CrashSimGen: Generating Dangerous Scenarios with Diffusion Models for self-Driving

CS329 Machine Learning (H)

Project of Scenario Generation via Diffusion Model

12211612 Haibin Lai, 12211717 Wenkun Shen

2024 Dec 8th

1 Preface

Source code of our project: https://github.com/HaibinLai/CrashSimGen

Website: https://haibinlai.github.io/CrashSimGen/

Video Demo: [Machine Learning Grand Finale] https://www.bilibili.com/video/BV1ZPcxeFEP2

2 Abstract

CrashSimGen is a project that generates critical road scenarios using diffusion models for autonomous driving risk assessment. The project begins by encoding a large set of driving scenes as image data, and training a dangerous scenario recognizer that automatically identifies and labels potentially hazardous situations. These recognized dangerous scene thumbnails are then fed into a diffusion model to generate new dangerous scene thumbnails.

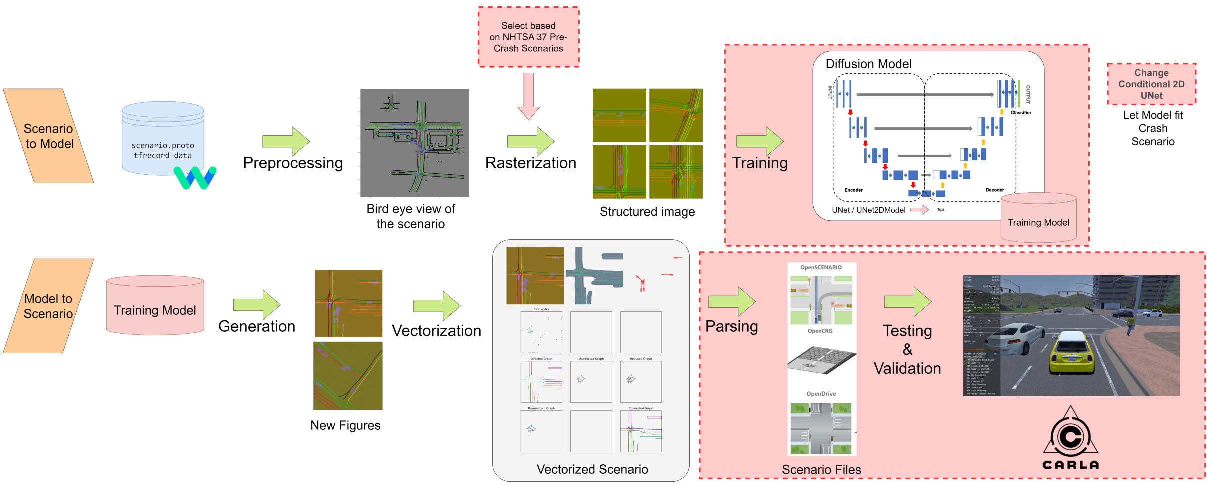


Figure 1: Project Workflow and Main Contribution

To transform the generated thumbnails into a format usable by autonomous driving systems, we try to develop a parser that utilizes OpenCV image processing tools to expand the scene thumbnails into OpenDrive formatted driving scenes. These scenes are then imported into the autonomous driving simulation software Carla, where they are tested to assess and improve the vehicle's response to dangerous scenarios in a simulated environment.

3 Introduction

In this part, we will show the **Background and Significance** in section 3.1 of data-driven approaches in autonomous driving research, and give a brief **analysis of Current Research Status** in section 3.2.

3.1 Data-driven Critical Driving Scenario Generation

In recent years, data-driven methods, particularly deep learning models, have become increasingly popular in the autonomous driving research community. As figure 2 displays, these models rely heavily on vast amounts of data collected from real-world driving scenarios to improve their performance[13]. The key advantage of such approaches lies in their ability to generalize across diverse situations, making them highly effective for tasks such as perception, decision-making, and control in autonomous systems.

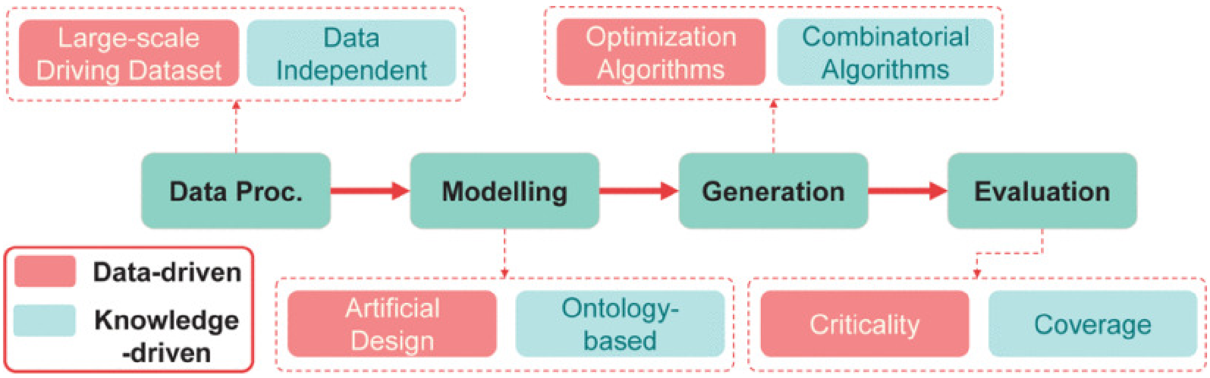


Figure 2: Comparison between data-driven and knowledge-driven scenario generations.[8]

Nowadays as the development of AI and self-driving models continues to grow, the significance of safety-critical data-driven techniques becomes ever more apparent , as they enable the development of systems that are robust to real-world complexity and variability [20].

The significance of studying critical dangerous self-driving scenarios is multifaceted. Firstly, the safety and reliability of autonomous vehicles (AVs) hinge on their ability to perceive, understand, and react appropriately to a wide array of traffic situations.[2]Real-world data, while valuable, is often limited in its diversity and quantity, especially when it comes to rare as figure 4 shows, but critical scenarios that are pivotal for testing the limits of AV safety. This is where the generation of diverse and realistic driving

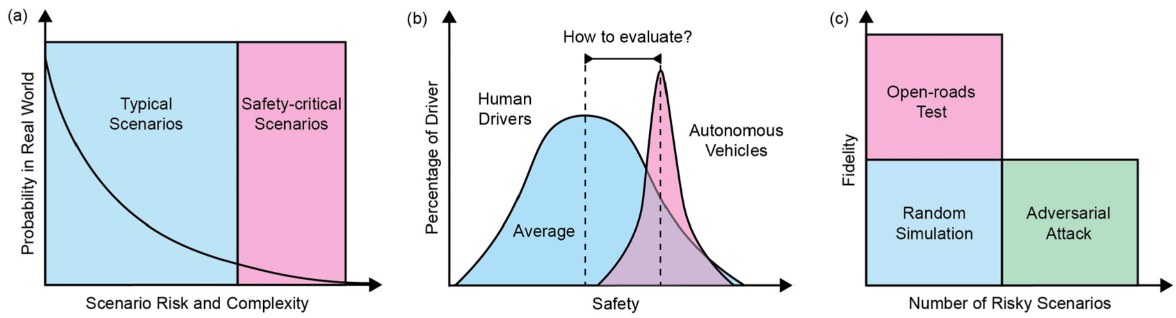


Figure 3: The importance and high value of critical scenario [5]

scenarios becomes crucial, as it allows for the simulation of these rare events in a controlled environment, thereby enhancing the robustness of AV systems [3].

Secondly, the generation of critical self-driving scenarios facilitates the development and validation of AV technologies in a cost-effective and time-efficient manner. Physical testing of every possible scenario is impractical due to the sheer scale and variability of traffic conditions.[22] Data-driven approaches on various scenarios, on the other hand, enable the creation of synthetic scenarios that can mimic real-world complexities, offering a scalable solution for AV testing and evaluation [7].

Furthermore, the study of critical self-driving scenarios is essential for advancing the field of artificial intelligence and machine learning. It provides a platform for testing and refining algorithms under conditions that are representative of the real world, thus pushing the boundaries of what is achievable with current technology[9]. The ability to generate diverse and realistic scenarios is not just about creating more data; it's about creating the right kind of data—data that can effectively train AV systems to handle uncertainty and make reliable decisions in unpredictable environments [6].

In summary, the study of self-driving scenarios is worth pursuing due to its direct impact on the safety, reliability, and advancement of autonomous driving technology.[15] It addresses the critical need for diverse and extensive data in AV development, offering a solution to the limitations of real-world data collection and providing a robust framework for testing AV systems under a multitude of conditions. The research is not only significant for the automotive industry but also for the broader field of AI, as it contributes to the development of more intelligent and adaptive systems capable of navigating the complexities of the real world [16].

3.2 Current Research Status

There have been several ways on data-driven critical scenarios generation with the rapid development of Generative Model as figure. These methods are mainly divided into two parts.

$$\begin{aligned}\mathcal{L} &= \mathbf{E}_{q,t}\left[D_{KL}\left(q(\mathbf{F}_{t-1} \mid \mathbf{F}_t, \mathbf{F}_0) \parallel p_\theta(\mathbf{F}_{t-1} \mid \mathbf{F}_t)\right)\right] \\ &= \mathbf{E}_{q,t}\left[D_{KL}\left(\mathcal{N}(\mathbf{F}_{t-1}; \mu_q, \Sigma_q(t)) \parallel \mathcal{N}(\mathbf{F}_{t-1}; \mu_\theta, \Sigma_t)\right)\right] \\ &= \mathbf{E}_{q,t}\left[|||\mu_\theta - \mu_q||_2^2\right]\end{aligned}$$

First, it samples scenarios from dataset, then it will use an estimation model $p_\theta(x)$ to learn the distribution of scenarios, usually measured by minimizing KL-Divergence or maximizing the log-likelihood [5].

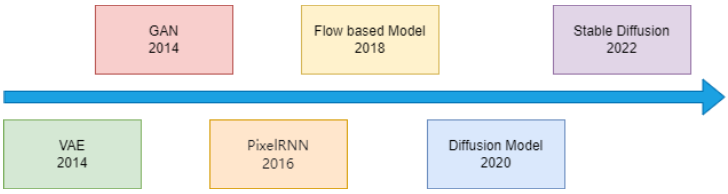


Figure 4: The Development of Generative Model

With the power of VAE, [4] learns a latent space of encounter trajectories and generates unseen scenarios by sampling from the latent space. However, with less understanding of the latent code, the generation is not controllable. As for the usage of GAN, [1] introduces recurrent models to generate realistic scenarios of highway lane changes. They use real-world data in the discriminator to help the improvement of the generator. SurfelGAN is proposed in [18] to directly generate point cloud data to represent scenarios from the view of the AV. However, GAN model needs more fine-tuning and maintains a bad KL-Divergence with the real critical data.

In recent years more models like Diffusion Model [10] comes to the tip of the Generative Models, enumerating more strong and powerful models in critical scenario generation. [14] introduces DriveSceneGen, it uses a classical diffusion model to generate novel driving scenarios that align with real-world data distributions with diversity. However, most of the generative scenario haven't been denoised enough and most scenario are too common

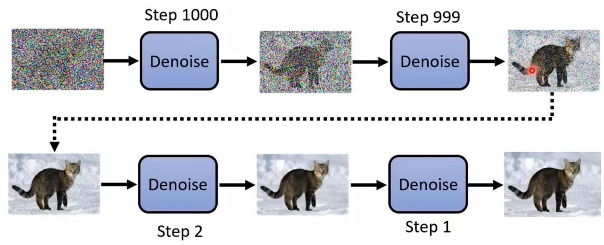


Figure 5: Diffusion Model's denoising example

or trival rather than critical. [17] use Denoising Diffusion Probabilistic Models (DDPM) enhanced with Diffusion with Transformer (DiT) blocks generating both realistic and diverse trajectories. However their works mainly focus on trajectory generation rather than more generalize critical scenarios as NHTSA 37 defined. [21] provides both controllability and realism traffic model using diffusion model, but it focus on safety scenarios rather than dangerous scenarios. [19] leverages the capabilities of LLMs to generate safety-critical scenarios for autonomous vehicles. But LLMs needs instructions and can only generate scenarios that close to the given limit tokens, while the real world scenario may be more various. [11] proposed VBD that utilizes diffusion generative models to predict scene-consistent and controllable multi-agent interactions in closed-loop settings. [12] combine latent diffusion, object detection and trajectory regression to generate distributions of synthetic agent poses, orientations and trajectories simultaneously.

