset is also suitable for closed-loop trajectory evaluation of end to end (E2E) models, making it a comprehensive benchmark that covers both reasoning and planning capabilities. DriveCombo's scenarios provide highly structured descriptions of traffic events, enabling plug-and-play integration with the Bench2Drive framework [7]. As shown in Figure 1, we can seamlessly convert DriveCombo's scenarios into closed-loop test cases to examine E2E models under real rule constraints. Specifically, we first align the original OpenScenario scenes to the Bench2Drive format and import them into CARLA. We then use the PDM-lite expert model [13] to generate ground-truth trajectories and evaluate various E2E models.

Using the Bench2Drive evaluation setup, we test E2E methods on fifteen short routes. All metrics follow the Bench2Drive evaluation standard. As shown in Table 1, both E2E models exhibit substantial performance drops on DriveCombo. For example, the Driving Score of UniAD Base [6] decreases from 45.81 to 27.59 and VAD [8] decreases from 42.35 to 26.68. Metrics related to efficiency and comfort also decline significantly. These results indicate that DriveCombo presents greater challenges in terms of rule complexity, traffic interaction and dynamic risk, and imposes stricter requirements on E2E methods. We plan to further expand the DriveCombo dataset to enable more comprehensive and systematic evaluation of trajectory planning capabilities.

3.2. Performance of MLLMs across countries on DriveCombo.

Results reported in the main paper are averaged over five countries to provide an overall assessment, while per-country performance is reported in Table 2 to reveal cross-country performance variations. The table reports average performance over 8 open-source models and 6 proprietary models evaluated in the main paper, highlighting performance variations across different national traffic rule systems. Overall, models achieve relatively higher accuracy on the USA and China subsets, while performance is comparatively lower on Japan and Australia, indicating potential challenges posed by country-specific regulatory structures and rule distributions.

3.3. Multi-Rule Compositional Reasoning

Table 3 evaluates the performance of MLLMs in multi-rule driving scenarios. As shown, under the #Rules = 3 setting, both open-source and proprietary models achieve relatively high scores. For example, larger open-source models like GLM-4.5V [16] reach about 68.15 accuracy (acc.) on L2–L4, GPT-5 Pro [12] further improves to 71.09. However, when the number of rules increases to #Rules = 4, all models show noticeable degradation, particularly on the challenging L4 and L5 tasks. In the high-complexity setting of #Rules = 5, even the best proprietary model, GPT-5 Pro, achieves only 34.78 acc in L5; the best open-source model, GLM-4.5V, reaches 32.78, also showing a clear performance degradation. Overall, as the complexity of the scenario increases (i.e., #Rules from 3 to 5), all models exhibit performance declines, indicating that DriveCombo introduces substantial difficulty in traffic-rule understanding and cross-entity interaction reasoning, and that current MLLMs still struggle with high-complexity driving scenarios.

References

[1] Anthropic. System card: Claude sonnet 4.5. Anthropic Blog, 2025. https://www.anthropic.com/claude. 2, 3

[2] ASAM. Asam openscenario: User guide. https://www.asam.net/index.php?eID=dumpFile&t=f&f=4092&token=

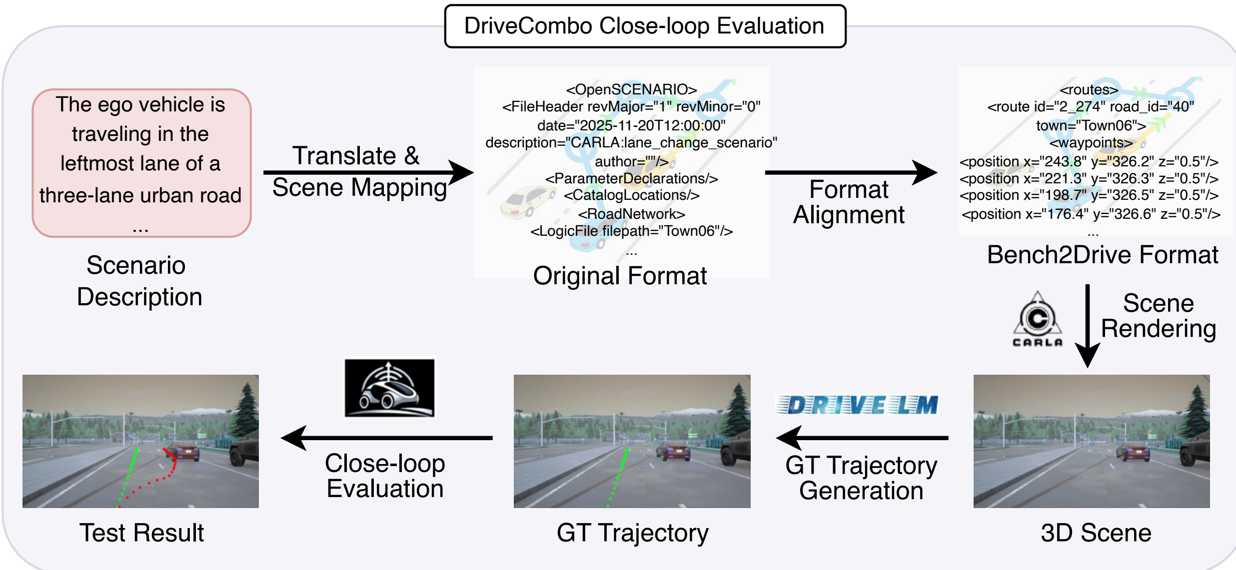


Figure 1. **DriveCombo's scenarios can be seamlessly converted into closed-loop test cases to evaluate end-to-end models under real rule constraints.** The green line means the GT trajectory and the red means the test trajectory.

Table 1. **Closed-loop Results of E2E-AD Methods in Bench2Drive and our DriveCombo.**

Method	**Benchmark**	**Closed-loop Metrics**			
		Driving Score ↑	**Success Rate** (%) ↑	**Efficiency** ↑	**Comfortness** ↑
UniAD-Base [6]	Bench2Drive [7]	45.81	16.36	129.21	43.58
	DriveCombo	27.59	13.33	101.32	31.92
VAD [8]	Bench2Drive [7]	42.35	15	157.94	46.01
	DriveCombo	26.68	13.33	116.53	33.96

d3b6a55e911b22179e3c0895fe2caae8f5492467, 2021. 3

[3] Yao Deng, Zhi Tu, Jiaohong Yao, Mengshi Zhang, Tianyi Zhang, and Xi Zheng. Target: Traffic rule-based test generation for autonomous driving systems. *IEEE Transactions on Software Engineering*, 2025. 3

[4] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. In *Conference on robot learning*, pages 1–16. PMLR, 2017. 3

[5] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022. 5

[6] Yihan Hu, Jiazhi Yang, Li Chen, Keyu Li, Chonghao Sima, Xizhou Zhu, Siqi Chai, Senyao Du, Tianwei Lin, Wenhai Wang, et al. Planning-oriented autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17853–17862, 2023. 5, 6

[7] Xiaosong Jia, Zhenjie Yang, Qifeng Li, Zhiyuan Zhang, and Junchi Yan. Bench2drive: Towards multi-ability benchmarking of closed-loop end-to-end autonomous driving. *Advances in Neural Information Processing Systems*, 37:819–844, 2024. 5, 6

[8] Bo Jiang, Shaoyu Chen, Qing Xu, Bencheng Liao, Jiajie Chen, Helong Zhou, Qian Zhang, Wenyu Liu, Chang Huang, and Xinggang Wang. Vad: Vectorized scene representation for efficient autonomous driving. *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8306–8316, 2023. 5, 6

[9] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *NeurIPS*, 2020. 4

[10] Standing Committee of the National People's Congress of the People's Republic of China. Road traffic safety law of the people's republic of china (2021 amendment). https://jtgl.beijing.gov.cn/jgj/jgxx/flfg/fl/205308/index.html, 2021. Originally adopted on Oct. 28, 2003; amended on Apr. 29, 2021; effective from May 1, 2024. 2

[11] State Council of the People's Republic of China. Regulations for the implementation of the road traffic safety

Table 2. **Performance of MLLMs across countries on DriveCombo.** We report average results of 8 open-source and 6 proprietary models used in the main paper.

