ChatScene: Knowledge-Enabled Safety-Critical Scenario Generation for Autonomous Vehicles

Jiawei Zhang
UIUC
jiaweiz7@illinois.edu

Chejian Xu
UIUC
chejian2@illinois.edu

Bo Li
UChicago & UIUC
bol@uchicago.edu

Abstract

We present ChatScene, a Large Language Model (LLM)-based agent that leverages the capabilities of LLMs to generate safety-critical scenarios for autonomous vehicles. Given unstructured language instructions, the agent first generates textually described traffic scenarios using LLMs. These scenario descriptions are subsequently broken down into several sub-descriptions for specified details such as behaviors and locations of vehicles. The agent then distinctively transforms the textually described sub-scenarios into domain-specific languages, which then generate actual code for prediction and control in simulators, facilitating the creation of diverse and complex scenarios within the CARLA simulation environment. A key part of our agent is a comprehensive knowledge retrieval component, which efficiently translates specific textual descriptions into corresponding domain-specific code snippets by training a knowledge database containing the scenario description and code pairs. Extensive experimental results underscore the efficacy of ChatScene in improving the safety of autonomous vehicles. For instance, the scenarios generated by ChatScene show a 15% increase in collision rates compared to state-of-the-art baselines when tested against different reinforcement learning-based ego vehicles. Furthermore, we show that by using our generated safety-critical scenarios to fine-tune different RL-based autonomous driving models, they can achieve a 9% reduction in collision rates, surpassing current SOTA methods. ChatScene effectively bridges the gap between textual descriptions of traffic scenarios and practical CARLA simulations, providing a unified way to conveniently generate safety-critical scenarios for safety testing and improvement for AVs. The code is available at https://github.com/javyduck/ChatScene.

1. Introduction

Although machine learning (ML), particularly deep neural networks (DNNs), has shown remarkable performance across a myriad of applications such as image recognition [24], natural language processing [12], and healthcare [16], they also exhibit a surprising susceptibility to subtle and adversarial perturbations. These perturbations can yield erroneous predictions [3, 39], posing potentially fatal consequences in safety-critical applications like Autonomous Driving (AD) [6, 28]. For example, by attaching seemingly innocuous stickers to a *Stop Sign* in the real world, an autonomous vehicle (AV) can readily misinterpret it as a *Speed Limit 80 Sign* [17], which can lead to some hazardous driving behaviors and potential accidents.

Therefore, given the potential for such adversarial manipulations, it is crucial that AVs undergo exhaustive testing across all conceivable safety-critical scenarios to ensure their safe and reliable operation before large-scale deployment. However, traditional real-world testing is not only prohibitively expensive but also demands extensive data collection, often requiring vehicles to be driven hundreds of millions of miles to accumulate sufficient safety-critical scenarios. Consequently, the generation of simulated scenarios for testing has been increasingly adopted as a cost-effective and efficient alternative.

For instance, Wachi *et al*. [43] employ multi-agent reinforcement learning to train adversarial vehicles, aiming to expose the vulnerabilities in rule-based driving algorithms within the CARLA platform [15]. Chen *et al*. [8] instead focus on generating adversarial scenarios for lane-changing maneuvers using ensemble deep reinforcement learning techniques, and Feng *et al*. [18] further offer a highway-driving simulation that incorporates further scenarios such as *Cut Following* and *Cut-in*. Nevertheless, a key challenge remains: these methods are limited to only a narrow range of safety-critical scenarios, which may still fall short of encompassing the complexity of real-world situations.

Meanwhile, the emergence of LLMs, trained with vast amounts of data from the internet and encompassing billions of parameters, has demonstrated a remarkable aptitude for capturing human knowledge [4, 9, 41, 52]. These models have established themselves as effective tools for the extraction of knowledge. For instance, LLMs begin to play a pivotal role in pedagogical processes [26], and they are becoming instrumental in synthesizing clinical knowledge to

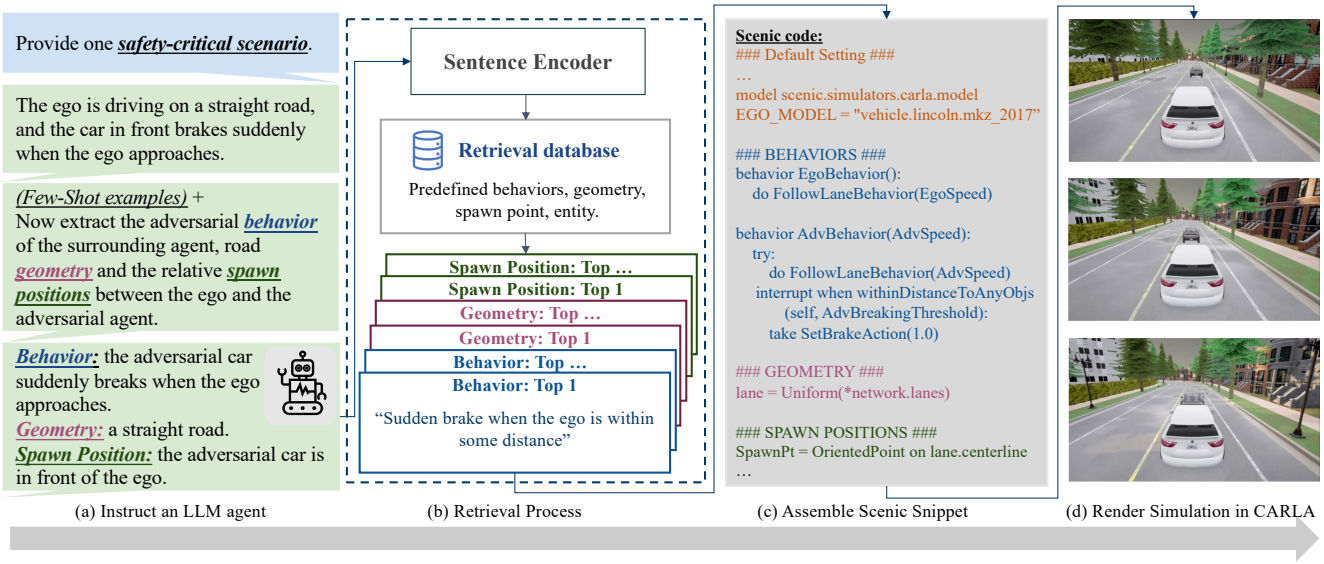

Figure 1. Overview of our LLM-based knowledge-enabled safety-critical scenario generation agent ChatScene.

support medical practice [38]. The legal field also benefits from LLMs, with tools like Chatlaw [11] interpreting legal regulations and judicial decisions, while in the financial sector, models such as BloombergGPT [45] are being harnessed to decode complex economic data.

This naturally leads to several compelling questions: *Is it possible to build an LLM-based agent that automatically generates safety-critical scenarios, capturing a broader and more intricate array of descriptions? Moreover, can the agent automatically convert these textual descriptions into real simulations to bolster the diversity and comprehensiveness of scenarios available for AV testing?*

Addressing the initial query of generating safety-critical driving scenario descriptions vian LLM agent is a process that is relatively straightforward; one can simply prompt the model with requests such as, *"Provide some descriptions for safety-critical driving scenarios."* While for the second question, the recent advancements of *Scenic* [19, 20], a domain-specific probabilistic programming language, allow for the scripting of scenes within CARLA [15] using syntax akin to Python, opens up two promising research directions: first, the possibility of guiding LLMs to autonomously script in Scenic, and second, the potential for finetuning a language-to-code model, such as *CodeGen* [32], to craft Scenic code derived from textual scenario descriptions. Nonetheless, these methods often encounter obstacles, such as the generation of non-executable code or calls to APIs that do not exist within Scenic, primarily due to the scarcity of available code examples for training.

To address the challenges of direct code generation by large language models (LLMs), we instead adopt an indirect approach that leverages LLMs to first curate a retrieval database comprising Scenic code snippets. These snippets encapsulate fundamental elements of driving scenarios, such as adversarial behaviors of surrounding vehicles, road geometries, etc. The details of the construction process will be introduced in Section 3.2. Then, during the evaluation phase, as illustrated in Figure 1, our agent ChatScene maps the description into the corresponding simulation through a four-step process:

i. Given instruction from the user, ChatScene generates a natural language description of a safety-critical scenario leveraging the intrinsic wide knowledge in LLMs.
ii. ChatScene further parses this description, extracting detailed characteristics that align with critical scenario components, such as the adversarial behaviors of surrounding vehicles.
iii. ChatScene then encodes these characteristics into embeddings to retrieve the corresponding Scenic code snippets from our pre-constructed database.
iv. Finally, the retrieved snippets are assembled into a complete and executable Scenic script, which is capable of enacting the described scenario within the CARLA simulation environment.