Current Related Works	Use Diffusion Model?	Sampling on Biggest Dataset?	Critical Scenario?	CARLA Simulation and Validation ?
WCDT (CVPR 2024)	✅	✅	❌	❌
DriveSceneGen (RA-L 2024)	✅	✅	❌	❌
ChatScene (CVPR 2024)	❌	❌	✅	✅
Scenario Diffusion (NeurIPS 2023)	✅	❌	❌	❌
DiffScene	✅	❌	✅	❌
CTG (ICRA 2023)	✅	❌	❌	❌
CrashSceneGen (Our Work)	✅	✅	✅	✅

Figure 6: Our work's contribution compares to Different projects works on scenario generation

Compare to current works, our works use Diffusion Model to train on Waymo Open Dataset, which is one of the largest dataset in Automous Driving. We select Critical Scenario in Waymo Motion, then build up a fine-tuned Attention based Diffusion Model using DDIM Scheduler and tuned with Bayesian Optimization.

In our project, members are responsible for distinct, complementary areas to ensure

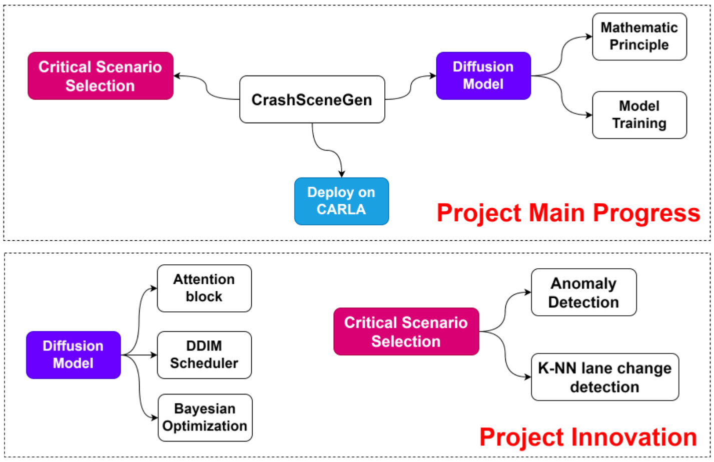


Figure 7: Project Progress and Innovation

comprehensive development and effective implementation.

Haibin Lai focuses on Diffusion Model Survey, where they explore the latest advancements in diffusion models relevant to generating critical road scenarios. They also handle Model Training & Fine-tuning, ensuring that the models are appropriately trained and optimized for our specific use case. Additionally, they work on the OpenScenario Parser, which is essential for transforming road scenarios into a format suitable for simulation environments like CARLA.

Wenkun Shen is responsible for conducting a Critical Scenario Survey, identifying and analyzing high-risk driving situations that are essential for autonomous vehicle testing. They also perform Anomaly Selection on Waymo Motion, where they examine anomalous events in the Waymo dataset to identify rare but critical scenarios. Finally, they manage CARLA Simulation, using the CARLA environment to test and evaluate the generated scenarios for realistic autonomous driving behavior and risk assessment.

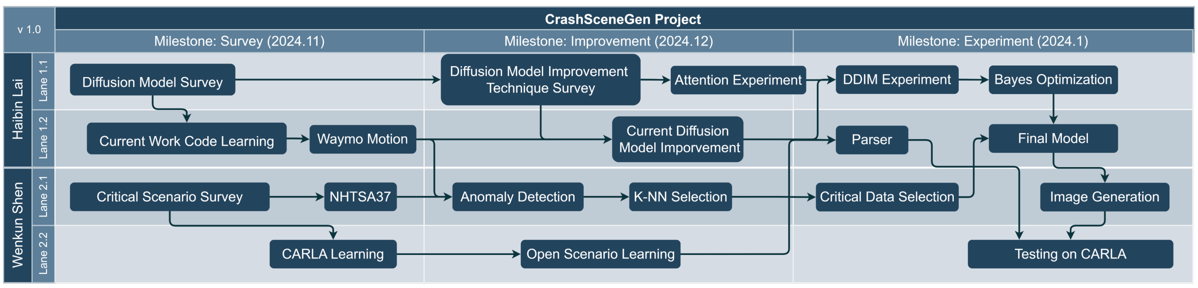


Figure 8: Our work's jobs allocation

4 Related Work

Critical Scenario recognition. Critical Scenario Recognition in autonomous driving faces several challenges. First, accurately identifying and classifying complex road situations, such as sudden pedestrian crossings or emergency braking, requires advanced perception technologies. Second, handling the vast amount of sensor data from cameras, LiDAR, and radar and effectively fusing it to understand the environment is computationally demanding. Third, real-time processing is crucial to make immediate decisions and avoid accidents in high-risk situations. Fourth, predicting the behavior of dynamic objects, like other vehicles and pedestrians, is a difficult task due to their unpredictable nature. Finally, ensuring the system's reliability across a wide range of driving conditions, including rare or edge cases, remains a significant hurdle.

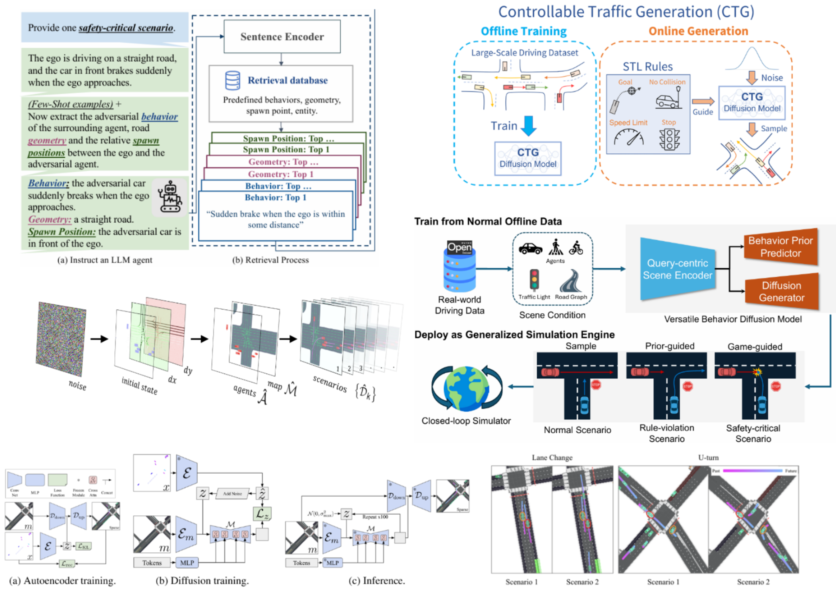


Figure 9: Different projects works on scenario generation using Diffusion Model

Generative Artificial Intelligence. Generative Artificial Intelligence (Generative AI) refers to a subset of AI techniques that focus on creating new content, such as text, images, and scenarios, based on patterns learned from existing data. This is achieved through advanced models like Generative Adversarial Networks (GANs), Variational Autoencoders (VAEs), and Transformer-based models.

Diffusion Model. Diffusion models are built on probabilistic principles and Markov Chain, and there are a few key formulas that describe their core components: the forward diffusion process, the reverse process, and how they are trained.

We will show the mathmatical process of Diffusion Model in section 5.1 .

Waymo Motion Dataset. The Waymo Motion Dataset is a large-scale dataset designed to support the development and evaluation of algorithms for autonomous driving. It is provided by Waymo, a self-driving technology company, and is part of their efforts to advance research in motion prediction, planning, and understanding the behavior of road users.

Waymo Motion	Parameters
Recorded Data length	574 hours
Data Size	About 860 GB
10Hz Segments	103,354
Scenarios/Frames	Over 200,000 / Over 2000 millions

Figure 10: Waymo Motion Dataset

5 CrashSimGen Workflow

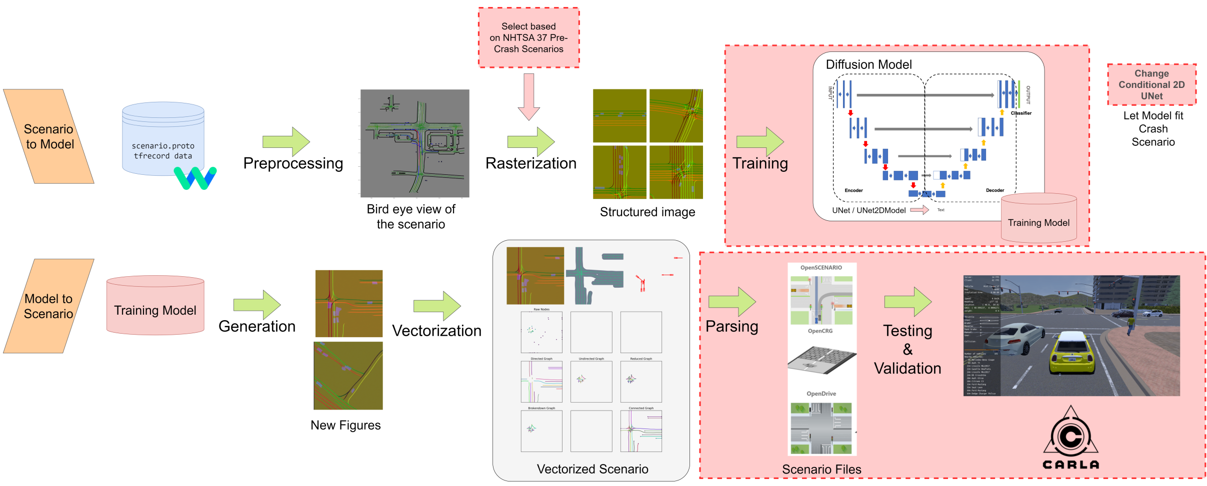


Figure 11: Project Workflow and Main Contribution

The CrashSimGen project focuses on generating critical and potentially hazardous road scenarios to assess the safety and performance of autonomous driving systems. The key objective is to enhance the ability of self-driving vehicles to respond to dangerous situations by simulating real-world risks in a controlled environment.

1. **Dangerous Scenario Recognition on dataset**

By encoding a large set of real-world driving scenes into image data, we plan to train or build a dangerous scenario recognizer. It can automatically identify and label scenes that are potentially hazardous for autonomous driving based on rules like NHTSA 37, or having many intersections with cars etc. These images represent various road scenarios, and the recognition system flags those that could pose a risk to vehicle safety.

2. Critial Scenario Diffusion Model Training

Once the dangerous scene thumbnails are identified, they are fed into a diffusion model. The model generates new thumbnails of dangerous road scenarios by utilizing its noise generation capability. This process helps create novel potential hazards that might not have been captured in the original dataset, providing additional scenarios for testing and evaluation. And We may change the model architecture, like adding Latent Diffusion Model or changing the architecture from current UNet2DModel to new Models.

3. Image Parsing and OpenDrive & OpenScenario Generation

To make the generated thumbnails usable for autonomous driving systems, we will develop a parser. This parser uses OpenCV, a popular image processing tool, to expand the scene thumbnails into OpenDrive-formatted driving scenarios. OpenDrive is a standardized format widely used in the autonomous driving industry to represent road networks and traffic environments. And OpenScenario can tell CARLA to imitate the scene based on the OpenDrive maps. They all can be generate from python package with information grabbed by OpenCV.