	Open-source Models (Avg.)					**Proprietary Models (Avg.)**				
Country	**L1**	**L2**	**L3**	**L4**	**L5**	**L1**	**L2**	**L3**	**L4**	**L5**
USA	71.36	67.69	62.26	58.98	35.11	86.29	79.50	72.93	68.69	42.71
China	72.38	67.25	63.01	59.99	34.76	85.89	79.22	73.03	68.95	42.41
UK	70.07	66.56	62.48	56.33	33.58	83.55	78.80	72.66	67.91	41.29
Japan	63.60	58.06	55.77	52.28	30.11	73.86	71.56	65.36	63.03	37.26
Australia	61.75	55.18	51.29	49.96	29.68	79.05	72.09	64.03	62.39	37.90
Average	67.83	62.95	58.96	55.51	32.65	81.73	76.23	69.60	66.20	40.31

Table 3. **Performance of MLLM in Complex Driving Scenarios with Different Numbers of Rules.** "#Rules" denotes number of traffic rules in each scenario. Green and light green mark the best and second-best open-source models, while blue and light blue indicate the best and second-best proprietary models.

Model	**Size**	**L2**	**L3** **#Rules=3**	**L4**	**L5**	**L2**	**L3** **#Rules=4**	**L4**	**L5**	**L2**	**L3** **#Rules=5**	**L4**	**L5**
Open-source Models													
Gemma 3	4B	47.11	44.57	41.89	20.05	44.40	42.52	39.91	16.57	40.93	38.12	32.06	13.89
Gemma 3	12B	56.30	54.85	50.14	24.41	45.97	43.82	43.31	23.43	43.41	41.57	37.72	20.45
Gemma 3	27B	60.74	58.18	54.23	33.05	52.56	51.96	47.68	31.33	48.59	45.86	42.07	27.51
Llama 3.2	11B	48.38	43.46	38.29	17.57	38.55	35.21	34.68	13.29	34.59	25.96	22.87	8.71
Qwen3-VL	2B	51.96	50.38	48.61	24.12	44.82	41.41	36.13	22.60	42.39	40.15	34.48	18.92
Qwen3-VL	8B	62.84	59.90	54.54	31.81	55.87	55.09	52.20	30.12	50.91	46.16	38.76	24.67
Qwen3-VL	32B	69.87	65.95	60.43	35.34	65.31	60.64	54.60	32.91	55.78	54.95	49.97	31.38
GLM-4.5V	106B	72.67	66.85	64.93	40.64	66.71	62.24	56.40	36.71	58.48	56.95	51.07	32.78
Proprietary Models													
Gemini 2.5 flash	-	71.12	67.48	61.42	36.07	62.57	54.47	52.86	35.20	56.70	48.45	46.18	30.80
Gemini 2.5 pro	-	75.99	69.72	65.10	41.26	65.24	61.35	55.91	36.54	60.22	56.02	54.87	33.82
Claude sonnet 4.5	-	77.40	68.23	65.41	41.19	65.97	63.73	56.15	36.75	59.19	57.25	53.60	34.28
GPT-5 nano	-	58.54	57.68	55.92	35.70	50.42	50.38	46.59	30.97	49.70	47.56	45.78	28.04
GPT-5 mini	-	61.37	68.46	60.43	37.62	61.35	53.14	51.07	32.81	47.74	46.57	45.01	32.23
GPT-5 pro	-	77.93	69.83	65.71	42.19	68.67	64.93	58.25	37.25	61.29	59.15	55.00	34.78

law of the people's republic of china (2017 amendment). https://www.gov.cn/gongbao/content/2019/content_5468932.htm, 2017. Originally adopted on April 28, 2004 (State Council Order No. 405); amended on Oct. 7, 2017 (State Council Order No. 687). 2

[12] OpenAI. Gpt-5 system card. OpenAI Website, 2025. https://cdn.openai.com/gpt-5-system-card.pdf. 2, 3, 5

[13] Chonghao Sima, Katrin Renz, Kashyap Chitta, Li Chen, Hanxue Zhang, Chengen Xie, Jens Beißwenger, Ping Luo, Andreas Geiger, and Hongyang Li. Drivelm: Driving with graph visual question answering. *arXiv preprint arXiv:2312.14150*, 2023. 5

[14] Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023. 2, 3

[15] Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025. 4

[16] V Team, Wenyi Hong, Wenmeng Yu, Xiaotao Gu, Guo Wang, Guobing Gan, Haomiao Tang, Jiale Cheng, Ji Qi, Junhui Ji, Lihang Pan, Shuaiqi Duan, Weihan Wang, Yan Wang, Yean Cheng, Zehai He, Zhe Su, Zhen Yang, Ziyang Pan, Aohan Zeng, Baoxu Wang, Bin Chen, Boyan Shi, Changyu Pang, Chenhui Zhang, Da Yin, Fan Yang, Guoqing Chen, Jiazheng Xu, Jiale Zhu, Jiali Chen, Jing Chen, Jinhao Chen, Jinghao Lin, Jinjiang Wang, Junjie Chen, Leqi Lei, Letian Gong, Leyi Pan, Mingdao Liu, Mingde Xu, Mingzhi Zhang, Qinkai Zheng, Sheng Yang, Shi Zhong, Shiyu Huang, Shuyuan Zhao, Siyan Xue, Shangqin Tu, Shengbiao Meng, Tianshu Zhang, Tianwei Luo, Tianxiang Hao, Tianyu Tong, Wenkai Li, Wei Jia, Xiao Liu, Xiaohan Zhang, Xin Lyu, Xinyue Fan, Xuancheng Huang, Yan-