For a further quantitative analysis, we utilize ChatScene to produce a range of text descriptions for safety-critical scenarios. These narratives are then processed by our framework to generate simulations, which are then evaluated using the Safebench platform [46]. Within this platform, the ego vehicle is controlled by a model trained under reinforcement learning, while our scenario generation agent aims to manage the adversarial objects (e.g., pedestrian, cyclist, or vehicle) surrounding it. Some examples of text-to-simulation mappings are provided in Figure 2, illustrating the practical application of our framework.

Our contributions can be concluded as:

- We introduce ChatScene, a novel LLM-based agent capable of generating safety-critical scenarios by first providing textual descriptions and then carefully transforming them into executable simulations in CARLA

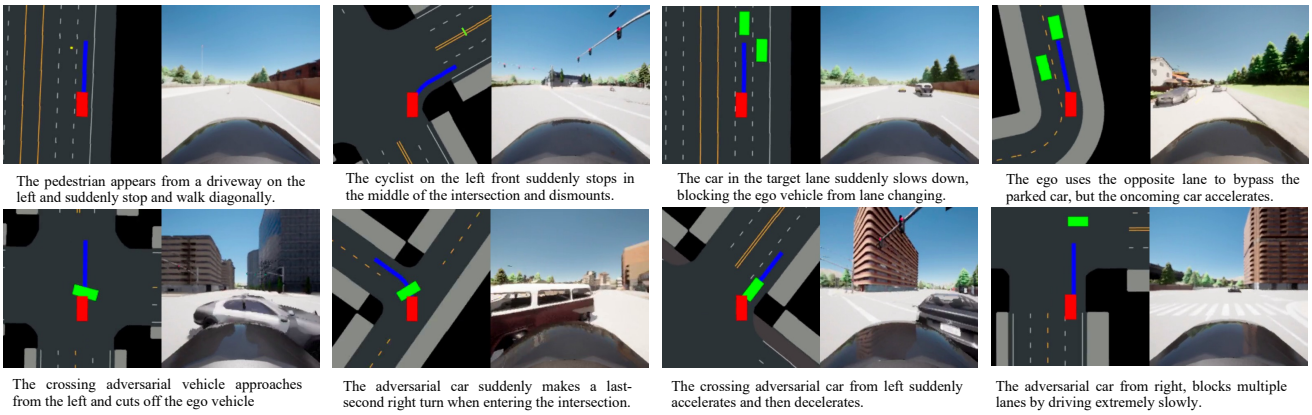

Figure 2. Eight selected text-simulation cases using our agent ChatScene, the descriptions here are shortened for clarity.

via Scenic programming language.

- An expansive retrieval database of Scenic code snippets has been developed. It catalogs diverse adversarial behaviors and traffic configurations, utilizing the rich knowledge stored in LLMs, which significantly augments the variety and critical nature of the driving scenarios generated.
- In Safebench's evaluation of eight CARLA Challenge traffic scenarios [7], our method's adversarial scenes increased the collision rate by 15% compared to four state-of-the-art (SOTA) baselines, demonstrating the superior safety-critical capabilities of our framework.
- Subsequent experiments involving the finetuning of the ego vehicle with a subset of our generated adversarial scenarios, followed by comparative evaluations against both the remaining scenarios we created and those from established baselines, demonstrated an additional reduction in average collision rates by at least 9%.
- Our framework, in conjunction with the retrieval database, not only facilitates direct code generation but also holds potential for future adaptations in multimodal conversions, including text, image, and video, specifically for autonomous driving applications.

2. Related Work

Generation of safety-critical scenarios. The generation of safety-critical scenarios for autonomous vehicles (AVs) generally falls into three main categories. The first is *data-driven generation* [14, 36, 40, 42, 48], which relies on real-world data to guide vehicle behavior. While realistic, this approach usually suffers from the scarcity and high cost of gathering pertinent data and the infrequency of genuinely risky scenarios within the collected dataset since the safety-critical scenarios usually lie on the long-tail distribution of the real-world scenario distribution. The second category, *adversarial generation* [1, 29, 33, 34, 50], intentionally conducts malicious attacks against the AVs by manipulating the behaviors of surrounding vehicles, such as pedestrians or other vehicles. While effective at creating challenging environments, this method may be computationally inefficient and may lack diversity in the generated scenarios. The third approach, *knowledge-based generation* [2, 5, 27, 35], uses predefined traffic rules or physical constraints to create scenarios. Although it is more systematic and can provide more diverse scenarios, this approach can be challenging to implement, as encoding these rules into simulations can be complex. Besides, the manually created rules are hard to cover all safety-critical situations, resulting in less risky scenarios since they typically do not incorporate adversarial attacks or unexpected behaviors that are crucial for testing the robustness of AVs.

Our work, instead, synergies the advantages of the latter two categories by integrating diverse real-world knowledge rules sourced from LLMs while using Scenic to adversarially optimize parameters of the surrounding environment, such as the speed of nearby pedestrians and vehicles, to enhance the risk and complexity of the generated scenarios.

LLMs for autonomous driving. LLMs have increasingly been explored for their potential in autonomous driving, particularly in their ability to interpret complex scenarios in a way that resembles human understanding. For instance, Fu *et al*. [21] utilize GPT-3.5 in a reasoning and action prompt style [49] to generate API-wrapped text descriptions of decisions made in response to observations in a highway environment. On the other hand, Xu *et al*. [47] propose to leverage a multimodal language model to interpret driving scenarios and provide corresponding descriptions with the prediction for the next control signals based on the driving video frames and human questions. Additionally, Zhong *et al*. [53] introduces the use of the LLM to transform a user's query about safety-critical scenarios into the corresponding differentiable loss function of a diffusion model to generate the query-compliant trajectories.

Our work differs from the first two studies in that we focus primarily on the generation of safety-critical scenarios rather than using LLMs to provide descriptions or actions for specific driving situations. Besides, in contrast to the last work, our work mainly aims to create more realistic safety-critical scenario generations on a platform like CARLA and

use the corresponding driving record to train or test the ego vehicle controlled trained with reinforcement learning.

3. Methodology

In this section, we delineate the approach of ChatScene, an LLM agent, for the generation of safety-critical scenarios through the application of Scenic programming language and a targeted retrieval mechanism. We begin with a concise motivation, including definitions of key terminologies. Then, we focus on the construction of the retrieval database, and following this, we explain the process of converting text descriptions of safety-critical scenarios, sourced from either humans or LLMs, into Scenic code. Specifically, we will begin by breaking down the original text description into sub-descriptions for each component (e.g., adversarial behavior). Next, we encode these components into vectors, which will serve as keys to retrieve the corresponding snippets from our retrieval database. These snippets will then be assembled to form a comprehensive Scenic script, which is then executed to run simulations within the CARLA environment.

3.1. Motivation and Notations

We notice that the Scenic [19, 20] programming language is highly effective and flexible for rendering simulations in CARLA. A pertinent question naturally arises: *"Can we directly prompt large language models like ChatGPT to write corresponding Scenic code based on descriptions of safety-critical scenarios?"* Although this approach seems promising, it often leads to issues like generating non-compilable code or using APIs not present in the codebase. This might be due to the scarcity of Scenic examples for training the LLMs and the complexity and breadth of code generation, which can induce hallucinations in the model.

However, upon examining Scenic code, we find that it can generally be segmented into four components as shown in Figure 3: (1) default map and model settings, typically fixed; (2) definition of *adversarial behavior* for surrounding vehicles; (3) the *road geometry*, which also influences the ego vehicle's spawn point; and (4) the *relative spawn position* of surrounding vehicles to the ego vehicle. Notably, the ego vehicle is controlled by models trained via reinforcement learning, so we don't need to define its behavior here.

Based on this, we propose a more efficient method: collecting a database of code snippets for the last three components with corresponding descriptions instead. This approach would likely reduce hallucination and allow for flexible assembly of these snippets into a complete Scenic code. Then, during evaluation, for input descriptions like *"The ego vehicle is driving on a straight road, and the car in front brakes suddenly as the ego approaches"*, our LLM agent can first decompose it into the sub-descriptions for each component via demonstrations easily, such as *"Behavior: the adversarial car suddenly brakes as the ego approaches"* for the behavior component. Then, our agent will do the corresponding retrieval and assemble the relevant code snippets based on the embedding of these descriptions to generate comprehensive Scenic code for simulation in CARLA. To provide a clearer understanding of our methodology, we define the following foundational concepts:

Route. A *'route'* is essentially a sequence of waypoints, each marking a specific location that the vehicle is intended to pass through during its trajectory. In the CARLA simulation environment, this route represents a pre-determined path for the ego vehicle that includes both the starting and the terminal points.

Base Scenario. A *'base scenario'* is utilized to conceptualize a high-level adversarial driving situation. It provides an abstract framework, such as *"a straight obstacle ahead of the ego vehicle"*, without delving into specifics regarding the identity or adversarial behavior of the obstacle. This abstraction allows for a generalized approach to categorizing various driving challenges.

Scenario. In contrast, a *'scenario'* builds upon the *'base scenario'* by infusing it with detailed attributes concerning the obstacle and its specific behaviors. For instance, the previous example *"The ego vehicle is driving on a straight road, and the car in front brakes suddenly as the ego approaches"* is a scenario derived from the foundational base scenario as shown above, which enriches the initial description by introducing specific dynamics of the adversarial situation.