4. Testing and Evaluation in Simulation

The final step involves importing the generated and formatted dangerous driving scenes into Carla. In this simulated environment, we will test and evaluate current AI on vehicle's response to these hazardous scenarios. The simulation helps assess how well the autonomous driving system can handle and mitigate potential risks in real-world driving conditions.

In next part, we will show our data selection technique, fundamental knowledge acquisition, including learning about models and its internal principles, and our CARLA Parser design.

6 Critical Scenario Data Selection

In our work, Critical Scenario Data Selection followed the **NHTSA37** rules to identify dangerous scenarios within the Waymo Motion dataset. The NHTSA37 guidelines provide a set of 37 key factors that define hazardous or critical driving scenarios, focusing on factors such as vehicle speed, road conditions, and traffic interactions. By adhering to these rules, we ensured that our selection process captured a comprehensive range of potential risk scenarios relevant for autonomous driving safety assessment.

- **Isolation Forest** on Speed Detection
- **K-NN** Based Lane Change Detection

Table 1. List of 44 Crash Scenarios

No.	Title	Scenario Definition
1	Struck Human	A pedestrian crossing a multi-lane roadway was struck by vehicle. The driver was looking for other vehicles and traffic controls, but did not see the pedestrian. This crash occurs more frequently in urban areas. The weather is typically clear and the road is usually dry.
3	Struck Animal	A male driving home after dark on a rural two-lane country road in November struck a deer crossing the road. The driver could not avoid hitting the deer.
9	Drowsy	The driver fell asleep and drifted off the right side of the road and struck a telephone pole. Witnesses say that there was no attempt to brake or steer away from the pole. The crash occurred in a rural area at night.
10	Aggressive, Departure	The male driver was driving too fast, as well as cutting in and out of traffic, maneuvering the vehicle to the limits of control. The driver lost control of the vehicle and went into a skid. The driver left the roadway and struck the guardrail and then a tree.
11	Slick Road Departure	The driver lost control while driving on an icy, wet road. The driver tried to bring the vehicle back under control by braking and steering. The vehicle spun out and came to rest in the ditch.
12	Rough Road Departure	Due to the patched and eroded condition of the road surface, the driver lost control of the vehicle and left the roadway.
13	Avoidance, Departure	The driver was alert and driving along a surface street. Suddenly something appeared in the driver's path (e.g., child, bicyclist, or animal). The driver slammed on the brakes and swerved to avoid the immediate threat. The vehicle drove over a curb and into an object.
18	Impaired, Departure	The young (under 25) male driver, who was legally impaired, was driving too fast. He lost control of the vehicle, which left the roadway and overturned. The crash occurred in a rural area between midnight and 2 a.m. on a weekend.
19	Back Into Object	Vehicle A was backing out of a driveway and struck Vehicle B that was parked along the side of the road. Driver A did not see the other vehicle.
22	Ran Red "T-Bone"	Driver ran the red light. The driver saw the light turn yellow but decided to continue through the intersection. The majority of these crashes occur during daylight hours in urban areas.

Figure 12: Critical Data Selection based on NHTSA37

Overview of NHTSA37 Rules

NHTSA37 refers to the 37 rules developed by the National Highway Traffic Safety Administration (NHTSA) that categorize and define critical driving scenarios based on various risk factors. These rules cover a broad spectrum of driving conditions, including vehicle behaviors (e.g., speeding, lane changes), environmental factors (e.g., weather, road hazards), and interactions with other vehicles or pedestrians. The goal of these rules is to provide standardized guidelines for recognizing and assessing potentially dangerous situations on the road. By leveraging these rules, we ensure that our scenario selection process is aligned with industry standards and effectively highlights the most critical risks that autonomous vehicles need to address.

Scenario Detection Using Machine Learning Models For the detection of dangerous scenarios within the Waymo Motion dataset, we employed two machine learning techniques:

Isolation Forest:

We used Isolation Forest, a popular anomaly detection algorithm, to identify high-risk scenarios related to overspeeding and high-speed driving.

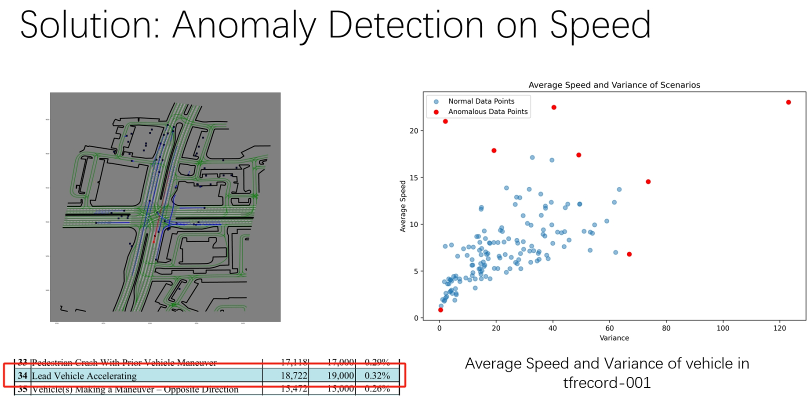


Figure 13: Anomaly Detection on speed based on NHTSA37

This model isolates observations by recursively partitioning the data, which allows it to detect anomalies that differ significantly from the typical driving patterns. In the context of dangerous road scenarios, the Isolation Forest was effective in recognizing instances where vehicles were traveling at dangerous speeds, which is a critical risk factor in autonomous driving.

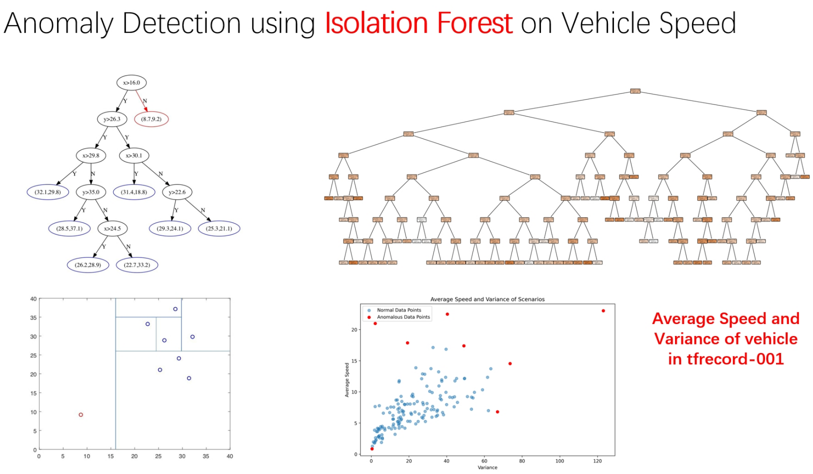


Figure 14: Mathematics Principle of Isolation Forest

K-Nearest Neighbors (KNN):

For detecting lane changes and steering maneuvers, we applied the KNN algorithm, which classifies scenarios based on the similarity to other instances in the dataset. By evaluating the proximity of various driving points and lane points, KNN was able to identify risky lane changes and turns that could lead to collisions or other hazardous events. This helped us isolate situations where vehicles performed sudden or aggressive movements, highlighting potential danger for autonomous vehicles.

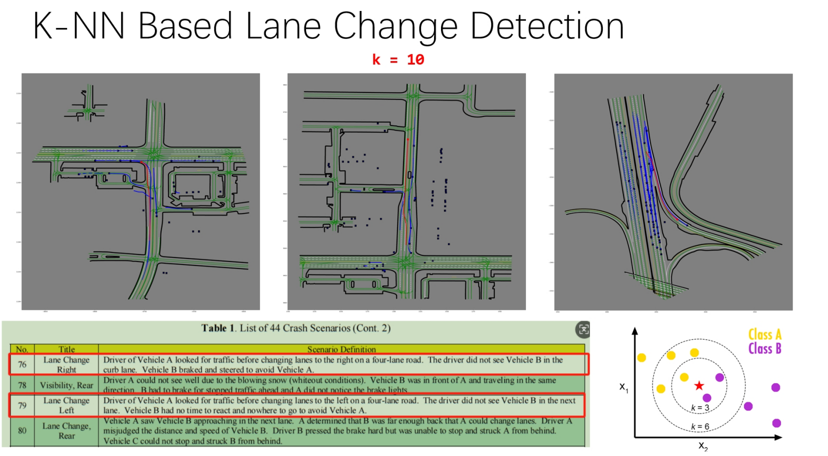


Figure 15: KNN for lane changing

Through this process, we were able to systematically identify and categorize critical road scenarios, allowing for better risk assessment and scenario generation for autonomous driving systems.

7 A fine-tuned Attention Based Diffusion Model with DDIM Scheduler

7.1 Mathematical Principle of Diffusion Model

Diffusion models are built on probabilistic principles and Markov Chain . Inside it are three processes : the forward diffusion process, the reverse process, and training process.

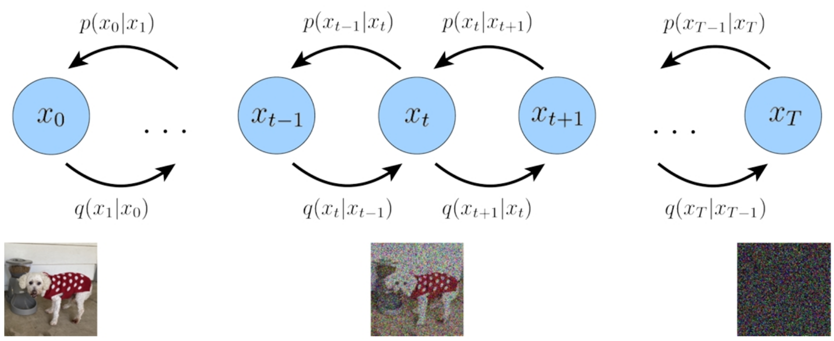


Figure 16: Diffusion Model's forward and reverse process

Forward Diffusion Process The forward diffusion process gradually adds noise to an image over a series of time steps. This process is modeled as a Markov chain that gradually corrupts a clean image into pure noise.

Let $\mathbf{x}_0$ be the original data sample (e.g., an image), and $\mathbf{x}_T$ be the noisy image at time step T (where T is the total number of steps). The forward process can be defined as:

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I})$$

Here: β_t is a noise schedule (a small positive value). $\mathbf{x}_t$ is the data sample at time t, which becomes increasingly noisy as t increases. $\mathcal{N}(\mu, \sigma^2)$ denotes a Gaussian distribution with mean μ and variance σ^2.

The above equation models the addition of Gaussian noise at each time step, with the standard deviation controlled by β_t.

Total Forward Process (Corrupting the Data Over Time) Forward diffusion process Describes a single step in the process of adding noise, and Total Forward Process describes the entire sequence of transitions. The forward process is iterated over multiple time steps, resulting in the corrupted sample $\mathbf{x}_T$ after T steps. The final noisy sample is given by:

$$q(\mathbf{x}_T|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1})$$

This equation expresses the total probability of transitioning from $\mathbf{x}_0$ to $\mathbf{x}_T$ via a series of noisy steps.

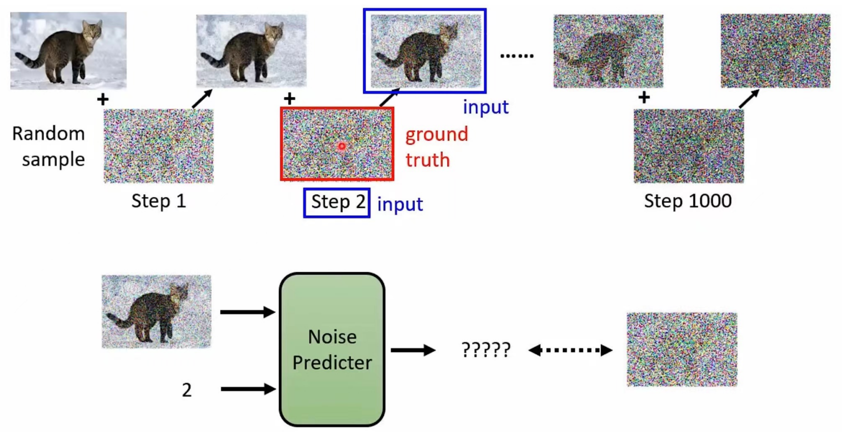


Figure 17: Diffusion Model's forward diffusion process

Reverse Process (Denoising the Sample) The reverse process is where the generative model learns to reverse the forward process, starting with random noise and gradually denoising it to recover the original data sample. The reverse process is modeled as a Markov chain:

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \sigma_\theta^2(t)\mathbf{I})$$

Where: $\mu_\theta(\mathbf{x}_t, t)$ is the mean of the predicted denoised image at time t, and it is learned by the model. $\sigma_\theta^2(t)$ is a learned variance at each step.

