CTF 2 - Report

Question 1: Overview of CTF 2 Test

- **What did you do? What experiments did you perform based on your homework 2 assignment?**
  - **Training AWS-DeepRacer Model:** For our Homework 2 assignment, our team conducted experiments using AWS DeepRacer-trained models. The best object avoidance model submitted was trained for approximately 3 hours and featured both LiDAR and Stereo Camera on board. Notably, these models exhibited strong performance in a virtual environment, effectively completing the track within the training period. However, challenges emerged when deploying the models physically, revealing discrepancies between simulation and reality.
  - **Stop-sign Detection DL Model custom training on a dataset:** We collected a dataset from Caltech to do a custom training of VGG16 from Tensorflow using bounding box regression. The plan was to load this object detection model on the locally set Follow the Leader project in CTF 2.
  - **Preparing our model to detect and stop upon seeing stop signs:** We studied the COCO dataset and the prediction labels of OpenVINO IR of the ssd_mobilenet_v2_coco object detection model. Then, we tried to install the aws-deepracer-follow-the-leader-sample-project locally. However, we could not install it locally due to many technical issues. Later, we studied the object_detection_pkg to prepare our trained DeepRacer model to detect stop signs. Upon seeing and detecting a stop sign, the DeepRacer will just stop.
  - **Generating Adversarial Images for other Red Team models:** We generated adversarial examples using the Fast Gradient Signed Method (FGSM) attack described in Explaining and Harnessing Adversarial Examples. We generated various adversarial images from one image based on the noise parameter tuning. Our goal was to put the generated images from our laptop in front of other Red Team models to check whether the models got fooled by the adversarial images and reacted abnormally.
  - **Building the track:** Our team also contributed to building the track for running the Deepracer in a physical environment for the CTF.

❖ **What worked and why?**

- ➢ **Training Duration:** Models trained for longer durations (approximately 3 hours) performed better. This aligns with the general understanding in machine learning that longer training periods often lead to improved model performance.
- ➢ **Sensor Configuration:** Models with LiDAR and Stereo Camera on board performed well. This suggests that the combination of these sensors contributed to the model's ability to navigate the track effectively.
- ➢ **Virtual Environment Performance:** The model performed well in the virtual environment, completing the track effectively. This demonstrates the effectiveness of the training process in a controlled setting.
- ➢ **Object Detection:** The model designed for object detection worked somewhat when deployed on the track with objects. This indicates that the model was able to detect and navigate around obstacles.

❖ **What didn't work and why?**

- ➢ **Sensitivity to Track Changes:** Even small changes in the track design significantly impacted the Deepracer's performance. This suggests that the model might be sensitive to environmental variations, and adapting to these changes is challenging.
- ➢ **Virtual vs. Physical Environment Discrepancies:** The model's success in the virtual environment did not directly translate to success in the physical environment. This could be due to the inherent differences between simulation and reality, emphasizing the importance of testing in the actual environment.
- ➢ **Unidirectional Training:** The model trained for counter-clockwise movement in the virtual environment exhibited a limitation in the physical scenario, being able to run counter-clockwise only. This indicates a lack of generalization or adaptability in the model's training.
- ➢ **Short Training Duration Impact:** The model trained for only 1 hour had noticeable performance issues, struggling with the first turn in the track. This emphasizes the importance of sufficient training time for achieving optimal performance.

Question 2: Based on your experiments and what you initially proposed in your homework 2 assignment, what do you recommend for future CTF scenarios?

There is some room for improvement in the future AWS DeepRacer CTF. The suggestions include:

- Model Training: During the CTF event, we realized that the models trained for a brief period on our DeepRacer weren't sufficiently effective for classroom use. A possible alternative to the current model training approach is to encourage implementing a local training environment. Although this might be complex, motivating students to establish their local setups seems beneficial. This would enable them to troubleshoot and refine their models more efficiently. Of course, it would also help if we obtain more student credit for model training. Many of us hesitated to experiment with various training implementations due to the concern of depleting the credits.
- Adversarial Perturbations and object misclassification: AWS DeepRacer primarily relies on a single front-facing camera, the primary sensor for perception and decision-making. It captures images of the environment, and these images are processed by the model to make decisions about how the car should navigate the track and react to a particular incident. Competitors can craft adversarial inputs to fool the perception system by introducing subtle perturbations to the camera input. The goal should be to make the DeepRacer misinterpret the environmental obstacles, leading to incorrect decisions. The DeepRacer must demonstrate robustness against these attacks.
- Gradient occlusion and beam manipulation: AWS DeepRacer, a simplified and educational platform, lets learners use lidar to make decisions based on sensor readings. It provides a high-resolution 3D view of the surroundings. Modifying the lidar beams to create phantom obstacles or removing existing obstacles will challenge the DeepRacer while making decisions based on potentially altered spatial information. Also, the model's ability to filter out irrelevant information can be tested by simulating interference or noise in the lidar signals. Deepracer should demonstrate resilience to environmental disturbances and accurately represent the surroundings.

Depending on the complexities of the above CTF scenarios, it can be a tricky and very challenging task. But designing such scenarios and implementing a robust DeepRacer model will also present a unique and convivial opportunity for the participants.

Question 3: Team member contributions

All of us worked collaboratively on this CTF. Only our team members built the track before and on the CTF day. We have utilized the AWS DeepRacer to train our model for object detection. While training the model for object detection, we tried to use OpenVino, but the resources were outdated, so it didn't work for us. So, instead of using OpenVino, we opted for the Tensorflow Object Detection API to train the model for object detection successfully. Also, we trained the VGG16 model on a custom dataset of stop signs using a bounding box regression algorithm. We employed the Fast Gradient Signed Method (FGSM) to generate adversarial images and deployed the MSF-ADV as our adversarial attack method. We used anaconda and ensured all the necessary conditions were met. We aimed to assess the vulnerabilities in the Multi-Sensor Fusion (MSF) of autonomous driving (AD) perception, ultimately enhancing the system's robustness and avoiding potential crashes.

Due to time limits, our team could not implement more scenarios for which we prepared, but the one we implemented was a team effort.