Scene. A *'scene'* represents the practical instantiation of a *'scenario'*, detailing the specifics of an adversarial event. This includes the route of the ego vehicle, the characteristics of the surrounding adversarial vehicle (*e.g.*, vehicle type), and environmental context (*e.g.*, buildings or traffic signals). Furthermore, it specifies parameters such as the positions, velocities, and initial placements of the agents involved. Notably, Scenic actually acts as a *probabilistic* programming language as shown in Figure 3; it will sample the parameters like `ADV_SPEED` before running the corresponding simulation in CARLA. This capability enables a single Scenic script to produce various distinct scenes for the same scenario by varying parameter values like speed within a predefined range.

3.2. Construction of the Retrieval Database

This section details our methodology for building the retrieval database, emphasizing the systematic collection and integration of Scenic code snippets.

Collection of Snippets. Our snippet collection process initiates with sourcing initial examples from the Scenic repository[1]. These examples, covering a range of adversarial behaviors and geometric configurations (e.g., straight roads, intersections), are then decomposed manually into description-snippet pairs. Utilizing this initial combined dataset, we engage LLMs for few-shot learning to generate more diverse snippets. This iterative process involves

[1] https://github.com/BerkeleyLearnVerify/Scenic/tree/main/examples/carla

```
## DEFAULT MAP AND MODEL SETTING
param map = localPath('../maps/Town05.xodr')
param carla_map = 'Town05'
model scenic.simulators.carla.model
EGO_MODEL = "vehicle.lincoln.mkz_2017"
```

```
## ADV BEHAVIOR OF THE SURROUNDING OBJECT
## The adversarial car suddenly breaks when the ego
approaches.
behavior AdvBehavior():
    do FollowLaneBehavior(target_speed=globalParameters.
         ADV_SPEED) until (distance from ego to self) <
         globalParameters.ADV_DISTANCE
    while True:
        take SetBrakeAction(globalParameters.BRAKE)
param ADV_SPEED = Range(0, 10)
param ADV_DISTANCE = Range(0, 20)
param BRAKE = Range(0, 1)
```

```
""" The ego vehicle is driving on a straight road and the
car in front brakes suddenly when the ego approaches.
"""
```

```
## GEOMETRY
## Ego agent drives in a straight road.
lane = Uniform(*network.lanes)
EgoSpawnPt = OrientedPoint on lane.centerline
Ego = Car at EgoSpawnPt,
    with blueprint EGO_MODEL
require (distance to intersection) > 50
```

```
## RELATIVE SPAWN POSITION OF THE ADVAGENT
## The adversarial car is in front of the ego.
param GEO_Y_DISTANCE = Range(0, 30)
AdvAgent = Car following roadDirection from
         EgoSpawnPt for globalParameters.GEO_Y_DISTANCE,
    with behavior AdvBehavior()
```

Figure 3. An example for the snippets in Scenic for a given safety-critical scenario description.

generating snippets for different components, including adversarial behaviors, geometric layouts, and relative spawn points. Each newly generated snippet is rigorously evaluated and corrected manually for its compatibility and compilability within Scenic's varying API contexts (such as differences across agent types like pedestrians, motorcycles, and cars). In this work, we consistently use `GPT-4`[2] to collect our snippets. Some examples of the prompts employed to generate these new snippets are provided in Appendix A.1.

Database Construction and Query Optimization. For every description-snippet pair, we continue to prompt `GPT-4` to generate several more distinct rephrasings of each description, maintaining the original snippet in each pair. This approach is designed to enhance the accuracy of retrieval. The descriptions are then encoded using `Sentence-T5`[3] [31], and the database construction and querying processes are facilitated by `faiss` [25]. We construct and manage the databases for different components, *i.e.,* adversarial behavior, road geometry, and relative spawn positions, independently.

3.3. Safety-Critical Scenario Generation

Upon the completion of the retrieval database, our LLM agent, ChatScene can now first generate a variety of descriptions for safety-critical scenarios and then convert them into the corresponding simulation via Scenic code during the evaluation. The detailed process is shown as follows:

i. **Instruct the LLM agent**: We start by instructing our LLM agent to generate potential adversarial scene descriptions. An example of such a query is: *"Provide a description of a safety-critical scenario where the ego vehicle is driving on a straight road."*

ii. **Description Extraction for Each Component**: To guarantee structured output for each scenario component, our LLM agent will then automatically employ a set of few-shot examples to guide it in generating organized sub-descriptions, which are formatted as "*Behavior: ...\n Geometry: ...\n Spawn Position: ...*" as illustrated in Figure 1 (a). Consequently, the agent can then efficiently extract the corresponding descriptions for each component using regular expressions. The prompts for the extraction are detailed in Appendix A.2.

iii. **Retrieving Scenic Code Snippets**: After extracting descriptions, our agent will utilize the `Sentence-T5` model to encode them. The embeddings will serve as keys for retrieving the relevant Scenic code snippets for each component, as shown in Figure 1 (b) and (c).

iv. **Scenario Rendering and Evaluation**: The Scenic code snippets are then assembled into a complete script and executed to run simulations in CARLA as demonstrated in Figure 1 (d). More text and simulation pairs are shown in Figure 2. Then, different parameter values, such as `ADV_SPEED` set between `[0, 10]` and `ADV_DISTANCE` between `[0, 20]`, will be sampled for collecting multiple scenes. The comprehensive details such as position, speed, acceleration, and collision information for all vechiles in each frame.

v. **Refinement of Collision-Prone Parameters**: To enhance the generation of the scenes that lead to the collision of the ego vehicle, the sampling ranges will be dynamically adjusted based on the information gained from previously collected data. Specifically, we will keep recording the parameters associated with collision cases and simply assume that parameters leading to collisions roughly align with a Gaussian distribution $\mathcal{N}(\mu, \sigma^2)$. Consequently, the sampling range will then be adjusted to $[\mu-\sigma, \mu+\sigma]$ for subsequent simulations. This iterative strategy has proven effective in increasing the probability of generating more collision-prone scenes. In the end, the most adversarially significant scenes will be kept for testing on each scenario.

The complete set of simulations along with comprehensive statistics are released for all involved vehicles. This data will support future research in the bidirectional conversions

[2] https://chat.openai.com/?model=gpt-4

[3] https://huggingface.co/sentence-transformers/sentence-t5-large

Table 1. **Statistics of scenario generation on selected scenarios**. We report *collision rate* (CR), the *overall score* (OS), and the *average displacement error* (ADE) to measure the effectiveness of different scenario generation algorithms; we test three differently trained ego vehicles, and the record herein represent the mean performance across these agents with all the scenes for the same base scenario. The last column shows the average over all the base scenarios, with bold numbers indicating the best performance among the 5 generation algorithms. LC: Learning-to-collide, AS: AdvSim, CS: Carla Scenario Generator, AT: Adversarial Trajectory Optimization, ↑/↓: higher/lower the better.

Metric	Algo.	Base Traffic Scenarios								Avg.
		Straight Obstacle	Turning Obstacle	Lane Changing	Vehicle Passing	Red-light Running	Unprotected Left-turn	Right-turn	Crossing Negotiation	
CR ↑	LC	0.30	0.09	0.87	0.83	0.71	0.69	0.59	0.58	0.584
	AS	0.51	0.33	0.86	0.87	0.57	0.70	0.29	0.57	0.586
	CS	0.45	0.61	0.89	0.87	0.63	0.69	0.68	0.60	0.676
	AT	0.50	0.31	0.78	0.82	0.71	0.68	0.59	0.62	0.627
	ChatScene	**0.89**	**0.70**	**0.95**	**0.93**	**0.79**	**0.75**	**0.78**	**0.86**	**0.831**
OS ↓	LC	0.761	0.830	0.505	0.507	0.601	0.615	0.548	0.588	0.619
	AS	0.673	0.707	0.507	0.490	0.675	0.607	0.705	0.593	0.620
	CS	0.698	0.567	0.489	0.490	0.641	0.613	0.505	0.579	0.573
	AT	0.668	0.714	0.538	0.505	0.607	0.620	0.545	0.569	0.596
	ChatScene	**0.470**	**0.522**	**0.434**	**0.440**	**0.537**	**0.560**	**0.474**	**0.421**	**0.482**
ADE ↑	LC	0.467	0.178	0.330	0.000	0.866	0.585	1.476	0.805	0.588
	AS	0.291	0.073	0.242	0.000	0.365	0.754	0.628	0.398	0.344
	CS	0.348	1.668	0.410	0.282	0.324	0.338	0.385	0.299	0.507
	AT	0.683	1.236	3.762	0.000	1.931	1.720	1.921	2.301	1.694
	ChatScene	**4.398**	**4.063**	**5.706**	**7.383**	**3.848**	**3.740**	**3.613**	**3.784**	**4.567**

among text, image, and video for autonomous driving.