The reverse process is learned by the model and parameterized by a neural network (often a U-Net architecture).

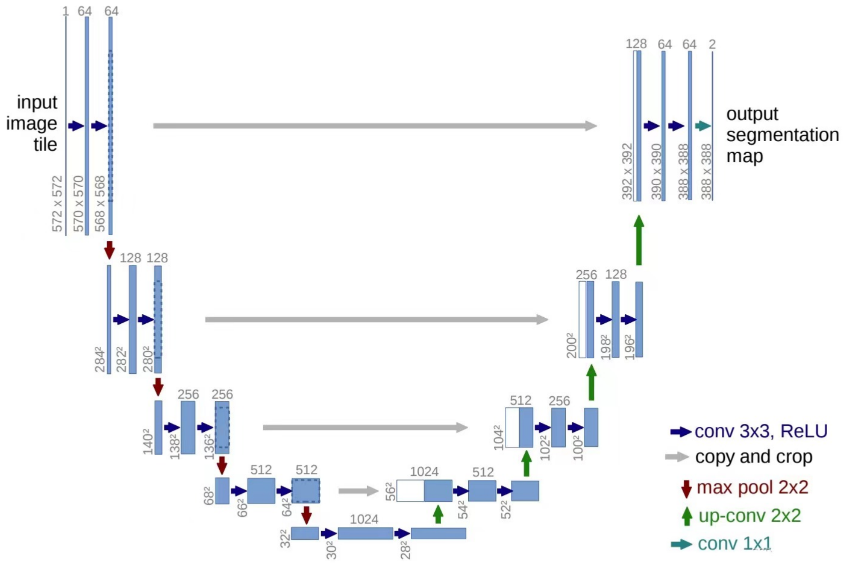


Figure 18: Diffusion Model's UNet architecture

Training Objective (Learning the Reverse Process) The goal of training the model is to learn the parameters θ of the reverse process such that it can accurately predict the denoised image at each step. To achieve this, we optimize a loss function that minimizes the difference between the predicted denoised image and the true data sample.

The standard objective is based on variational inference and the evidence lower bound (ELBO). The loss function is typically the **denoising score matching** objective, which can be written as:

$$L(\theta) = E_{q(\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_T)} \left[\sum_{t=1}^{T} \|\mathbf{x}_t - \hat{\mathbf{x}}_t\|^2 \right]$$

Where: $\hat{\mathbf{x}}_t$ is the predicted denoised version of $\mathbf{x}_t$ at time step t. The expectation is taken over the distribution of all noisy data sequences $q(\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_T)$.

This loss function encourages the model to accurately recover the clean image from noisy samples at each time step.

Inference (Sampling from the Model) Once the model is trained, generating new data samples involves sampling from the reverse process, starting with pure noise $\mathbf{x}_T$ and iteratively applying the learned denoising steps:

$$\mathbf{x}_{T-1} \sim p_\theta(\mathbf{x}_{T-1}|\mathbf{x}_T)$$

This process is repeated from $t = T$ down to $t = 1$, generating a new sample $\mathbf{x}_0$ after T steps.

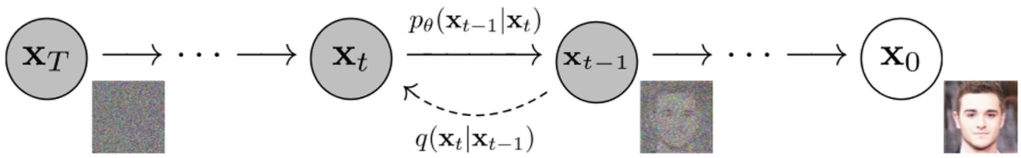


Figure 19: Diffusion Model's inference workflow

7.2 Model Under-fit: Add Attention Blocks for Critical feature on Data

Here we analyse the current diffusion model as figure **??**, with a learning rate of 1e-5, trained over 30 epochs and consisting of 56,574,595 parameters, appears to be underfitting the data. This is evident from the noisy nature of the generated images and the incorrect capture of certain features. One possible explanation for this underfitting could be the insufficient complexity of the model, given the large volume of image data. The model may not be adequately capturing the nuances of the data, leading to noisy outputs and the failure to preserve key scene characteristics.

The issue may stem from the model's inability to effectively learn from such a large dataset within the given training configuration, as figure 22 displays. Despite the substantial number of parameters, the architecture might not be sufficiently complex or sufficiently tuned to model the intricate details of the images. Additionally, the relatively low learning rate could be causing the model to converge too slowly, preventing it from

Training Diffusion Model: Pic in Mixture!
Underfitting

Selected Dataset Characteristic	Parameters
Image Size	256 x 256
Number of Scenarios	5102

Selected Diffusion Model	Parameters
Learning Rate	1e5
Epoch	30
Parameter	56,574,595

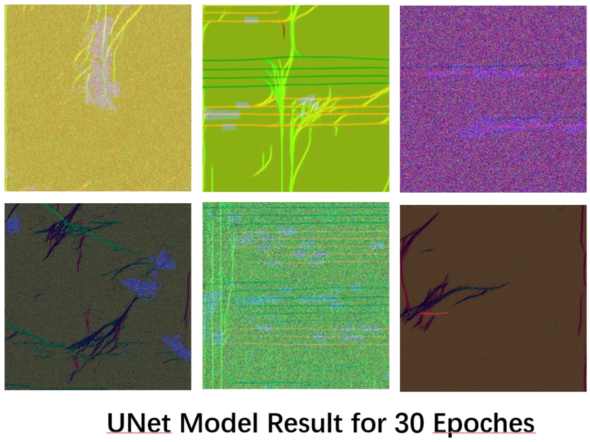


Figure 20: Current Diffusion Model is under-fitting

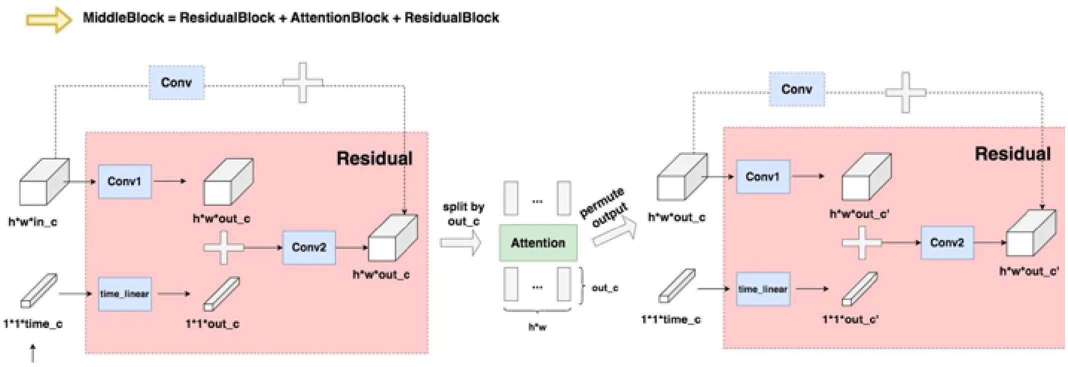


Figure 21: Attention Mechanism in Diffusion Model

achieving optimal performance within the 30 epochs. Thus, improving the model's capacity and potentially increasing the number of training epochs or adjusting the learning rate could help mitigate the underfitting issue and enable the model to better capture and preserve important scene features.

To further train the model and reduce the loss during training, we incorporated an attention mechanism module. This module is added on top of the existing residual neural network structure used in the sampling module, along with the upsampling and downsampling layers. By integrating an attention layer, the model gains the ability to focus on more relevant features of the input data, helping to better capture important patterns while reducing noise and improving the overall output quality. This addition aims to enhance the model's capability to effectively learn from the data and improve the generation of images.

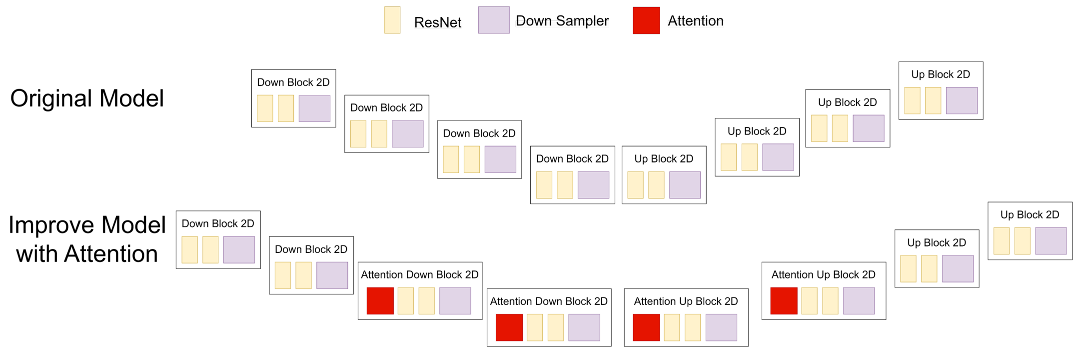


Figure 22: Attention Based UNet

The attention mechanism in diffusion models works by selectively emphasizing certain regions of the input data that are more informative or relevant to the task at hand. It allows the model to weigh different parts of the image differently, giving higher importance to crucial details and less importance to irrelevant or noisy areas. In diffusion models, this mechanism enables the network to focus on key features during the iterative denoising process, improving the accuracy and quality of generated images. By modulating the influence of different features at each step, the attention mechanism helps the model better preserve important scene characteristics and capture fine-grained details, thus reducing the likelihood of underfitting and enhancing the model's performance.

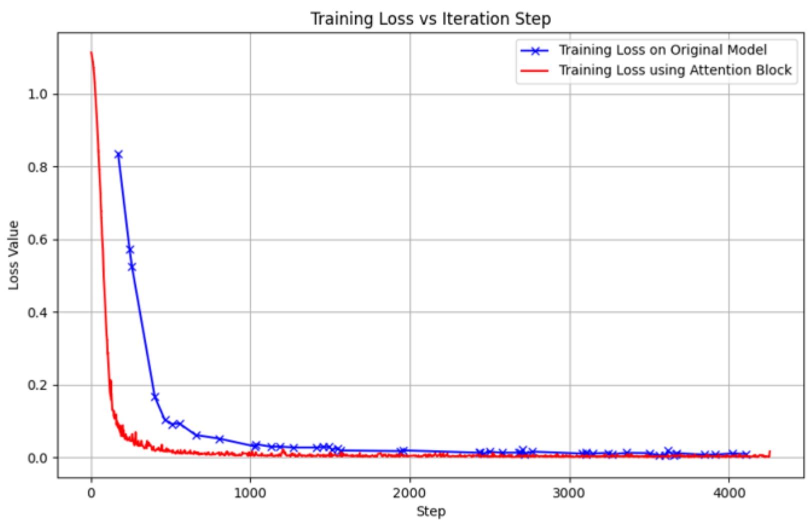


Figure 23: Training Improvement after using Attention block

After incorporating four attention layers into the model, we observed a rapid decline in the training loss within the first 1000 steps as 23 shows. The convergence speed of the model improved significantly compared to the original standalone UNet, achieving faster and more efficient training. This suggests that the attention mechanism effectively enhanced the model's ability to focus on key features of the data, allowing it to learn and adapt more quickly. By selectively emphasizing the most relevant areas during training, the attention layers contributed to a more refined learning process, leading to a quicker reduction in loss and improved overall model performance.