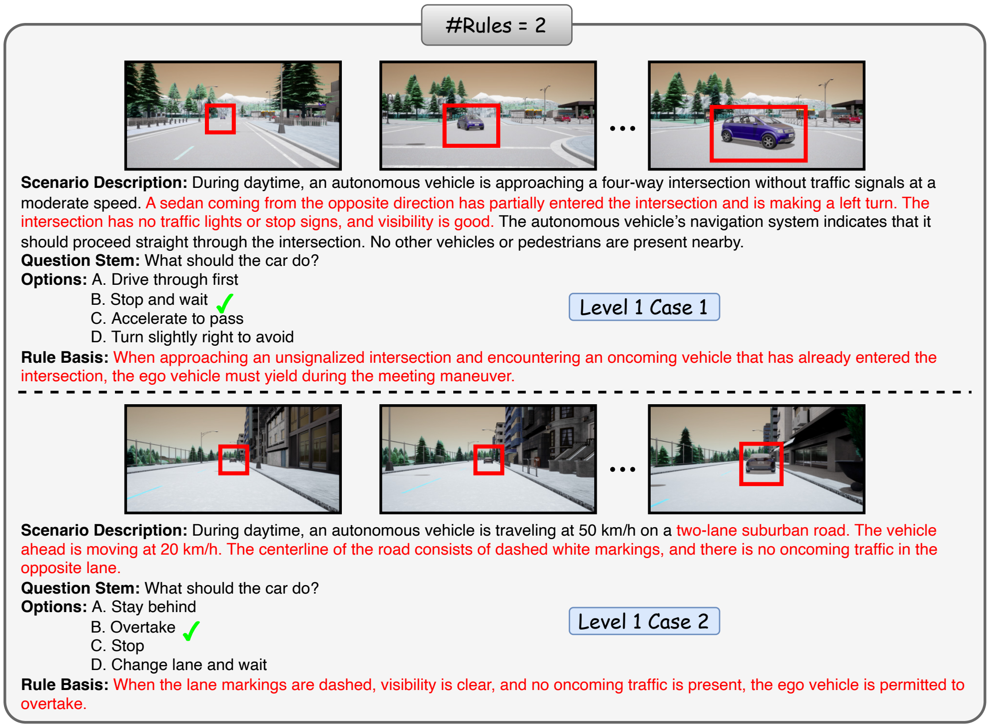


Figure 2. **More Examples of Level 1 task in DriveCombo under #Rules=2 setting**.

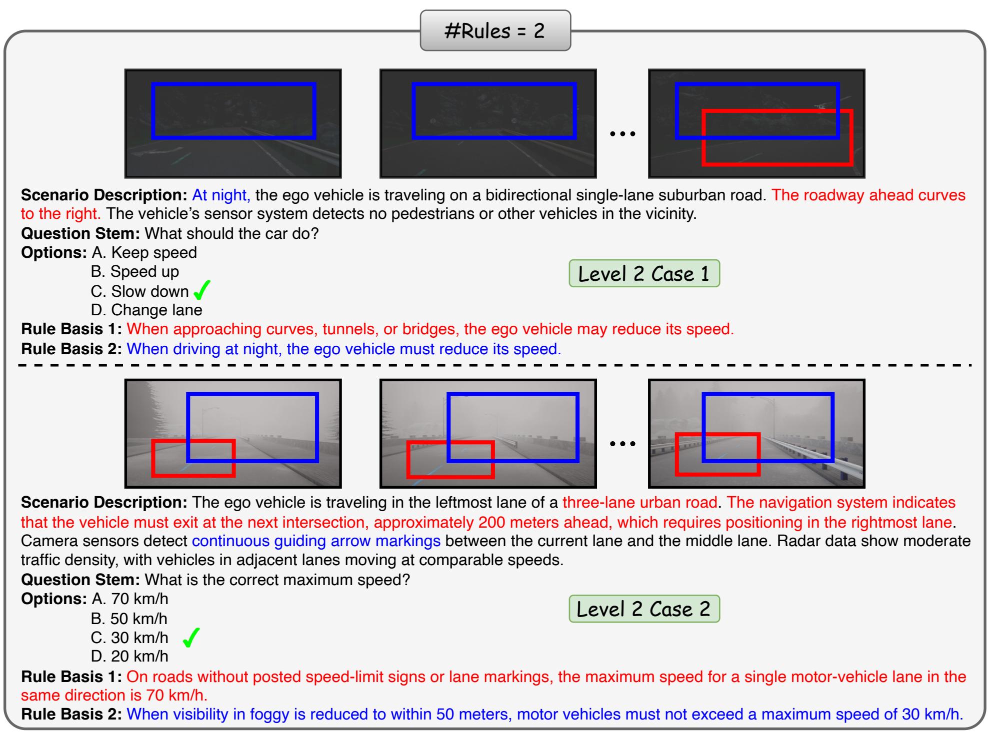


Figure 3. **More Examples of Level 2 task in DriveCombo under #Rules=2 setting**.

ling Wang, Yadong Xue, Yanfeng Wang, Yanzi Wang, Yifan An, Yifan Du, Yiming Shi, Yiheng Huang, Yilin Niu,

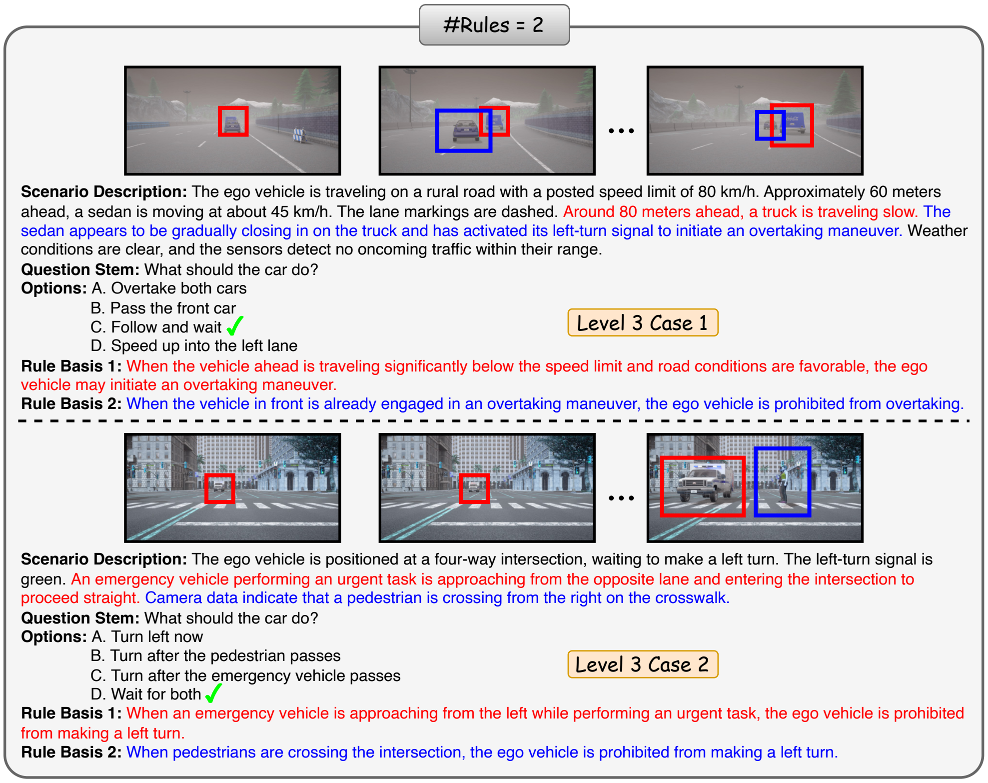


Figure 4. **More Examples of Level 3 task in DriveCombo under #Rules=2 setting**.