4. Experiment

In this section, we conduct a quantitative evaluation of our agent in generating safety-critical scenarios. Our assessment is twofold: First, we test the actual safety-critical nature of the scenes produced by our agent, specifically their potential to provoke collisions involving the ego vehicle. Second, we evaluate the performance of the ego vehicle, which has undergone adversarial retraining using scenarios generated by our agent, to ascertain whether these scenarios contribute significantly to enhancing the robustness of ego vehicle.

4.1. Setup

In this work, to simulate autonomous driving, we control the ego vehicle with a reinforcement learning-based model and employ Scenic to guide the surrounding adversarial vehicle. Besides, for a more flexible and convenient implementation, we integrate Scenic into the Safebench platform [46] and conduct all the evaluations on Safebench.

AD algorithms. Safebench provides three prominent deep RL methodologies to train our ego vehicle. These are: Proximal Policy Optimization (PPO) [37], an on-policy stochastic algorithm; Soft Actor-Critic (SAC) [23], an off-policy stochastic technique; and Twin Delayed Deep Deterministic Policy Gradient (TD3) [22], a deterministic off-policy approach. The observation of the ego encompasses four essential dimensions: the distance to the next waypoint, longitudinal speed, angular speed, and a detection signal for front-facing vehicles.

Baselines. We employ two main categories of scenario-generation techniques for evaluation: Adversary-based and Knowledge-based. Adversary-based approaches, like **Learning-to-collide (LC)** [13] and **AdvSim (AS)** [44], challenge AD systems by altering initial poses of agents or perturbing trajectories. Knowledge-based approaches, such as **Carla Scenario Generator (CS)** [10] and **Adversarial Trajectory Optimization (AT)** [51], focus on scenarios adhering to real-world traffic rules and physical principles.

Metrics. Evaluation under Safebench encompasses three categories: *Safety level* (including collision rate and adherence to traffic signals), *Functionality level* (route adherence and completion), and *Etiquette level* (smoothness of driving and lane discipline). Our focus primarily lies on the *collision rate* and a composite *overall score (OS)*, with the latter aggregating all metrics, further details are deferred to Appendix C.1. We also leverage the *average displacement error* (ADE) to measure scene diversity generated by each algorithm. Specifically, it is calculated as the average of the mean of the Euclidean distances between the positions of the adversarial objects at each corresponding time step across the trajectories for each pair of scenes generated for the same scenario.

4.2. Safety-Critical Scenarios Generation

This section explores the capabilities of our agent in generating the most adversarial safety-critical scenarios when compared to the baselines.

Experiment Setup. Following Safebench [46], we leverage a surrogate ego vehicle trained using SAC to choose the most challenging scenes generated by various methods. Following this, we then evaluate these selected adversarial scenes using three distinct ego vehicles trained via SAC,

Table 2. **Diagnostic Report:** This report presents the average test results conducted using three distinct ego vehicles across eight base scenarios. These tests are evaluated on three different performance levels for each scenario generation algorithm, offering a comprehensive overview of agent efficacy. CR: collision rate, RR: frequency of running red lights, SS: frequency of running stop signs, OR: average distance driven out of road, RF: route following stability, Comp: average percentage of route completion, TS: average time spent to complete the route, ACC: average acceleration, YV: average yaw velocity, LI: frequency of lane invasion, OS: overall score, ↑/↓: higher/lower the better.

Algo.	Safety Level				Functionality Level			Etiquette Level			OS ↓
	CR ↑	RR ↑	SS ↑	OR ↑	RF ↓	Comp ↓	TS ↑	ACC ↑	YV ↑	LI ↑	
LC	0.584	**0.326**	0.158	0.032	0.894	0.731	0.216	0.211	0.243	0.112	0.619
AS	0.586	0.300	0.160	0.025	0.891	0.745	0.261	0.203	0.245	0.127	0.620
CS	0.676	0.313	**0.161**	**0.036**	0.890	0.741	0.244	0.215	0.243	0.131	0.573
AT	0.627	0.312	0.158	0.028	0.893	0.726	**0.279**	0.219	0.248	0.137	0.596
ChatScene	**0.831**	0.179	0.143	0.035	**0.833**	**0.544**	0.223	**0.705**	**0.532**	**0.243**	**0.482**

PPO, and TD3. This assesses the effectiveness and generality of different algorithms for generating safety-critical scenarios.

Base Scenario and Route. We adopt eight key base traffic scenarios from the Carla Challenge [7], whose texts are summarized from the NHTSA report [30], each with 10 diverse routes for the ego vehicle. These base scenarios are: *Straight Obstacle*, *Turning Obstacle*, *Lane Changing Vehicle*, *Passing Red-light Running*, *Unprotected Left-turn*, *Right-turn*, and *Crossing Negotiation*.

Scenario. In contrast to the baseline methods, which offer only one single scenario per base scenario, our approach demonstrates greater diversity. We consistently instruct our agent to generate five unique descriptions of scenarios under each base scenario, which are then mapped into corresponding Scenic scripts for simulation. Detailed descriptions of these scenarios can be found in Appendix B.

Scene. For each route and scenario, Safebench selects approximately 9 to 10 of the most adversarial scenes based on testing with a SAC-trained surrogate model on each route, resulting in about 98 to 100 scenes for each base scenario. For our approach, the agent first generate 50 simulations per scenario and route, updating parameter ranges every 10 steps. From these, our agent then selects the two simulations that not only lead to a collision but also yield the lowest *overall score* using the same surrogate model. Consequently, this method also yields a total of 100 scenes per base scenario, calculated as $2 \times 10 \times 5 = 100$. We report the average performance of all the selected scenes tested on the ego vehicles trained with three different AD algorithms for each base scenario.

Evaluation Results: Our experimental results, detailed in Table 1, provide a thorough evaluation of various scenario generation algorithms. These are assessed based on *collision rate* (CR), *overall score* (OS), and *average displacement error* (ADE), with metrics derived from testing three distinct ego vehicle training paradigms across various base traffic scenarios. Notably, our agent, ChatScene, consistently outperforms existing benchmarks across all metrics for each base scenario.

Specifically, ChatScene significantly enhances the generation of safety-critical scenarios, evidenced by a marked 15% increase in collision rates over the most competitive existing baselines. This substantial improvement in scenario complexity effectively challenges and evaluates autonomous driving systems in more adversarial environments.

Regarding overall performance, our agent achieves a significant relative reduction in the overall score, amounting to 16% more compared to the leading baseline. This reduction underscores the heightened complexity and challenge presented by our scenarios. Additionally, a detailed diagnostic report for the average performance on the overall eight base scenarios is provided in Table 2. This report provides a detailed breakdown of the overall score across three distinct levels, encompassing a broader range of evaluations beyond the collision rate. Notably, our generated scenarios considerably diminish the average route completion rate, and they compel the ego vehicle to maintain higher average acceleration and yaw velocity, alongside frequent lane invasions, to avoid collisions with surrounding adversarial objects. These dynamics further demonstrate the effectiveness and safety-critical nature of our agent, establishing its potential to create scenarios that rigorously test autonomous driving systems.

Moreover, ChatScene's superiority in scenario diversity is also confirmed by achieving the highest score in ADE metrics. This outcome, indicative of the variability in adversarial objects' trajectories, highlights the comprehensive and diverse nature of our generation approach, which is essential for a thorough assessment of autonomous driving systems.

In conclusion, our results demonstrate that ChatScene not only elevates collision rates across all base traffic scenarios but also significantly lowers the overall performance scores of ego vehicles. The enhanced scenario diversity also reinforces the effectiveness of our approach. This comprehensive performance underlines the potential of our agent to set new benchmarks in the evaluation and testing of autonomous driving systems. Detailed performance for each ego vehicle trained with different AD algorithms is provided in Appendix C.2.

Table 3. **Evaluating ego vehicle Performance Post-Finetuning:** We assess the effectiveness of various scenario generation methods based on two key metrics: *collision rate* (CR) and *overall score* (OS). For this evaluation, we finetuned the surrogate SAC-trained ego vehicle using the previously selected adversarial scenes for the first eight routes. The reported data represents the mean performance across the scenes from the last two routes, as provided by all methods. The 'PP' is shorted for 'Pre Pretraining,' which represents the corresponding performance of the surrogate ego vehicle on the scenes for the last two routes before finetuning. The last column provides an average across all scenarios. ↑/↓: higher/lower the better.