7.3 DDIM Scheduler: Stride Sampling

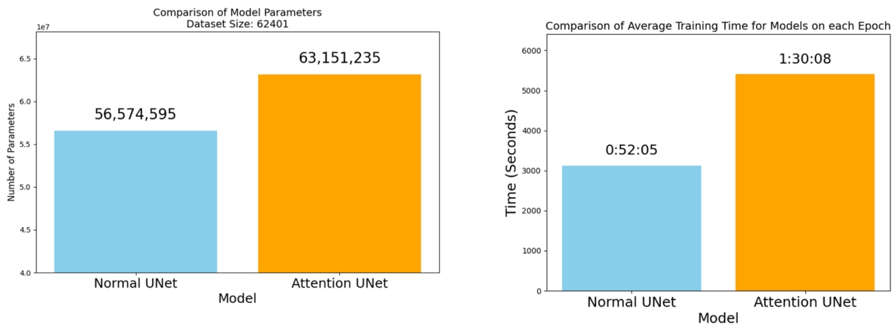


Figure 24: Model Parameter Explosion leads to training time increasing

After adding the attention layers, the model did indeed show promising results. However, this improvement brought about a new challenge. The model's parameter count increased from 56,574,595 to 63,151,235 due to the additional parameters introduced by the attention blocks. This significant increase in parameters resulted in a substantial slowdown in training speed. The training time per epoch escalated from 52 minutes to 90 minutes, making the training process considerably more expensive. The extensive number of parameters in the attention blocks made the training time longer, and it became clear that the model still required further denoising to achieve optimal performance.

To address this issue, we replaced the original DDPM (Denoising Diffusion Probabilistic Models) Scheduler with the DDIM (Denoising Diffusion Implicit Models) Scheduler. The DDIM scheduler is designed to accelerate the training process by introducing more efficient sampling techniques, which can reduce the number of steps required for denoising and thereby speed up convergence. By utilizing DDIM, we aim to maintain or even improve the quality of the generated images while significantly decreasing the training time, thus mitigating the cost of training the model with the added attention mechanism.

DDIM (Denoising Diffusion Implicit Models) is an advanced variant of the Denoising Diffusion Probabilistic Models (DDPM) that introduces a more efficient way of sampling from the diffusion process, aiming to accelerate training while preserving the quality of generated images.

The key principle behind DDIM is to provide a more flexible, implicit sampling process that allows for fewer diffusion steps while still effectively denoising the data. Unlike DDPM, which requires a fixed number of steps for each forward and reverse diffusion process, DDIM introduces a method where the reverse process is not strictly probabilistic but instead uses a more deterministic sampling approach. This reduces the computational cost by requiring fewer steps to reach a high-quality sample.

In DDIM, the model is trained in a similar way to DDPM, but during the generation phase, the model uses a non-Markovian reverse process to denoise the data more efficiently. This results in a significant reduction in the number of sampling steps needed, leading to faster generation times without sacrificing image quality.

The reverse diffusion process in DDPM is modeled by:

$p\theta(\mathbf{x}0 : T) = p(\mathbf{xT}) \prod t = 1^T p\theta(\mathbf{xt-1}|\mathbf{x_t})$

where:

$(p\theta(\mathbf{xt-1}|\mathbf{x_t}))$ is typically parameterized by a neural network to predict the noise at each timestep, $(p(\mathbf{x_T}))$ is usually a Gaussian distribution representing the noisy image at the final step.

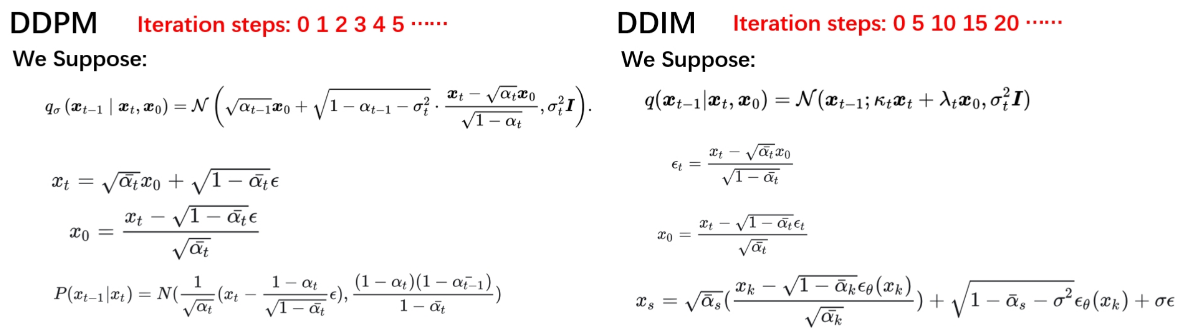


Figure 25: Training Improvement after using Attention block

3. DDIM Reverse Process

In DDIM, the reverse process is expressed as:

$\mathbf{xt-1} = \sqrt{\alpha t - \bar{1}}\left(\frac{\mathbf{xt} - \sqrt{1-\bar{}}\epsilon\theta(\mathbf{xt}, t)}{\sqrt{}}\right) + \sqrt{1 - \alpha t - \bar{1}}\,\epsilon\theta(\mathbf{xt}, t)$

where:

$(\mathbf{x_t})$ is the noisy image at time step (t), $(\mathbf{x_{t-1}})$ is the less noisy image at the previous time step, $(\epsilon\theta(\mathbf{xt}, t))$ is the predicted noise at time step (t), $()and(\bar{})$ are parameters that control the noise schedule at each time step.

4. DDIM Sampling (Non-Markovian)

DDIM introduces a more efficient reverse diffusion process with fewer steps. The reverse process equation can be written as:

$\mathbf{xt - 1 = xt} - \Delta\mathbf{x_t}$

where:

$\Delta\mathbf{xt}$ is the difference between the noisy image ($\mathbf{xt}$) and the predicted clean image based on$\epsilon\theta(\mathbf{xt}, t)$.

By using a more deterministic process and fewer steps, DDIM achieves faster sampling without losing image quality.

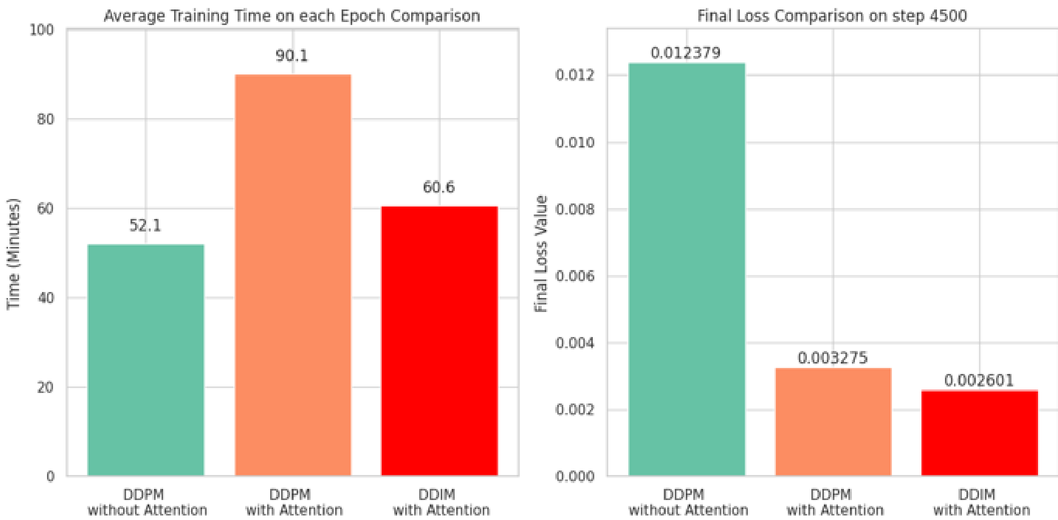


Figure 26: Improvement using DDIM Scheduler

In this work, we introduced an improvement to the model by incorporating DDIM, which significantly accelerated the sampling process and enhanced the training efficiency. As a result, the training time was reduced from 90 minutes to 60.6 minutes. Additionally, the loss at 4500 steps was lower compared to the original Attention module, demonstrating that DDIM not only improved the speed of learning but also led to better performance in terms of loss reduction. The key advantage of DDIM lies in its ability to reduce the number of required sampling steps, allowing the model to converge faster and learn more effectively.

Figure 27: Improvement on Training Loss using DDIM Scheduler

7.4 Bayesian Optimization to fine tune Model

To further improve the model's performance, we used Bayesian Optimization. This technique helps to optimize hyperparameters more efficiently by treating the optimization process as a probabilistic model. It uses a surrogate model, typically a Gaussian Process, to make predictions about the objective function and guide the search for optimal hyperparameters in an intelligent manner.

Explanation of Bayesian Optimization and its Basic Working Formula: Bayesian Optimization is a global optimization technique particularly well-suited for optimizing functions that are expensive to evaluate. It is often used to optimize hyperparameters in machine learning models, where each function evaluation (e.g., training a model) can be computationally expensive. Bayesian Optimization involves the following key steps:

1.Surrogate Model (Gaussian Process): We use a probabilistic model, often a Gaussian Process (GP), to approximate the unknown objective function. This model helps us predict the value of the objective function at new points. The Gaussian Process is used to model the objective function ($f(\mathbf{x})$) as a distribution over functions, where ($\mathbf{x}$) represents the hyperparameters and ($f(\mathbf{x})$) represents the performance measure (such as validation loss). The key idea is that instead of evaluating the function at every possible point, Bayesian Optimization chooses the next point based on the predictions of the surrogate model.

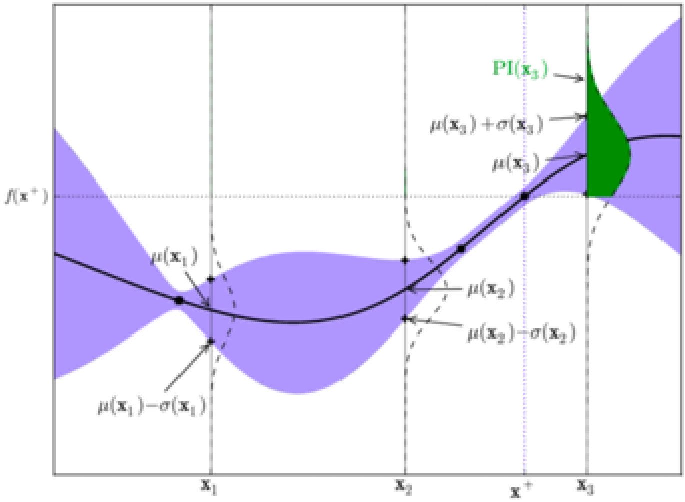


Figure 28: Gaussian Process on Bayesian Optimization

2.Acquisition Function: The acquisition function determines where to sample next by balancing exploration (trying areas of the parameter space with high uncertainty) and exploitation (choosing the points that are likely to perform well based on current knowledge). Common acquisition functions include Expected Improvement (EI), Probability of Improvement (PI), and Upper Confidence Bound (UCB).

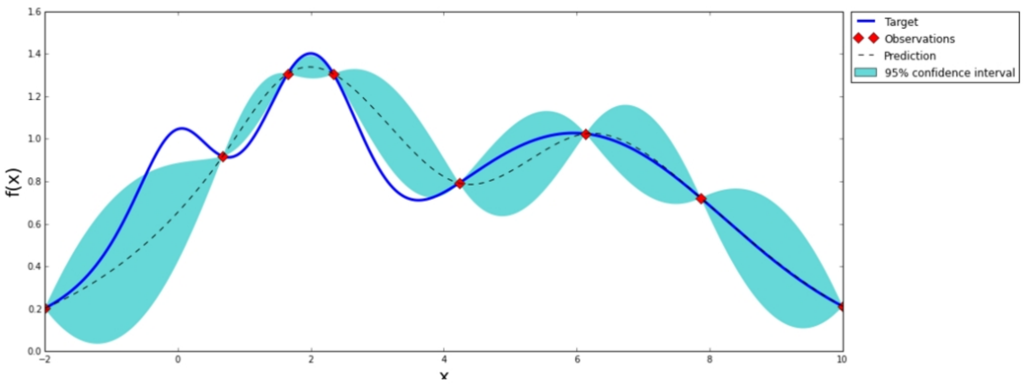
3.Updating the Model: After evaluating the objective function at a new point, we update the Gaussian Process model with the new data, refining our understanding of the objective function and improving the next prediction.