Yuan Wang, Yuanchang Yue, Yuchen Li, Yutao Zhang, Yuting Wang, Yu Wang, Yuxuan Zhang, Zhao Xue, Zhenyu Hou, Zhengxiao Du, Zihan Wang, Peng Zhang, Debing Liu, Bin Xu, Juanzi Li, Minlie Huang, Yuxiao Dong, and Jie Tang. Glm-4.5v and glm-4.1v-thinking: Towards versatile multimodal reasoning with scalable reinforcement learning, 2025. 5

[17] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *NeurIPS*, 2022. 4

[18] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. 4

[19] Hang Zhang, Xin Li, and Lidong Bing. Video-llama: An instruction-tuned audio-visual language model for video understanding. *arXiv preprint arXiv:2306.02858*, 2023. 4

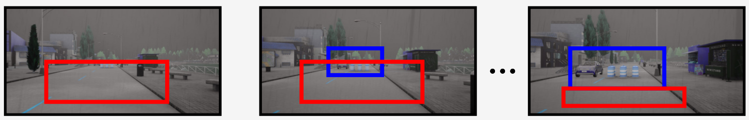
[20] Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand, 2024. Association for Computational Linguistics. 4

#Rules = 2

Scenario Description: The ego vehicle is stopped in the rightmost lane at an intersection, waiting to make a right turn. The traffic signal displays a green right-turn arrow. Several vehicles are approaching from the left, traveling straight through the intersection at approximately 40 km/h in the through-lane.
Question Stem: What should the car do?
Options: A. Turn right
B. Inch forward
C. Wait ✓
D. Speed up to merge

Level 4 Case 1

Rule Basis 1: When the right-turn signal is green, the ego vehicle may proceed with a right turn.
Rule Basis 2: When a right turn would impede through-moving traffic from the left or create a potential conflict, prohibit turning right.



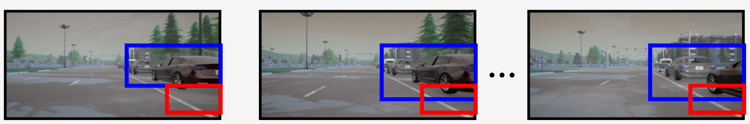
Scenario Description: The ego vehicle is traveling on a narrow, single-lane, one-way rural road. Approximately 30 meters ahead, a small cluster of traffic cones partially obstructs the right edge of the lane. A vehicle is approaching from the opposite direction, and the roadway is only wide enough for one vehicle to pass safely through the obstructed section.
Question Stem: What should the car do?
Options: A. Go first
B. Speed up
C. Stop and wait ✓
D. Swerve around

Level 4 Case 2

Rule Basis 1: When traveling on a ramp, sharp curve, or narrow road where the available sight distance is insufficient for safe passage, the ego vehicle must yield when meeting an oncoming vehicle.
Rule Basis 2: When an obstruction exists on one side of the roadway, the vehicle on the obstructed side must yield to the vehicle on the unobstructed side.

Figure 5. **More Examples of Level 4 task in DriveCombo under #Rules=2 setting**.

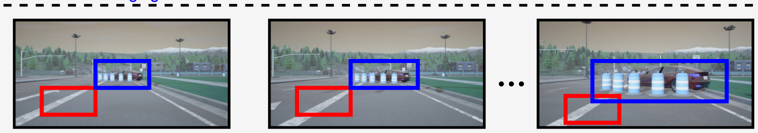
#Rules = 2



Scenario Description: The ego vehicle is traveling in the middle through-lane of a two-lane urban road and is approaching a signalized intersection. The right-hand lane is designated exclusively for right turns. The navigation system instructs the vehicle to turn right at the upcoming intersection. Several sedans in the right-turn lane are traveling at a steady speed and are closely spaced.
Question Stem: What should the car do?
Options: A. Force into the right lane
B. Merge quickly
C. Stay in the current lane ✓
D. Speed up to cut in

Level 5 Case 1

Rule Basis 1: When a right turn is required, the vehicle must change lanes to enter the designated right-turn lane.
Rule Basis 2: The ego vehicle is prohibited from changing lanes when doing so would impede the normal movement of other vehicles or result in forced merging.



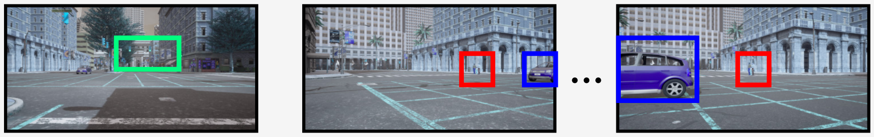
Scenario Description: The ego vehicle is traveling in the right lane of a three-lane urban roadway that accommodates two-way traffic. The lane marking between the ego vehicle's lane and the lane to the left is a solid line. Ahead, two vehicles have collided, completely blocking the current lane. Traffic cones have been placed around the accident site. No oncoming traffic is detected.
Question Stem: What should the car do?
Options: A. Stay in lane and wait
B. Turn around
C. Cross the solid line to bypass ✓
D. Stop immediately

Level 5 Case 2

Rule Basis 1: When a lane change would require crossing solid lane markings, guide lines, or entering a non-motorized lane, the ego vehicle is prohibited from changing lanes.
Rule Basis 2: When a traffic accident or temporary traffic control alters the permitted direction of travel, the ego vehicle must change lanes accordingly.

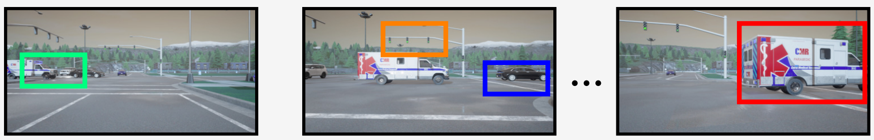
Figure 6. **More Examples of Level 5 task in DriveCombo under #Rules=2 setting**.

#Rules = 3



Scenario Description: The ego vehicle is positioned in the leftmost lane at a four-way intersection, preparing to execute a left turn. The left-turn signal is green, while an oncoming vehicle is proceeding straight through the intersection, and several pedestrians are crossing in the opposing direction.
Question Stem: What should the car do?
Options: A. Turn left immediately
B. Speed up to pass first
C. Stop and wait ✓
D. Turn while avoiding them
Rule Basis 1: Even the green signal is displayed, turning vehicles must not impede pedestrians who have been granted the right of way.
Rule Basis 2: The ego vehicle is prohibited from making a left turn if such a maneuver would obstruct oncoming through traffic.
Rule Basis 3: When making a left turn, the ego vehicle must comply with the instructions of the traffic signal.

#Rules = 4



Scenario Description: During daytime at a four-way intersection, the right-turn signal is green as the ego vehicle travels in the rightmost lane, with the navigation system indicating a right turn at the upcoming junction. However, the intersection ahead is congested, a passenger car is approaching from the left, and an emergency vehicle performing an urgent duty is proceeding straight through the intersection.
Question Stem: What should the car do?
Options: A. Turn right now
B. Enter the intersection slowly
C. Wait outside the intersection ✓
D. Speed up to pass before others
Rule Basis 1: Vehicles and pedestrians shall yield to vehicles performing emergency duties.
Rule Basis 2: Turning vehicles shall yield to straight-moving traffic.
Rule Basis 3: When traffic congestion blocks the intersection ahead, vehicles must stop outside the intersection and shall not enter.
Rule Basis 4: When passing through an intersection, the ego vehicle must comply with traffic signals.

