Challenge 3: Evaluating Planning and Control Modules in SafeBench

24-784 Special Topics: Trustworthy AI Autonomy

Prof. D. Zhao

Due: Friday Apr 21st, 23:59:59 EST

- Your online version and its timestamp will be used for assessment.
- We will use Gradescope to grade. The link is on the panel of CANVAS. This homework is to be completed **in a group**. Please abide by the ACADEMIC INTEGRITY POLICY on the syllabus.
- Submit your solutions in **.pdf** format to **Challenge-3**. Insert the generated images and plots in the **.pdf**. We will check all of your answers in the report.
- Please familiarize yourself with Python programming, if you have never used it previously. We only accept Python for the coding of all homeworks. To that end, attending Python recitation bootcamp at the beginning of this course is highly recommended.
- We advise you to start with the assignment early. All the submissions are to be done before the respective deadlines. For information about available credits for late days, refer to the syllabus in CANVAS.

1 Introduction

In this challenge assignment C3, you are required to form a team in 3 to tackle a real-world problem in autonomous driving.

In C2, we add patches to a stop sign to attack perception modules in C2, now we turn to the planing and control module. Planning and control modules are also important in the autonomous driving system, which usually take results of perception, such as detected obstacles and tracking of surrounding objects, and output control commands to low-level hardware of the system. These two modules may contain a lot of complicated components in real-world autonomous driving systems, so in this assignment, we only consider using a reinforcement learning (RL) algorithm to control the vehicle.

We've already seen an example that trains a model-based RL agent to solve the parking problem in Problem 2. In this assignment, we plan to train another RL method, called Soft Actor Critic (SAC) to control a vehicle in the Safebench platform.

The goals of this challenge are:

- Learn about the Carla simulator, understand the basic frameworks of SafeBench.
- Train a SAC agent to control a autonomous vehicle to follow given routes.
- Evaluate the trained agent in safety-critical scenarios.
- Compare the evaluation results of different scenarios and adjust the SAC agent to achieve better performance in safety-critical scenarios.

2 Resources

Please refer to C2 for installation of Carla and Safebench. For all the following exercises, you will use the Safebench platform as you did in C2. Note that we will use a different branch for C3, which is located in https://github.com/trust-ai/SafeBench/tree/24784_s23_c3.

3 Problem Sets

Exercise 1: Evaluate a Rule-based Agent

Figure 1: Screenshot of Safebench. The left figure is the bird-eye-view image, where the red block is ego vehicle, the green block is other vehicles or cyclists, and the blue line is the route for the ego vehicle to follow.

In Safebench, we provide a simple rule-based agent (named *Basic*) that uses a PID controller to follow a given route and uses the distance between objects to avoid obstacles. You are requested to run this agent in safety-critical scenarios with the following script and tune the parameters of the lateral PID controller and longitudinal PID controller. You can start the evaluation with the following command.

```
python scripts/run.py
    --agent_cfg basic.yaml
    --scenario_cfg standard.yaml
    --num_scenario 4
    --mode eval
    --save_video
```

You should be able to see similar window to Figure 1. The parameters of PID controllers can be found in `Safebench/safebench/agent/config/basic.yaml`. You can change the following parameters to make the vehicle stably follow the given trajectory (shown in blue color). There is no numerical metric for this question, thus you just need to make the vehicle have reasonable behavior.

Parameter Name	Description
d_t	timestep interval
lateral_KP	parameter P of the lateral controller
lateral_KI	parameter I of the lateral controller
lateral_KD	parameter D of the lateral controller
longitudinal_KP	parameter P of the longitudinal controller
longitudinal_KI	parameter I of the longitudinal controller
longitudinal_KD	parameter D of the longitudinal controller
max_steering	max value for the steering action
max_throttle	max value for the throttle action

What to submit

Please find the videos in folder `Safebench/log/exp/exp_basic_standard_seed_0/video`. Then, you can find multiple .mp4 files under the folder named by the timestamp of the experiment. Since we run 4 scenarios in parallel in each batch, the name of the videos are in this format: `video_n_id_a_b_c_d.mp4`, where n is the number of iteration, a, b, c, d are the scenario id. You should have 8 files in total if 32 scenarios are finished. Please upload all videos to google drive and insert the link into the submitted .pdf.