The Basic Formula for Gaussian Process: Given a set of data points ($\mathbf{X} = \mathbf{x1}, \mathbf{x2}, \ldots, \mathbf{xn}$) and corresponding values ($\mathbf{y} = f(\mathbf{x1}), f(\mathbf{x2}), \ldots, f(\mathbf{xn})$), the Gaussian Process is defined as a joint Gaussian distribution:

$$\left[f(\mathbf{x1}) \; f(\mathbf{x2}) \; \vdots \; f(\mathbf{x_n}) \right] \sim \mathcal{N}\left(0, \mathbf{K}(\mathbf{X}, \mathbf{X})\right)$$

Where ($\mathbf{K}(\mathbf{X}, \mathbf{X})$) is the covariance matrix between the input points, and the function values ($f(\mathbf{x})$) are assumed to be jointly Gaussian. The goal of Bayesian Optimization is to find the set of hyperparameters ($\mathbf{x}^*$) that maximize the expected value of the objective function ($f(\mathbf{x})$), which is modeled by the acquisition function.

By using Bayesian Optimization, the model can more efficiently search the hyperparameter space and find the optimal set of parameters with fewer evaluations, saving both



Probability of Improvement

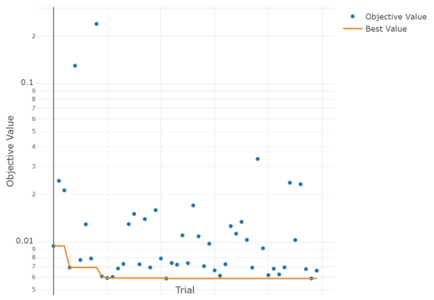
$$POI(X) = P(f(X) \geq f(X^+) + \xi) = \Phi(\frac{\mu(x) - f(X^+) - \xi}{\sigma(x)})$$

Expected Improvement

$$EI(x) = \begin{cases} (\mu(x) - f(x^+))\Phi(Z) + \sigma(x)\phi(Z), if\ \sigma(x) > 0 \\ 0, if\ \sigma(x) = 0 \end{cases}$$

Figure 29: Bayesian Optimization Main process

time and computational resources while improving the model's performance.



Hyper Parameter	Optimal Value
Number of Epochs	30
Learning Rate	0.00009934870757755

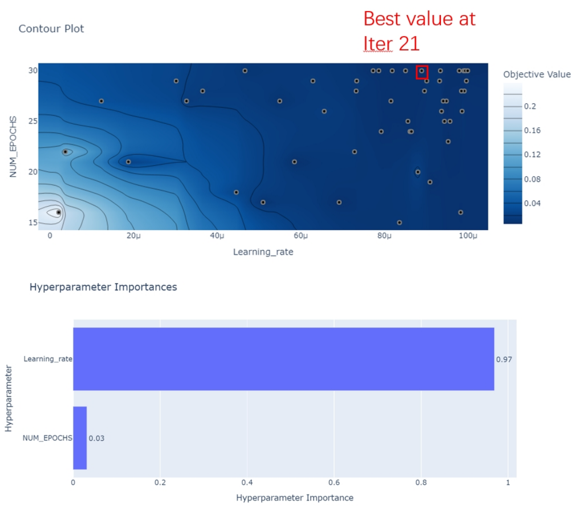


Figure 30: Bayesian Optimization Main process

Here we applied Bayesian Optimization to fine-tune two critical hyperparameters of the model: the learning rate and the number of epochs. The model was trained 50 times to identify the optimal hyperparameters, and the best values were found after 21 iterations: a learning rate of 0.00009934870757755 and 30 epochs. This process demonstrated the effectiveness of Bayesian Optimization in efficiently searching for optimal hyperparameter values, reducing the computational cost and time compared to traditional search methods.

The results also revealed that the learning rate had a more significant impact on the model's performance than the number of epochs. This highlights the crucial role the learning rate plays in guiding the gradient descent process and its influence on the optimization of the loss function. A well-chosen learning rate can greatly enhance the model's ability to converge efficiently to the optimal solution, whereas the number of epochs has a secondary impact on the training process.

8 Parser for Carla

After generating high-quality images, our next step is to identify the roads in the images and convert them into a map format that CARLA can recognize. To achieve this, we designed a parser based on the graph-based recognition approach from the original paper. The parser matches the complex road networks and transforms them into a map in OpenDrive format.

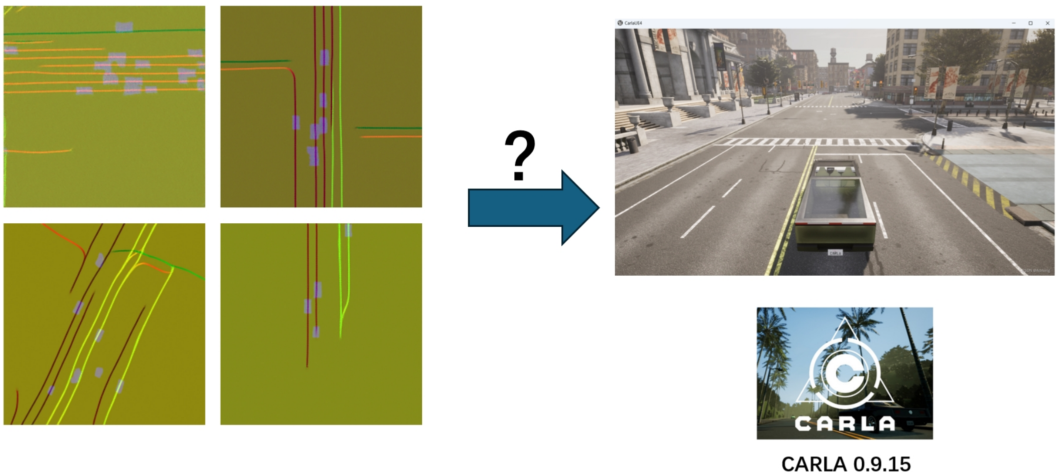


Figure 31: Need CARLA Parser

OpenDrive is an open standard file format designed for the representation of road networks in driving simulation environments. It provides a structured way to describe roads, intersections, lanes, traffic signs, and other road-related elements, which are essential for realistic driving simulations. OpenDrive files are typically in XML format and include precise geometric and semantic data about the road network, enabling simulation systems like CARLA to interpret and navigate the environment effectively.

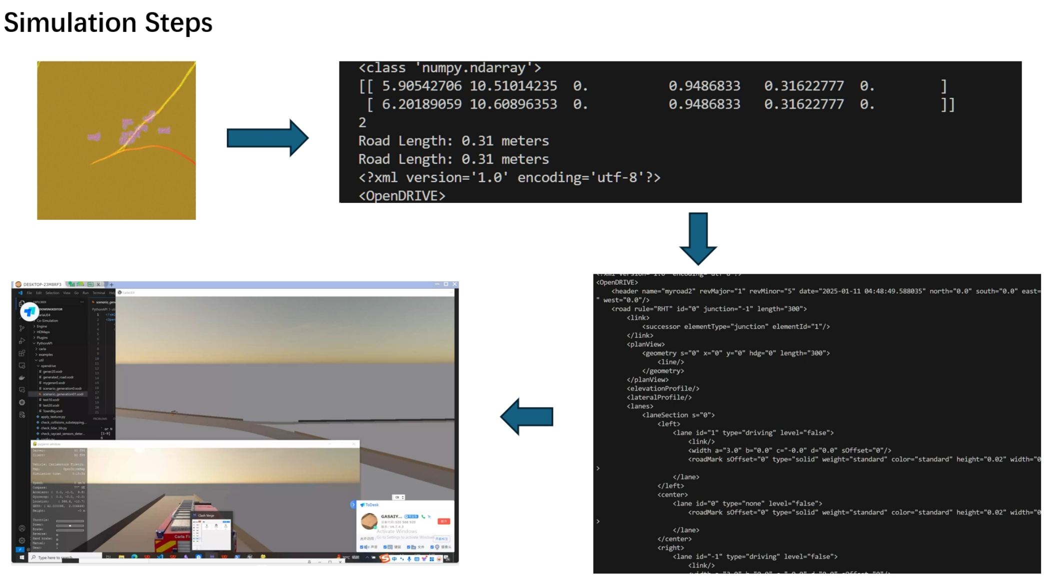


Figure 32: CARLA OpenDrive Generation Process

After our graph-based recognizer identifies lanes, vehicles, and crossroads, it records their positions and IDs. Then, we use OpenDrive to calculate the specific lengths and deviation angles of these lanes and stitch them together one by one.

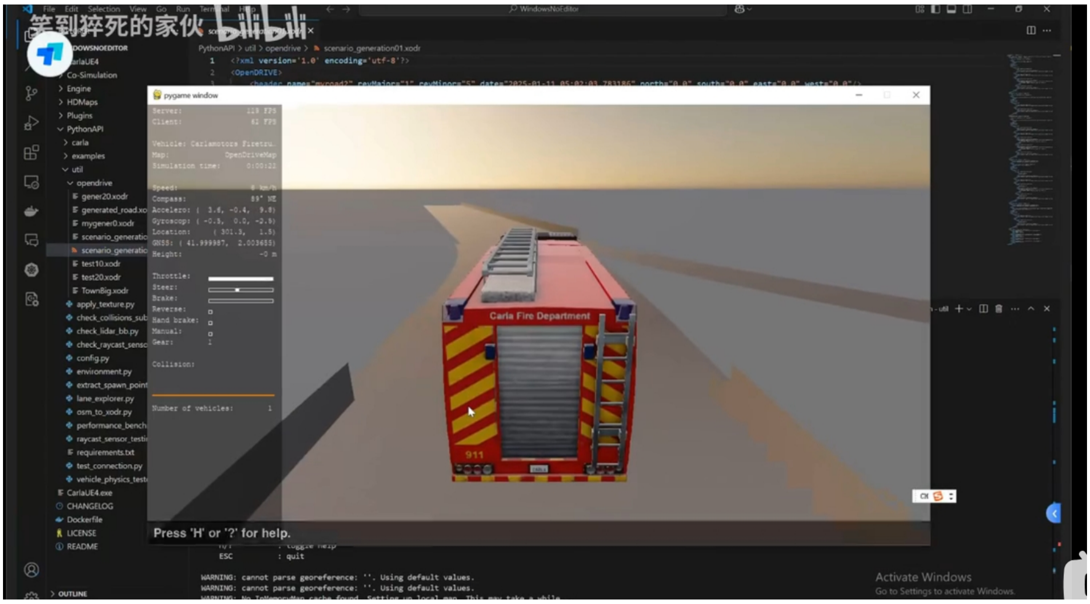


Figure 33: CARLA OpenDrive Simulation Process

For the crossroads, we add the center in OpenDrive and connect them to the adjacent roads. Due to time constraints, we were unable to include additional information such as traffic signals and pedestrians. We suggest that these features will be added in our future work.