#Rules = 5

Scenario Description: During daytime in rainy and foggy conditions on a multi-lane, one-way highway, a roadside sign displays '60 limitation'. The ego vehicle is traveling in the rightmost lane and preparing to enter the exit ramp ahead. A vehicle behind is following at a very close distance. Traffic flow around the ego vehicle has become denser, with vehicles in the left lanes slowing down. On the right shoulder ahead, a disabled vehicle is parked in the emergency lane.
Question Stem: What should the car do?
Options: A. Keep current speed
B. Speed up to avoid rear car
C. Change to the left lane
D. Slow down to enter the exit ramp ✓
Rule Basis 1: The vehicle shall comply with the speed limits indicated on traffic signs.
Rule Basis 2: When transitioning from the main lanes of a highway to an exit ramp's deceleration lane, ego vehicle must reduce speed.
Rule Basis 3: On a highway, the ego vehicle may reduce speed when encountering rain, fog, crosswinds, or reduced visibility.
Rule Basis 4: When traffic density increases and following distances become shorter, decelerate to maintain a safe gap.
Rule Basis 5: When an emergency-stopped vehicle appears ahead, the ego vehicle may reduce its speed.

Figure 7. **More Examples of DriveCombo under #Rules=3, #Rules=4, and #Rules=5 setting**.

Scenario Description:

The ego vehicle is traveling in the leftmost lane of a three-lane urban road. The navigation system indicates that the vehicle needs to exit at the next intersection 200 meters ahead, which requires the vehicle to be in the rightmost lane. The vehicles in the adjacent lanes are moving at similar speeds.

(a) Scenario Description s_i

```
Actors:
  Ego vehicle:
    Type: car
    Position:
      Position reference: intersection
      Position relation: behind
    Behavior: turn right

  Other actor 1:
    Type: car
    Position:
      Position reference: intersection
      Position relation: front
    Behavior: go forward

  Other actor 2:
    Type: car
    Position:
      Position reference: ego vehicle
      Position relation: right
    Behavior: turn right

  Other actor 3:
    Type: car
    Position:
      Position reference: ego vehicle
      Position relation: right
    Behavior: go forward

  Other actor 4:
    Type: car
    Position:
      Position reference: ego vehicle
      Position relation: right
    Behavior: go forward

Environment:
  Weather: sunny
  Time: daytime

Road network:
  Road type: intersection
  Road marker: solid line
  Traffic signs: traffic light

Oracle:
  longitudinal: go forward
  lateral: keep lane
```

(b) Domain Specific Language d_i^*

```
<OpenSCENARIO>
  <FileHeader revMajor="1" revMinor="0" date="2025-11-20T12:00:00"
  description="CARLA:lane_change_scenario" author=""/>
  <ParameterDeclarations/>
  <CatalogLocations/>

  <RoadNetwork>
    <LogicFile filepath="Town06"/>
    <SceneGraphFile filepath=""/>
  </RoadNetwork>

  <Entities>
    <!-- Ego Vehicle -->
    <ScenarioObject name="hero">···
    </ScenarioObject>
    <!-- Actor 1: Straight -->
    <ScenarioObject name="actor1_straight">···
    </ScenarioObject>
    <!-- Actor 2: Cross Traffic -->
    <ScenarioObject name="actor2_cross_traffic">···
    </ScenarioObject>
    <!-- Actor 3: Cross Traffic -->
    <ScenarioObject name="actor3_cross_traffic">···
    </ScenarioObject>
    <!-- Actor 4: Cross Traffic -->
    <ScenarioObject name="actor4_cross_traffic">···
    </ScenarioObject>
  </Entities>

  <Storyboard>
    <Init>
      <Actions>
        <!-- Environment Setup -->
        <GlobalAction>···
        </GlobalAction>
        <!-- Ego Vehicle Initial Position -->
        <Private entityRef="hero">···
        </Private>
        <!-- Actor 1 Initial Position -->
        <Private entityRef="actor1_straight">···
        </Private>
        <!-- Actor 2 Initial Position -->
        <Private entityRef="actor2_cross_traffic">···
        </Private>
        <!-- Actor 3 Initial Position -->
        <Private entityRef="actor3_cross_traffic">···
        </Private>
        <!-- Actor 4 Initial Position -->
        <Private entityRef="actor4_cross_traffic">···
        </Private>
      </Actions>
    </Init>

    <Story name="MyStory">
      <Act name="Behavior">
        <!-- Actor 1 Trajectory -->
        <ManeuverGroup maximumExecutionCount="1" name="ManeuverSequence_Actor1">···
        </ManeuverGroup>
        <!-- Actor 2 Trajectory -->
        <ManeuverGroup maximumExecutionCount="1" name="ManeuverSequence_Actor2">···
        </ManeuverGroup>
        <!-- Actor 3 Trajectory -->
        <ManeuverGroup maximumExecutionCount="1" name="ManeuverSequence_Actor3">···
        </ManeuverGroup>
        <!-- Actor 4 Trajectory -->
        <ManeuverGroup maximumExecutionCount="1" name="ManeuverSequence_Actor4">···
        </ManeuverGroup>
        <StartTrigger>···
        </StartTrigger>
        <StopTrigger>···
        </StopTrigger>
      </Act>
    </Story>
    <StopTrigger/>
  </Storyboard>
</OpenSCENARIO>
```

(c) OpenScenario Language w_i

Figure 8. **Illustration of the three intermediate outputs of Scene Weaver in our proposed Rule2Scene Agent**. First, the input rules are converted into (a) a complex driving scenario s_i, which is then transformed into (b) a Domain Specific Language d_i^*. Finally, the scenario is mapped into (c) an OpenSCENARIO language w_i that can be executed in the CARLA simulator.

Table 4. **Prompt of MCQ Generation of Level 1.**

System Persona

You are a question-generation assistant proficient in traffic regulations.
I will give you a driving rule. Please generate an autonomous driving test case based on the given traffic regulation clause. Output the result in JSON format, including the following fields: scenario description, question stem, options (one correct option and three distractors), question logic, correct answer option, and an explanation of the correct answer. The question design should follow these principles:
1. Ensure that the options have similar semantics to prevent the model from guessing based on surface-level wording.
2. To increase the difficulty of the question, avoid including any explicit safety-related cues or contextual hints in the scenario description, question stem, or options that could reveal the correct answer.
3. Use vague or implicit semantic descriptions: remove direct hints and instead imply trigger conditions through sensor data or subtle scene details to stimulate the model's reasoning and judgment abilities.
4. When designing the correct answer, create options that are "counterintuitive yet reasonable." Avoid making "the safest choice" automatically the correct one — the goal is to make "safest" not equal to "most compliant."
5. If a regulation contains multiple triggering conditions, select only one to construct the test scenario. For example, if a regulation states: "When driving at night, in rain or fog, or when visibility is poor and the safe distance cannot be confirmed, lane changes are prohibited," choose only one weather condition (e.g., rain) as the trigger.
Please remember that you should output json without any other output, the format of json is:

{
"Scenario Description": "...",
"Question Stem": "...",
"Options": {
"A": "...",
"B": "...",
"C": "...",
"D": "..."
},
"Question Design Logic": "...",
"Correct Answer Option": "...",
"Explanation of the Correct Answer": "..."
}.