Please also report a table to show the final evaluate results (output of the last batch). The table should include

Evaluation Metric	Value
collision_rate	
out_of_road_length	
distance_to_route	
incomplete_route	
running_time	
final_score	

Exercise 2: Train a SAC Agent in Ordinary Scenarios

Now you have seen the performance of a rule-based agent in safety-critical scenarios. Let's try to train a SAC agent that can also follow given routes. During the training, we use ordinary scenarios, where a lot of surrounding vehicles are controlled by auto-polit. You can start the training with the following command.

```
python scripts/run.py
    --agent_cfg sac.yaml
    --scenario_cfg ordinary.yaml
    --num_scenario 4
    --mode train_agent
```

Hyper-parameters of SAC agent is located in `Safebench/safebench/agent/config/sac.yaml`. The default parameters should be good enough to train the agent. The trained model will

be stored in folder `Safebench/safebench/agent/model_ckpt/sac`. During the training, we can use the episode reward to check the learning performance, which is calculated by:

$$R = \sum_{t=0}^{T} r_t, \tag{1}$$

where r_t is the reward for each step. We store episode reward as a python dictionary into the file `Safebench/log/exp/exp_sac_ordinary_seed_0/training_results/results.pkl`. The content of the dictionary is {`'episode':List, 'episode_reward':List`}.

What to submit

Please plot the episode reward during training using the store results in `results.pkl`. The x-axis is the training episode and the y-axis is the episode reward. The final reward should be around 1200. The figure you obtain will be similar to Figure 2.

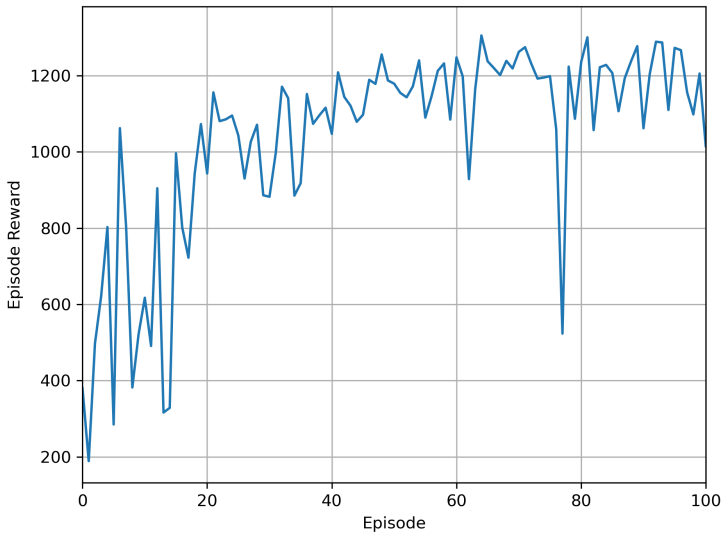

Figure 2: Episode reward figure for SAC agent training. The maximium reward should be around 1200.

Exercise 3: Evaluate the Trained Agent in Safety-critical Scenarios

Since we train the SAC agent in ordinary scenarios, the agent may not perform well in safety-critical scenarios. You can use the following command to evaluate the performance of trained agent in safety-critical scenarios.

```
python scripts/run.py
    --agent_cfg sac.yaml
    --scenario_cfg standard.yaml
    --num_scenario 4
    --mode eval
    --save_video
```

What to submit

Similar to the requirement to Exercise 1, please upload the videos to google drive and insert the link into the submitted .pdf.

Please also report a table to show the evaluate results based on the final output from Safebench. Compared with the results obtained by the rule-based agent, what conclusion can you get? Please write a short answer in the .pdf.

Bonus: Attending the Safebench competition at CVPR

(Up to 10 points of the whole course)

Register Safebench Challenge (2 points)

We are hosting a chllenge in CVPR 2023 using the Safebench platform (Official Website). Since you are already familiar with this platform after doing both C2 and C3, you are welcome to register the challenge and submit your results. We are now collecting the information of participation. Please fill in this form to register. We will release the instruction of result submission soon.

Achieve better performance in safety-critical scenarios (8 points)

As you can see from the results in Exercise 1 and Exercise 3, training a safe autonomous vehicle is not a easy task. In this bonus question, you need to design a planning and control agent (not necessarily a RL agent) to achieve as high final score as possible in safety-critical scenarios. We will evaluate your algorithm based on the performance and novelty. You may get up to 8 points of the whole course.

What to submit

Submit the solution to the Gradescope. Upload the source code and evaluation videos to google drive and insert the link to the .pdf.

The submission deadline the competition is June 1st, 2023. You may still keep improving your performance even after the conclusion of this course. Awards with cash will be given to the competition winners.

4 Reference

1. AWS setup, https://www.youtube.com/watch?v=Yn-w3sfJil4
2. SafeBench code: https://github.com/trust-ai/SafeBench/tree/24784_s23_c3
3. SAC: https://arxiv.org/abs/1801.01290
4. Safety-critical Scenario Generation: https://ieeexplore.ieee.org/document/10089194