After identifying key road elements (such as lanes, vehicles, and crossroads), the

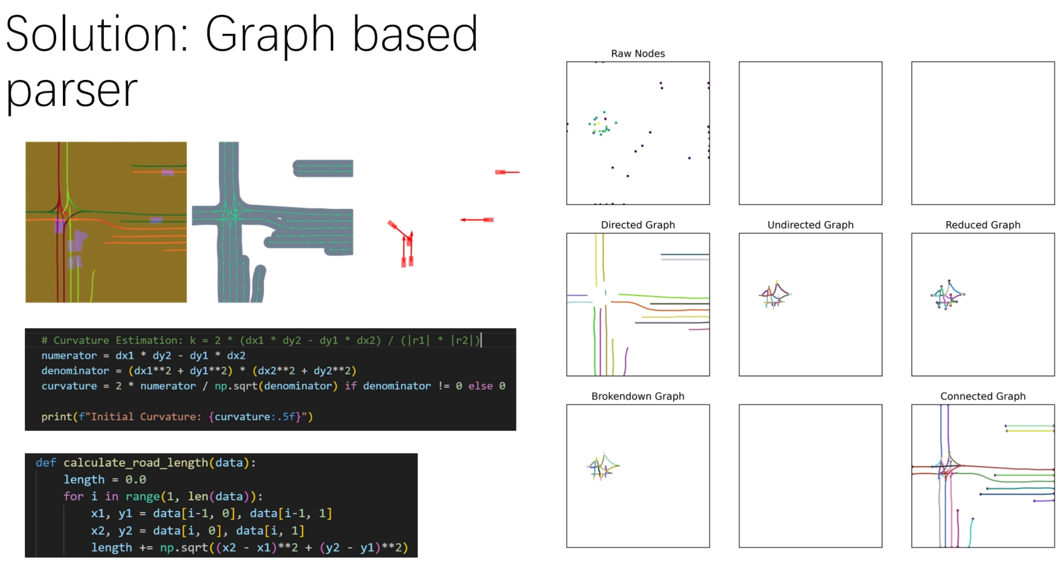


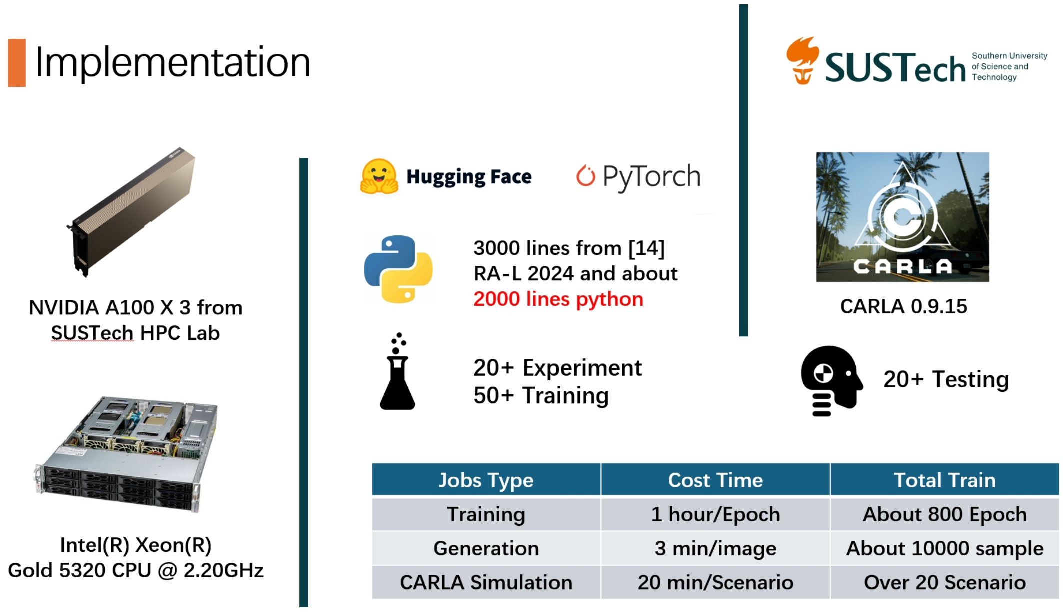
Figure 34: Graph based OpenDrive CARLA Parser

system tracks their positions and assigns unique IDs to each. The OpenDrive format is then used to calculate critical lane characteristics, such as length and deviation angle, and piece them together to form a complete road network. For crossroads, their centers are added in the OpenDrive map, and connections to nearby roads are made to ensure accurate representation. Due to time constraints, some elements, such as traffic signals and pedestrians, were not included in the current implementation but are planned for future updates.

9 Experiment

We conduct our experiment on the cluster at SUSTech HPC lab. We implenment our design with about 2000lines python codes, and train about 50 times model with using 3 NVIDA A100 GPU. And we test the openDrive simulation for about 20 times on CARLA 0.9.15 version.

The following part contains 4 experiment: Critical Scenario Comparison, Attention Comparison, DDIM Comparison, and Bayesian Optimization Comparison. We test if our tuning takes improvement.



Jobs Type	Cost Time	Total Train
Training	1 hour/Epoch	About 800 Epoch
Generation	3 min/image	About 10000 sample
CARLA Simulation	20 min/Scenario	Over 20 Scenario

Figure 35: Experiment Overview

9.0.1 Critical Scenario Comparison

In this experiment, we conducted a comparison of various critical road scenarios identified by our system against existing datasets and methodologies. This allowed us to assess if we can get the critical scenario recognition approach. By comparing scenarios from different sources on waymo motion as figure 36, we ensured that our system could detect a wide range of hazardous situations, providing a robust foundation for autonomous driving risk assessment.

-

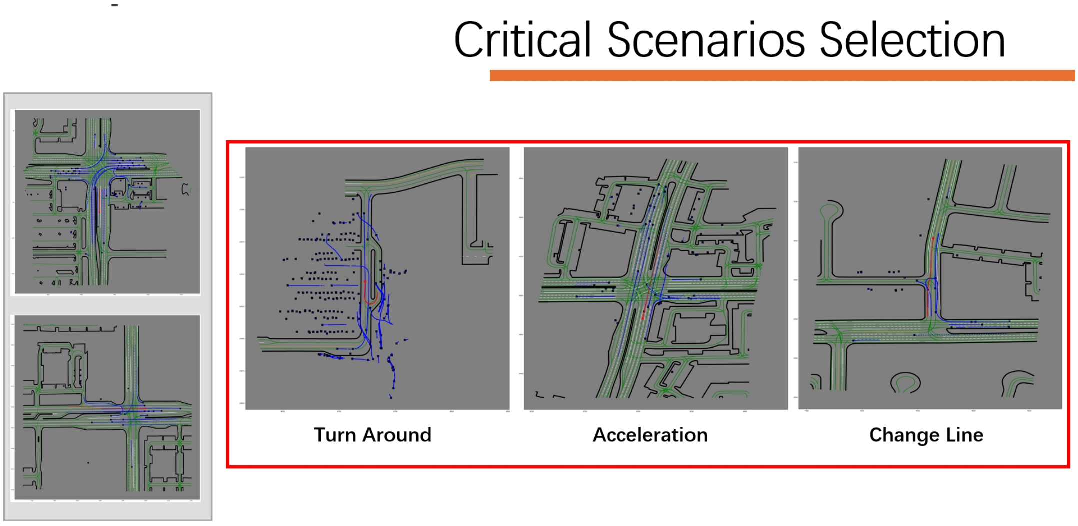


Figure 36: Experiment on Critical Data Selection

9.0.2 Attention Comparison

We carried out an attention mechanism as figure 37 shows, we make a comparison to evaluate different attention-based models for processing road scenario data. This experiment aimed to determine how different attention strategies—such as self-attention or multihead attention—impact the model's ability to capture spatial and temporal dependencies in driving scenes. The goal was to improve our model's performance by selecting the most effective attention mechanism for enhancing scenario recognition and simulation.

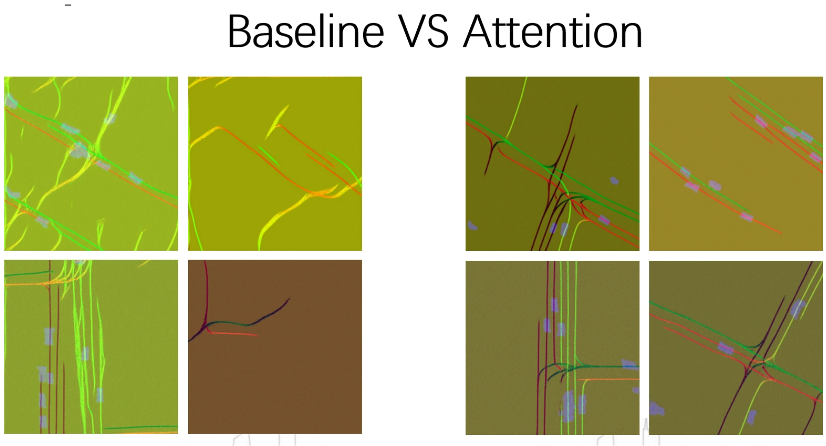


Figure 37: Experiment on Attention Blocks

9.0.3 DDIM Comparison

In this test, we compared the performance of the DDIM (Denoising Diffusion Implicit Models) scheduler against other diffusion models in generating driving scenarios as figure 38 displays. The experiment was focused on understanding how well DDIM performs in terms of generating realistic and coherent road situations compared to alternative diffusion methods. This comparison helped us fine-tune our model for more accurate and efficient scenario generation.

9.0.4 Bayesian Optimization Comparison

This experiment involved comparing Bayesian Optimization methods to improve the hyperparameter tuning of our diffusion models as figure 39 displays. Bayesian Optimization was used to optimize the model parameters, balancing exploration and exploitation to identify the most effective configuration for generating dangerous road scenarios. By com-

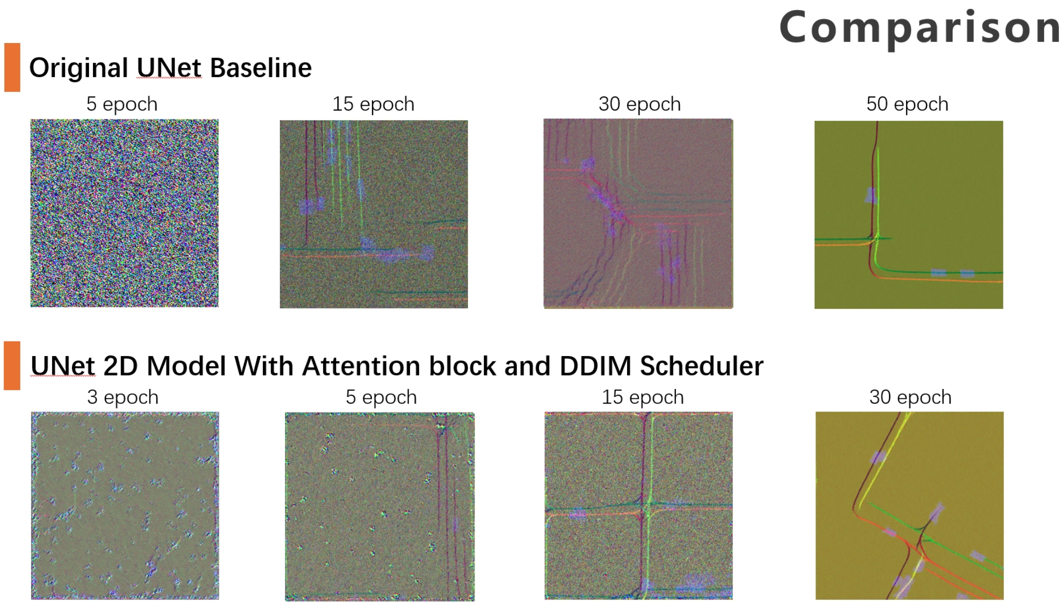


Figure 38: Experiment on DDIM Scheduler

paring different optimization strategies, we were able to fine-tune our model for better performance and efficiency.

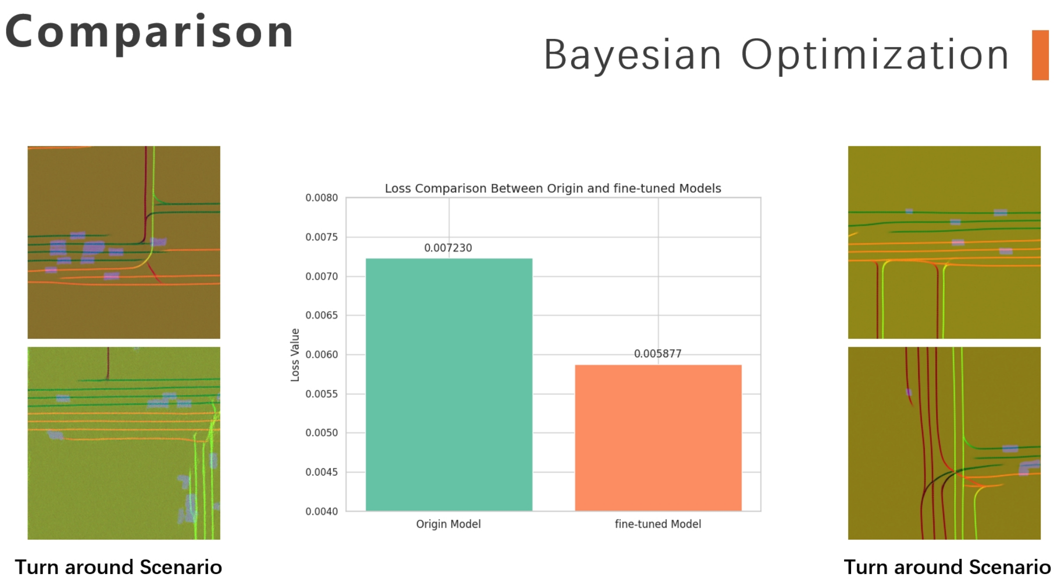


Figure 39: Experiment on Bayesian Optimization

9.0.5 Parser Demo

A demonstration of the OpenScenario Parser(figure 40) was conducted to showcase how the system can convert road scenario data into a format suitable for simulation platforms like CARLA. This demo illustrated how the parser processes the raw data, extracts relevant road features (such as lanes, vehicles, and traffic signs), and outputs it in a standardized format that can be easily utilized in simulation environments for further testing and validation.


```xml
<OpenDRIVE>
    <header name="myroad2" revMajor="1" revMinor="5" date="2025-01-11 04:48:49.588035" north="0.0" south="0.0" east=
" west="0.0"/>
    <road rule="RHT" id="0" junction="-1" length="300">
        <link>
            <successor elementType="junction" elementId="1"/>
        </link>
        <planView>
            <geometry s="0" x="0" y="0" hdg="0" length="300">
                <line/>
            </geometry>
        </planView>
        <elevationProfile/>
        <lateralProfile/>
        <lanes>
            <laneSection s="0">
                <left>
                    <lane id="1" type="driving" level="false">
                        <link/>
                        <width a="3.0" b="0.0" c="-0.0" d="0.0" sOffset="0"/>
                        <roadMark sOffset="0" type="solid" weight="standard" color="standard" height="0.02" width="0
>

                    </lane>
                </left>
                <center>
                    <lane id="0" type="none" level="false">
                        <roadMark sOffset="0" type="solid" weight="standard" color="standard" height="0.02" width="0
>

                    </lane>
                </center>
                <right>
                    <lane id="-1" type="driving" level="false">
                        <link/>
```


Figure 40: Experiment on Parser Generation

10 Conclusion

Project Recording	Status	links
Report	✅	CrashSimGen: Final ML Project - Online LaTeX Editor Sharelatex by SUSTech CRA
PPT	✅	https://www.haibinlaiblog.top/wp-content/uploads/2025/01/ML_DM.pptx
Demo	✅	https://github.com/HaibinLai/CrashSimGen/tree/main/Demo
Code	✅	https://github.com/HaibinLai/CrashSimGen
Video	✅	【机器学习 Grand Finale】 https://www.bilibili.com/video/BV1ZPcxeFEP2
Website	✅	https://haibinlai.github.io/CrashSimGen/
Poster	WIP	WIP

Figure 41: Our Project Review

CrashSimGen is a project designed to generate critical road scenarios for autonomous driving risk assessment using diffusion models.

The workflow begins by encoding a diverse set of driving scenes into image data. A dangerous scenario recognizer is then trained to automatically identify and label potentially hazardous situations from these images.

Then a diffusion model is used to trained for better critical scenario generation. The model is improved using Attention blocks, DDIM schecduler, and bayesian optimization.

After that, the model's generated picture is transformed into the openDrvie Scenario.(The scenario demo can be viewed on www.bilibili.com)

Once the dangerous scenes are recognized, their thumbnails are used as inputs for a diffusion model. This model generates new dangerous scene thumbnails, simulating critical road scenarios that can be used for evaluating the safety and performance of autonomous driving systems. The overall goal of this project is to enhance risk assessment and improve the reliability of autonomous vehicles in real-world traffic conditions.

11 Future Work

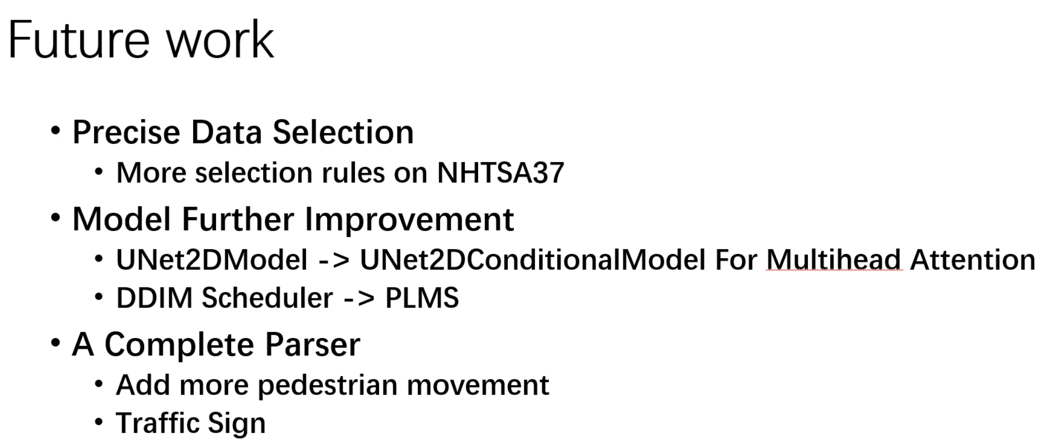


Figure 42: Our Project Future work

Data Selection and Model Improvement

In the future, we plan to refine the data selection process by introducing more selection rules on NHTSA37, ensuring the chosen data more accurately reflects a variety of hazardous driving scenarios. This step will help improve the quality and diversity of the training dataset, which is crucial for generating a wide range of dangerous road situations. Additionally, we aim to enhance the model by upgrading from the current UNet2DModel to the UNet2DConditionalModel, incorporating multihead attention mechanisms to better capture complex spatial and temporal dependencies in the driving scenes.

Advanced Scheduler and Parser Enhancements

To further improve the quality and realism of the generated scenarios, we will switch from the DDIM scheduler to the PLMS (Pseudo Likelihood Maximization Sampling) scheduler. This change is expected to yield more accurate and coherent scene generation by providing better control over the diffusion process. Additionally, we plan to enhance our parser to handle more complex road features, such as pedestrian movement and traffic signs. By incorporating dynamic pedestrian behavior and a more detailed representation of traffic signs, we aim to simulate more realistic and varied driving environments.

Expanding Simulation Scope and Integration

Our future work will focus on expanding the scope of the generated driving scenarios to cover a broader range of real-world traffic conditions. This includes incorporating more diverse pedestrian movements, which will help the model simulate real-time interactions between pedestrians and vehicles. We will also integrate traffic signs into the simulation, adding an additional layer of complexity that will be important for assessing the risk in autonomous driving. These enhancements will contribute to a more comprehensive and accurate risk assessment model for autonomous vehicle testing.

References

[1] Andreas Demetriou, Henrik Allsvåg, Sadegh Rahrovani, and Morteza Haghir Chehreghani. Generation of driving scenario trajectories with generative adversarial networks. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–6, 2020.

[2] W. Ding, B. Chen, B. Li, K. J. Eun, and D. Zhao. Multimodal safety-critical scenarios generation for decision-making algorithms evaluation. *IEEE Robotics and Automation Letters*, 6(2):1551–1558, 2021.

[3] W. Ding, B. Chen, M. Xu, and D. 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.

[4] Wenhao Ding, Wenshuo Wang, and Ding Zhao. A multi-vehicle trajectories generator to simulate vehicle-to-vehicle encountering scenarios. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 4255–4261, 2019.

[5] Wenhao Ding, Chejian Xu, Mansur Arief, Haohong Lin, Bo Li, and Ding Zhao. A survey on safety-critical driving scenario generation—a methodological perspective. *IEEE Transactions on Intelligent Transportation Systems*, 24(7):6971–6988, 2023.

[6] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572, 2014.

[7] Y. Guo, V. V. Kalidindi, M. Arief, W. Wang, J. Zhu, H. Peng, and D. Zhao. Modeling multi-vehicle interaction scenarios using gaussian random field. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 3974–3980. IEEE, 2019.

[8] Kunkun Hao, Wen Cui, Lu Liu, Yuxi Pan, and Zijiang Yang. Integrating data-driven and knowledge-driven methodologies for safety-critical scenario generation in autonomous vehicle validation. In *2024 IEEE 24th International Conference on Software Quality, Reliability, and Security Companion (QRS-C)*, pages 970–981, 2024.

[9] G. Harshvardhan, M. K. Gourisaria, M. Pandey, and S. S. Rautary. A comprehensive survey and analysis of generative models in machine learning. *Computer Science Review*, 38:100285, 2020.

[10] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020.

[11] Zhiyu Huang, Zixu Zhang, Ameya Vaidya, Yuxiao Chen, Chen Lv, and Jaime Fernández Fisac. Versatile behavior diffusion for generalized traffic agent simulation, 2024.

[12] Ethan Pronovost, Meghana Reddy Ganesina, Noureldin Hendy, Zeyu Wang, Andres Morales, Kai Wang, and Nicholas Roy. Scenario diffusion: Controllable driving scenario generation with diffusion, 2023.

[13] S. Riedmaier, T. Ponn, D. Ludwig, B. Schick, and F. Diermeyer. Survey on scenario-based safety assessment of automated vehicles. *IEEE Access*, 8:87456–87477, 2020.

[14] Shuo Sun, Zekai Gu, Tianchen Sun, Jiawei Sun, Chengran Yuan, Yuhang Han, Dongen Li, and Marcelo H. Ang. Drivescenegen: Generating diverse and realistic driving scenarios from scratch. *IEEE Robotics and Automation Letters*, 9(8):7007–7014, 2024.

[15] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT Press, 2018.

[16] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[17] Chen Yang, Yangfan He, Aaron Xuxiang Tian, Dong Chen, Jianhui Wang, Tianyu Shi, and Arsalan Heydarian. Wcdt: World-centric diffusion transformer for traffic scene generation, 2024.

[18] 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, 2020.

[19] Jiawei Zhang, Chejian Xu, and Bo Li. Chatscene: Knowledge-enabled safety-critical scenario generation for autonomous vehicles, 2024.

[20] Z. Zhong, Y. Tang, Y. Zhou, V. d. O. Neves, Y. Liu, and B. Ray. A survey on scenario-based testing for automated driving systems in high-fidelity simulation. *arXiv preprint arXiv:2112.00964*, 2021.

[21] Ziyuan Zhong, Davis Rempe, Danfei Xu, Yuxiao Chen, Sushant Veer, Tong Che, Baishakhi Ray, and Marco Pavone. Guided conditional diffusion for controllable traffic simulation, 2022.

[22] M. Zhou, J. Luo, J. Villella, Y. Yang, D. Rusu, J. Miao, W. Zhang, M. Alban, I. Fadakar, and Z. Chen. Smarts: Scalable multi-agent reinforcement learning training school for autonomous driving, 2020.