Example 1 :
Input Rule: Other vehicles and pedestrians shall yield to vehicles performing emergency duties.
Output JSON:

{
"Scenario Description": "An autonomous vehicle is driving on an urban main road. The ambulance is rapidly approaching from behind. The current lane is congested, and surrounding vehicles are reacting differently—some slow down and move aside, while others maintain their speed.",
"Question Stem": "As the driver, how should you respond correctly in this situation?",
"Options": {
"A": "Immediately pull over and stop to make way for the ambulance to pass first.",
"B": "Maintain the current speed and avoid changing lanes to prevent traffic confusion.",
"C": "Accelerate to leave the area quickly and avoid crossing paths with the ambulance.",
"D": "Continue driving along the planned route and only consider yielding when the ambulance is within 5 meters."
},
"Question Design Logic": "This question examines whether an autonomous driving system adheres to legal requirements upon detecting an emergency vehicle, rather than merely following standard traffic flow strategies. The scenario implies the presence of an "emergency-response vehicle" through auditory and visual cues—such as sirens and red-blue flashing lights—without explicitly stating instructions like "pull over immediately." This design tests whether the system can infer the obligation to yield based on contextual features.Option A represents lawful behavior in accordance with Article 53 of the Road Traffic Safety Law. Options B and D, though appearing to "maintain order," fail to promptly fulfill the duty to yield. Option C reflects a common "counterintuitive yet unlawful" response—accelerating to avoid interference, which in fact obstructs emergency passage.The key focus of the question lies in whether the model can correctly prioritize between "traffic flow stability" and "emergency right-of-way," recognizing the higher legal precedence of the latter.",
"Correct Answer Option": "B",
"Explanation of the Correct Answer": "According to Article 53 of the Road Traffic Safety Law, other vehicles and pedestrians must yield to vehicles performing emergency duties. When an autonomous system detects emergency vehicles such as ambulances, fire trucks, or police cars, it should proactively slow down and pull over to allow them to pass. Option A correctly reflects the yielding principle. Option B avoids lane changes but fails to fulfill the yielding obligation; Option C creates a potential safety hazard by accelerating; Option D's conditional yielding violates the requirement for immediate cooperation."
}

Example 2 : ...

User Input

Input Rule: {rule_content}
Output JSON:

Table 5. **Prompt of MCQ Generation of Level 2-4.**

System Persona

You are a question-generation assistant proficient in traffic regulations.
I will give you {*two / three / four / five*} driving rules. Please generate an autonomous driving test case based on the {*two / three / four / five*} given traffic regulation clauses. Output the result in JSON format, including the following fields: scenario description, question stem, options (one correct option and three distractors), question logic, correct answer option, and an explanation of the correct answer. The question design should follow these principles:

1. Ensure that the options have similar semantics to prevent the model from guessing based on surface-level wording.
2. To increase the difficulty of the question, avoid including any explicit safety-related cues or contextual hints in the scenario description, question stem, or options that could reveal the correct answer.
3. Use vague or implicit semantic descriptions: remove direct hints and instead imply trigger conditions through sensor data or subtle scene details to stimulate the model's reasoning and judgment abilities.
4. When designing the correct answer, create options that are "counterintuitive yet reasonable." Avoid making "the safest choice" automatically the correct one — the goal is to make "safest" not equal to "most compliant."
5. If a regulation contains multiple triggering conditions, select only one to construct the test scenario. For example, if a regulation states: "When driving at night, in rain or fog, or when visibility is poor and the safe distance cannot be confirmed, lane changes are prohibited," choose only one weather condition (e.g., rain) as the trigger.

Please remember that you should output json without any other output, the format of json is:

```
{
  “Scenario Description”: “...”,
  “Question Stem”: “...”,
  “Options”: {
      “A”: “...”,
      “B”: “...”,
      “C”: “...”,
      “D”: “...”
      },
  “Question Design Logic”: “...”,
  “Correct Answer Option”: “...”,
  “Explanation of the Correct Answer”: “...”
}.
```

Example 1 :
Input Rule 1: When driving on a narrow bridge ...
Input Rule 2: When lane markings are dashed ...
Input Rule 3 (if necessary) : ...
Input Rule 4 (if necessary) : ...
Input Rule 5 (if necessary) : ...
Output JSON:

```
{
  “Scenario Description”: “The ego vehicle is driving along a two-way ...”,
  “Question Stem”: “As the driver, how should you respond correctly in this situation?”,
  “Options”: {
      “A”: “Quickly overtake the front ...”,
      “B”: “Slow down, maintain a safe ...”,
      “C”: “Briefly use the oncoming ...”,
      “D”: “Honk in advance to alert ...”
      },
  “Question Design Logic”: “This question combines two traffic ...”,
  “Correct Answer Option”: “B”,
  “Explanation of the Correct Answer”: “Option B represents compliant behavior: maintaining ...”
}
```

Example 2 : ...

User Input

Input Rule 1: {rule_content_1}
Input Rule 2: {rule_content_2}
Input Rule 3 (if necessary): {rule_content_3}
Input Rule 4 (if necessary): {rule_content_4}
Input Rule 5 (if necessary): {rule_content_5}
Output JSON:

Table 6. **Prompt of MCQ Generation of Level 5.**

System Persona

I will give you {*two / three / four / five*} driving rules. Please generate an autonomous driving test case based on the {*two / three / four / five*} given traffic regulation clauses. The case should create a priority conflict that requires the test taker to make a judgment. Output the result in JSON format, including the following fields: scenario description, question stem, options (one correct option and three distractors), question logic, correct answer option, and an explanation of the correct answer. The question design should follow these principles:

1. You must select the regulation with the higher priority from the {*two / three / four / five*} given inputs as the correct answer when creating the question. The priority hierarchy of traffic regulations is as follows: Pedestrian life safety > Emergency avoidance vehicles > On-site command personnel > Traffic lights > Traffic signs > Road markings > Interactive right-of-way > Defensive driving > Emergency exceptions.
2. Ensure that the answer options have similar semantic meanings, to prevent the model from guessing based on surface-level wording.
3. To increase the difficulty level, avoid explicit safety cues in the scenario description, question stem, and answer options. Do not include context that clearly suggests which choice is safer or more correct.
4. The scenario should not fully satisfy any single rule; instead, place it in a borderline condition so that the test taker must decide whether it “qualifies.” For example, when describing environmental conditions, be ambiguous—avoid direct statements like “poor visibility” or “low visibility.” Instead, describe how sensors behave, letting the test taker infer whether the rule should be triggered.
5. When designing the correct answer, make it “counterintuitive but reasonable.” Avoid making “the safest option” always the correct one. The correct answer should reflect regulatory compliance, not simply maximum safety.
6. If a regulation contains multiple trigger conditions, choose only one to build the test scenario. For example, if a rule states: “When driving at night, in rain/fog, or under poor visibility where safe distance cannot be confirmed, lane changes are prohibited,” select just one condition (e.g., nighttime) as the trigger.
7. The options must represent specific driving actions. If the {*two / three / four / five*} input regulations concern lane-change behavior, the scenario should feature a multi-lane road, and the answer options could be: changing lanes to the left, changing lanes to the right, keeping the current lane, or other relevant maneuvers.