Metric	Algo.	Base Traffic Scenarios								Avg.
		Straight Obstacle	Turning Obstacle	Lane Changing	Vehicle Passing	Red-light Running	Unprotected Left-turn	Right-turn	Crossing Negotiation	
CR ↓	PP	0.48	0.39	0.58	0.59	0.69	0.67	0.46	0.60	0.559
	LC	0.12	0.22	0.51	0.03	0.29	**0.00**	0.37	0.14	0.210
	AS	0.23	0.05	0.53	**0.00**	0.22	0.05	0.41	0.23	0.216
	CS	0.22	0.20	0.39	**0.00**	0.04	0.22	0.19	0.14	0.176
	AT	0.14	0.13	0.30	**0.00**	0.18	**0.00**	0.23	0.09	0.135
	ChatScene	**0.03**	**0.01**	**0.11**	0.05	**0.03**	0.10	**0.01**	**0.00**	**0.043**
OS ↑	PP	0.673	0.684	0.648	0.607	0.609	0.620	0.651	0.560	0.632
	LC	0.827	0.778	0.684	0.944	0.824	0.954	0.696	0.795	0.813
	AS	0.784	0.840	0.666	**0.958**	0.838	0.937	0.677	0.750	0.806
	CS	0.816	0.787	0.715	0.957	**0.934**	0.820	0.767	0.806	0.825
	AT	0.849	0.783	0.803	0.955	0.850	**0.948**	0.809	**0.915**	0.864
	ChatScene	**0.905**	**0.905**	**0.906**	0.929	**0.934**	0.903	**0.893**	0.862	**0.905**

4.3. Adversarial Training on Safety-Critical Scenarios

The aim of these experiments was to assess the effectiveness of safety-critical scenarios generated by various algorithms in enhancing the resilience of an ego vehicle. The findings substantiate our hypothesis that the nature of adversarial scenarios is crucial to the robustness of the ego vehicle.

Experiment Setting. For maintaining the consistency, we conduct finetuning on the same surrogate SAC-trained ego vehicle under each base scenario independently, using scenes on the first eight routes generated by each algorithm, and test the adversarially finetuned ego vehicle with the selected scenes from the last two routes from all algorithms, which also resulted in around 100 test cases for each base scenario in total. The surrogate model is finetuned with 500 epochs, utilizing a learning rate of 0.0001. We report the optimal performance based on evaluations conducted every 50 epochs. Further details on finetuning settings and selecting the checkpoints can be found in Appendix C.3.

Evaluation Results. Table 3 presents the performance outcomes of post-adversarial training, showcasing the efficacy of our agent in strengthening the robustness of the ego vehicle. Notably, when finetuned adversarially with scenarios generated by our method, the ego vehicle consistently surpassed the performance of agents trained with alternative approaches in most base scenarios. We observed a 51% reduction in collision rates compared to the original ego vehicle without finetuning, and the overall score improved by 43% relatively. More significantly, the collision rate was reduced by an additional 9% compared to the SOTA, which indicates that our agent can effectively contribute to improving the safety and reliability of autonomous driving systems. By exposing the ego vehicle to more challenging and diverse scenarios, we are directly aiding in the advancement of robust autonomous vehicle algorithms.

In conclusion, the experimental results demonstrate the tangible benefits of our adversarial finetuning approach. The substantial reduction in collision rates, coupled with the marked enhancement in overall performance, underscores the potential of our agent in fortifying autonomous agents against adversarial perturbations. These findings signify a pivotal step towards establishing safer and more resilient autonomous driving systems, thereby fostering greater trust and reliability in real-world deployment scenarios.

5. Conclusion

In this work, we introduce ChatScene, an LLM-based agent skilled at safety-critical scenario generation by automatically generating descriptions of safety-critical scenarios, decomposing these descriptions to retrieve the appropriate Scenic code, and subsequently compiling it to run simulations within the CARLA environment. Our experiments reveal that the scenarios produced by ChatScene pose greater challenges, substantially elevating the collision rates for the ego vehicle under the same scenario compared to other methods. Moreover, these generated scenarios have proven to be more effective in fine-tuning the ego vehicles to avoid collisions in safety-critical situations, demonstrating the agent's utility in enhancing the robustness of autonomous vehicles.

Acknowledgment

This work is partially supported by the National Science Foundation under grant No. 2046726, No. 2229876, DARPA GARD, the National Aeronautics and Space Administration (NASA) under grant No. 80NSSC20M0229, and Alfred P. Sloan Fellowship.

References

[1] Yasasa Abeysirigoonawardena, Florian Shkurti, and Gregory Dudek. Generating adversarial driving scenarios in high-fidelity simulators. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8271–8277. IEEE, 2019. 3

[2] Gerrit Bagschik, Till Menzel, and Markus Maurer. Ontology based scene creation for the development of automated vehicles. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 1813–1820. IEEE, 2018. 3

[3] Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 387–402. Springer, 2013. 1

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020. 1

[5] Panpan Cai, Yiyuan Lee, Yuanfu Luo, and David Hsu. Summit: A simulator for urban driving in massive mixed traffic. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4023–4029. IEEE, 2020. 3

[6] Yulong Cao, Chaowei Xiao, Benjamin Cyr, Yimeng Zhou, Won Park, Sara Rampazzi, Qi Alfred Chen, Kevin Fu, and Z Morley Mao. Adversarial sensor attack on lidar-based perception in autonomous driving. In *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*, pages 2267–2281, 2019. 1

[7] CARLA. Carla autonomous driving challenge, 2019. 3, 7

[8] Baiming Chen, Xiang Chen, Qiong Wu, and Liang Li. Adversarial evaluation of autonomous vehicles in lane-change scenarios. *IEEE transactions on intelligent transportation systems*, 23(8):10333–10342, 2021. 1

[9] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022. 1

[10] Scenario Runner Contributors. Carla Scenario Runner. https://github.com/carla-simulator/scenario_runner, 2019. 6

[11] Jiaxi Cui, Zongjian Li, Yang Yan, Bohua Chen, and Li Yuan. Chatlaw: Open-source legal large language model with integrated external knowledge bases. *arXiv preprint arXiv:2306.16092*, 2023. 2

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 1

[13] Wenhao Ding, Baiming Chen, Minjun Xu, and Ding Zhao. Learning to collide: An adaptive safety-critical scenarios generating method. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2243–2250. IEEE, 2020. 6

[14] Wenhao Ding, Mengdi Xu, and Ding Zhao. Cmts: A conditional multiple trajectory synthesizer for generating safety-critical driving scenarios. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4314–4321. IEEE, 2020. 3

[15] 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. 1, 2

[16] Bradley J Erickson, Panagiotis Korfiatis, Zeynettin Akkus, and Timothy L Kline. Machine learning for medical imaging. *Radiographics*, 37(2):505, 2017. 1

[17] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1625–1634, 2018. 1

[18] Shuo Feng, Xintao Yan, Haowei Sun, Yiheng Feng, and Henry X Liu. Intelligent driving intelligence test for autonomous vehicles with naturalistic and adversarial environment. *Nature communications*, 12(1):748, 2021. 1

[19] Daniel J Fremont, Tommaso Dreossi, Shromona Ghosh, Xiangyu Yue, Alberto L Sangiovanni-Vincentelli, and Sanjit A Seshia. Scenic: a language for scenario specification and scene generation. In *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation*, pages 63–78, 2019. 2, 4

[20] Daniel J Fremont, Edward Kim, Tommaso Dreossi, Shromona Ghosh, Xiangyu Yue, Alberto L Sangiovanni-Vincentelli, and Sanjit A Seshia. Scenic: A language for scenario specification and data generation. *Machine Learning*, pages 1–45, 2022. 2, 4

[21] Daocheng Fu, Xin Li, Licheng Wen, Min Dou, Pinlong Cai, Botian Shi, and Yu Qiao. Drive like a human: Rethinking autonomous driving with large language models. *arXiv preprint arXiv:2307.07162*, 2023. 3

[22] Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *Proceedings of the 35th International Conference on Machine Learning*, pages 1587–1596. PMLR, 2018. 6

[23] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning*, pages 1861–1870. PMLR, 2018. 6

[24] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1

[25] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547, 2019. 5

[26] Enkelejda Kasneci, Kathrin Seßler, Stefan Küchemann, Maria Bannert, Daryna Dementieva, Frank Fischer, Urs Gasser, Georg Groh, Stephan Günnemann, Eyke Hüllermeier, et al. Chatgpt for good? on opportunities and challenges of large

language models for education. *Learning and individual differences*, 103:102274, 2023. 1

[27] Moritz Klischat, Edmond Irani Liu, Fabian Holtke, and Matthias Althoff. Scenario factory: Creating safety-critical traffic scenarios for automated vehicles. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–7. IEEE, 2020. 3

[28] Zelun Kong, Junfeng Guo, Ang Li, and Cong Liu. Physgan: Generating physical-world-resilient adversarial examples for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14254–14263, 2020. 1

[29] Ritchie Lee, Mykel J Kochenderfer, Ole J Mengshoel, Guillaume P Brat, and Michael P Owen. Adaptive stress testing of airborne collision avoidance systems. In *2015 IEEE/AIAA 34th Digital Avionics Systems Conference (DASC)*, pages 6C2–1. IEEE, 2015. 3

[30] Wassim G Najm, John D Smith, Mikio Yanagisawa, et al. Pre-crash scenario typology for crash avoidance research. Technical report, United States. National Highway Traffic Safety Administration, 2007. 7

[31] Jianmo Ni, Gustavo Hernandez Abrego, Noah Constant, Ji Ma, Keith Hall, Daniel Cer, and Yinfei Yang. Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1864–1874, 2022. 5

[32] Erik Nijkamp, Bo Pang, Hiroaki Hayashi, Lifu Tu, Huan Wang, Yingbo Zhou, Silvio Savarese, and Caiming Xiong. Codegen: An open large language model for code with multi-turn program synthesis. In *The Eleventh International Conference on Learning Representations*, 2022. 2

[33] Aayush Prakash, Shaad Boochoon, Mark Brophy, David Acuna, Eric Cameracci, Gavriel State, Omer Shapira, and Stan Birchfield. Structured domain randomization: Bridging the reality gap by context-aware synthetic data. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 7249–7255. IEEE, 2019. 3

[34] Davis Rempe, Jonah Philion, Leonidas J Guibas, Sanja Fidler, and Or Litany. Generating useful accident-prone driving scenarios via a learned traffic prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17305–17315, 2022. 3

[35] Elias Rocklage, Heiko Kraft, Abdullah Karatas, and Jörg Seewig. Automated scenario generation for regression testing of autonomous vehicles. In *2017 ieee 20th international conference on intelligent transportation systems (itsc)*, pages 476–483. IEEE, 2017. 3

[36] John M Scanlon, Kristofer D Kusano, Tom Daniel, Christopher Alderson, Alexander Ogle, and Trent Victor. Waymo simulated driving behavior in reconstructed fatal crashes within an autonomous vehicle operating domain. *Accident Analysis & Prevention*, 163:106454, 2021. 3

[37] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 6

[38] Karan Singhal, Shekoofeh Azizi, Tao Tu, S Sara Mahdavi, Jason Wei, Hyung Won Chung, Nathan Scales, Ajay Tanwani, Heather Cole-Lewis, Stephen Pfohl, et al. Large language models encode clinical knowledge. *arXiv preprint arXiv:2212.13138*, 2022. 2

[39] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *2nd International Conference on Learning Representations, ICLR 2014*, 2014. 1

[40] Shuhan Tan, Boris Ivanovic, Xinshuo Weng, Marco Pavone, and Philipp Kraehenbuehl. Language conditioned traffic generation. In *Conference on Robot Learning*, pages 2714–2752. PMLR, 2023. 3

[41] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. 1

[42] Robin van der Made, Martijn Tideman, Ulrich Lages, Roman Katz, and Martin Spencer. Automated generation of virtual driving scenarios from test drive data. In *24th International Technical Conference on the Enhanced Safety of Vehicles (ESV) National Highway Traffic Safety Administration*, number 15-0268, 2015. 3

[43] Akifumi Wachi. Failure-scenario maker for rule-based agent using multi-agent adversarial reinforcement learning and its application to autonomous driving. In *International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence, 2019. 1

[44] Jingkang Wang, Ava Pun, James Tu, Sivabalan Manivasagam, Abbas Sadat, Sergio Casas, Mengye Ren, and Raquel Urtasun. Advsim: Generating safety-critical scenarios for self-driving vehicles. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9909–9918, 2021. 6

[45] Shijie Wu, Ozan Irsoy, Steven Lu, Vadim Dabravolski, Mark Dredze, Sebastian Gehrmann, Prabhanjan Kambadur, David Rosenberg, and Gideon Mann. Bloomberggpt: A large language model for finance. *arXiv preprint arXiv:2303.17564*, 2023. 2

[46] Chejian Xu, Wenhao Ding, Weijie Lyu, Zuxin Liu, Shuai Wang, Yihan He, Hanjiang Hu, Ding Zhao, and Bo Li. Safebench: A benchmarking platform for safety evaluation of autonomous vehicles. *Advances in Neural Information Processing Systems*, 35:25667–25682, 2022. 2, 6, 12

[47] Zhenhua Xu, Yujia Zhang, Enze Xie, Zhen Zhao, Yong Guo, Kenneth KY Wong, Zhenguo Li, and Hengshuang Zhao. Drivegpt4: Interpretable end-to-end autonomous driving via large language model. *arXiv preprint arXiv:2310.01412*, 2023. 3

[48] Zhenpei Yang, Yuning Chai, Dragomir Anguelov, Yin Zhou, Pei Sun, Dumitru Erhan, Sean Rafferty, and Henrik Kretzschmar. Surfelgan: Synthesizing realistic sensor data for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11118–11127, 2020. 3

[49] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing

reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*, 2022. 3

[50] Linrui Zhang, Zhenghao Peng, Quanyi Li, and Bolei Zhou. Cat: Closed-loop adversarial training for safe end-to-end driving. In *Conference on Robot Learning*, pages 2357–2372. PMLR, 2023. 3

[51] Qingzhao Zhang, Shengtuo Hu, Jiachen Sun, Qi Alfred Chen, and Z Morley Mao. On adversarial robustness of trajectory prediction for autonomous vehicles. *arXiv preprint arXiv:2201.05057*, 2022. 6

[52] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *arXiv preprint arXiv:2306.05685*, 2023. 1

[53] Ziyuan Zhong, Davis Rempe, Yuxiao Chen, Boris Ivanovic, Yulong Cao, Danfei Xu, Marco Pavone, and Baishakhi Ray. Language-guided traffic simulation via scene-level diffusion. *arXiv preprint arXiv:2306.06344*, 2023. 3

A. Few-Shot Prompts

A.1. Prompt for generating snippets

In this section, we present an example to illustrate our process for collecting snippets. This particular example focuses on generating more adversarial behaviors of surrounding vehicles within the base scenario of a *"Straight Obstacle"*. The specific prompt employed in this process is detailed in Table 4. Upon receiving responses from the GPT-4.0, we meticulously review each description-snippet pair for adversarial behavior. This review includes manual verification and correction of any errors, such as the use of unusable APIs, ensuring the accuracy and applicability of the snippets.

A.2. Prompt for extracting descriptions

In this section, we present the few-shot prompt, as detailed in Table 5, designed for extracting component-specific descriptions from a comprehensive input description. While for extracting the specified descriptions for each component from the text, we will employ regular expressions.

B. Detailed Scenario descriptions

In this section, we provide the detailed descriptions for the scenarios generated by our method for each base scenario, the full descriptions are shown in Table 6 and Table 7.

C. Additional Experiment Details and Results

We run all our experiments on one NVIDIA RTX A6000; we consistently use the online version of GPT-4 as the underlying LLM for manually monitoring the generation quality, and it can be easily adapted to use the API with version `gpt-4-1106-preview` for making it more automatic.

C.1. Detailed Metric

We adopt the metrics from Safebench [46], and we provide the intuition for each metric here:

- CR (collision rate): Evaluates the rate of collisions, reflecting the autonomous system's accident avoidance capability.
- RR (frequency of running red lights): Measures the frequency of running red lights, an essential aspect of traffic law adherence.
- SS (frequency of running stop signs): Assesses how often the vehicle fails to stop at stop signs, a key traffic rule compliance metric.
- OR (average distance driven out of road): Quantifies the average distance the vehicle deviates from its intended roadway, indicating lane discipline.
- RF (route following stability): Examines the stability with which the vehicle follows its planned route.
- Comp (average percentage of route completion): Represents the average percentage of the planned route successfully completed by the vehicle.
- TS (average time spent to complete the route): Evaluates the time efficiency of the vehicle in completing its assigned routes.
- ACC (average acceleration): Measures the average acceleration, providing insights into the smoothness of the vehicle's operation.
- YV (average yaw velocity): Assesses the average yaw velocity, indicating the vehicle's turning and handling characteristics.
- LI (frequency of lane invasion): Quantifies the frequency of lane invasions, a measure of lane-keeping accuracy.

While overall score (OS) is an aggregated metric that combines all metrics above to provide a comprehensive performance overview. We also adopt the same weights in Safebench [46].

C.2. Detailed Performance for Differently Trained ego vehicles

We provide a detailed assessment of adversarial performance in the scenes generated by different methods in each test ego vehicle (trained with SAC, PPO, TD3, respectively). Detailed statistics on the collision rate can be found in Table 8, while the overall score is reported in Table 9. The results demonstrate that our method exhibits greater generalizability across diverse training algorithms for ego vehicles, consistently achieving the best average performance.

C.3. Adversarial Finetuning Details

We utilize a pretrained surrogate SAC-trained model from Safebench [46] for our experiment. The fine-tuning details are as follows:

- We finetune the model with 500 epochs (one epoch just represents one simulation for one scene).
- The policy learning rate and Q-value learning rate are both set at 0.0001.
- The experience replay buffer is divided into two: one for non-collision cases with a size of $20,000$, and another for collision cases with a size of 200.
- The batch size is fixed at 512. In each batch, 80% of the samples are drawn from the non-collision buffer, while the remaining 20% are from the collision buffer. If collision cases are less than $20\% \times 512$, all available collision cases are selected.
- The entropy regularization coefficient, analogous to the inverse of the reward scale in the original SAC paper, is set at 0.01.
- The discount factor is 0.99, and the Q-ensemble critic comprises 2 models.

During evaluation, it was observed that prolonged training epochs led the model to adopt a stopping strategy. Therefore,

to measure the performance accurately, we assess all checkpoints post 100 epochs at intervals of every 50 epochs. The optimal performance is reported based on the checkpoint exhibiting the lowest collision rate while maintaining a reasonable route completion rate (> 0.3), as adversarial events typically occur after 30% completion of the total route.

```
Your task is to provide descriptions of adversarial behaviors exhibited by surrounding objects
(pedestrians, cars, cyclists, motorcycles) that may lead to a collision with the ego vehicle on a
straight road. The behaviors should be safety-critical. Ensure that the provided code snippets adhere
to the Scenic API without creating new APIs. Here are some refined examples:

Behavior Description: A pedestrian suddenly starts crossing the road without looking.
Snippet:
behavior AdvBehavior():
    do CrossingBehavior(ego, globalParameters.ADV_SPEED, globalParameters.ADV_DISTANCE)
param ADV_SPEED = Range(0, 5)
param ADV_DISTANCE = Range(0, 20)

Behavior Description: A pedestrian steps onto the road right in front of the ego vehicle and stops.
Snippet:
behavior AdvBehavior():
    try:
        do CrossingBehavior(ego, globalParameters.ADV_SPEED, globalParameters.ADV_DISTANCE)
    interrupt when network.laneAt(self) is network.laneAt(ego):
        take SetWalkingSpeedAction(0)
param ADV_SPEED = Range(0, 5)
param ADV_DISTANCE = Range(0, 20)

Behavior Description: A car in an adjacent lane suddenly merges into the ego's lane.
Snippet:
behavior AdvBehavior():
    laneChangeCompleted = False
    try:
        do FollowLaneBehavior(target_speed=globalParameters.ADV_SPEED)
    interrupt when withinDistanceToAnyCars(self, globalParameters.ADV_DISTANCE) and not
    laneChangeCompleted:
        current_laneSection = network.laneSectionAt(self)
        leftLaneSec = current_laneSection._laneToLeft
        do LaneChangeBehavior(
            laneSectionToSwitch=leftLaneSec,
            target_speed=globalParameters.ADV_SPEED)
        laneChangeCompleted = True
param ADV_SPEED = Range(0, 10)
param ADV_DISTANCE = Range(0, 30)

Behavior Description: An adversarial cyclist sprints from behind a bus stop onto the road and stops in
front of the ego vehicle.
Snippet:
behavior AdvBehavior():
    do CrossingBehavior(ego, globalParameters.OPT_ADV_SPEED, globalParameters.OPT_ADV_DISTANCE) until
    (distance from self to network.laneAt(ego)) < globalParameters.OPT_STOP_DISTANCE
    while True:
        take SetWalkingSpeedAction(0)
param OPT_ADV_SPEED = Range(0, 10)
param OPT_ADV_DISTANCE = Range(0, 15)
param OPT_STOP_DISTANCE = Range(0, 1)

Now, based on these examples, your task is to provide additional Scenic code snippets that simulate
adversarial behaviors in traffic scenarios. Each snippet should be accompanied by a concise and clear
behavior description, similar to the provided examples. Your code must follow the existing Scenic
repository's API structure without introducing new APIs.
```

Table 4. A few-shot prompt for generating adversarial behavior in “Straight Obstacle” base scenario.

```
Your task is to decompose full descriptions of safety-critical scenarios into sub-descriptions for the
following distinct components:

Behavior: Describe the behavior of the adversarial object (you should also indicate the type of the
object like pedestrians, cars, cyclists, and motorcycles).
Geometry: Specify the road condition where the scenario occurs (e.g., straight road, three-way
intersection).
Spawn Position: Indicate the initial relative position of the adversarial object to the ego vehicle.

Here are refined examples:
Scenario: The ego vehicle is driving on a straight road, and the car in front brakes suddenly as the
ego approaches.
Behavior: The adversarial car suddenly brakes when the ego approaches.
Geometry: A straight road.
Spawn Position: The adversarial car is directly in front of the ego vehicle.

Scenario: The ego vehicle attempts a right turn at a four-way intersection, and an adversarial
pedestrian steps onto the road in front of it.
Behavior: The adversarial pedestrian deliberately steps onto the road right in front of the ego
vehicle.
Geometry: Lanes for turning right on a four-way intersection.
Spawn Position: The adversarial pedestrian is on the right front of the ego.

Scenario: The ego vehicle navigates around a parked car, and an oncoming car suddenly turns into its
path.
Behavior: The adversarial car suddenly turns into the ego's path without signaling.
Geometry: A two-lane road.
Spawn Position: The adversarial car is oncoming from the left lane of the ego.

Scenario: The ego vehicle is traveling along a straight road when a pedestrian, initially hidden behind
a bus stop on the sidewalk to the right, unexpectedly dashes onto the road directly in front of the ego
vehicle and comes to an abrupt stop.
Behavior: The adversarial pedestrian suddenly sprints from right, stopping abruptly in front of the ego
vehicle.
Geometry: A straight road.
Spawn Position: The adversarial pedestrian is initially stationed behind a bus stop on the right front.

Scenario: The ego vehicle is changing to the right lane when an emergency vehicle approaches rapidly
from behind.
Behavior: The adversarial car approaches rapidly from behind.
Geometry: A lane with right lanes on a straight road.
Spawn Position: The adversarial car is approaching from the rear on the right lane.

Scenario: The ego vehicle is turning right at an intersection, and a crossing car from left suddenly
brakes.
Behavior: The adversarial car suddenly brakes near the ego vehicle.
Geometry: Lanes for turning right on an intersection
Spawn Position: The adversarial car is crossing the intersection from the left.
```

Table 5. A few-shot prompt for extracting descriptions for different component given the full description of the safety-critical scenarios.

Table 6. The descriptions of scenarios generated in our method for the first four base scenarios.

Base Scenario	Description of Scenarios
Straight Obstacle	1. The ego vehicle is driving on a straight road; the adversarial pedestrian suddenly crosses the road from the right front and suddenly stops in front of the ego. 2. The ego vehicle is driving on a straight road; the adversarial pedestrian stands behind a bus stop on the right front, then suddenly sprints out onto the road in front of the ego vehicle and stops. 3. The ego vehicle is driving on a straight road; the adversarial pedestrian appears from a driveway on the left and suddenly stop and walk diagonally. 4. The ego vehicle is driving on a straight road; the adversarial pedestrian suddenly appears from behind a parked car on the right front and suddenly stop. 5. The ego vehicle is driving on a straight road; the adversarial pedestrian is hidden behind a vending machine on the right front, and abruptly dashes out onto the road, and stops directly in the path of the ego vehicle.
Turning Obstacle	1. The ego vehicle is turning left at an intersection; the adversarial motorcyclist on the right front pretends to cross the road but brakes abruptly at the edge of the road, causing confusion. 2. The ego vehicle is turning left at an intersection; the adversarial pedestrian on the opposite sidewalk suddenly crosses the road from the right front and stops in the middle of the intersection. 3. The ego vehicle is turning right at an intersection; the adversarial pedestrian on the left front suddenly crosses the road and stops in the middle of the intersection, blocking the ego vehicle's path. 4. The ego vehicle is turning left at an intersection; the adversarial cyclist on the left front suddenly stops in the middle of the intersection and dismounts, obstructing the ego vehicle's path. 5. The ego vehicle is turning right at an intersection; the adversarial motorcyclist on the opposite sidewalk abruptly crosses the road from the right front and comes to a halt in the center of the intersection.
Lane Changing	1. The ego vehicle is attempting to change lanes to avoid a slow-moving leading vehicle; the adversarial car in the target lane suddenly merges into the ego vehicle's original lane, blocking the ego from returning to its initial position. 2. The ego vehicle is changing to the right lane; the adversarial car is driving parallel to the ego and blocking its path. 3. The ego vehicle is performing a lane change to evade a slow-moving vehicle; the adversarial car in the target lane on the right front suddenly brakes, causing the ego vehicle to react quickly to avoid a collision. 4. The ego vehicle is preparing to change lanes to evade a slow-moving leading vehicle; the adversarial car in the target lane starts weaving between lanes, making it difficult for the ego to predict its position and safely execute the lane change. 5. The ego vehicle is attempting to change lanes to avoid a slow-moving leading vehicle; the adversarial car in the target lane suddenly slows down, matching the speed of the leading vehicle, and effectively blocks the ego from completing the lane change.
Vehicle Passing	1. The ego approaches a parked car that is blocking its lane and must use the opposite lane to bypass the vehicle, cautiously monitoring oncoming traffic, and suddenly encounters a jaywalking pedestrian, requiring the ego to quickly assess the situation and respond appropriately to avoid a collision. 2. The ego encounters a parked car blocking its lane and must use the opposite lane to bypass the vehicle, carefully assessing the situation and yielding to oncoming traffic, when an oncoming motorcyclist swerves into the lane unexpectedly, necessitating the ego to brake or maneuver to avoid a potential accident. 3. The ego approaches a parked car obstructing its lane and must use the opposite lane to go around when an oncoming car suddenly turns into the ego's path without signaling, requiring the ego to react quickly and take evasive action to prevent a collision. 4. The ego encounters a parked car blocking its lane and must use the opposite lane to bypass the vehicle when an oncoming car suddenly accelerates, closing the gap for the ego to safely return to its lane, necessitating the ego to quickly decide whether to accelerate or brake to avoid a collision. 5. The ego encounters a parked car blocking its lane and must use the opposite lane to bypass the vehicle when an oncoming pedestrian enters the lane without warning and suddenly stops, necessitating the ego to brake sharply or steer to avoid hitting the pedestrian.

Table 7. The descriptions of scenarios generated in our method for the last four base scenarios.

Base Scenario	Description of Scenarios
Red-light Running	1. The ego is driving straight through an intersection when a crossing vehicle runs the red light and unexpectedly accelerates, forcing the ego to quickly reassess the situation and perform a collision avoidance maneuver. 2. The ego vehicle is moving straight through the intersection; the adversarial vehicle, initially on the left front, runs the red light and makes an abrupt right turn, forcing the ego vehicle to perform a collision avoidance maneuver. 3. The ego vehicle is moving straight through the intersection; the adversarial vehicle, initially on the left front, runs the red light and makes an abrupt left turn, forcing the ego vehicle to perform a collision avoidance maneuver. 4. The ego moves straight at an intersection when a crossing vehicle runs the red light from the right and brakes abruptly, causing the ego to rapidly adapt its trajectory and perform a collision avoidance maneuver. 5. The ego vehicle is going straight through the intersection; the adversarial vehicle approaches from the left front and cuts off the ego vehicle.
Unprotected Left-turn	1. The ego starts an unprotected left turn at an intersection while yielding to an oncoming car when the oncoming car's throttle malfunctions, leading to an unexpected acceleration and forcing the ego to quickly modify its turning path to avoid a collision. 2. The ego attempts an unprotected left turn at an intersection while yielding to an oncoming car when the oncoming car suddenly brakes, necessitating the ego to rapidly reassess the situation and adjust its turn. 3. The ego vehicle is making an unprotected left turn; the adversarial vehicle approaches the intersection at a normal speed but then suddenly attempts to make a last-second right turn. 4. The ego attempts an unprotected left turn at an intersection while yielding to an oncoming car when the oncoming car veers erratically. 5. The ego commences an unprotected left turn at an intersection while yielding to an oncoming car when the adversarial car, coming from the right, blocks multiple lanes by driving extremely slowly, forcing the ego vehicle to change lanes.
Right-turn	1. The ego is performing a right turn at an intersection when the crossing car suddenly speeds up, entering the intersection and causing the ego to brake abruptly to avoid a collision. 2. The ego vehicle is turning right; the adversarial car (positioned ahead on the right) blocks the lane by braking suddenly. 3. The ego vehicle is turning right; the adversarial car (positioned ahead on the right) reverses abruptly. 4. The ego vehicle is turning right; the adversarial car (positioned behind on the right) suddenly accelerates and then decelerates. 5. The ego vehicle is turning right; the adversarial vehicle enters the intersection from the left side, swerving to the right suddenly.
Crossing Negotiation	1. The ego vehicle is approaching the intersection needs crossing negotiation; the adversarial car (on the left) suddenly accelerates and enters the intersection first and suddenly stops. 2. The ego vehicle is approaching the intersection needs crossing negotiation; the adversarial car (on the right) suddenly accelerates and enters the intersection first and suddenly stops. 3. The ego vehicle is entering the intersection needs crossing negotiation; the adversarial vehicle comes from the opposite direction and turns left and stops, causing a near collision with the ego vehicle. 4. The ego vehicle is entering the intersection needs crossing negotiation; the adversarial vehicle comes from the right and turns left and stops, causing a near collision with the ego vehicle. 5. The ego vehicle is entering the intersection needs crossing negotiation; the adversarial car, coming from the right, blocks multiple lanes by driving extremely slowly, forcing the ego vehicle to change lanes.

Table 8. **Collision Rate (CR) Performance Across Different Models**. This table presents the detailed analysis of the *collision rate* (CR) for various test ego vehicles, each trained with distinct RL algorithms. We showcase the mean CR for each model, demonstrating how they perform in the selected scenes under the same base scenario. Bold values denote the best performance for each scenario. Algorithms include: LC: Learning-to-collide, AS: AdvSim, CS: Carla Scenario Generator, AT: Adversarial Trajectory Optimization. Higher values of CR (↑) is preferable here.

Model	Algo.	Base Traffic Scenarios								Avg.
		Straight Obstacle	Turning Obstacle	Lane Changing	Vehicle Passing	Red-light Running	Unprotected Left-turn	Right-turn	Crossing Negotiation	
SAC (4D)	LC	0.40	0.11	0.60	0.80	0.92	0.86	0.70	0.75	0.642
	AS	0.53	0.40	0.75	**0.90**	0.62	0.85	0.21	0.53	0.599
	CS	0.53	0.68	0.67	**0.90**	0.76	0.90	0.69	0.68	0.725
	AT	0.73	0.48	0.77	**0.90**	**1.00**	**0.97**	0.76	**0.90**	0.814
	ChatScene	**0.94**	**0.73**	**0.92**	0.81	0.70	0.88	**0.84**	0.78	**0.825**
PPO (4D)	LC	0.09	0.11	**1.00**	**1.00**	0.22	0.20	0.28	0.00	0.363
	AS	0.39	0.19	**1.00**	**1.00**	0.67	**0.61**	0.62	**0.92**	0.675
	CS	0.22	**0.61**	**1.00**	**1.00**	0.47	0.37	0.52	0.44	0.579
	AT	0.10	0.11	0.98	0.87	0.13	0.21	0.03	0.05	0.310
	ChatScene	**0.87**	**0.61**	0.97	0.98	**0.89**	0.52	**0.74**	0.91	**0.811**
TD3 (4D)	LC	0.42	0.06	**1.00**	0.70	**1.00**	**1.00**	0.79	**1.00**	0.746
	AS	0.60	0.39	0.83	0.70	0.41	0.65	0.03	0.26	0.484
	CS	0.61	0.53	**1.00**	0.70	0.67	0.80	0.83	0.67	0.726
	AT	0.67	0.35	0.59	0.70	**1.00**	0.85	**0.99**	0.90	0.756
	ChatScene	**0.87**	**0.75**	0.96	**0.99**	0.77	0.86	0.75	0.90	**0.856**

Table 9. **Overall Score (OS) Performance Across Different Models**. This table presents the detailed analysis of the *overall score* (OS) for various test ego vehicles, each trained with distinct RL algorithms. We showcase the mean OS for each model, demonstrating how they perform in the selected scenes under the same base scenario. Bold values denote the best performance for each scenario. Algorithms include: LC: Learning-to-collide, AS: AdvSim, CS: Carla Scenario Generator, AT: Adversarial Trajectory Optimization. Lower values of OS (↓) is preferable here.

Model	Algo.	Base Traffic Scenarios								Avg.
		Straight Obstacle	Turning Obstacle	Lane Changing	Vehicle Passing	Red-light Running	Unprotected Left-turn	Right-turn	Crossing Negotiation	
SAC (4D)	LC	0.716	0.824	0.617	0.515	0.491	0.521	0.497	0.500	0.585
	AS	0.663	0.673	0.552	**0.466**	0.650	0.538	0.746	0.617	0.613
	CS	0.661	0.532	0.569	**0.466**	0.578	0.508	0.503	0.539	0.544
	AT	0.565	0.633	0.546	**0.466**	**0.462**	**0.475**	0.461	**0.423**	0.504
	ChatScene	**0.450**	**0.505**	**0.451**	0.489	0.582	0.492	**0.426**	0.461	**0.482**
PPO (4D)	LC	0.858	0.823	0.457	0.445	0.850	0.858	0.702	0.887	0.735
	AS	0.726	0.780	0.457	0.444	0.617	**0.646**	0.535	**0.408**	0.577
	CS	0.806	0.570	0.468	0.444	0.722	0.771	0.582	0.656	0.627
	AT	0.853	0.819	**0.439**	0.487	0.896	0.852	0.826	0.861	0.754
	ChatScene	**0.497**	**0.578**	0.444	**0.421**	**0.503**	0.705	**0.504**	0.417	**0.509**
TD3 (4D)	LC	0.708	0.843	0.442	0.561	**0.461**	**0.466**	0.447	0.378	0.538
	AS	0.631	0.668	0.511	0.561	0.757	0.638	0.834	0.755	0.669
	CS	0.627	0.599	0.430	0.561	0.622	0.559	0.430	0.542	0.546
	AT	0.587	0.689	0.629	0.561	0.462	0.534	**0.348**	0.423	0.529
	ChatScene	**0.462**	**0.483**	**0.407**	**0.410**	0.527	0.483	0.493	**0.385**	**0.456**