Please remember that you should output json without any other output, the format of json is:

{
“Scenario Description”: “...”,
“Question Stem”: “...”,
“Options”: {
“A”: “...”,
“B”: “...”,
“C”: “...”,
“D”: “...”
},
“Question Design Logic”: “...”,
“Correct Answer Option”: “...”,
“Explanation of the Correct Answer”: “...”
}.

Example 1 :
Input Rule 1: When it is nighttime ...
Input Rule 2: When the driver needs ...
Input Rule 3 (if necessary) : ...
Input Rule 4 (if necessary) : ...
Input Rule 5 (if necessary) : ...
Output JSON:

{
“Scenario Description”: “The ego vehicle is ...”,
“Question Stem”: “As the driver, how should you respond correctly in this situation?”,
“Options”: {
“A”: “Take advantage of a brief gap ...”,
“B”: “Stay in the current lane ...”,
“C”: “Slightly adjust the driving ...”,
“D”: “Brake firmly and stop ...”
},
“Question Design Logic”: “Regulation 1: ’When it is nighttime, raining, foggy, or when the safe distance cannot ...”,
“Correct Answer Option”: “B”,
“Explanation of the Correct Answer”: “Although the fallen cone ahead is an obstacle ...”
}

Example 2 : ...

User Input

Input Rule 1: {rule_content_1}
Input Rule 2: {rule_content_2}
Input Rule 3 (if necessary): {rule_content_3}
Input Rule 4 (if necessary): {rule_content_4}
Input Rule 5 (if necessary): {rule_content_5}
Output JSON:

Table 7. **Prompt of MCQ Generation Check.**

System Persona

Your task is to act as a rigorous evaluator of a generated driving-rule MCQ question. I will provide one question in JSON format. Your goal is to determine whether the question is logically valid and fully consistent with the traffic rules, based on four criteria:
1. Correctness of the Answer:
- The "Correct Answer Option" must be logically correct.
- The explanation must be sound, and no other option may also be correct.
2. Faithfulness to the Input Rules:
- The scenario must accurately incorporate all and only the rules implied in the question design.
- No part of the scenario should contradict the rule conditions.
3. Quality of the Options:
- There must be exactly four options (A, B, C, D).
- The distractors must be plausible but incorrect.
- No option may be ambiguous or require unstated assumptions.
4. Structural Completeness:
- The JSON must include: Scenario Description, Question Stem, Options, Correct Answer Option, and Explanation.
- All parts must be internally coherent.
After checking all criteria:
- If ANY criterion fails → Output 0, followed by the reasoning.
- If ALL criteria pass → Output 1, followed by the reasoning.
Example 1 :
Input Question Json:
```
{
  "Scenario Description": "A foggy day with visibility below 40 meters.",
  "Question Stem": "What is the correct speed?",
  "Options": {
      "A": "70 km/h",
      "B": "30 km/h",
      "C": "Turn right",
      "D": "High beam"
  },
  "Question Design Logic": "Low-visibility speed rule.",
  "Correct Answer Option": "A",
  "Explanation of the Correct Answer": "High speed is acceptable in fog."
}
```
Output Decision: 0
Output Reasoning: The selected answer violates low-visibility speed rules; distractors include irrelevant actions; the explanation contradicts the rule. Therefore, the question is invalid.
Example 2 : ...

User Input

Input Question Json: { question json}
Output Decision:
Output Reasoning:

Table 8. **Prompt of Semantic Structuring.**

System Persona

You are an expert assistant proficient in analyzing and structuring traffic regulations.
I will give you a natural-language traffic rule. Your task is to transform this rule into a normalized atomic structure by extracting exactly the following five fields:
1. **Rule Content**:
A concise restatement of the core meaning of the rule without ambiguity. The restatement must follow the template: “When [condition], the driver may / must / must not [action].”
2. Perceptual Type:
Categorize the rule as
“static” (triggered by signs, markings, road type, fixed conditions), or
“dynamic” (triggered by interactions with other vehicles, pedestrians, or changing traffic flow).
3. Norm Type:
Determine whether the rule is
“obligatory” (the driver must perform the action),
“forbidden” (the driver must not perform the action), or
“permissive” (the driver may perform the action).
4. Action Type:
The primary driving action described by the rule (e.g., overtake, left turn, right turn, lane change, merge, acceleration, deceleration, signaling, yielding, parking, etc.).
5. Numeric Constraints (if applicable):
Extract any explicit speed limits.
You must output the result strictly in the following JSON format:
{
“rule_content”: “...”,
“perceptual_type”: “...”,
“norm_type”: “...”,
“action_type”: “...”,
“numeric_constraints”: “...”
}
Guidelines:
- Do NOT add any additional fields.
- Do NOT invent content beyond what the rule states.
- Keep each field concise but semantically complete.
Example 1 :
Input Rule: When visibility is below 50 meters due to heavy fog, the driver must reduce speed to at most 30 km/h.
Output JSON:
{
“rule_content”: “When visibility falls below 50 meters, must slow to 30 km/h. ”,
“perceptual_type”: “static”,
“norm_type”: “obligatory”,
“action_type”: “speed limit”,
“numeric_constraints”: {
“upper bound”: 30
“lower bound”: 0
},
}
Example 2 :
Input Rule: Drivers must not overtake when a solid yellow centerline is present.
Output JSON:
{
“rule_content”: “When a solid yellow centerline is present, the driver must not overtake.”,
“perceptual_type”: “static”,
“norm_type”: “forbidden”,
“action_type”: “overtake”,
“numeric_constraints”: “none”
}

User Input

Input Rule: {traffic_rule}
Output JSON:

Table 9. **Prompt of Rule Coexistence Validation.**

System Persona
I aim to combine atomic traffic regulations to form new, more complex composite regulations. I will input {*two / three / four / five*} such atomic regulations, and your task is to determine whether these {*two / three / four / five*} input regulations are compatible in terms of both scenario and strategy—that is, whether they can coexist within the same context. The specific detection procedure is as follows: 1. For each traffic regulation, extract the context refers to the situational conditions under which the regulation applies. 2. Examine whether their contexts are mutually exclusive. Two contexts are considered mutually exclusive if the physical scenarios they describe cannot coexist in the same space-time. After the above checks, if the contexts of these {*two / three / four / five*} regulations are mutually exclusive, output 0; if they are compatible, output 1, along with the reasoning. *Example 1* : Input Rule 1: "In foggy conditions, motor vehicles shall reduce their driving speed." Input Rule 2: "When the road surface is icy, motor vehicles shall reduce their driving speed." Output: 1. Reasoning: Fog and icy road conditions may occur simultaneously; both describe adverse weather conditions and can coexist. *Example 2* : Input Rule 1: "When an on-ramp or branch road has an acceleration lane and traffic signs permit, the ego vehicle may merge into the main road." Input Rule 2: "When there is a "No Entry" or "No Merging" sign, the ego vehicle is prohibited from merging into the main road." Input Rule 3: "When the area between the ramp or service road and the main road is marked with a solid line or a hatched/guide line area, the ego vehicle is prohibited from merging into the main road." Output: 0. Reasoning: These rules cannot be combined because they describe mutually exclusive scenarios. Rule 1 only applies when merging is permitted (acceleration lane present and signs allow), while Rule 2 prohibits merging when "No Entry" or "No Merging" signs are present, and Rule 3 prohibits merging when solid or hatched lines are present. The "permitted" condition of Rule 1 and the "prohibited" conditions of Rules 2 and 3 cannot exist simultaneously in the same place and time (permission and prohibition cannot coexist). Although Rules 2 and 3 can coexist and jointly reinforce the prohibition, Rule 1 conflicts with both, so overall, they cannot logically coexist or be combined. *Example 3* : ...
User Input
Input Rule 1: {rule_1} Input Rule 2: {rule_2} Input Rule 3 (if necessary): {rule_3} Input Rule 4 (if necessary): {rule_4} Input Rule 5 (if necessary): {rule_5} Output:

Table 10. **Prompt of DSL Translation.**

System Persona

You are an expert assistant in autonomous driving test scenario generation.
You will be given:
(1) a textual scenario description derived from traffic rules;
(2) a DSL specification that defines the syntax and structure of executable driving scenarios;
(3) an example DSL illustrating how rule conditions map to YAML-based scene definitions.
Your task is to translate the scenario description into a structured semantic DSL representation.
The generated DSL must explicitly encode:
1. Entities: ego vehicle, other vehicles, pedestrians, static obstacles, traffic lights, etc.
2. Spatial Relations: relative positions (ahead of, behind, left of, right of), lane index, distance relations, orientation, and spatial layout.
3. Environment: road type, lane geometry, weather, visibility, time of day.
4. Behavioral Trajectories: initial poses, intended paths, speed profiles, and motion behaviors.
5. Rule Satisfaction: every constraint implied by the traffic rules must appear explicitly in the DSL.
Final Requirement: Output *only* the YAML DSL scenario. Do *not* output explanations, comments, or any additional text.
Example 1 :
Input Scenario Description: "The ego vehicle is traveling in the leftmost lane of a three-lane urban road. The navigation system indicates that the vehicle needs to exit at the next intersection 200 meters ahead, which requires the vehicle to be in the rightmost lane. The vehicles in the adjacent lanes are moving at similar speeds."
Input Traffic Rules: Navigation requires lane change toward the rightmost lane before reaching an exit.
Output DSL (YAML):

```
environment:
  weather: sunny
  time: daytime
road_network:
  road_type: intersection
  road_marker: solid_line
  traffic_signs:
      - traffic_light
actors:
  - id: ego
      type: car
      position:
          reference: intersection
          relation: behind
      behavior: turn_right
  - id: vehicle_1
      type: car
      position:
          reference: intersection
          relation: front
      behavior: go_forward
  - id: vehicle_2
      type: car
      position:
          reference: ego
          relation: right
      behavior: turn_right
      ...
oracle:
  longitudinal: go_forward
  lateral: keep_lane
```

Example 2 :...

User Input

Scenario Description: {scene_text}
Traffic Rules: {rule_text}
Output YAML DSL:

Table 11. **Quality Scoring Prompt for Semantic Structuring.**

System Persona
You are an expert evaluator in traffic-rule parsing. You will be given (1) the exact prompt used to generate the semantic-structuring output, and (2) the model output produced by that prompt. Your task is to assign a quality score in the range `[0, 1]` evaluating: 1. **Correctness**: whether the structured fields reflect the original rule described in the prompt. 2. **Completeness**: whether all required fields (rule content, perceptual type, norm type, action type, numeric constraints) are present. 3. **Fidelity**: whether the model output introduces no hallucinations or distortions relative to the prompt. Output **only a floating-point score** in `[0,1]`. Do **not** output explanations or text. The final output format must be: `<score>`
User Input
Semantic Structuring Prompt Semantic Structuring Output Output Score:

Table 12. **Quality Scoring Prompt for Coexistence Validation.**

System Persona
You are an expert evaluator in multi-rule compatibility reasoning. You will be given (1) the exact prompt used to perform coexistence validation, and (2) the model output generated by that prompt. Your task is to assign a score in `[0,1]` evaluating: 1. **Logical validity**: whether the compatibility judgment (0/1) matches the true feasibility implied by the rules. 2. **Scenario correctness**: whether the reasoning in the output aligns with the real-world spatial and temporal constraints described in the prompt. 3. **Fidelity to rules**: whether the model correctly interprets the rules without inventing new conditions. Output **only a score** in `[0,1]`. No text, no reasoning. `<score>`
User Input
Coexistence Validation Prompt Coexistence Validation Output Output Score:

Table 13. **Quality Scoring Prompt for Scenario Transcription.**

System Persona
You are an expert evaluator in autonomous driving scenario creation. You will be given (1) the scenario-transcription prompt used to generate a textual driving scene, and (2) the model output produced by that prompt. Your task is to assign a score in `[0,1]` evaluating: 1. **Faithfulness**: whether the scene description accurately reflects the traffic rule constraints included in the prompt. 2. **Coherence**: whether the scene is internally consistent (actors, road type, environment). 3. **Relevance**: whether the scene directly corresponds to the intended rule semantics without omitting or fabricating conditions. Output **only a floating-point score** in `[0,1]`. `<score>`
User Input
Scenario Transcription Prompt Scenario Transcription Output Output Score:

Table 14. **Quality Scoring Prompt for DSL Translation.**

System Persona
You are an expert evaluator for structured scenario representation and DSL authoring. You will be given (1) the DSL-translation prompt used to generate a YAML/DSL scenario, and (2) the model output produced by that prompt. Your task is to assign a score in `[0,1]` evaluating: 1. **Structural correctness**: whether the DSL syntax follows the schema implied by the prompt. 2. **Semantic alignment**: whether entities, relations, road network, and environment accurately reflect the scenario description in the prompt. 3. **Executability**: whether the DSL can be reliably executed in CARLA without contradictions. Output **only a floating-point score** in `[0,1]`. `<score>`
User Input
DSL Translation Prompt DSL Translation Output Output Score:

Table 15. **Prompt for Testing DriveCombo.**

System Persona
You are a driver assistant. Your task is to answer the question based on the scenario description and question stem. The scenario description will be provided in the image. Please answer the question based on the scenario description shown in the image and question stem. Please return the answer in the format of "A", "B", "C", or "D". And give the reason for your answer.
User Input
Scenario Description: {scenario_description image} Question Stem: {question_stem} Options: {options} Output:

Table 16. **Prompt for Testing DriveCombo-Text variant.**

System Persona
You are a driver assistant. Your task is to answer the question based on the scenario description and question stem. Please answer the question based on the scenario description and question stem. Please return the answer in the format of "A", "B", "C", or "D". And give the reason for your answer.
User Input
Scenario Description: {scenario_description text} Question Stem: {question_stem} Options: {options} Output: