



GUARANTEEING AI ROBUSTNESS AGAINST DECEPTION

Multi-Perspective, Spatiotemporal-Coherent, and Multi-Modal Aware Object Detection for Adversarial Robustness

Phase 3 Final Report

May 30th, 2024

Legal Organization: Intel Federal LLC	
Organization Type: Large Business	
Agreement Number: HR0011203001	
Technical Point of Contact:	Cory Cornelius
Title:	AI Research Engineer/Scientist, Intel Labs
Unclassified Email:	cory.cornelius@intel.com
Phone:	(503) 712-4160
Company Name:	Intel Federal LLC
Address:	4100 Monument Corner Drive, Suite 540
City, State, Zip:	Fairfax, VA 22030
Contracts Point of Contact:	Jannine Rhee
Title:	Contracts Manager, Intel Federal
Unclassified Email:	jannine.rhee@intel.com
Other Team Member:	Duen Horng (Polo) Chau
Title:	Associate Professor, College of Computing, Georgia Tech
Unclassified Email:	polo@gatech.edu
Phone:	(404) 385-7682

Distribution Statement A. Approved for public release: distribution is unlimited.

Copyright © 2024, Intel Corporation and Georgia Institute of Technology. Intel®, the Intel® logo, and others are trademarks of Intel Corporation in the U.S. and other countries. Other names and brands may be claimed as the property of others.

Contents

1	Phase 3 Executive Summaries	1
1.1	Q3 2022 Update	2
1.2	Q4 2022 Update	3
1.3	Q1 2023 Update	4
1.4	Q2 2023 Update	5
1.5	Q3 2023 Update	6
1.6	Q4 2023/Q1 2024 Update	7
2	Phase 1 & 2 Summaries and Phase 3 Plan	8
2.1	Phase 1 Summary	8
2.2	Phase 2 Summary	9
2.3	Phase 3 Plan	10
2.3.1	Promoting Robust Features and Representations	10
2.3.2	Refining the Physically Plausible Threat Model	11
2.3.3	Tasks	12
3	Phase 3 Technical Progress	13
3.1	Q3 2022 Update	14
3.1.1	Overhead Detection Dataset Exploration	14
3.1.2	Robust Architectures	15
3.1.3	Differentiable Rendering	16
3.2	Q4 2022 Update	19
3.2.1	Uni-modal Overhead Object Detection	19
3.2.2	Multi-modal Overhead Object Detection	21
3.2.3	Multi-object Overhead Object Tracking	23
3.2.4	Modular Adversarial Robustness Toolkit	25
3.3	Q1 2023 Update	27
3.3.1	Diffusion Priors for Object Detection	27
3.3.2	Robust Architectures	31
3.3.3	Modular Adversarial Robustness Toolkit	33
3.4	Q2 2023 Update	36
3.4.1	Score-Based Generative Modeling for Adversarial Defense	36
3.4.2	Background Subtraction Defense for Object Tracking	42
3.4.3	CARLA Data Collection Tools	44
3.4.4	Modular Adversarial Robustness Toolbox	46

3.5	Q3 2023 Update	48
3.5.1	White-box adversarial robustness of score-based density estimation	48
3.5.2	Evaluating and designing adversarial patches	51
3.5.3	Improving and combining defenses for the final evaluation	55
3.5.4	LLM Self Defense: By self examination, LLMs know they are being tricked	59
3.6	Q4 2023/Q1 2024 Update	61
3.6.1	Simulation in the Attack Loop	61
3.6.2	Procedural Camouflage Attack	65
3.6.3	Semantic Attacks on CLIP-based Anomaly Segmentation	65
3.6.4	Release of Score-based Diffusion Models and Attacks	70
3.6.5	LLM Self Defense	71
4	Phase 3 Significant Accomplishments	73
4.1	Graded Q4 2022 Accomplishments	73
4.2	Graded Q1 2023 Accomplishments	73
4.3	Graded Q2 2023 Accomplishments	74
4.4	Graded Q3 2023 Accomplishments	74
4.5	Graded Q4 2023/Q1 2024 Accomplishments	74
5	Phase 3 Publications	76
5.1	Q3 2022 Update	76
5.2	Q4 2022 Update	76
5.3	Q1 2023 Update	76
5.4	Q2 2023 Update	76
5.5	Q3 2023 Update	77
5.6	Q4 2023/Q1 2024 Update	78
6	Phase 3 Meetings and Events	79
6.1	Q3 2022 Update	79
6.2	Q4 2022 Update	80
6.3	Q1 2023 Update	80
6.4	Q2 2023 Update	82
6.5	Q3 2023 Update	82
6.6	Q4 2023/Q1 2024 Update	85
	Bibliography	86

1 Phase 3 Executive Summaries

In Phase 3, we will fully utilize simulation and our synthetic dataset tools to construct and understand our robust features and representations approach. Synthetic data enables us to independently collect multi-modal data from simulation, while our auto-labeling tools enables us to construct multi-task annotations. Together multi-modal data and multi-task annotations enable implement our robust features and representations. Simulation also enables us to instantiate physically plausible attacks. This enables us to refine our physically plausible threat models and understand our defenses on a more level playing field. In the following sections, we summarize our progress for each quarter.

1.1 Q3 2022 Update

In the 3rd quarter of 2022, we spent time understand the 6th evaluation datasets. We found that these datasets are missing annotations with less than 100 pixels of area, which includes a non-trivial amount of pedestrian and vehicle annotations. We were able to use our auto-labeling tools to re-generate these “missing” annotations. We are currently training our defenses on the “full” annotations and continue to spend time understanding the 6th evaluation datasets and scenarios.

We also spent time understanding how architectural choices affect adversarial robustness. Through the examining and combining these architectural choices, we adversarially trained 150 models and created guidelines for designing adversarially robust architectures. This work resulted in a new state-of-the-art result on the RobustBench ImageNet leaderboard for $\ell_\infty = 4/255$ untargeted attacks. This work is under submission to CVPR 2023.

Finally, we continue to examine differentiable rendering techniques. We continue to invest in differentiable rendering because it enables us understand how one might compute physically plausible depth attacks. Differentiable rendering enables us to compute attacks on the mesh or shape, as opposed to texture or color, of an object. We simplified this problem by only considering mesh/shape attacks on a binary classifier. Our results to date, however, yield infeasible mesh perturbations, and so we plan to continuing exploring how to make these perturbations more physically realizable.

1.2 Q4 2022 Update

In the 4th quarter of 2022, we released our Modular Adversarial Robustness Toolkit (MART) as open source on Github under the BSD-3 license:

<https://github.com/IntelLabs/MART>

We used MART to train most of the models used in our 6th GARD evaluation submissions, as described in Section 3.2.4. We plan to continuing using it going forward both as a means to develop and train models and to evaluate the robustness of said models.

We also submitted 3 defenses to the 6th GARD evaluation. Our defenses covered three scenarios: uni-modal object detection, multi-modal object detection, and multi-object tracking. Our uni-modal object detector learned class-specific anchors that bias the Faster R-CNN object detector to only detect objects in places they were seen in the training set. We found this improves the benign performance of the model and marginally increases the robustness of the model. We describe this submission in more detail in Section 3.2.1. Our multi-modal object detector exploited our observation that the depth modality is more robust than the image modality and even the fusion of image-depth modalities. See Section 3.2.2 for more details. Finally, our multi-object tracker combined pre-processing filters with our prior work on homography-based background subtraction. We found that the application of our approach in the 5th GARD evaluation does work for the 6th GARD evaluation scenarios with a minor modification that takes into account the moving camera perspective. We describe the results of this submission in Section 3.2.3.

More generally, we found that new ideas are needed for defenses against the scenarios in the 6th GARD evaluation. These scenarios presented new challenges for our defenses, since most objects in the images and videos were smaller than the adversarial patches. Small objects are generally difficult to detect and track even in benign cases. The adversarial patches in 6th GARD evaluation surround many of these small objects, which gives more control to the attacker than in prior evaluations. However, since these patches do not overlap these objects, we believe it should still possible for defenders to correctly detect and track objects even in the presence of these background patches. Similarly, we found that the patches in the tracking scenario were dynamic in the sense that their location jitters despite being static patches. Finally, we observed that subtlety in the patch threat model for depth require further validation in order to ensure the implementation matches the desired threat model. As such, we plan to continue exploring additional techniques to mitigate this threat and improve the evaluation of these scenarios.

1.3 Q1 2023 Update

In the 1st quarter of 2023, we began an effort into robust, learned priors via generative models. Score-based diffusion is a powerful model that learns the gradient of the training data distribution, or score, unlike other methods that directly or implicitly learn the probability density function. After training said model, not only does this class of generative model allow us to sample new images, score-based diffusion models also enable us to evaluate the likelihood of any other image. These likelihoods enable us to detect anomalous inputs that do not match the training data distribution. In Section 3.3.1, we describe our efforts to train score-based diffusion models on the 6th GARD evaluation CARLA overhead dataset and how we use that model to detect perturbed regions in unseen samples that contain adversarial patches.

We also surveyed state-of-the-art architectures to understand how different architectural components affect robustness when adversarial training. More specifically, we found four architecture components that affect robustness, including three new components: the depth-width ratio, the type of input stem, and the use of squeeze-and-excitation blocks. We found there exists an optimal range of depth-width ratios that improve robustness. Similarly, we found that an overlapping convolutional input stem, which the ResNet family of architectures use, yields higher robustness than a non-overlapping one, which the Transformer-family of architectures use. Finally, we find that larger reduction ratios in squeeze-and-excitation blocks reduce robustness. We also confirm previous findings that non-parametric smooth activations consistently improve robustness. We describe these findings in more detail and how they generalize to different robust training procedures in Section 3.3.2.

Lastly, we also spent time adding new features and capabilities to our Modular Adversarial Robustness Toolkit (MART). We added the ability to generate static, universal adversarial patches. Such patches are static in the sense that the adversary cannot change it after deployment, and they are universal in the sense that they must work under different scenario configurations. This capability was necessary for a recent demo we gave at Intel Vision 2023. In our demo, attendees stood in front of adversarial patches that we designed to achieve different adversarial goals like hallucination objects, turning people into other objects, and suppressing objects. The setup forced us, the adversary, to generate patches *a priori* with only some knowledge of the placement of the patch and very limited knowledge of the rest of the scenario. In contrast to the existing scenarios, we believe this scenario represents a more realistic scenario that forces the adversary to generate patches that generalize across different scenario configurations. Adversaries do not control every aspect of a given scenario and thus must generate perturbations that survive those uncontrolled things, like lighting and relative placement of the patch. We describe these and other updates to MART and our Intel Vision 2023 demo in Sections 3.3.3 and 6.3.

1.4 Q2 2023 Update¹

In the 2nd quarter of 2023, we made several key advances in developing robust defenses for object detection and tracking. First, we submitted two novel defenses to the 7th GARD evaluation. For object detection, we leveraged generative models that we trained to estimate the likelihood of detections and filter adversarial hallucinations. This approach significantly reduced hallucinations compared to undefended models, though it increased the disappearance rate. For tracking, we optimized a background subtraction technique that utilizes homography to stabilize patch coordinates across frames. This allowed us to simplify our overall defense architecture. We describe these efforts in Sections 3.4.1 and 3.4.2.

A major enabler for training our generative defenses was the massive dataset we collected using our custom CARLA data collection tools, which we released as open source. We generated over 1.5 million annotations across 90,000 images—an order of magnitude more data than the provided training set. This diverse, large-scale dataset was critical for adequately training our own models. For more details see Section 3.4.3.

We continue to expand the capabilities of our Modular Adversarial Robustness Toolkit (MART). Our goal with MART is to serve as a training framework and alternative evaluation framework that supplements Armory and aids GARD TA 2 evaluators. We added configurable attacks, adversarial training features, and other improvements to increase the usability of MART. We outline these new features in Section 3.4.4.

We have also been assisting TwoSix’s engagement with West Point on developing physically realizable patches to fool object detection models. By drawing from our own experience with our ShapeShifter research [CCMC18], we shared best practices like ensuring high digital attack success rates before fabrication and using low-glare paper for printing. We also showed them how we use 3D simulation tools, like Blender, to refine patch placement strategies, prior to printing patches. We describe these efforts in Section 6.4.

Moving forward, we aim to enhance our work along multiple fronts. We identified several issues that we have shared with the GARD TA 2 evaluators: foliage occlusion problems in CARLA that cause inconsistencies in annotations and unnecessary memory usage in the Adversarial Robustness Toolkit (ART). We hope to resolve these issues, and thus enable more accurate benchmarking of adversarial scenarios in CARLA. We also plan to perform a white-box analysis of our generative models to understand their robustness against gradient-based attacks. Similarly, we plan to enable the re-insertion of adversarial patches into CARLA scenes and further increasing the modularity and ease-of-use of MART. We look forward to building on these advancements and collaborating closely with the GARD TA 2 evaluators in next and final evaluation.

¹This executive summary was written with the help of Claude, an AI assistant created by Anthropic.

1.5 Q3 2023 Update

In the 3rd quarter of 2023, we examined the robustness of the likelihood-estimation method we used in our 7th GARD evaluation submission. Our results show that generating samples from a diffusion that have simultaneous high likelihood and high complexity do not exist, even in a worst-case adversarial setting. The details of this work can be found in Section 3.5.1 and will appear as a poster at the NeurIPS 2023 Workshop on Diffusion Models.

In Section 3.5.2, we describe our efforts to evaluate and design more realistic adversarial patches using score-based diffusion models with classifier guidance and differentiable rendering. We found that the existing attacks in Armory are not effective when re-inserted and rendered the CARLA simulator. This indicates that we need more robust attacks that survive this digital-to-physical/simulation gap. We describe two approaches we took this quarter: attacks guided by score-based diffusion and attacks using differentiable rendering. These methods generate adversarial patches that appear more natural, which we believe will survive this gap. We are currently working to understand whether these methods do survive this gap.

For the 8th GARD evaluation, we submitted two defenses—one for the uni-modal object detection scenario and another for the multi-modal object detection scenario. Our approach for the final evaluation was to combine our existing defenses that we submitted in prior evaluations with the hope that the sum would yield robustness greater than the parts. As such, we combined an adversarially trained object detector with our likelihood estimation post-processor. Our adversarial trained detector uses an improved training method that we adapt to the patch threat model. We also fixed an issue that generally affected object detection models. Many detection models hard limit the number of detections they output, which enables an adversary to induce a “denial of service” attack by generating many high-confidence hallucinations. Overall, we find this combination significantly reduces hallucination rates, but we also find a trade-off between misclassification rate and disappearance rate. Defensive methods that overcome this trade off remains an open problem. The detail of these results can be found in Section 3.5.3.

Finally, our Adversarial AI Photobooth continues to be a hit demo. We updated it to include a diffusion-based defense that re-samples the input in real-time. This improved demo was featured as one of five “tech experiences” at Intel Innovation 2023. We attracted several hundred people, including some press, across two days. Many participants were delighted by to see themselves through the eyes of a diffusion model and quickly understood the efficacy of the diffusion-based defense. Photos of our tech experience at Intel Innovation 2023 and examples of this re-sample defense can be found in Section 6.5.

1.6 Q4 2023/Q1 2024 Update

This summary marks the end of Intel’s and Georgia Tech’s effort in the GARD program and serves as a reflection on our progress. Our pre-GARD work showed it was possible for adversarial attacks to affect an actual object detection system using real adversarial traffic signs [CCMC18]. Contrary to existing work of the time, our method did not follow the literature’s tendency to focus on small ℓ_p -bounded perturbations. Instead we recognized that what matters is whether the output from a model, in our case an object detector, aligned with human perception (Figure 5 in [CCMC18]). We started the program not knowing whether these ideas around physically realizable adversarial attacks and the proper evaluation of them would become a part of GARD.

In GARD, we pushed to evaluate adversarial attacks under this “physically realizable” threat model, where: 1) attackers do not precisely control the input to a model and thus must generalize to environmental effects; and 2) attackers do not comply with small ℓ_p -bounded perturbations but are often bound to some area in the real world (i.e., the “patch” threat model). To model these threats, we advocated for the use of simulation, where the simulator could be thought of as a proxy for the “real world.” Not only did simulation enable the GARD TA2 evaluators to create physically realizable scenarios, but it also enabled them, and us, to collect datasets and effectively evaluate defenses under the physically realizable threat model. This resulted in the use of CARLA [DRCLK17] to construct scenarios, collect datasets, and evaluate defenses as described in our collaborative paper with a GARD TA2 evaluator MITRE [Liu+22]. We officially open-sourced these tools and our models under the BSD-3 license on GitHub [Cor+24] as described in Section 3.6.4.

Physically realizable evasion attacks remain difficult to defend against. Our work made progress by leveraging alternative sensing modalities. The intuition here is simple: an attacker that controls one modality, like intensity values, cannot affect other modalities like depth values. We found that per-pixel depth can be robust to realistic depth perturbation budgets via adversarial training. We also found that training generative models on multi-modal data increases robustness [ACMH23]. Properly modeling and understanding the limits of physical attacks in other modalities remains an interesting open problem.

Our final effort closes the loop and returns to understanding physically plausible adversarial attacks. We describe in Section 3.6.1 how an attack can overcome the non-differentiability of CARLA. We do this by integrating the simulator into the attack loop itself and thus construct adversarial attacks that remain adversarial in simulation. In Section 3.6.2, we demonstrate how to construct adversarial attacks that fool both humans and machines. Finally, we describe in Section 3.6.3 how we apply our learning’s from GARD to understand the limits of a foundation model, where we evaluate a zero-shot anomaly detection use case. We hope these efforts can show what attacks are possible and can serve as the inspiration for future DARPA programs.

2 Phase 1 & 2 Summaries and Phase 3 Plan

2.1 Phase 1 Summary

In Phase 1, our work focused on how novel input representations impact model robustness. We explored this hypothesis by focusing on the UCF101 Video Action Recognition (UCF101) scenario, where we submitted several scenario defense configurations that attempted to understand this hypothesis. We chose the UCF101 scenario because it forces models to understand how people and objects interact through the lens of activity recognition.

To understand our hypothesis, we created multiple spatial coherence approaches that constrained the input via background segmentation and ablation. We also performed experiments on the temporal coherence properties exhibited by the baseline UCF101 model, Motion-Augmented RGB Stream for Action Recognition (MARS). We did not design the background ablation or instance segmentation model feature extractors to be robust. In fact, our results showed this component of our defenses to be the weakest link. We attempted to make this component, an instance segmentation model, robust by using randomized smoothing. We believe that robust optimization combined with our feature extraction techniques should yield higher end-to-end robustness, which we explored further in the second phase of the GARD program.

We also explored keypoint-based representations of the human actions in UCF101. To do this, we used a Graph Convolutional Network (GCN) to model skeletal semantic- and temporal-coherence. Prior work showed that skeleton-based representations can be a robust representation of the human body because it is less susceptible to manipulation when compared to manipulating pixels. Our defense combined a keypoint detection model with a GCN to track the spatio-temporal movement of human actions in UCF101 to evaluate this hypothesis.

Independent of our submissions, we also explored the Adversarial Patches Rearranged In Context (APRICOT) object detection scenario. We found it difficult to create defenses that remain faithful to the spirit of APRICOT object detection scenario. Because APRICOT does not enable adaptive attacks, simple pixel manipulations often reduced the efficacy of the adversarial patches. Ultimately, simulation replaced the approach taken with APRICOT because simulation enabled adaptive, physically realizable attacks.

Finally, our Phase 1 explorations were all dependent on the supplied datasets and Armory infrastructure, and we found early integration with these scenarios was key. For example, we updated many of our results to account for a distribution shift found we found in Armory’s use of the UCF101 dataset. We also encountered a few challenges with the APRICOT dataset, both in terms of labels and in terms of our inability to run adaptive attacks on that threat model.

2.2 Phase 2 Summary

In Phase 2 of GARD, we took a two-pronged approach towards understanding robustness. Firstly, we invested in synthetic data generation from the CARLA simulator; and, secondly, we enhanced our robust features approach to support multiple modalities.

Simulation enables practitioners to tailor datasets to their needs. To enable this, we created a data collection tool that creates datasets from YAML configuration files specifying what data to collect from the CARLA simulator. We also created a data annotation tool that extracts ground truth annotations from these collected datasets. To improve the utility of these annotations, we implemented an instance segmentation sensor for CARLA that was released in the 0.9.13 version of the simulator. This sensor assigns a unique identifier to every object in the simulator, thus enabling our tools, and other tools, to disambiguate between overlapping objects of the same class. To ease scaling of data collection, we put our tools and the CARLA client into a Docker image. Together this Docker image and scenarios configuration files enabled the GARD evaluators to create three new scenarios: unimodal object detection, multi-modal object detection, and object tracking. They also enabled us, and other performers, to collect auxiliary data and annotations from these scenarios. For example, we extracted keypoint annotations from the object tracking scenarios and examined whether optical flow might be a useful modality.

Simulation also levels the playing field between attackers and defenders. With cyber-physical systems, these systems use sensors to sense the world. Many attack methods, however, occur in the digital world, thus giving them an unfair advantage because they bypass the sensing mechanism. While there is some work, including our own, describing how to make these attacks real, physical things, such attacks remain difficult to execute because they usually require fabrication. More so, attackers usually have limited control of the real world, and thus need to make their attacks generalize to environmental effects like lightning, occlusion, and sensor movement just like defenders already have to do.

To enable more realistic evaluation of attacks, we packaged the GARD evaluators’ attack assets into a Docker container. This enabled us to show how to use our data collection tool to dynamically modify the texture of attack patches via YAML configuration. These techniques enable future evaluations to understand the efficacy of attacks (and defenses) against real world-like effects. The

substitution of simulation for the real world has proven to be a cost-effective way of leveling the playing field between attackers and defenders and has enhanced our collective understanding of adversarial examples.

Finally, we also enhanced our robust features approach to include multi-modal inputs. We extended the ideas we developed for the UCF101 action recognition task and showed how those defensive ideas could be extended to work on other tasks like object detection and object tracking. These new defenses exploit specific features of each scenario, whether it is the use of a static camera in the object tracking scenario or the use of the depth modality to localize objects. Our defenses pushed the evaluators to modify these scenarios to change these features to make it more difficult for our defenses to succeed. We even modified our own tools to “break” our defenses. For example, we added support for multiple pedestrians and camera movement to better understand our object tracking defense. Together our tools, our defensive ideas, and attack capabilities set the stage for a productive Phase 3.

2.3 Phase 3 Plan

In Phase 3, we plan to fully utilize simulation and our synthetic dataset tools to construct and understand our robust features and representations approach. Synthetic data enables us to independently collect multi-modal data from simulation, while our auto-labeling tools enables us to construct multi-task annotations. Similarly, simulation also enables us to better understand physically plausible attacks.

2.3.1 Promoting Robust Features and Representations

Multi-modal inputs enable models to extract modality-specific features that are, ideally, independent of the other modality. For example, the depth modality is useful for locating objects, while the image modality is useful for classifying those objects. Our semantic fusion technique combines these modality-specific features in a robust manner under a particular threat model (i.e., patch attacks targeting the image modality). We plan to extend our semantic fusion technique to mitigate stronger threat models (e.g., patch attacks targeting both the image and depth modalities). Our insight is that the depth modality enables us to make more assumptions about the signal (e.g., depth is locally smooth, and discontinuities represent object boundaries) that we can encode into the model itself. This contrasts with the image modality, where it is difficult to make any assumptions about the signal. For example, depth is correlated with the area of an object since objects further away are generally smaller in area. We plan to encode this type of spatial assumption into our models.

We also recognize it is not always the case that systems will be capable of multi-modal sensing.

For these cases, we plan to leverage multi-task learning. Multi-task learning enables models to learn features that are discriminative for each task. We previously leveraged multi-task learning when training object detection models, which are designed to both localize and classify objects. We plan to extend this work to include spatial tasks like depth estimation and object-part classification, and temporal tasks like flow estimation and object tracking. Our hypothesis is that optimizing a model to learn features that are discriminative for independent tasks makes those features more robust to attacks. We plan to add more modalities to the CARLA simulator to support this effort.

We believe that the combination of multi-modal inputs with multi-task learning makes models robust. In Phase 3 we plan to explore the integration of these approaches more completely using the tooling we developed in the previous phase. To understand whether these methods inspire robustness, we plan to continue comparing our methods with adversarial training. Our hope is that our methods will yield models with similar or better robustness than adversarial training.

2.3.2 Refining the Physically Plausible Threat Model

Our experience with creating physically plausible attacks has taught us the difficulty of generating digital attacks that have efficacy in the real world. Physically fabricating digital attacks is not only time consuming but they are also difficult to test and understand. For example, we cannot easily control the real world to understand all the factors of variation that might affect the efficacy of an attack, and we were often left guessing as to why a physical attack failed. Simulation, however, enables us to lower this cost of evaluating these attacks by substituting the real world with simulation. It also enables us to control the environment to better understand the efficacy of these attacks. Thus, the use of simulation enables us to cost effectively explore physically plausible threat models.

While the use of simulation enables the exploration of more threat models, we also believe it is important to align metrics with these new threat models. Our work in the previous phase found that some metrics can be misleading under specific threat models. For example, hallucinations that occur under a patch threat model are not accurately captured by a simple count of hallucinations. It will always be possible for an adversary to generate hallucinations where the adversary over the area of the patch (i.e., where the adversary directly controls the input). However, hallucinations that occur away from the patch are cause for more concern since this better aligns with what a human would deem a hallucination. Disentangling the hallucination metric into these two metrics (near and away from patch) will help us better understand the robustness of our methods. We suspect a similar analysis is necessary for other metrics and threats.

Finally, we believe it is important to co-develop defensive capabilities in-line with offensive capabilities. We have found that investing in attack capabilities has better positioned us to create robust defenses. Our understanding of attacks on multi-modal inputs is currently limited. As such,

we plan to investigate the physical plausibility of digitally generated depth attacks. For example, it is non-trivial to construct a 3D mesh from a digital depth perturbation. We expect, however, that understanding how to construct such perturbations will yield defensive insights about robust features of the depth modality.

2.3.3 Tasks

We formally define our Phase 3 tasks as follows:

- T3.1** Expand synthetic dataset tools to extract additional modalities.
- T3.2** Investigate spatial and temporal relationships between modalities.
- T3.3** Create improved adversarial training baseline models utilizing additional modalities.
- T3.4** Understand the efficacy of multi-modal, multi-task learning against attacks.
- T3.5** Create physically plausible attack for a non-image modality.
- T3.6** Define threat-model aware and tasks-specific robustness metrics.
- T3.7** Submit defenses to evaluations and continue to self-evaluate.

3 Phase 3 Technical Progress

In these sections we report quarter-by-quarter progress towards the Phase 3 tasks as described in Section [2.3.3](#).

3.1 Q3 2022 Update

The follow sections describe our technical progress in the 3rd quarter of 2022. Our technical progress includes work on the 6th evaluation datasets, an exploration of robust architectures and new state-of-the-art robustness result, and an investigation into how to attack the shape of objects using differentiable rendering.

3.1.1 Overhead Detection Dataset Exploration

In this quarter we spent time examining the CARLA overhead detection dataset used in the 6th evaluation. This dataset contains *person* and *vehicle* annotations of overhead images collected from CARLA. During our exploration, we discovered three issues with the dataset and resolved them appropriately. We also filed these issues with the GARD TA 2 performers.

The first issue we discovered was there are persons and vehicles in the dataset that are visible yet have no corresponding annotation. We pictorially show this in Figure 3.1, where people are walking on the green patch have no bounding box annotation. This is due to the fact that the annotations with segmentation area of less than 100 px were removed.



Figure 3.1: An example image with 18 original bounding box annotations (marked in blue) is missing person annotations that are visibly walking on the sidewalk and the patch (left). Augmenting these annotations to include all objects reveals all 33 annotations (marked in magenta, right).

Second, we found that applying a segmentation area threshold for this dataset may not be optimal. Intuitively, the two classes, *person* and *vehicle*, vary in size, as vehicles tend to be larger and differently shaped than people. A common threshold for both classes adversely affects the person annotations because it eliminates more person annotations than vehicle annotations. We pictorially show this in Figure 3.2, where the the histogram of missing person annotations has more mass

than the histogram of missing vehicle annotations. As Figure 3.2 shows, people are smaller than vehicles, so any thresholds need to be class specific.

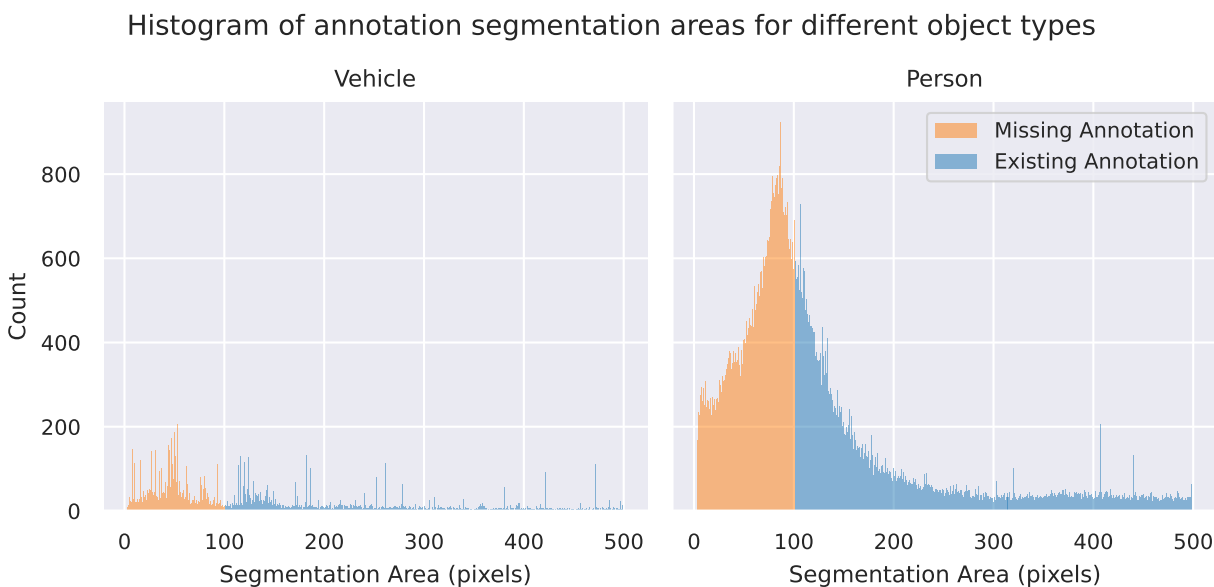


Figure 3.2: Histogram of segmentation mask areas (in pixels) for all objects filtered for (area less than 500 px) from the training dataset. Vehicle areas are more spread out over a wider range compared to person areas. The number of missing persons in the dataset is much higher when a threshold of 100 px is used, hence using a common threshold to filter out smaller objects may adversely affect the person class, also creating the issue we see in Figure 3.1.

We resolved both of these issues by generating our own version of annotations using the instance segmentation images and our data annotation tools. We disabled this threshold, thus including all objects in the scene regardless of segmentation area. This is useful because the task of overhead object detection necessarily includes tiny objects at lower resolutions. We are working to understand how this affects the baseline model from both a training and evaluation perspective.

Finally, the third issue we found is that some annotations include the `iscrowd` attribute that is set to 1. The original intention was to use this attribute to indicate objects with multiple disconnected parts (e.g., a partially occluded person). However, many Microsoft Common Objects in Context (MS-COCO)-specific data loaders by default ignore annotations with `iscrowd` set to 1 for unspecified reasons. We resolved this issue by setting `iscrowd` to 0 for all annotations so that the default data loaders do not ignore these annotations. Similar to the issues above, we are working to understand how this affects the training and testing of the baseline model.

3.1.2 Robust Architectures

Advancements in deep learning architectures, from VGG, to Inception and ResNet-50, to Vision Transformer (ViT) and Swin Transformer and ConvNeXt, have steadily increased in accuracy on

ImageNet-1K from 70% to around 85% [LH17; Sze+15; HZRS16; Dos+20; Liu+21; LMWFDX22]. However, little work has explored whether the increased accuracy of these models also increase their robustness against adversarial attack when using adversarial training [MMSTV18]. To bridge this gap, we designed a set of experiments to evaluate the contributions of different architectural components and design architectures for robustness while controlling for model capacity, since increasing capacity tends to improve robustness with adversarial training.

We studied the impact of network-level modifications (depth and width), stage-level modifications (stem stage, and dense connections), and block-level modifications (kernel size, dilation, activation, squeeze-and-excitation, and normalization) on robustness by adjusting the aforementioned modifications over 150 models on ImageNet.

From our explorations, we distilled robust network design guidelines that deepen our understanding of robust architectural components and generalize to new settings of adversarial attack. For instance, we find that when controlling for model capacity, the depth yields more robustness than width, while width yields more robustness than kernel size. We also find the following modifications boost robustness: adopting squeeze-and-excitation operation with a slight decrease in width, reducing the first normalization layer in a block, and reducing the down sampling factor in the stem stage.

Using these guidelines, we construct a family of architectures with 3 differing capacities that are comparable to cross-covariance image transformer (XCiT) models [Ali+21], a state-of-the-art models for for adversarial robustness. All three of our models—RobArch-S, RobArch-M, RobArch-L—outperform their XCiT counterparts as shown in Figure 3.3.

3.1.3 Differentiable Rendering

We extended previous work on creating physically realizable adversarial objects [CCMC18] to attack image classifiers and object detectors using a differentiable renderer. Differentiable rendering provides the advantage of bridging a 2D projection (i.e., images) to an underlying 3D scene representation by allowing differentiation through the rendering process and exposing the gradients of scene parameters like object textures, meshes, lighting, and sensors. Through the use of differentiable rendering, we can optimize 3D scene parameters with respect to an image or model outputs [Kat+20]. This technique allows a flexible way to manipulate the parameters of a scene for discovering attacks that could be materialized in the real world.

We constructed a white-box adversarial object creation pipeline, as shown in Figure 3.4, by rendering a scene and then passing the resulting image to the model. Next, we took the loss from the model and back propagated these errors through the renderer to obtain a gradient of the loss with respect to the scene parameters. This method allowed us to generate ℓ_2 and ℓ_∞ constrained

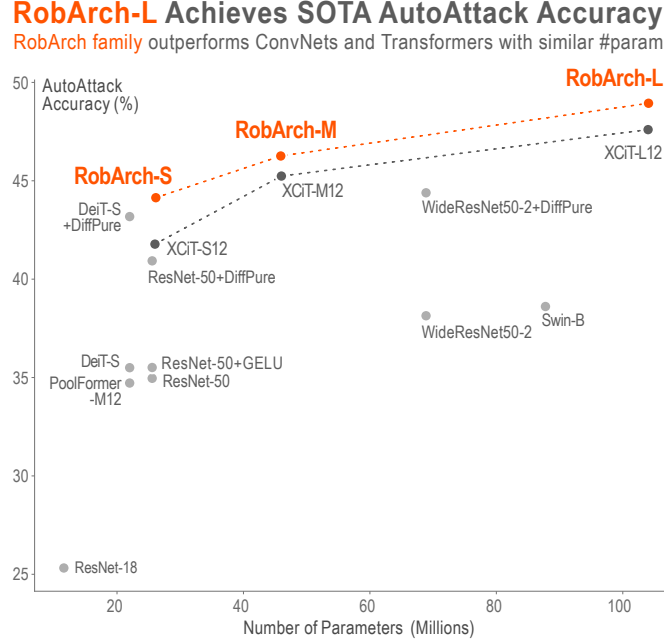


Figure 3.3: Accuracy of our designed architectures compared current state-of-the-art XCiT models and other well-known models. All three variants of our architecture achieve higher accuracy than models of similar capacity.

PGD attacks by perturbing select scene parameters, such as the vertices of a mesh or the colors in a texture. After perturbing the selected scene parameters, we re-rendered the scene using these perturbed parameters and passed the resulting image back to the model.

Currently, we explored producing adversarial objects by generating ℓ_2 and ℓ_∞ constrained PGD attacks on the vertex positions of a mesh. We use an undefended binary classifier with a ResNet-50 architecture trained on the classes “bicycle” and “bird” [BCZOCG18]. Using differentiable rendering, we perturb the mesh and then re-render the object using our proposed pipeline until the model mistakenly classifies the image or abstains from making a prediction, defined as making a low-confidence prediction. Figure 3.5 shows our preliminary results, demonstrating that it is possible to create meshes that successfully attack the classifier. However, some attacks on the “bird” class are successful only at high perturbation strength or are in scenes where the sensor angle is off-center. We also find that mesh perturbations create degenerate meshes that contain vertex self-intersections or are non-manifold. Future work on these mesh attacks could incorporate a mesh correction post-processing step, possibly by making use of Laplacian smoothing.

Motivated by the work of Athalye et al. [AEIK18] that produces robust 3D adversarial examples, we find that we can attack these models using differentiable rendering and generating adversarial textures. Figure 3.5 shows how the texture of a “bird” object can be perturbed using an ℓ_∞ -constrained PGD attack to optimize a texture that causes an image classifier to predict the “bike” class. Our future work will expand into using robust image classifiers and object detectors.

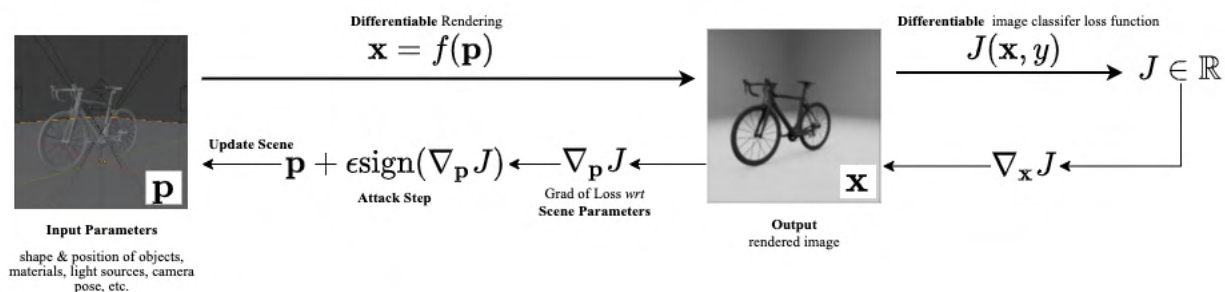


Figure 3.4: In our proposed differentiable rendering pipeline, we obtain the gradient of the image classifier loss with respect to the scene parameters and construct an adversarial perturbation of an object’s mesh vertices or texture in an effort to attack the model.

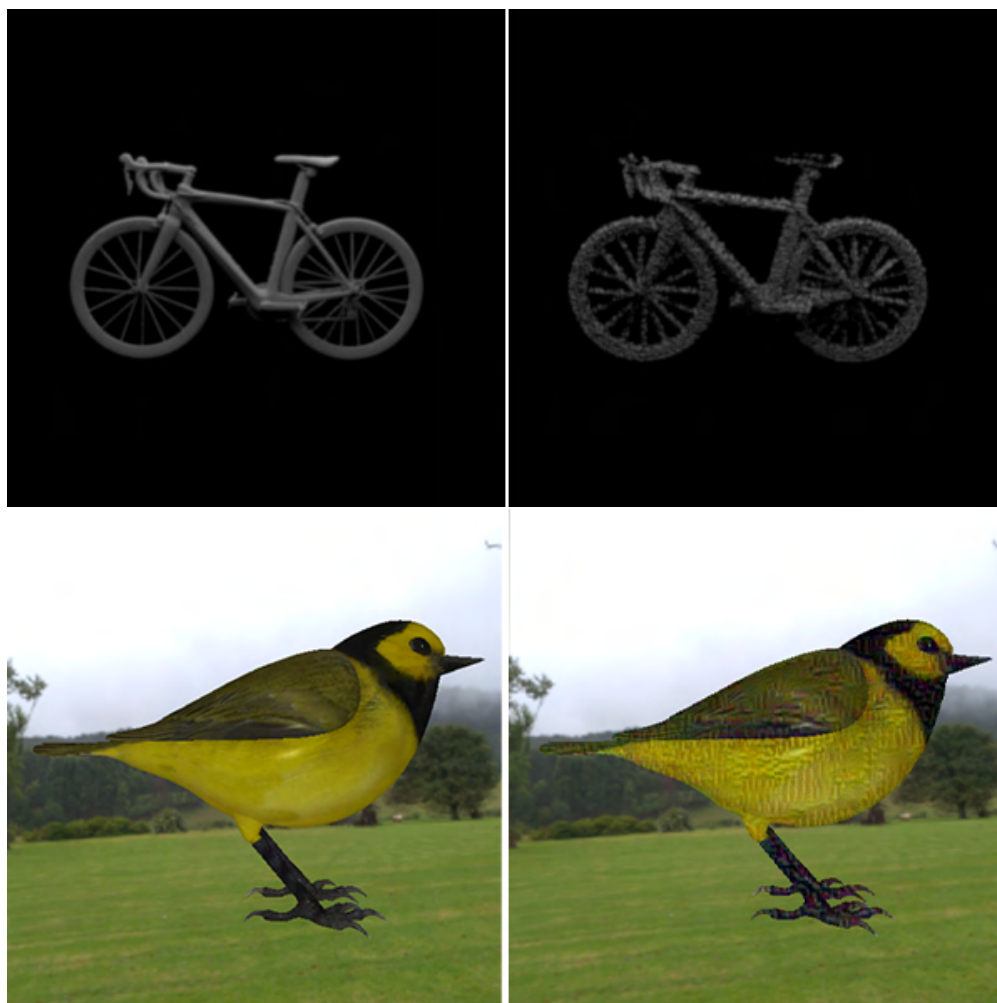


Figure 3.5: Top Left: Benign rendering of a bicycle. Top Right: Adversarial object created through differentiable rendering of a perturbed mesh. The model classifies this image as a “bird”. Bottom Left: Benign rendering of a bird. Bottom Right: Adversarial object produced by optimizing the texture map through differentiable rendering. The model classifies this image as a “bicycle”.

3.2 Q4 2022 Update

In the 4th quarter of 2022, we submitted 3 defenses to the 6th GARD Evaluation. Our defenses covered three different scenarios: uni-modal object detection, multi-modal object detection, and multi-object tracking. These scenarios presented new challenges for our defenses, since most objects in the images and videos are smaller than the adversarial patches, as shown in Figure 3.6. Because the scenarios placed the patch directly behind objects, this enables the attack to manipulate more pixels near and around objects. Additionally, objects in the scenarios are small relative to the input size, thus making it difficult to correctly detect objects in the benign case. For example, in Figure 3.6 it is difficult for humans to detect the pedestrians in the lower right corner of the patch because they are both small and blend into the patch. We describe the details of our submissions, the affect of this threat on our submissions, and our Modular Adversarial Robustness Toolkit (MART) in the following sections.

3.2.1 Uni-modal Overhead Object Detection

For uni-modal object detection, we developed a decision tree-based model for classifying object class based on bounding box features in images. We used this decision tree to learn the design of anchors for deep object detection models. The general motivation is to learn a prior from benign scenes in order to help design anchor widths and aspect ratios for each class. Anchors in the Faster R-CNN object detection pipeline are rectangular masks with a configurable size and aspect ratio that slide across the input sample and all extracted feature maps [RHGS15]. The features extracted from each anchors are potential candidates for objects, hence anchors should be designed to accommodate all potential objects in a scene. At the same time, over-provisioning anchors by e.g., including shapes that are unnecessarily large, may lead to increased adversarial vulnerability of the detection pipeline. Improving the design of the anchors is feasible in the CARLA overhead object detection scenario due to the small number of object classes, whereas the default anchor sizes—64 px, 128 px and 256 px at each feature level—were chosen with the MS-COCO dataset in mind and intended to handle a large number of objects of varying sizes and aspect ratios.

Figure 3.7 depicts a high-level block diagram of how we use this prior in the object detector. To design the anchors, we trained a decision tree that classifies two labels (pedestrian or vehicle), using four, manually extracted features of the ground-truth bounding box: height, width, area, and aspect ratio. This decision tree achieves 85% test accuracy, which tells us a correlation exists between these features and the class of object in the respective box. To extract relevant anchor widths and aspect ratios, we performed clustering on a subset of best performing tree leaves (with accuracy greater than 95%). These clusters formed the anchors of the anchor generator in the Faster R-CNN.

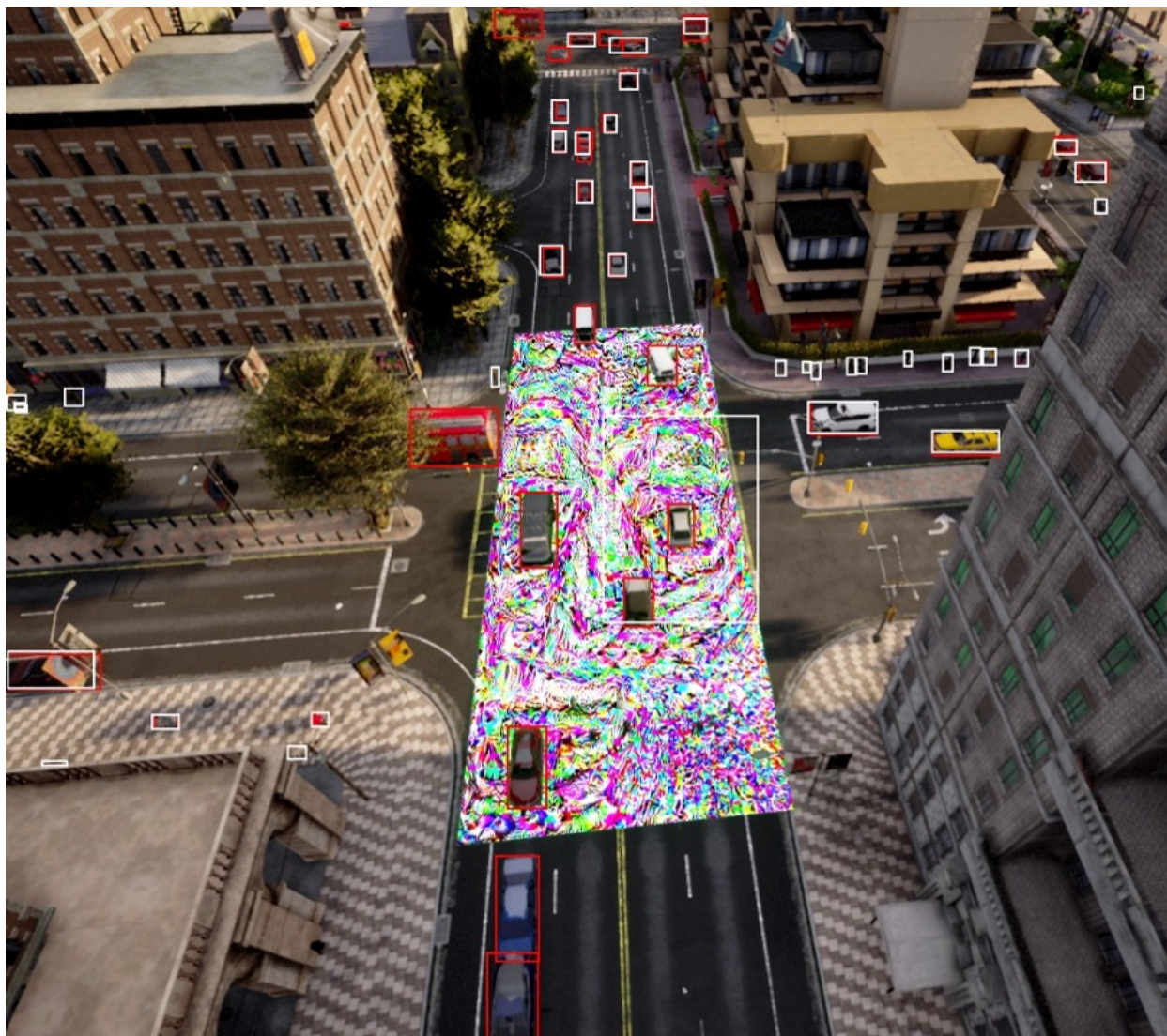


Figure 3.6: An example of an attacked test image using a large adversarial patch in the uni-modal (RGB camera only) scenario placed on the ground is shown, together with the boxes detected by the model under attack. While it may be hard to notice perceptually, there are six hidden pedestrians (crossing the street) in the lower part of the attacked patch.

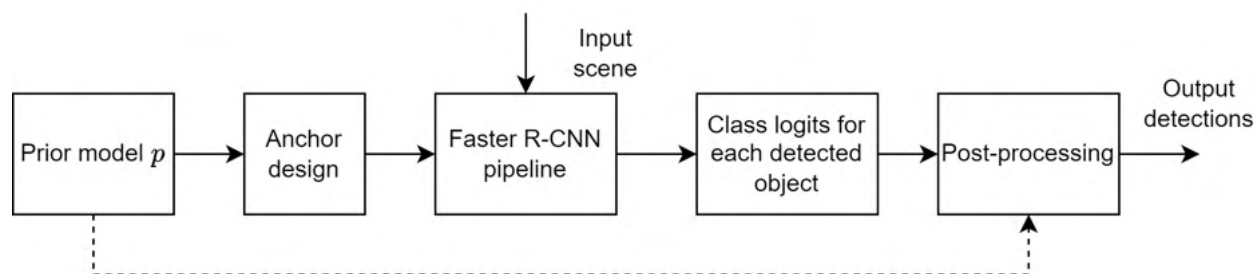


Figure 3.7: Block diagram of the high-level approach for the uni-modal object detection task using learned priors. The prior was used to design the anchor sizes for our submission, and can potentially be used for post-processing.

When evaluated in Armory, we obtained a 10% relative increase in robust performance when using the tree-based design of anchors (without any post-processing), from 22 AP50 for the baseline to 27 AP50. We also tried to constrain specific anchors such that they could only classify to one class, but this post-processing technique caused a benign performance drop to 38 AP50. Given that the patch can completely surround objects in this evaluation, we believe we need new techniques to handle this threat.

3.2.2 Multi-modal Overhead Object Detection

In multi-modal object detection scenario, the adversary can change the RGB pixels arbitrarily and the depth by ± 0.03 m of a green-screen patch to confuse the target model. However, our investigation suggests that the reference attack algorithm in Armory is not aligned with this threat model.

We ran an ablation study of the RGB-Depth fusion attack and found the initialization of the depth attack was likely incorrect. Table 3.1 shows the RGB attack initialization alone does not change AP50, while the 1000-iteration RGB attack is able to decrease AP50 from 69% to 42%. This is expected, because an adversarial perturbation is not just random noise. However, we found that the depth attack initialization alone decreases AP50 by 4 p.p. (from 69% to 65%), while the 1000-iteration depth-only attack further decreases by only 2 p.p. (from 65% to 63%). The initialization of the depth modality is so destructive because the attack algorithm initializes the depth value of a patch to the average of the original depth value of the patch. If a patch is flat and perpendicular to the camera, such initialization is acceptable. But this is not the case for most images in the test dataset, resulting in adversarial patches that we hypothesized were unrealistic.

Table 3.1: Ablation study on the reference RGB-Depth attack algorithm in Armory with the baseline model on the test dataset. We report benign AP50 in the “Benign” column, and adversarial AP50 in three initialization scenarios as well as three attack scenarios. The “Attack Initialization” columns show that RGB attack initialization works as expected while the depth attack initialization surprisingly decreases AP50 by 4 p.p. from 69% to 65%. The “1000-iteration Attack” columns show that the depth attack is not as effective as the RGB attack. The depth attack alone only further decreases AP50 by 2 p.p. from 65% to 63% after the attack initialization. The result indicates that the depth attack initialization is likely incorrect and is later confirmed in Figure 3.8.

	Benign	Attack Initialization			1000-iteration Attack		
		RGB + Depth	RGB	Depth	RGB + Depth	RGB	Depth
AP50 of Baseline Model	69%	65%	69%	65%	39%	42%	63%

To gather evidence for this hypothesis, we used Open3D [ZPK18] to visualize these adversarial examples, as shown in Figure 3.8. Figure 3.8a shows two parallel surfaces of titled patches, which we root caused to a quantization error in Armory’s example exporting code¹. After Armory fixed

¹<https://github.com/twosixlabs/armory/issues/1858>

this bug, we re-visualized the adversarial patch and found it was incorrectly initialized as shown in Figure 3.8b. The adversarial patch generated by the attack algorithm in Armory unrealistically protrudes above and below the street, thus not conforming to the threat model and verifying our hypothesis. This same initialization bug can be more clearly seen in Figure 3.9. We recognize that it is challenging to correctly implement an attack algorithm for RGB-Depth tasks, especially without a handy 3D visualization tool. As such, we plan to open-source our 3D adversarial example visualization toolkit in the next quarter.

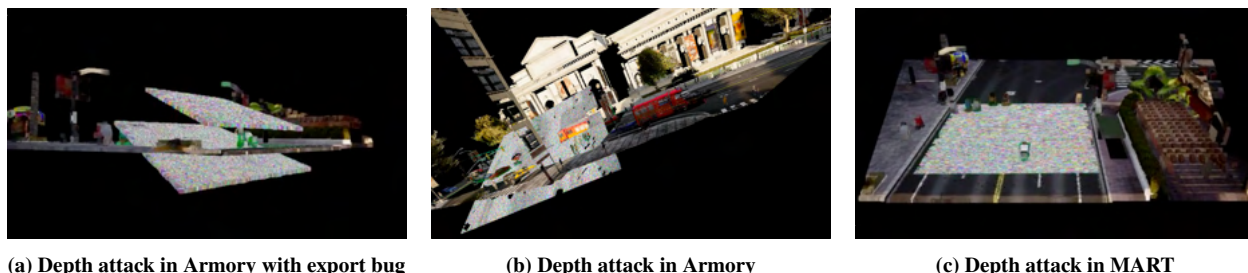


Figure 3.8: Visualization of RGB-Depth adversarial examples in the 3D space. Figure 3.8a demonstrates a faulty adversarial example with two tilted patch surfaces in parallel exported from Armory. Figure 3.8b demonstrates a faulty adversarial example with one tilted patch after Armory fixed the quantization error in exporting examples. Figure 3.8c demonstrates an adversarial example generated by the correct depth attack algorithm that we implemented in MART.



Figure 3.9: An example scene from the multi-modal object detection scenario, as viewed from the side. This scene shows a bug we discovered in the reference patch attack, which incorrectly initializes the depth input. This results in an unrealistic patch that protrudes from and into the ground. We used Open3D [ZPK18] to visualize this scene and attack.

In MART, however, it was straightforward to compose a depth attack with the mask additive threat model. We implemented such depth attack with correct initialization, as is visualized in Figure 3.8c. Our depth attack runs in the linear depth space rather than the three-channel raw depth space. This also avoids another bug in the Armory attack which results in incorrect depth clamping to the desired ± 0.03 m. We used this attack in MART to adversarially train a depth-only model that

achieves 90% AP50 under attack. We only use depth because it yields a model with higher AP50 than the counterpart that fuses image and depth, which again confirms our findings from the 5th GARD evaluation. During adversarial training, we compute Fast Gradient Sign Method (FGSM) attacks with depth epsilon of ± 0.0225 m, which is 25% smaller than the epsilon in test. We also fine-tuned this model using 20 images from the development dataset with the incorrect depth initialization generated by the Armory attack, so that the model adapts to the domain shift caused by the titled patches. We compare our model with the baseline in Table 3.2. Our model outperforms the baseline by a large margin. Indeed, the AP50 of our model under attack (73%) is even higher than the benign AP50 of the baseline model (69%). This is a result of architectural improvements (as described in Section 3.2.1) and proper adversarial training of the depth modality.

Table 3.2: Multi-modal object detection AP50 results as measured on the test dataset. Our submission model is very robust against a correct depth attack that we implemented in MART: 90% compared with 91% of benign. When evaluated with the faulty RGB-Depth attack in Armory, our model achieves 73% AP50, which is even higher than 69%, the benign AP50 of the Armory baseline model.

	AP50		
	Benign	MART Attack	Armory Attack
Armory Baseline Model	69%	–	39%
Our Submission Model	91%	90%	73%

3.2.3 Multi-object Overhead Object Tracking

Our multi-object tracking defense uses a combination of pre-processing filters and homography-based background subtraction to defend against patch attacks. In the broadest terms, our pre-processing defense is to remove the “stationary” background in the video, which may contain an adversarial patch, so that the tracker can focus on the foreground pixels which consist of the object(s) of interest.

Concretely, our approach consists of the following steps and closely follows our implementation from our 5th evaluation submission. It starts with the homography-based background subtraction. We warp all video frames in the batch into a common static perspective through homographic transformation. Without loss of generality, our approach uses the first frame of every video as the reference frame. That frame’s perspective becomes the common, static perspective for all video frames in that batch, i.e., all frames’ perspectives are warped to match that of the reference frame’s. While we can assume the defender controls the location and perspective of the camera, we do not necessarily have access to this information. Instead, we estimated the projective transformation matrix that warps a frame’s perspective to that of the reference frame’s perspective.

We implemented this frame-warping function using OpenCV [Bra00] as follows. First, we extract ORB keypoints [RRKB11] from both the frames to be warped and the reference frame. We then compute the homographic transformation matrix for the frame to be warped by using Random SAmple Consensus (RANSAC) (as implemented in OpenCV’s `findHomography` function) to find a matching between these keypoints. Finally, we warp each video frame using the computed homography matrix. Our implementation improves upon our 5th evaluation submission to by “zooming” out to a larger frame size by 150%. This ensures the warping procedure does not unintentionally “crop away” tracking targets near the edges of the frame. If we assume the defender controls the position of the camera, we could in theory, use non-image features to compute the homography.

Given each frame of the video now has the same perspective, we then extracted the static background across all the warped frames by computing the median pixel value over the time axis at each pixel location (each of the RGB channels were processed independently). This enabled us to mask out all pixels that lie within a certain threshold of these static background pixel values.



Figure 3.10: The patch “moves” between (a) Frame 1 and (b) Frame 2 as shown in (c) the pixel value “differences” between Frame 1 and Frame 2, thus showing the impact of this misalignment.

We followed background subtraction with a small-scale feature filter that removes any remnants of background remaining. In theory no such remnants should exist if the adversarial patch were perfectly aligned with the green screen. However, we observed that adversarial patches are not aligned with the green screen in some frames of the videos, even when the perspectives in those frames are identical. Figure 3.10 demonstrates this patch misalignment issue. This patch misalignment issue was also observed in the 5th evaluation, for which we filed an issue on the Armory repository ². Unfortunately, this issue persists in the 6th evaluation scenario. We believe that “re-injecting” these patches back into the CARLA simulator will rectify this issue. We are currently working to enable this in our data collection tool. Since this filtering is based on convolution operations alone, we believe that a determined attack should be able to overcome these filters and new ideas may be necessary to defend against 6th evaluation scenarios if the background remnants persist even after

²<https://github.com/twosixlabs/armory/issues/1532>

background subtraction.

Our submission evaluates whether the current implementation of our attack can readily discover informative gradients that may be used to effectively disrupt the background subtraction procedure when the adversarial pattern is stationary. We believe an attack strategy that discovers or approximates gradients that are more informative to the attackers can enable them to create adversarial patterns that will break the current defense (e.g., the background mimics the foreground and therefore, subtracting the background results in losing the foreground object to be tracked as well). In this case, our defense may also be susceptible to “hallucination” attacks (e.g., a pedestrian pattern appearing as part of the adversarial pattern that looks similar to the actual pedestrian target being tracked). The general idea of background subtraction relies upon the assumption that the “background” (including the patch) does not significantly change over time. An adversary could break this assumption by placing the patch on the side of truck that would drive through the scene. Whether such patches would remain effective while moving remains an open question.

3.2.4 Modular Adversarial Robustness Toolkit

We open-sourced our MART ³ under the BSD-3 license. Many existing libraries for adversarial robustness evaluation, like ART [Nic+18] and FoolBox [RBB17], collect many attack algorithms as separate implementations. MART attempts to unify these attacks under an optimization framework, as shown in Figure 3.11, so that users can compose existing components into novel attacks by using basic building blocks from the PyTorch library, as is shown in Figure 3.12. Indeed, MART made it straightforward for us to compose an attack for the depth modality as described in Section 3.2.2. As a result, we were able to avoid two semantic bugs in the depth modal which was discovered in the Armory/ART attack as is demonstrated in Figure 3.9. MART also makes it straightforward to run adversarial training, since we implement adversary modules as a layer that can be inserted in the target model. We used MART to adversarially train our 6th evaluation submission described in Section 3.2.2. We plan to keep improving Modular Adversarial Robustness Toolkit by implementing RGB-Depth fusion attacks and integrating them to Armory.

³<https://github.com/IntelLabs/MART>

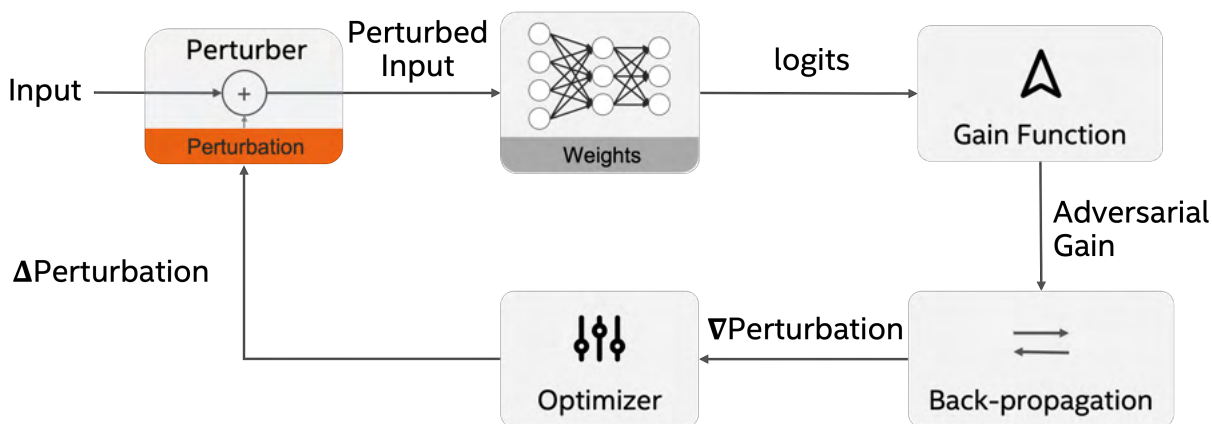


Figure 3.11: A unified optimization framework for composing attack algorithms in Modular Adversarial Robustness Toolkit.

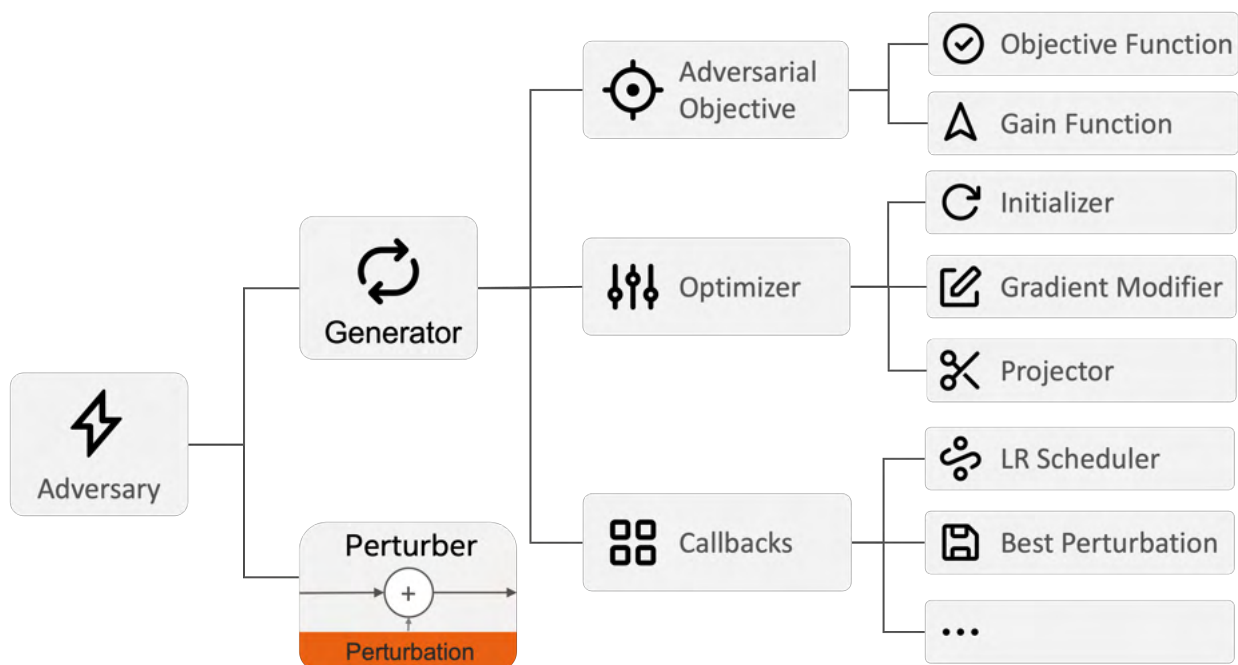


Figure 3.12: Modular attack composition in Modular Adversarial Robustness Toolkit.

3.3 Q1 2023 Update

In the 1st quarter of 2023, we invested in generative modeling as a means to detect anomalous inputs. We described our approach and preliminary results in Section 3.3.1. We also examined architectural components that affect adversarial training and describe three design principles in Section 3.3.2. Finally, in Section 3.3.3, we describe our efforts on using MART to evaluate our depth-only defense and how we utilized MART in a demo we create for Intel Vision 2023.

3.3.1 Diffusion Priors for Object Detection

Our goals were to investigate how feasible and how useful it is to train an unconditional score-based diffusion model from scratch on the 6th evaluation CARLA overhead dataset. We opted to train our own unconditional diffusion model from scratch in order to better match the distribution of specific CARLA scenes. We could have used or fine-tuned publicly available models like Stable Diffusion, but the use of latent diffusion does not allow for exact log-likelihood evaluation, as we discuss in Section 3.3.1.2.

3.3.1.1 Training score-based diffusion priors

We trained a generative, score-based diffusion model using the open-source framework of Song et al. [SSKKEP20]. These models were trained to predict the time-dependent score (the high-dimensional gradient of the log-likelihood function $p(x)$) function of a family of perturbed data distributions, with an increasing amount of perturbation as time increases. A pictorial representation of these distributions and the forward diffusion process which converts the data distribution p_0 to a pure noise distribution p_T is shown in Figure 3.13.

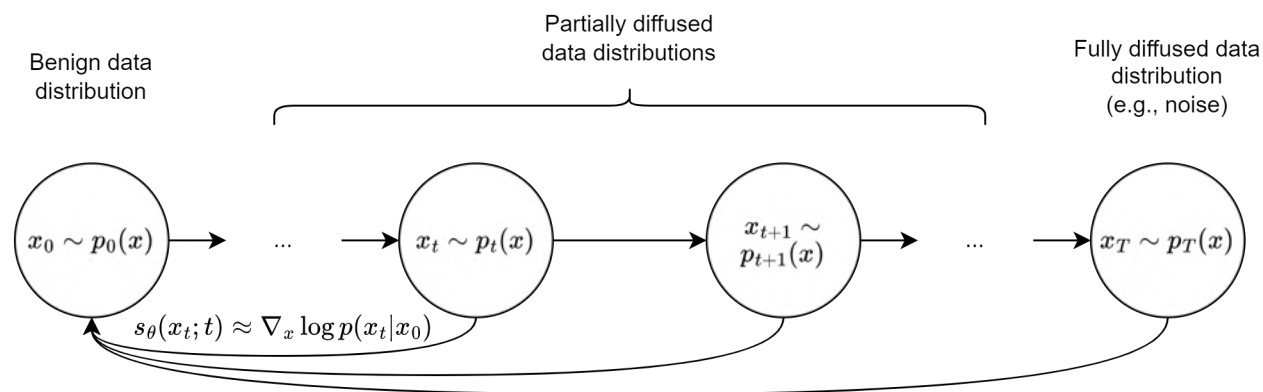


Figure 3.13: State diagram of the discretized forward diffusion process and the time-dependent score function learned by the diffusion model.

Due to the high resolution of full CARLA scenes (up to 960×1280 px), we trained our diffusion model on patches of these images. This increases the net amount of training data, leads to a better specified distribution, and also increases data diversity of the considered distribution. We trained a model on image patches of size 160×160 px that were randomly cropped from larger 224×224 px patches centered around ground-truth object annotations, thus avoiding patches of rooftops or buildings and only retaining patches with pedestrians, vehicles, or empty ground surfaces (because of the random crop, pedestrians in some patches may be missed) in them. Such patches yield an accurate prior for detecting and defending against the adversarial patch attacks.

Table 3.3 provides more details on the used architecture and the learning process, with full code planned for release alongside our future submissions to evaluations. Figure 3.14 shows examples of patches from the training data, as well as generated patches. Interestingly, we can see that the yellow vehicle in the training data (left side, fourth row) is perfectly replicated in the sampled data (right side, first row), up to a horizontal flipping, which was used as augmentation. This is conducive to both a very strong prior learned, but also to overfitting on specific vehicle shapes, colors, and positions relative to road markings, because of the relatively low vehicle diversity in the training set.

Name	Value
Network s_θ	NCSN++ [SSKKEP20]
Depth (in blocks)	4
Parameters	101M
Train patches	96458
Train epochs	200
Sampling algorithm	Euler
Sampling steps	2000

Table 3.3: Architecture and hyper-parameters used for training and sampling score-based diffusion models.

3.3.1.2 Adversarial patch detection performance using score-based diffusion priors

Given a trained score-based diffusion model s_θ , we can estimate the log-likelihood $p_0(x)$ of an arbitrary sample x using any ordinary differential equation (ODE) solver method and the methodology in Song et al. [SSKKEP20]. In our case, x is a patch of size 160×160 px from the RGB scene, for which we can estimate a scalar value corresponding to its log-likelihood: a higher log-likelihood will indicate that the patch under test is more likely to be sampled under the training distribution of benign patches, whereas a lower log-likelihood indicates a more anomalous sample.

Our first experiments were focused on using models trained on patches to estimate the spatial likelihood of a complete scene, with or without adversarial patches present, and to quantify the performance of a single, constant detection threshold for spatially localizing the patch. To accomplish this, we slide a window with the size of the patch (160×160 px) across the entire image with a stride of 40 px and estimate the log-likelihoods of each resulting patch. We then form a pixel-wise (but still with an effective resolution of 40 px) likelihood image by averaging all obtained likelihoods



Figure 3.14: (Left) 16 randomly chosen patches from the 6th evaluation CARLA overhead training data. (Right) 16 randomly generated samples from one of our score-based diffusion models trained on patches using a stochastic sampling procedure.

for all patches in which a pixel is present.

The results for two benign and adversarial samples, and three different solvers, respectively are shown in Figure 3.15. The threshold value of -1.8 was chosen based on the pair of samples in the first and fourth row and the first column, one benign and one adversarial, using the adaptive Runge-Kutta (RK) solver in `scipy`⁴. We observe that the threshold generalizes and accurately localizes the adversarial patch for the other pair of samples in the fourth column, and additionally to two other non-adaptive solvers (Euler and fixed grid RK) with different step size and varying complexity.

Motivated by the result in Figure 3.15, we plan to continue investigating the feasibility of score-based diffusion models for adversarial defense purposes. There are two important pieces that we will work. First, integrating the diffusion prior in the object detection pipeline, such that it is compatible with the evaluation requirements. Our current investigations indicate that the most feasible way of doing this is by estimating the log-likelihood of patches extracted around the detected objects, and overriding classification to the background class if anomalous patches are detected. Second, rigorously evaluating the white-box adversarial robustness of the entire log-likelihood estimation pipeline. We used the open-source `torchdiffeq` [Che18] library for implementing fully differentiable ODE solvers, which allows for memory-efficient back-propagation along the entire procedure via the adjoint ODE method. We are currently investigating whether the adjoint ODE method itself can cause gradient masking and looking into using gradient checkpointing for more accurate gradients.

⁴https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.solve_ivp.html

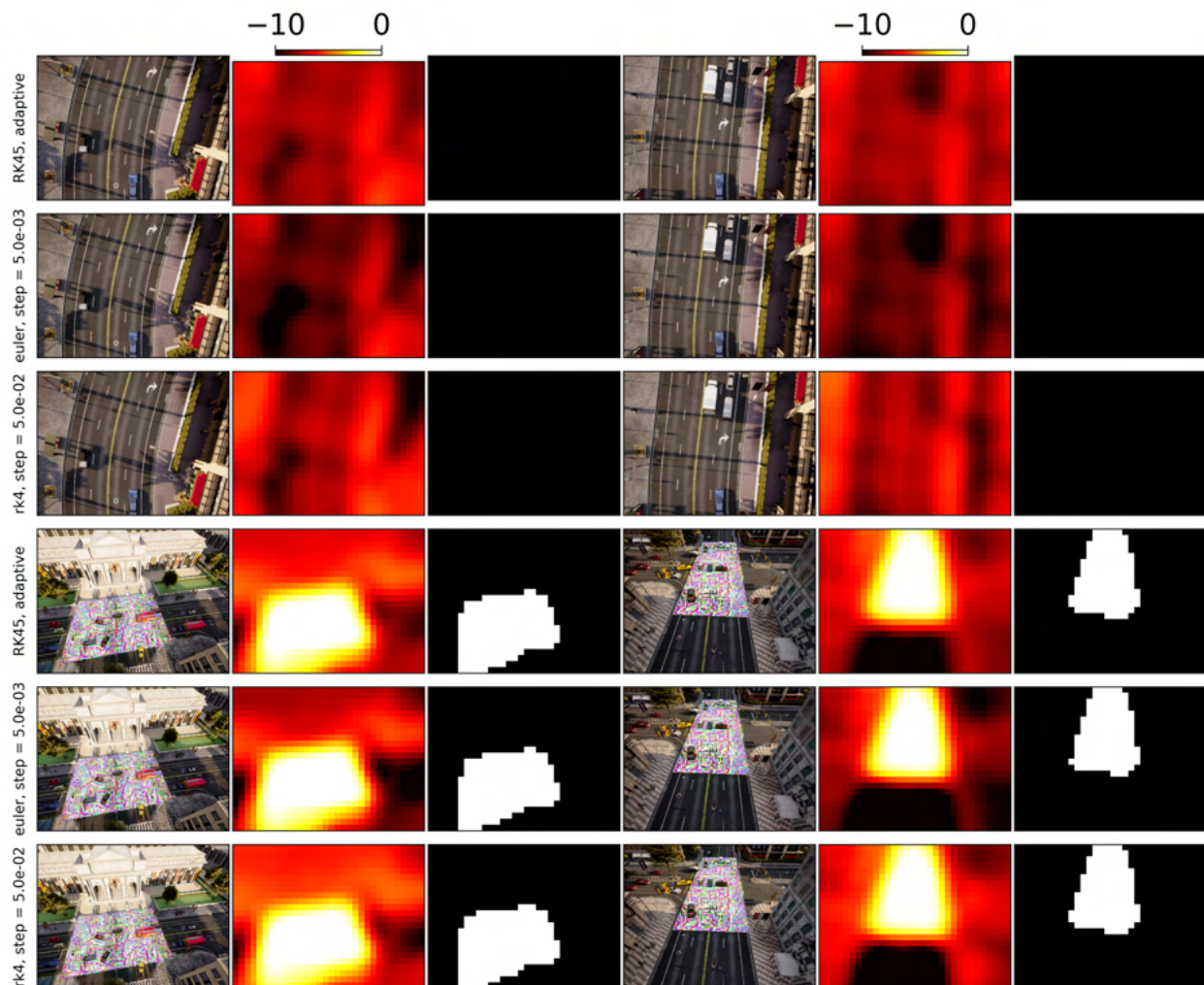


Figure 3.15: The first three rows show two benign samples (first and fourth columns), their estimated negative log-likelihood maps (second and fifth columns), and the binarized maps with a threshold of -1.8 (third and sixth columns). The following three rows show two adversarially attacked samples and their estimated maps. The white color in the binary map marks an anomaly detection according to the diffusion prior.

3.3.2 Robust Architectures

We aim to unify existing works’ diverging opinions on how architectural components affect the adversarial robustness of Convolutional Neural Networks (CNNs). To accomplish our goal, we survey and distill four key architectural components underpinning state-of-the-art (SOTA) CNNs and Transformers that boost adversarial robustness. These components span a network’s macro and micro designs: *depth and width*, *stem stage*, *squeeze-and-excitation block*, and *activation*. Different from previous research on robust architectures that solely relies on the small-scale CIFAR-10, we explore the adversarial training of all four components on the large-scale ImageNet and evaluate on its full validation set. Through extensive experiments over the wide spectrum of *dataset scales*, *adversarial training methods*, *model parameters*, and *network design spaces*, we make the following key contributions.

3.3.2.1 A suite of generalizable robust architectural design principles for CNNs

We found three design principles described below and shown in Figure 3.16.

(A) Optimal Range for Depth and Width Configurations.

Despite the popularity of 4-stage residual networks on ImageNet, existing exploration has been constrained to 3-stage designs [HWEGBM21; HLDB22]. We discover a flexible depth and width scaling rule that does not place a restriction on the total number of stages, and we verify its generalizability and optimality through extensive experiments.

(B) Preferring Convolutional over Patchify Stem Stage.

Convolutional stem and *patchify* stem are commonly used in CNNs and Transformers to downsample input images due to the redundancy inherent in natural images. *Convolutional* stem uses overlapped convolution kernel to slide the input image, while *patchify* stem patchifies the input image into $p \times p$ non-overlapping patches. We discover that convolutional stem, especially with a postponed downsampling design, outperforms patchify stem due to its less-aggressive stride-two downsampling and overlapped convolution kernels.

(C) Robust Residual Block Design.

Our investigations on how squeeze-and-excitation block and activations affect robustness present new findings on ImageNet that differ from previous research on CIFAR. Through a hyperparameter sweep, we find that a larger reduction ratio r in squeeze-and-excitation block reduces robustness, a new discovery not previously reported as prior work was based on a fixed r [HLDB22]. We also confirm that non-parametric smooth activations consistently improve robustness on CIFAR-10, CIFAR-100 and ImageNet [XTGYL20; DMM22].

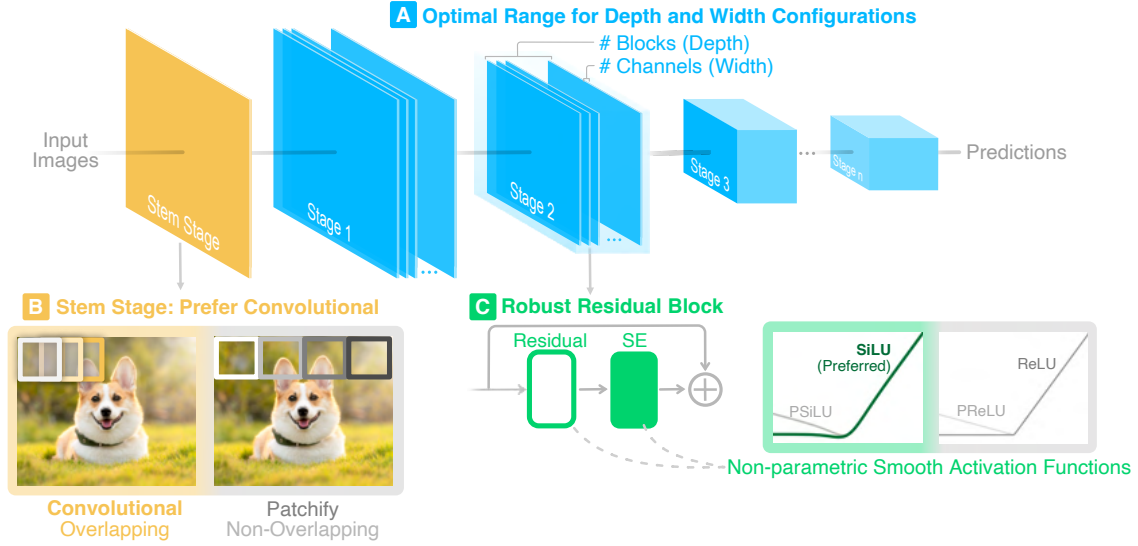


Figure 3.16: We synthesize a suite of generalizable architectural design principles to robustify CNNs, spanning a network’s macro and micro designs: (A) optimal range for depth and width configurations, (B) preferring *convolutional* over *patchify* stem stage, and (C) robust residual block design by adopting squeeze-and-excitation blocks, and non-parametric smooth activation functions. The principles consistently and markedly improve AutoAttack accuracy for CIFAR-10, CIFAR-100, and ImageNet over the wide spectrum of adversarial training methods, model parameters, and network design.

3.3.2.2 Consistent and marked improvements on adversarial robustness.

We verify the generalization of the three design principles across a wide spectrum of *dataset scales* (CIFAR-10, CIFAR-100 [KH+09], ImageNet [DDSLF09]), *adversarial training methods* (standard adversarial training [MMSTV18], TRADES [ZYJXEJ19], fast adversarial training [WRK20], MART [WZYBMG20], and diffusion-augmented adversarial training [WPDLLY23]), *model parameters* (from 26M to 267M), and *network design spaces* (variants of Wide ResNet (WRN) [ZK16] and ResNet [HZRS16]). Our experiments demonstrate that all design principles consistently and markedly improve AutoAttack accuracy by 1 p.p. to 3 p.p. on CIFAR-10 and CIFAR-100—boosting even the SOTA diffusion-augmented adversarial training [WPDLLY23] by such amounts—and 4 p.p. to 9 p.p. on ImageNet. In particular, our robustified WRN-70-16 boosts the AutoAttack accuracy by 1.31 p.p. (65.02% $\rightarrow$ 66.33%) on CIFAR-10, and 1.12 p.p. (37.77% $\rightarrow$ 38.89%) on CIFAR-100. On ImageNet, the AutoAttack accuracy is boosted by 6.48 p.p. (39.78% $\rightarrow$ 46.26%) through robustified ResNet-101, and 6.94 p.p. (42.00% $\rightarrow$ 48.94%) through robustified WRN-101-2. Our findings unify prior works’ diverging opinions on how architectural components affect robustness and highlight the benefits of exploring intrinsically robust architectural components.

3.3.3 Modular Adversarial Robustness Toolkit

Our Modular Adversarial Robustness Toolkit (MART) framework has made it much easier to implement adversarial machine learning features. In this quarter, we implemented the RGB-Depth fusion attack, the generalized adversarial patch, and the live adversarial patch. We are working to clean up our implementation and release it in the next version of MART. We plan to integrate MART attacks into ARMORY Adversarial Robustness Evaluation Test Bed (ARMORY) and make MART accessible to Adversarial Robustness Toolkit (ART) users by working with the two teams in next quarter.

3.3.3.1 Generalized Adversarial Patch

In GARD, we evaluate defenses where the adversary knows everything about the scenario, model, and defense. For example, in the 6th evaluation, the adversary generates a patch that targets pedestrians and vehicles in a known positions. In practice, though, the adversary would not know the exact positioning of these pedestrians after they have deployed their perturbation. It is not always the case that an adversary would know the exact configuration of the scenario. In such situations adversaries are therefore forced to make their perturbations robust to these different configurations. Much like how we train models to generalize to new data, adversaries must generate perturbations to generalize to new scenario configurations. This was such the case in our recent demo (see Section 6.3), since we could not know *a priori* who would come by the demo and how they would position themselves within the demo.

The standard technique for adversaries to make their perturbations generalize to such new configurations is Expectation over Transformation (EoT) [AS18]. We added support for EoT to MART in order to meet the needs of our demo. This feature enabled us to generate patches using images from MS-COCO training dataset. That is, we find patches, via gradient descent, that, in expectation, work well against MS-COCO training at meeting the adversary’s goals (see Figure 3.17). We then tested these patches by digitally compositing them into a set of images we collected that mimic the setup of the demo. This setup is very similar to how we currently train machine learning models: instead of modifying the parameters of the model, we modify the parameters of the patch. This is contrast to GARD evaluations where defenders need to generalize to new scenario configurations but adversaries do not.

3.3.3.2 Live Adversarial Patch

We also used MART to implement a live adversarial patch generation program. While the pre-computed patches work well at the demo in general, they do require some calibration on the camera and the patches look a little strange compared with a benign background. In contrast, our adversarial



Figure 3.17: The patches we generated using MART. Each patch had a different adversary goal as described in their respective captions. See Figure 6.3 for example detections from our demo. The real-time perturbation (h) mimics the setup of the current GARD evaluations. As such, it requires less perturbation budget than the other patches and thus looks more like the benign banner (a).

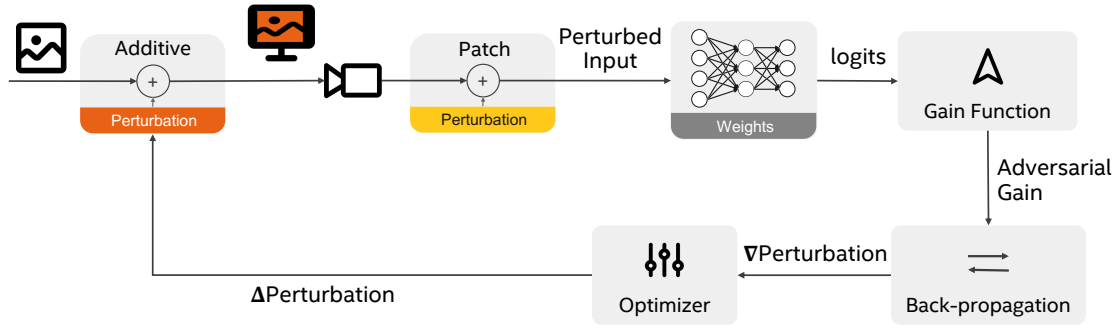


Figure 3.18: The pipeline for generating a live adversarial patch. Note the addition of the display/capture devices as compared to the non-physical setting in Figure 3.11. Due to the broken gradient flow in the display/capture component, we can only compute the gradient of the yellow perturbation and project it to the red perturbation.

patch generated in real time is robust to various environmental conditions, such as different screens, cameras and lighting conditions. The live adversarial patch also requires less budget in perturbation because the patch is generated for a specific screen-camera setup that generalization is not expected. As a result, the live adversarial patch looks more like the benign image compared with the pre-computed patches, as in Figure 3.17h.

Figure 3.18 depicts the pipeline for generating adversarial patch in the live setting. It’s important to note that the camera imaging system is not differentiable. Thus we compute the gradient of the patch perturbation in one attack iteration, and later project the patch perturbation to the additive perturbation of the target image. In next iteration, the camera captures the perturbed image, and we reset the gradient of patch perturbation. The loop continues by projecting the patch perturbation into the additive perturbation. Even though the threat model in the live adversarial patch is not realistic, we believe the demo is useful in two ways. First, the live adversarial patch is pedagogically useful because it shows audience the process of generating adversarial patches. Second, the live adversarial patch could be a good starting point of physical patches if we intend to generate L_p -bounded patches.

3.3.3.3 RGB-Depth Fusion Attack

We added support for multi-modal inputs to MART by implementing a modality dispatch mechanism that works with composers, gradient modifiers, projectors and threat model enforcers. This makes it straightforward to compose a RGB-Depth fusion attack in MART. Our fusion attack achieves 18.53% AP50, while the ARMORY attack only achieves 33%. We didn’t have time to investigate the difference between the two implementation since the ARMORY attack was released a few days before the quarterly report due date, but we plan to do so in next quarter for the 7th evaluation.

3.4 Q2 2023 Update

In the 2nd quarter of 2023, we submitted two defenses to the 7th GARD evaluation, one for the object detection scenario and one for the object tracking scenario. We describe our generative model-based object detection defense in Section 3.4.1 and our background subtraction-based object tracking defense in Section 3.4.2. In Section 3.4.3, we describe how we used our CARLA data collection tools to collect an order of magnitude more images, which we used to train our defenses. Finally, in Section 3.4.4 we describe our efforts to improve the usability of MART.

3.4.1 Score-Based Generative Modeling for Adversarial Defense

This quarter we focused on using pre-trained score-based diffusion models for adversarial sample analysis and defensive purposes in overhead object detection from single images. The basic principles of the score-based diffusion models we used can be found in Section 3.3.1 of our Q1 2023 update and have remained the same. Our work this quarter was integrating these learned generative priors in the object detection pipeline and more thoroughly investigating their black-box and white-box adversarial robustness.

We used two distinct, unconditional, score-based diffusion models trained on 160×160 px patches extracted around objects from two types of scenes: uni-modal (RGB) and multi-modal (RGB-D). We used the training data from 6th and 7th evaluations, and trained the models completely from scratch, exactly as described in Section 3.3.1.1. To train a score-based model on multi-modal data, we converted the three depth channels into a single log-scale channel (using a custom log conversion that is only part of the defense pipeline, but not of the detection one) and concatenated it with the three RGB channels to obtain a four-channel image. Training the model was otherwise straightforward and took approximately three days on an 8-GPU machine. Figure 3.19 shows synthetic multi-modal samples, where RGB samples look like CARLA vehicles or pedestrians and the corresponding depth sample is consistent with the RGB sample.

While diffusion models are known for their sampling capabilities (sometimes also referred to as “denoising” when used for other types of defensive purposes [CTDRSK22]), we have opted for a different use in the form of *log-likelihood estimation*. Given a pre-trained score-based model s_θ , the log-likelihood of a query sample x can be estimated via the following process from [SSKKEP20], which we used in its entirety.

First, we create a wrapper around the model which outputs $\nabla_t \log p_{X_t}(x, t) = \text{div}_{s_\theta}(x, t)$ —the divergence of the model at a query point x and a time variable t with arbitrary values between 0 and 1. Next, we use the above wrapper with off-the-shelf numerical integration software to leverage the

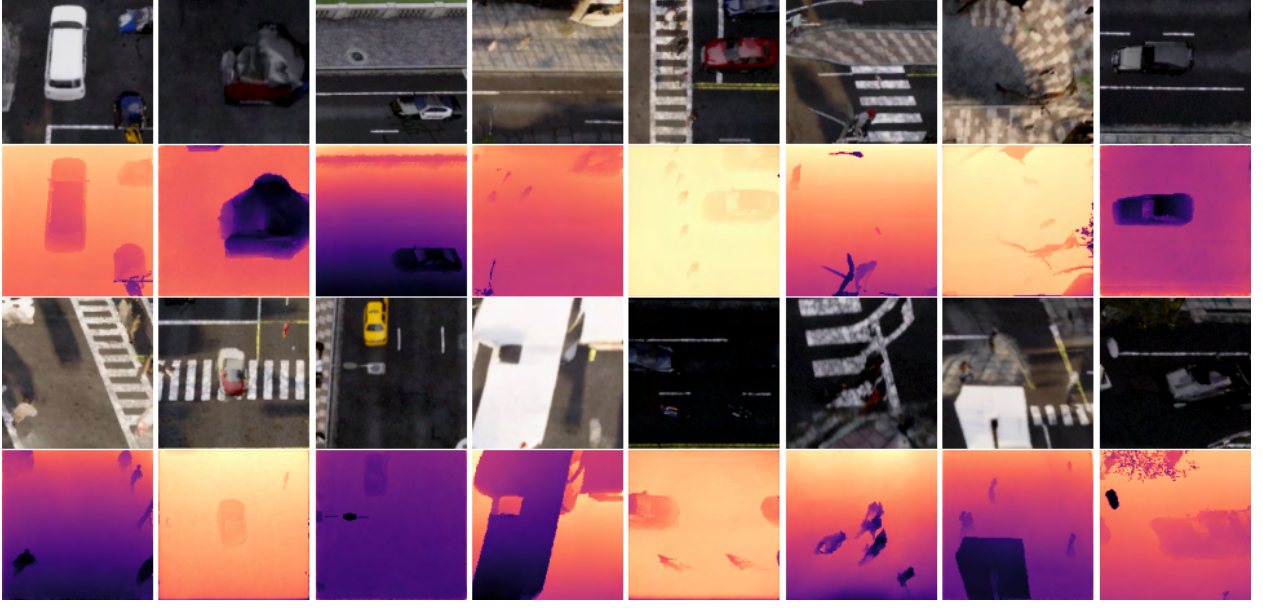


Figure 3.19: Synthetic patches sampled from our multi-modal score-based diffusion model. The first and third rows show the RGB image and the second and fourth rows show the matching depth maps. It can be noticed that even when the samples are imperfect, the depth dimension is consistent with the RGB image.

identity:

$$\int_0^1 \nabla_t \log p_{X_t}(x, t) = \log p_{X_1}(x_1) - \log p_{X_0}(x_0). \quad (3.1)$$

In the above, x_0 is the query sample, and x_1 is its latent representation under a Gaussian prior. Off-the-shelf integration software allows one to compute both the left-hand side of the above equation, as well as obtain x_1 , which in turn leads to a closed-form evaluation of $\log p_{X_1}(x_1)$. Once both terms are computed, the estimated log-likelihood of the query patch x_0 under the benign data distribution p_{X_0} can be computed using the expression:

$$\log p_{X_0}(x_0) = \log p_{X_1}(x_1) - \int_0^1 \nabla_t \log p_{X_t}(x, t). \quad (3.2)$$

Equipped with accurate log-likelihood estimation, we created a defensive post-processor that ***detects and filters on-patch hallucinations***. Figure 3.20 shows the high-level block-diagram of our approach, which we used for the 7th evaluation submission and is Armory-compatible. Our submissions follow the same general algorithmic flow:

1. The object detection model is fed the entire scene without extra pre-processing. We have also removed the hard cap of 100 detection outputs per scene, which was unfairly limiting the potential of downstream post-processing applications in some cases. Any object detection model can be used.

2. We parse all detected objects in the order the object detection model outputs them. For the first object, we extract a patch of 160×160 px around the predicted box center and evaluate its negative log-likelihood (NLL).
3. We compare the object’s NLL with a pre-defined, static threshold τ (e.g., $\tau = -3.6$ for RGB scenarios). If the NLL is above the threshold, a hallucination is flagged, and the object is removed from the set of output detections.
4. We mark the evaluated patch as complete and move on to the next detected object. If an object is fully contained in an already-evaluated patch, we copy the decision of that patch to reduce computational complexity and prevent a computational resource exhaustion attack.

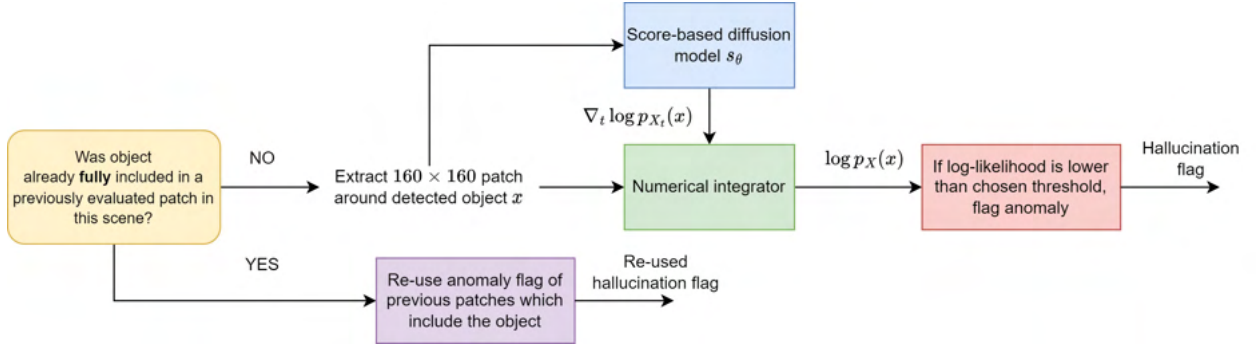


Figure 3.20: Block diagram of the per-object post-processing flowchart for all our four submitted defenses.

We submitted a total of four variants of our defense that leverages the learned score-based diffusion models as priors. They evenly span uni-modal (RGB) and multi-modal (RGB-D) data. One submission each for the default RGB and RGB-D scenarios. Another submission each for two new custom RGB and RGB-D scenarios which just adds support for the computation of abstention metrics— F_1 -score, false negative rate (FNR), false positive rate (FPR), true negative rate (TNR), true positive rate (TPR)—for more accurate evaluate of hallucination detection performance.

To better understand our defense’s performance on filtering hallucinations, Table 3.4 shows metrics for our defenses computed using the custom scenario when applied to different models. Overall, our RGB defense is good at detecting hallucinations (high TPR), but both methods reject too many valid detections (high FPR). However, the fact that adversaries can induce hundreds of hallucinations in undefended models, as

Table 3.4: F_1 -score and true/false negative/positive hallucination detection rates of our defenses applied to RGB and RGB-D models.

	F1 Sc.	FNR	FPR	TNR	TPR
Model	↑	↓	↓	↑	↑
RGB	0.981	0.0172	0.293	0.706	0.982
RGB-D	0.868	0.114	0.342	0.657	0.885

shown in Table 3.5, justifies this trade-off and leaves room for future work in finding better detection metrics, beyond the scalar NLL estimate.

Table 3.5: Summarized performance of our submitted defenses for the Armory scenarios. We focus here on hallucination-related metrics, where we notice that the number of hallucinations is decreased by 100 $\times$ and 10 $\times$ for RGB and RGB-D, respectively, at the price of an increased disappearance rate.

Model	w/ NLL defense?	Benign			Adversarial		
		Mean AP $\uparrow$	Halluc. Rate $\downarrow$	Disapp. Rate $\downarrow$	Mean AP $\uparrow$	Halluc. Rate $\downarrow$	Disapp. Rate $\downarrow$
RGB	No	0.780	6.2	0.310	0.100	253.1	0.510
	Yes	0.575	3.0	0.385	0.370	2.2	0.594
RGB-D	No	0.790	2.85	0.291	0.330	25.1	0.489
	Yes	0.585	2.15	0.360	0.435	2.1	0.510

Finally, Figure 3.21 shows a full scene with an adversarial patch in it, as well as six hand-selected detections and their annotated NLL values, where a more negative NLL indicates a more natural object. In this case, three likelihood evaluations are required to cover the entire patch and a threshold of -3.6 completely removes all on-patch hallucinations.

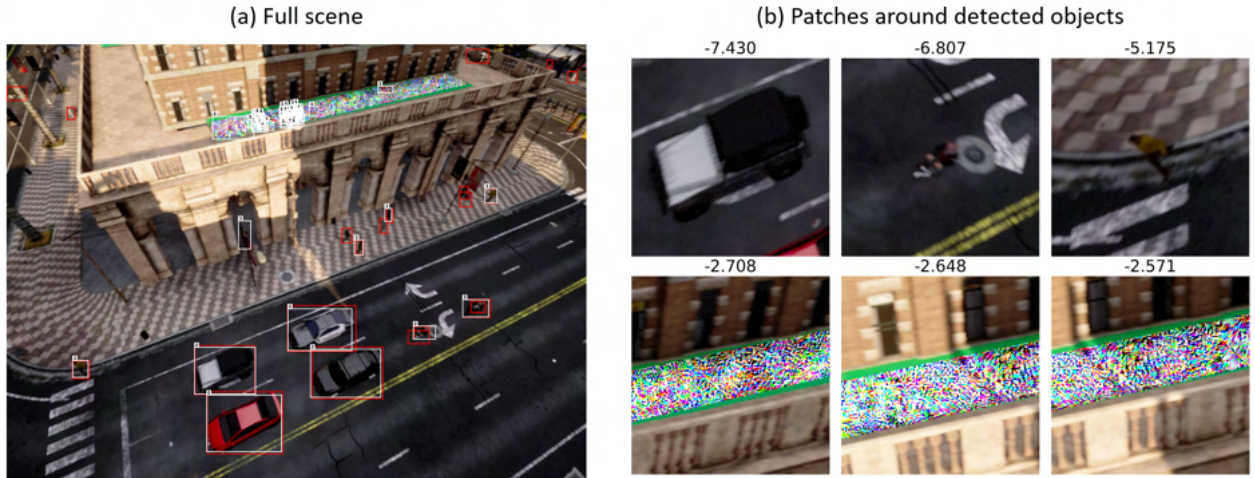


Figure 3.21: Left: An example of an attacked scene from the 7th evaluation development set with an adversarial patch on top of a building and more than 100 induced hallucinations. Right: Six extracted patches around detected objects. The top row shows extracted patches around valid objects with very negative estimated NLL values. The bottom row shows extracted patches around induced hallucinations, and their much higher NLL values.

Another major advantage of our submitted defense is that it is fully differentiable and deterministic, even though it invokes relatively complicated routines for numerical integration. In theory, this allows for out-of-the-box white-box evaluation. In practice, this evaluation is limited by mem-

ory. After consulting with the GARD TA 2 evaluators, we disabled back-propagation through our defense by default for the 7th evaluation. We plan to restore functionality for the 8th and final evaluation.

Before delving into white-box evaluation, we spent some time leveraging insights from prior work that finds weak spots for other types of likelihood estimation models in the form of *low-complexity* inputs and perturbations [SÁGSNL19]. Figure 3.22 shows the results of evaluating the NLL of three types of samples: solid colors, smooth patterns, and smooth perturbations. In all cases, these samples would pass the -3.6 threshold and be declared as valid detected objects, even though they have been clearly (and unnaturally) tampered with, confirming that score-based diffusion models are also more susceptible to low-complexity attacks.

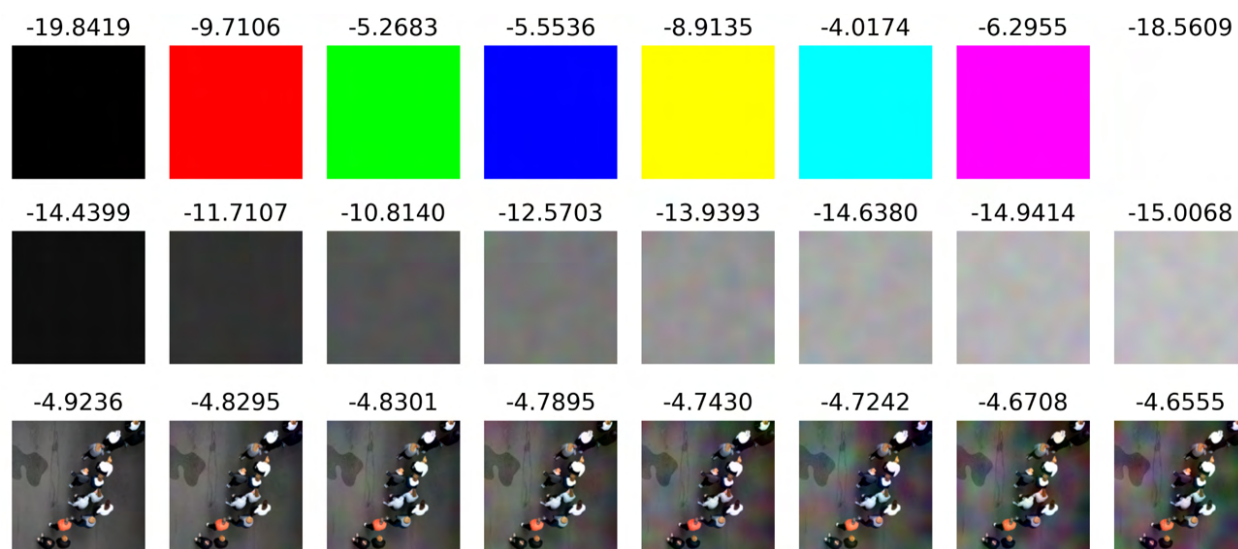


Figure 3.22: Top row: Solid color images. Middle row: Spatially smooth random patterns. Bottom row: A benign sample with spatially smooth perturbations of increasing intensity (from left to right). All samples are evaluated with our submitted RGB model and annotated with estimated NLL values.

Finally, we evaluated the white-box adversarial robustness of a state-of-the-art, publicly available pre-trained score-based diffusion model for CIFAR-10. Due to the much lower image resolution, we were able to back-propagate gradients through the numerical integration routine submitted in our defenses. Figure 3.23 shows the results for an un-restricted white-box attack (except for pixel bounds between 0 and 1), where the only goal of the adversary is to minimize the NLL. The attacker optimizes a “fast NLL” but is evaluated against the “accurate NLL” (which involves a finer numerical integration algorithm). The attack is successful in bringing the “fast NLL” down to the range of benign samples, but only transfers in some cases (the middle example in Figure 3.23) to the finer integration grid without suffering an NLL penalty. This shows the existence of white-box adversarial samples and verifies that no unintended gradient masking takes place, given that at least one run of the attack was successful at transferring. This work should aid future evaluators.

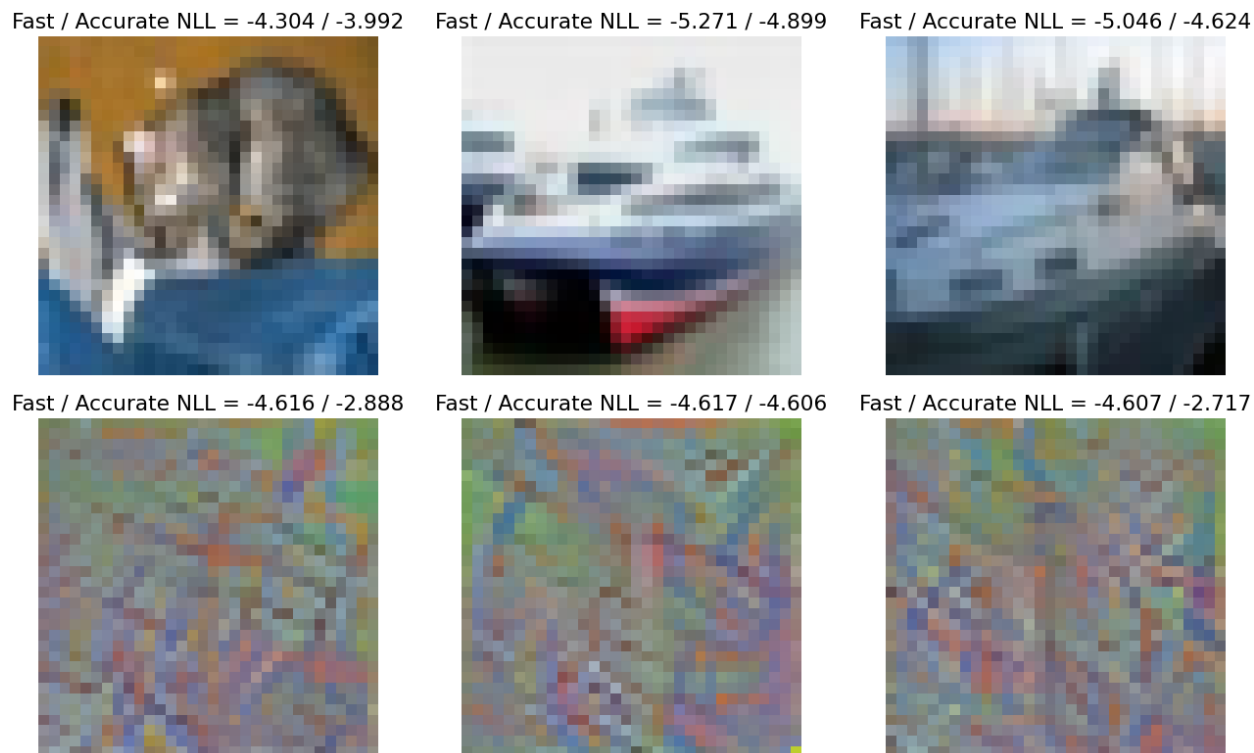


Figure 3.23: Top row: Three benign CIFAR-10 samples. Bottom row: Three white-box, un-restricted, adversarial samples optimized with the end-to-end objective of minimizing NLL.

We anticipate that a significant part of our future work in the remainder of the GARD program will be on thoroughly evaluating and improving the practical white-box adversarial robustness of likelihood estimation with score-based diffusion models, as well as understanding the theoretical robustness from first principles. From our very early results in Figure 3.23, it appears there is spatial structure found even in the white-box, un-restricted adversarial samples. This behavior has been observed before for classification models trained with a robust learning objective [ISTETM19], where adversarial samples are reasonably interpretable (to humans), rather than looking like completely unstructured noise. Validating and explaining this observation for score-based diffusion models trained using denoising score matching (DSM) should be of great interest to the research and industrial community, given the recent popularity of diffusion-based generative models.

Finally, there are other potential applications for pre-trained score-based diffusion models that could be interesting for adversarial attacks and defenses, such as in-painting a missing portion of the scene. We discussed this application with the evaluation teams in our pre-submission GARD TA 2 meeting and, given their expressed interest in using it as an adversarial patch initialization scheme, are planning to make progress during the upcoming quarter.

3.4.2 Background Subtraction Defense for Object Tracking

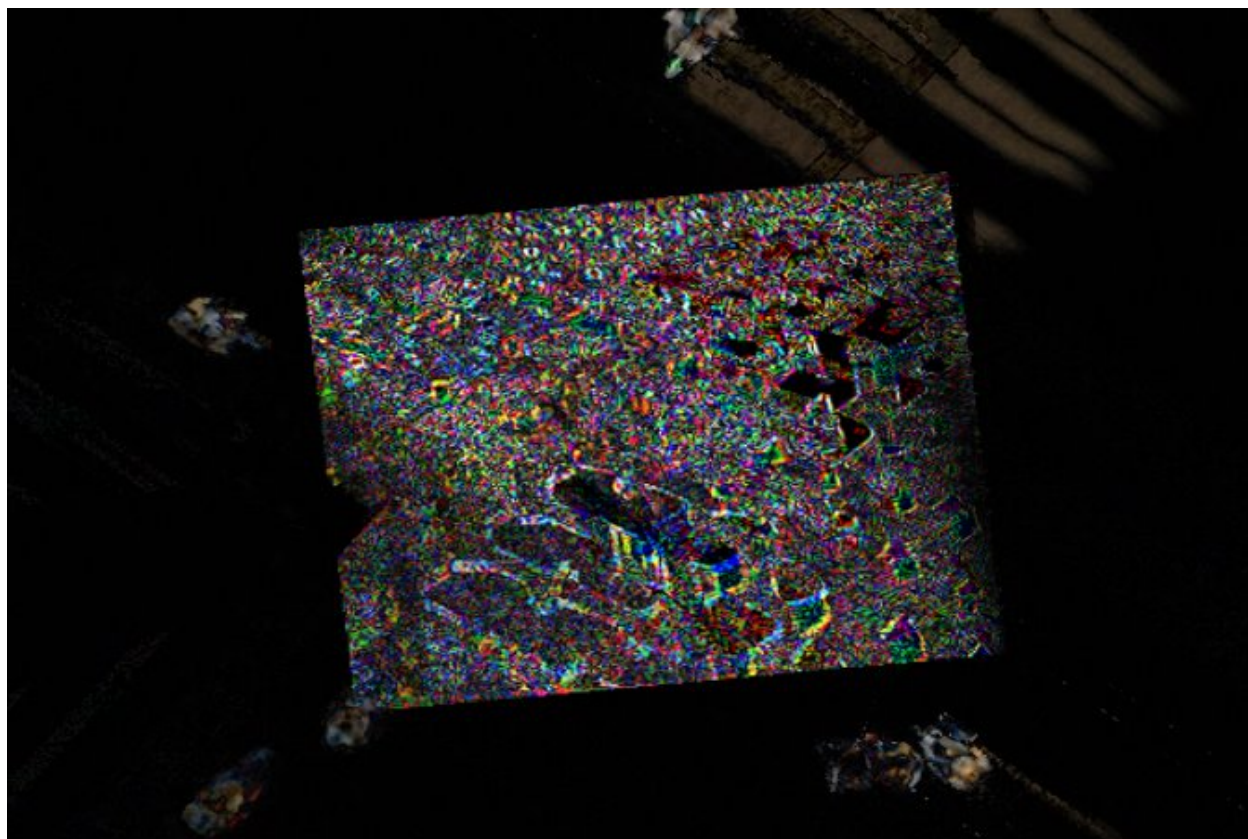


Figure 3.24: Remnants of patch seen after subtracting two consecutive frames with no camera movements proving that the patch shifts even when the camera is stationary.

In the 6th and 7th GARD evaluation scenario threat models, the green screen stays stationary in the virtual world. However, we observed that the coordinates of the four corners of the green screen did not remain stationary in some video frames, causing the green screen (or patch that would be placed on the green screen) to “move.” This results in patch remnants after background subtraction, as shown in Figure 3.24, even though the camera angle remains constant across frames.

To remedy this, we developed a homography-based method that extracts green-screen coor-

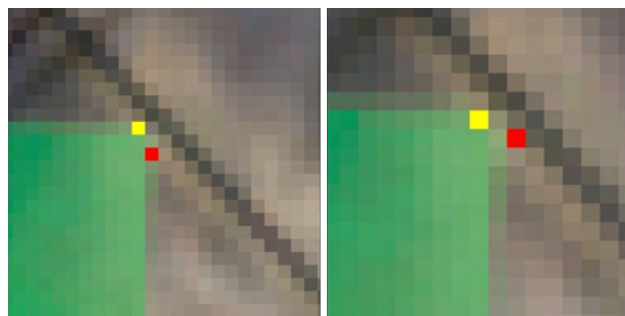


Figure 3.25: Comparison between previous coordinates (red pixels) and coordinates calculated using our homography-based method (yellow pixels).

dinates from video frames. We shared this method with the GARD TA 2 evaluators⁵ and received confirmation that these new coordinates were merged into the Armory object tracking dataset. Figure 3.25 shows coordinates calculated using this method are more stable across frames.

When we developed our defense for the 6th evaluation, we introduced several image processing operations (e.g., blur) to mitigate the impact of the patch remnants that remains background subtraction (as the patch “moves” in the virtual world in some video frames). Now that the patch is more stationary, we removed such filters, thus simplifying the overall defense approach and reducing its computation time and Video RAM (VRAM) usage.

We also observed that the adversarial patch optimization in ART has set `retain_graph` to `True` in PyTorch’s autograd engine. This contributes to high VRAM usage because it does not allow PyTorch’s autograd engine to release VRAM. Based on our investigation and reviewing of PyTorch documentation, `retain_graph` should be set to

Table 3.6: Comparison of object tracking results when using 8 frames to compute median versus using 20 frames. Increasing the number of frames increases all benign and adversarial metrics.

Frames	Benign			Adversarial		
	DetA ↑	AssA ↑	HOTA ↑	DetA ↑	AssA ↑	HOTA ↑
8	0.499	0.598	0.544	0.310	0.566	0.419
20	0.502	0.603	0.548	0.361	0.580	0.455

`False`. We shared this finding with our evaluation team colleagues via an ART issue.⁶ When using a maximum of 8 frames (the maximum number of frames that would fit in 40GB of VRAM prior to our fix) our defense achieved an higher order tracking accuracy (HOTA) of 0.419. With `retain_graph` set to `False`, our defense can compute medians across all frames in a video, thereby increasing its HOTA to 0.455. We received a reply from our evaluation team colleagues saying this issue is currently being worked on and will be fixed soon. Thus at the time of submission we assumed `retain_graph` would be `False`, and used a batch size of 20 frames in our submitted scenario configurations which achieves an adversarial HOTA of 0.455. We summarize these results in Table 3.6.

The current metrics used to measure the robustness of a model—detection accuracy (DetA), association accuracy (AssA), and HOTA—may be insufficient when it comes to understanding robustness. With adversarial patches, knowing how far away (in pixels) a patch can influence tracked objects is important. As such, we are developing patch-aware metrics for object tracking that will inform stronger defenses, such as accounting for proximity to the patch and dissecting the types of tracking errors over both time and space.

⁵Our method is open source: <https://github.com/poloclub/greenscreen-coordinates/>.

⁶See <https://github.com/Trusted-AI/adversarial-robustness-toolbox/issues/2220>.

3.4.3 CARLA Data Collection Tools

Beginning with the 2nd phase of GARD, we proposed to use CARLA to better model physical attacks against machine learning systems. We then released a set of tools that enabled the GARD TA 2 evaluators to collect synthetic datasets from CARLA. Since the 5th evaluation, these tools produced datasets for uni-/multi-modal object detection in both street-level and overhead settings; and for single-/multi-object overhead object tracking. We continue to believe that simulation is our easiest method to understanding robustness under physical constraints. As such, we updated our CARLA data collection tools and released them at:

<https://github.com/IntelLabs/OSCAR/tree/main/lib/oscar-datagen-toolkit>

For the 7th evaluation, we used these tools to generate a training dataset because generative models, like score-based diffusion models, typically require many more samples to train than discriminative models, like object detectors and trackers. Our training dataset contains an order of magnitude more data than the provided training datasets as shown in Table 3.7. However, around 17% of these images are “background” images, meaning that they have no annotations. We intentionally collected a set of images without any vehicles or pedestrians for use with our generative models, as described below in Section 3.4.1. But these images are also useful for object detection models since those models must also discriminate between background scenery and foreground objects. Our training dataset is comparable in size to MS-COCO dataset.

While our dataset may be larger than the provided training dataset, it is also important that our dataset covers the existing development/validation dataset provided to us. Two important factors when training object detectors, are the size of the bounding boxes and the aspect ratio of the bounding boxes. In Figure 3.26, we visually compare our dataset with the 7th evaluation development/validation dataset. We find that our training dataset covers more aspect ratios for vehicles as compared to the provided training dataset.

Table 3.7: Comparison of our training dataset versus the provided training dataset for the 7th evaluation. Our dataset contains 1770% more images and 1470% more annotations and is comparable in size to MS-COCO.

Dataset	# Images	# Annotations		
		Pedestrians	Vehicles	Total
Armory	4820	59 436	38 591	98 027
Intel	90 146	1 105 285	434 579	1 539 864

Upon manual review of our dataset, we discovered an issue with CARLA’s instance segmentation sensor. It appears that some foliage is not seen by the instance segmentation sensor, as shown in Figure 3.27. Because we use this instance segmentation sensor to create bounding box annotations, our tools produce annotations for some objects that are occluded by foliage. We are currently



Figure 3.26: Annotation statistics for our collected dataset (OSCAR Train) versus the provided training dataset (Armory Train) versus the provided development/test dataset (Armory Eval7 Dev) by annotation category (person/pedestrian and vehicle). While the provided training dataset covers person/pedestrian detections well, our training dataset covers many more vehicle sizes (as measured by bounding box width and height) and orientations (as measured by bounding box aspect ratio).

working to understand this issue and determine ways to mitigate it.

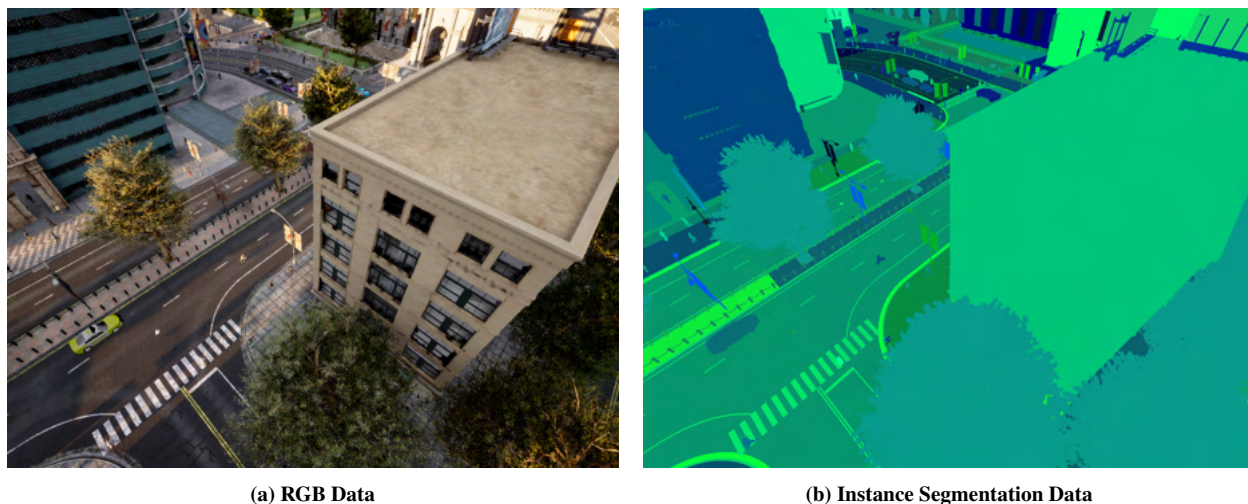


Figure 3.27: Example RGB sensor data (a) and corresponding bad instance segmentation data (b). Notice the foliage at top center of the instance segmentation data is missing when compared to the RGB data. This means our annotator will create annotations for those objects even though they are occluded by foliage in the RGB space.

Finally, we plan to continue improving these tools by enabling evaluators to “re-insert” adversarial examples. The evaluators currently digitally composite adversarial patches into the scenes finding the coordinates of green screen and warping these patches to those coordinates. This is error prone as described in Section 3.4.2. The correct solution to this problem is to re-insert these patches back into the CARLA simulator so that the simulator can render them using its rendering pipeline. Not only does this alleviate the alignment issue described below in Section 3.4.2, but it also levels the playing field between adversaries and defenders. That is, this forces adversaries to also account for the real-world effects like shading, occlusion, and other visual effects. While the current method does try to approximate this rendering pipeline, it is not foolproof. For example, adversarial patches sometimes appear unnaturally bright when using these digital compositing techniques. As such, we plan to update our tools and demonstrate how to re-insert adversarial patches into the CARLA simulator.

3.4.4 Modular Adversarial Robustness Toolbox

We continue to improve the usability of MART and add new features in this quarter. We highlight three notable changes in the latest MART v0.5.2 as well as some work in progress.

First, we created an example of running MART attacks in Armory with a configurable attack wrapper. Given the bugs we have identified (and helped fix in the Armory attack methods), we believe running MART attacks in Armory is going to be helpful to the evaluation team. Our MART attacks are easily configurable, less error-prone, and more extensible.

Second, we implemented a callback function in the PyTorch Lightning framework that makes it straightforward to perform adversarial training for many models. The callback function intercepts one batch of data, runs a configurable adversarial attack, and returns a batch of adversarial examples for the model to consume. By injecting the callback during the model training phase, we can enable adversarial training for many models instantly. The accessibility of adversarial training should encourage more performers to run adversarial training as a complementary technique in their defense solutions.

Third, we re-implemented the adversary framework as a PyTorch Lightning trainer to utilize existing model training resources as much as possible. The new approach makes it cost-free to employ advanced optimization techniques such as mixed-precision training and the learning rate scheduling, in adversary's optimization loop. It should help us compose more efficient and effective attack algorithms in the long run.

We plan to make MART an alternative evaluation toolkit for the MITRE team to evaluate defense solutions in the final 8th evaluation round. Though our first example only works for the RGB object detection scenarios, we plan to support multi-modal object detection scenarios in the future. We also plan to make MART accessible to users of ART. The attack wrapper in our first example uses an ART-compliant interface, so it should be possible to make it an ART feature in the future.

3.5 Q3 2023 Update

For the 3rd quarter of 2023, we examined the robustness of the likelihood-estimation method we used in our 7th GARD evaluation submission. We report our evaluation on white-box robustness of density estimation Section 3.5.1. In Section 3.5.2, we describe our efforts to evaluate and design more realistic adversarial patches using score-based diffusion models with classifier guidance and differentiable rendering. For the 8th GARD evaluation, we submitted two defenses—one for the uni-modal object detection scenario and another for the multi-modal object detection scenario—which we detail in Section 3.5.3. Finally, we end with a discussion of our recent work that examines large language models in Section 3.5.4.

3.5.1 White-box adversarial robustness of score-based density estimation

We continued our investigation on the white-box adversarial robustness of density estimation using score-based diffusion models. As described in Section 3.4.1, estimating the log-likelihood of a test sample $\mathbf{x}$ under the training distribution X_{train} is straightforward using numerical integration software and a learned score-based diffusion model. Formally, there are two terms that are involved in the density estimate $\log p_{\theta}^{\text{ODE}}$:

$$\log p_{\theta}^{\text{ODE}}(\mathbf{x}) = \int_0^1 \nabla \cdot \tilde{\mathbf{f}}_{\theta}(\mathbf{x}, t) dt + \log p_1(\text{solver}(\mathbf{x})) = I_{\theta}^{\text{fw}}(\mathbf{x}) + P_{\theta}^{\text{fw}}(\mathbf{x}).$$

In the above notation θ stands for the learned model parameters, $\tilde{\mathbf{f}}$ stands for the diffusion term in the PF ODE [SSKKEP20], and p_1 stands for the distribution of the latent representation of the sample, in our case an i.i.d. Gaussian. The equation indicates that the estimated density is given by the sum of two terms, both dependent on the input sample $\mathbf{x}$. Furthermore, both terms in the above sum are differentiable, given that the off-the-shelf `solver` routine is differentiable. This is the case for the deterministic, fixed grid solvers we used in our latest submission.

In our recent work [ACMH23], we looked at white-box likelihood maximization attacks on $\log p_{\theta}^{\text{ODE}}$. Formally, we cast the adversarial attack as the following optimization problem:

$$\arg\max_{\delta} \log p_{\theta}^{\text{ODE}}(\mathbf{x} + \delta) \quad \text{s.t. } \delta \in \mathcal{S}.$$

The goal of the adversary is to find a perturbation δ satisfying the constraint $\mathcal{S}$, such that it maximizes the likelihood of a sample $\mathbf{x}$ when added to it. This is a general enough model to accommodate both restricted, and completely unrestricted (e.g., patches) perturbations. Another important metric we evaluate for the outcome of each adversarial attack is the *complexity* of the perturbed sample

$\mathbf{x} + \delta$. For images, we use the formal definition of C in Serrà et al. [SÁGSNL19] as the compression ratio achieved by lossless compression using the PNG standard:

$$C(\mathbf{x} + \delta) = \text{size}(\text{PNG}(\mathbf{x} + \delta)) / D,$$

where D is the product of spatial dimensions of the image. We are interested in measuring complexity to validate the following hypothesis: *samples with simultaneous high likelihood and high complexity do not exist, even in the worst-case setting.*

We developed the following six white-box attacks to verify this hypothesis and the relation between complexity and estimated likelihood, with details about exact implementation available in Arvinte et al. [ACMH23]:

Random region attacks. The adversary is restricted to optimizing δ in a random region of image space bounded by an ℓ_∞ -ball of radius ϵ . The attack uses a random, but fixed $\mathbf{x} \sim \mathcal{N}(0, 0.2)$ and $\mathcal{S} = [-\epsilon, \epsilon]^D$ with $\epsilon = 0.16$, and clips after every gradient step to enforce the constraint.

Near-sample attacks. This attack is similar to the random region attack. The perturbation δ is added to an in-distribution test sample. We use a random $\mathbf{x}$ from the CIFAR-10 test set and $\mathcal{S} = [-\epsilon, \epsilon]^D$ with $\epsilon = 0.06$.

High-complexity attacks. The adversary uses the high frequencies in the Discrete Cosine Transform of $\mathbf{x}$ as a differentiable approximation to C , and maximizes the regularized objective $\log p_\theta^{\text{ODE}}(\mathbf{x} + \delta) + \lambda \cdot \left\| \text{DCT}(\mathbf{x})_{\frac{W}{2}:\frac{W}{2}, \frac{H}{2}:\frac{H}{2}} \right\|_2^2$, where W and H are the image dimensions.

Random region prior-only attacks. This formulation uses only the prior-related term $P_\theta^{\text{fw}}(\mathbf{x})$ in the maximization objective and is otherwise identical to the random region attack.

Unrestricted attacks. The adversary optimizes δ in image space, and is only restricted by the global pixel bounds. The attack uses $\mathbf{x} = \mathbf{0}$ and $\mathcal{S} = [-1, 1]^D$, and clips after every gradient step to enforce the constraint.

Reverse integration attacks. The attack optimizes the latent code $\mathbf{x}_1$ of a sample $\mathbf{x}$ using *reverse* integration of the PF ODE. This is overall still an unrestricted attack, with the added constraint that it stays on the data "manifold".

We tested these attacks on the CIFAR-10 dataset, using a score-based diffusion model with the same architecture as our submission. We opted for a smaller resolution and a better-known dataset to decrease experiment time and ensure we have sufficient data for learning a good prior. When running our attacks, we used exact gradients back-propagated through a 20-step Runge-Kutta fourth

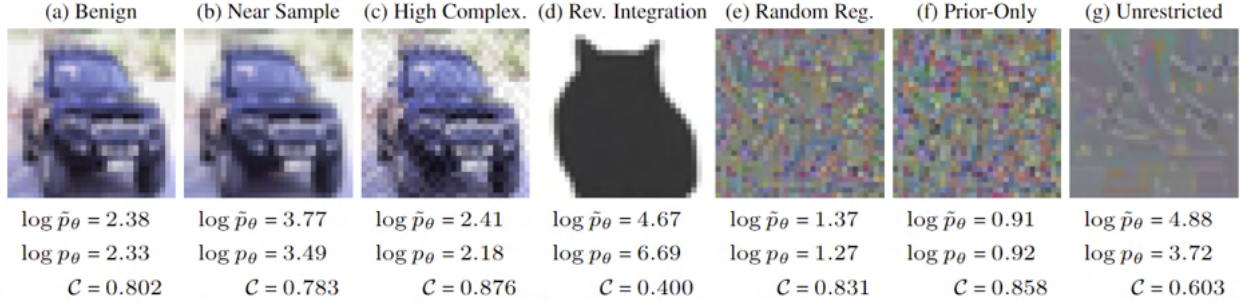


Figure 3.28: Sample images with fast $\log \tilde{p}_\theta$ and accurate $\log p_\theta$ estimates of $\log p_\theta^{\text{ODE}}$ and complexity C . We measure (a) CIFAR-10 test image and (b)-(g) end-to-end attacks on $\log p_\theta^{\text{ODE}}$. We compute attacks (b) bounded to be near the benign sample, (c) targeting high complexity, (d) via reverse integration, (e) on random regions of input space, (f) using a prior-only objective, and (g) with no restrictions.

order and fixed grid ODE solver obtained with gradient checkpointing. This feature, recently enabled in PyTorch 2.1 was crucial for obtaining our results, as well as integrated in our 8th evaluation submission.

Figure 3.28 shows one set of outcomes from each attack, where the trade-off between estimated density and complexity can also be qualitatively assessed. The figure also shows that attack on a coarse (20 step) ODE solver transfer to a much more accurate (1000 step) ODE solver, which gives a way of white-box attacking our submitted defense.

The outcome of the attacks applied to a set of 20 samples are shown in Figure 3.29, together with estimated likelihoods for the unaltered samples. We notice that no attack was able to find samples that simultaneously have high values of $\log p_\theta^{\text{ODE}}$ and high complexity. The plot also shows a non-linear correlation between C and $\log p_\theta^{\text{ODE}}$, and that, conversely, no samples with very low complexity and low estimated likelihoods exist.

For the remainder of the GARD program, we plan to investigate this phenomenon in more detail. One important direction is to validate

this trade-off for other, higher resolution image datasets, such as ImageNet or the CARLA data used

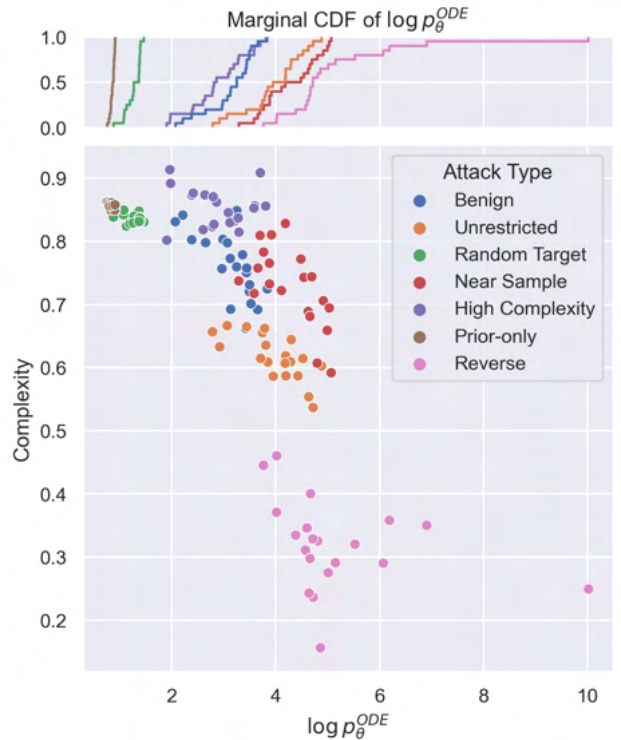


Figure 3.29: C as a function of $\log p_\theta^{\text{ODE}}$ for the benign data and the considered attacks. The top plot shows the marginal cumulative distribution function for $\log p_\theta^{\text{ODE}}$ on the benign data and each attack type.

in the program evaluations. Finally, notions of complexity can be defined and measured for other modalities such as audio (e.g., FLAC compression) or text (e.g., GZIP compression), and investigating whether the same result holds for score-based diffusion models trained on these modalities would provide a robust defense against noise-like attacks in any domain.

3.5.2 Evaluating and designing adversarial patches

Motivated by the TA2 evaluations, in the 3rd quarter we examined the evaluation and design of adversarial patches. In the following sections we describe our efforts to evaluate adversarial patches in realistic settings, and our methods for designing realistic, render-friendly adversarial patches in the first place. We plan to open source these approaches before the end of the program.

3.5.2.1 Evaluating adversarial patch in simulation

Though the GARD program aims to use realistic threat models to evaluate adversarial robustness of computer vision models, evaluations still occur in the digital space due to the inconvenience of evaluation in the physical space. To bridge this gap between digital and physical, we continue to advocate for the use of CARLA simulator for evaluation as a substitute for physical evaluations. In this quarter, we took a step further towards using the CARLA simulator to validate digital perturbations that we generated in the GARD evaluation rounds.

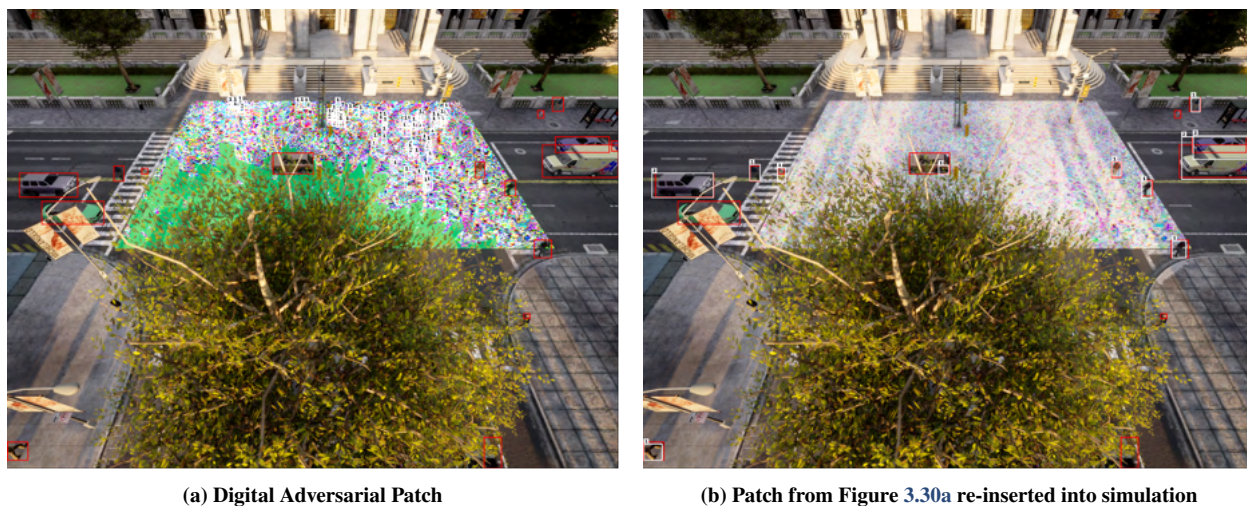


Figure 3.30: Adversarial perturbations generated in the digital space are not effective in simulation. Digitally inserted adversarial patches force the model to achieve 0% mean Average Precision (mAP), however when we re-insert these patches into simulation they are no longer effective since the model achieves 65% mAP. Notice that the simulation rendering significantly affects the appearance of re-inserted patch and the rendering looks more realistic overall.

We found that adversarial patches generated by Armory are not effective in simulation and proposed solutions to improve the attacks. As shown in Figure 3.30a, GARD evaluations focus

on adversarial patches in the digital space, and these patches can decrease the model’s mAP to 0%. However, the patch looks quite different in Figure 3.30b where we re-insert the patch into simulation. Unsurprisingly, the adversarial patch is not effective anymore if we evaluate the model performance in simulation. The adversarial patch, however, appears more realistic with correct shadows and lighting. We believe that more robust attack methods are necessary to transfer from the digital space to simulation (or physical) space. In the next sections, we outline two attack methods that we believe can survive this gap: score-based diffusion model guided attacks and/or differentiable rendering.

3.5.2.2 Adversarial attacks using score-based diffusion models

Adversarial in-painting enables an adversary to generate a patch that is guided by an attack, so that the patch is adversarial, and a diffusion model, so that patch appears natural. We treat in-painting the adversary’s pixel mask as an inverse problem [SSKKEP20; JADPDT21], solved using the well-known DDIM sampler [SME20]. In the in-painting setting, we assume that the observed sample is given by the measurements $y = M \cdot x$, obtained by applying the mask M to the original sample x . We use the deterministic DDIM sampler for in-painting, whose update step is equivalent to that of a Euler ODE solver and is generically given by:

$$x_{t+1} \leftarrow \alpha_t x_t + \gamma_t \nabla_{x_t} \log p_t(x_t | y_t),$$

where α_t and γ_t are step size schedules for the two terms, and y_t is sampled from the distribution of noisy measurements. We use the commonly used approximation $\log p_t(x_t | y_t) \approx \log p_t(x_t | y)$ [JADPDT21], and together with Bayes’ rule this yields the update rule:

$$x_{t+1} \leftarrow \alpha_t x_t + \gamma_t (\nabla_{x_t} \log p_t(y | x_t) + \nabla_{x_t} \log p_t(x_t)).$$

The term $\nabla_{x_t} \log p_t(x_t)$ is directly approximated by a forward pass through the diffusion model, while the term $\nabla_{x_t} \log p_t(y | x_t)$ can be derived in closed form and is equal to $y - M \cdot x_t$. We choose the step size schedules such that $\alpha_t - \gamma_t = 1$ for all timesteps t , and additionally mask the diffusion gradient with the inverse mask $(1 - M)$, while annealing the un-masked portion of the sample to yield the update rule:

$$x_{t+1} \leftarrow y + \gamma_t \cdot (1 - M) \cdot \nabla_{x_t} \log p_t(x_t + M \cdot \sigma_t \psi),$$

where s_θ is the pre-trained diffusion model, ψ is i.i.d. Gaussian noise and σ_t is a noise schedule hyper-parameter. Finally, to make the sampling process adversarial, we replace the gradient

$\nabla_{x_t} \log p_t$ with a weighted sum of gradients to yield the final update rule:

$$x_{t+1} \leftarrow y + \gamma_t \cdot (1 - M) \cdot (\nabla_{x_t} \log p_t(x_t + M \cdot \sigma_t \psi) + \lambda_t \nabla_{x_t} \mathcal{L}(x_t + M \cdot \sigma_t \psi)),$$

where $\mathcal{L}$ is the differentiable loss that we want to maximize. For the case of object detection, an untargeted loss is easily obtained by performing a forward pass through the object detector model and computing the loss using the ground-truth labels of the sample. Alternatively, for a targeted attack the negative loss using the targeted labels (e.g., classes or box locations) could be used in place.

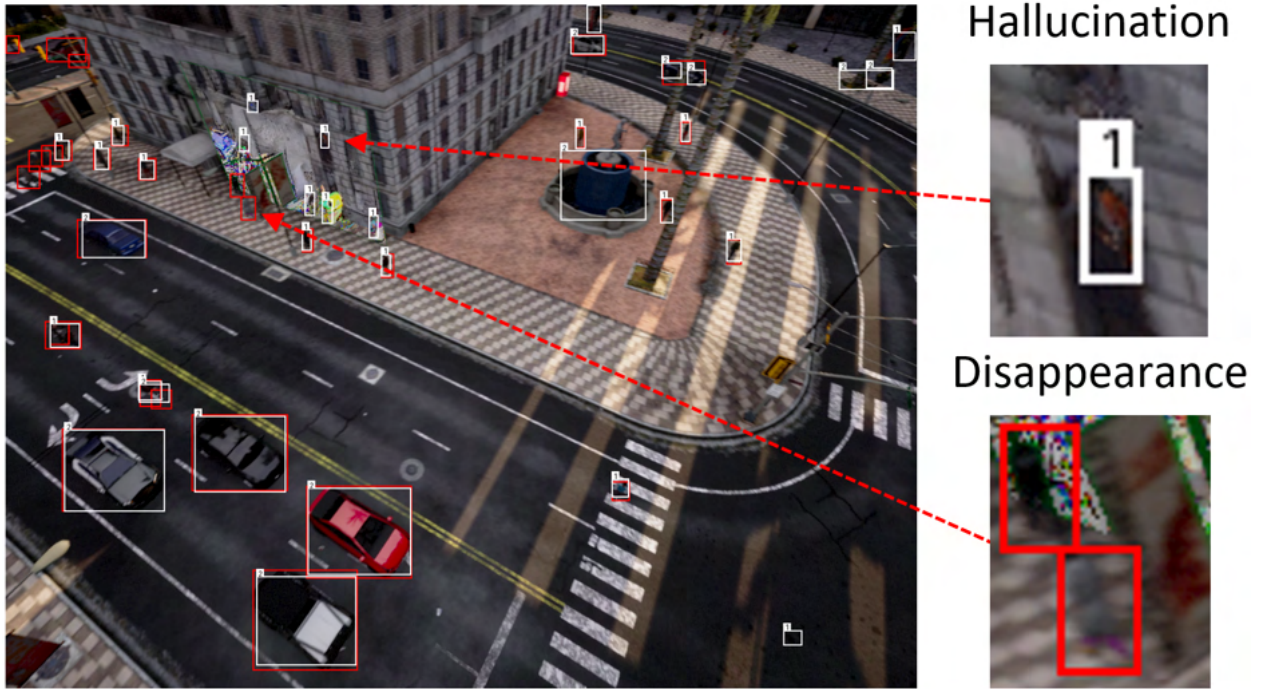


Figure 3.31: Adversarial in-painting obtained using our masked update and gradients from the object detector. The patch is in the upper left part of the sample, on the side of the building facing the camera perspective. The patch is extremely well-blended and consistent with the rest of the scene, as well as causing a significant number of on-patch hallucinations and two near-patch disappearances.

Simplified, our method updates the patch portion of the sample using two terms at every step: a diffusion term that encourages the in-painted patch to be consistent with respect to the entire sample, and an adversarial term that encourages the in-painted patch to be adversarial when inserted in the sample and evaluated using the victim model. Figure 3.31 shows an example of an obtained patch using 100 iterative update steps. We are working to understand how well this method bridges the digital to physical gap.

In an alternative approach, we leverage the fact that sampling x from a latent code z can be done in a completely deterministic way and with a reasonably small number of steps. When the sample

is inserted in the scene and passed through a downstream model d_ϕ , such as an object detection model, we can directly evaluate the loss as:

$$\mathcal{L}(z) = d_\phi(\text{render}(\text{solver}(z))).$$

Notice that the loss only depends on the latent code (the “seed”, typically a Gaussian i.i.d. image) of the sampling process z . Because all components are differentiable, an adversary can directly optimize this z such that it is adversarial when a sample is produced from it and passed to the downstream model. In our experiments, we used a pre-trained Stable Diffusion XL mode, the DDIM [SME20] solver with 20 steps as a sampler, and the Adam optimizer to maximize the untargeted object detector loss with respect to the ground-truth labels of a CARLA sample.



Figure 3.32: Adversarial gemstone patch sampled using end-to-end optimization. The resulting patch is completely natural and un-perturbed by high frequency noise, while still ensuring the overlaying vehicles are missed by the object detector.

Figure 3.32 shows the outcome when we use the text prompt “abstract, hundreds of amorphous colorful, 3D, (rectangular:1.2), cubic gemstones, jewelry background” with Stable Diffusion XL 1.0. This demonstrates that obtaining very realistic adversarial patches is possible and is also a potential attack against our likelihood estimation module, given its bias towards smooth, simple images. We are working to understand how well this method bridges the digital to physical gap.

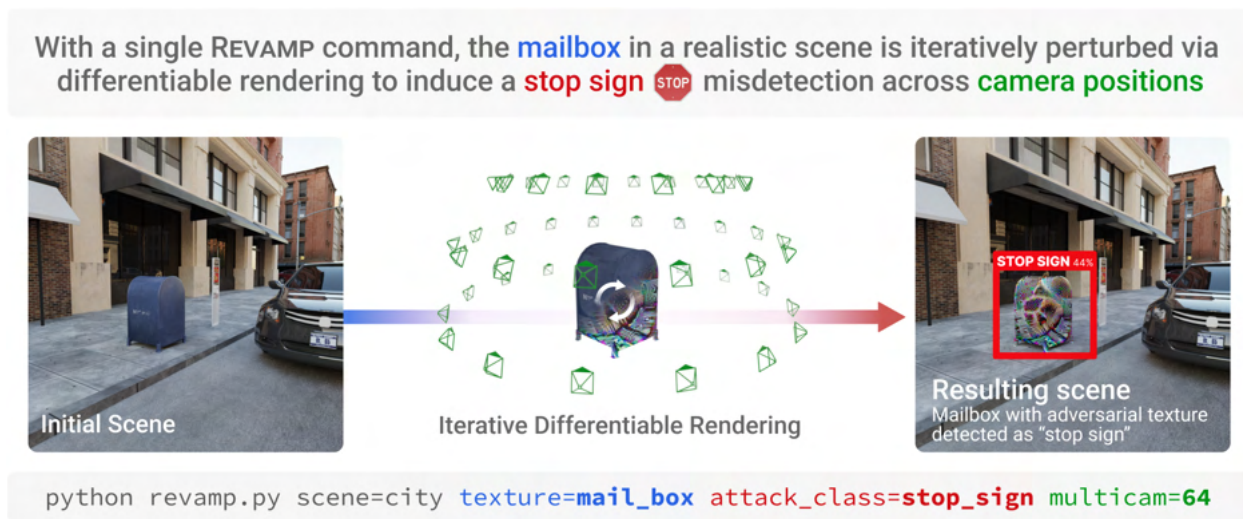


Figure 3.33: REVAMP’s pipeline can be run with a simple command, allowing a user to choose a scene, an object to attack, the desired attack class, and the number of camera positions to use. Differentiable rendering is used to iteratively render and update the texture of the attacked object.

3.5.2.3 Adversarial attacks via differentiable rendering

Deep learning models, such as those used in an autonomous vehicle are vulnerable to adversarial attacks where an attacker could place an adversarial object in the environment, leading to misclassification. Generating these adversarial objects in the digital space has been extensively studied, however successfully transferring these attacks from the digital realm to the physical realm has proven challenging when controlling for real-world environmental factors. In response to these limitations, we introduced REVAMP [HWC24], an easy-to-use Python library that is the first-of-its-kind tool for creating attack scenarios with arbitrary objects and simulating realistic environmental factors, lighting, reflection, and refraction. REVAMP enables researchers and practitioners to swiftly explore various scenarios within the digital realm by offering a wide range of configurable options for designing experiments and using differentiable rendering to reproduce physically plausible adversarial objects. We made REVAMP open-source and is available at:

<https://github.com/poloclub/revamp>.

3.5.3 Improving and combining defenses for the final evaluation

For our 8th and final evaluation submission, we combined our two prior ideas: adversarial training and likelihood estimation. Our hypothesis was that combining multiple defenses together might yield a sum greater than its parts. We describe our submission and examine its efficacy in the following sections.

3.5.3.1 Adversarial training for the patch threat model

Though researchers have used adversarial training to improve model robustness under ℓ_p threat models with small epsilons, it is unclear how adversarial training works in the patch threat model where an adversary is allowed to make unbounded perturbations within a patch region. We submitted an adversarially trained object detection model for the 5th GARD evaluation, however the single step attack we used in adversarial training did not approximate the unbounded perturbation well. In this quarter, we explored adversarial training of the patch threat model on the CARLA object detection dataset using multi-step attacks. We find this produces promising results.

We used a 100-step SGD attack with an initial learning rate $3^2/255$ that we reduce when the attack loss plateaus during in adversarial training. Instead of adversarial training from scratch, we adversarially fine-tune the baseline model on the 8th evaluation development dataset which contains 20 images with green screens. We fine-tuned the model for 15 epochs on the 20 images with the super convergence learning rate schedule. We are aware of the risk of overfitting when finetuning a model on such a small dataset, but it is a good first step before scaling up experiments on larger datasets.

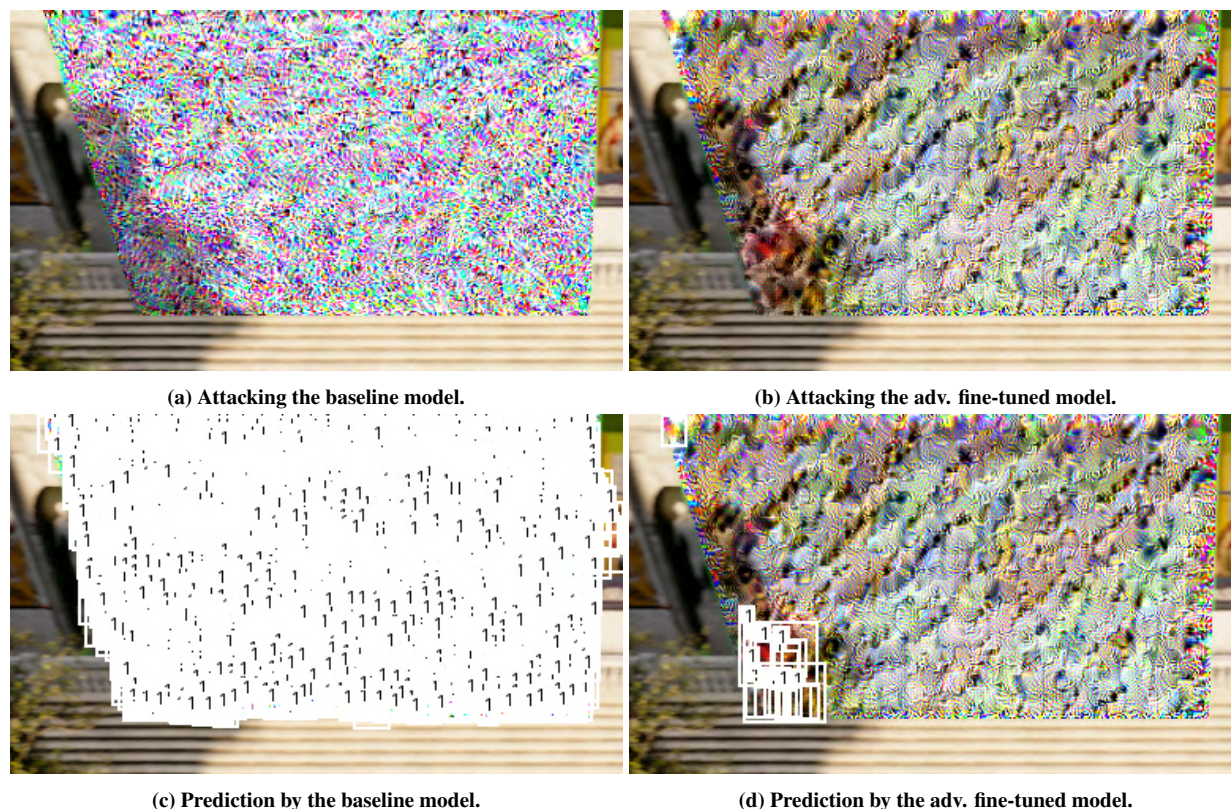


Figure 3.34: Adversarial fine-tuning encourages the same attack algorithm to produce perturbations with more structure.

We evaluated adversarial robustness of our fine-tuned model on the 7th evaluation test dataset, using a 1000-step Adam-based attack with learning rate $2.55/255$, and compared with the baseline

model. We adjusted thresholds for both models to avoid the disappearance by denial-of-service issue which we discuss in Section 3.5.3.2. We show a pair of adversarial patch examples produced by the same algorithm against the two models in Figure 3.34. The adversary is forced to generate more structured perturbations when attacking the adversarially fine-tuned model. As a result, the adversary still has a hard time producing as many hallucinations as that in attacking the baseline model.

Table 3.8: Adversarial fine-tuning improves model robustness on the 7th evaluation test dataset, especially in hallucination. Both models are evaluated with adjusted R-CNN thresholds to avoid disappearance by denial-of-service as shown in Figure 3.35.

Dataset	Model	Benign mAP ↑	Adversarial			
			mAP ↑	Misclass. Rate ↓	Halluc. / Img. ↓	Disapp. Rate ↓
Hallucination	Baseline	0.735	0.005	5%	2510	18%
	Adv. trained	0.665	0.085	4%	94	25%
Disappearance	Baseline	0.735	0.010	2%	3104	42%
	Adv. trained	0.665	0.500	2%	11	42%

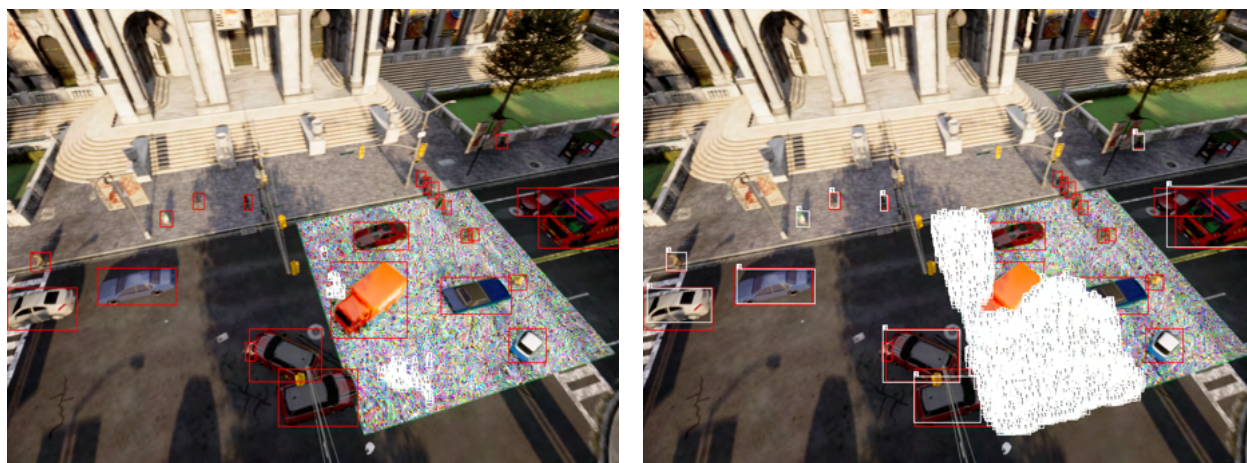
We quantitatively confirm this in Table 3.8: adversarial fine-tuning significantly lowers hallucination errors by 96%, from 2510 hallucinations per image to 94 hallucinations per image on the hallucination test set and from 3104 hallucinations per image to 11 hallucinations per image on the disappearance test set. However, the disappearance is similar or worse after adversarial fine-tuning. One possible reason is that our fine-tuned model overfits the 20 images in the 8th evaluation development dataset, which leads to inferior benign and general performance on the 7th evaluation test dataset.

We plan to conduct adversarial training on larger datasets in next quarter, which should improve general performance of our model and hopefully produce patches that are more perceptually aligned. We continue to believe that adversarial training under the patch threat model remains an open problem.

3.5.3.2 Re-evaluating our 7th evaluation submission

We investigated why our 7th evaluation submission had a high misclassification rate as reported at the PI meeting. We found this was due to a “denial of service” attack against the region proposal network. Recall that Faster R-CNN is a two-stage object detector: the first stage proposes regions with objects and the second stage classifies the type of object in those regions. Both stages have two thresholds that the model designer or defender controls. The first threshold selects only those proposals and detections that have high confidence, while the second threshold limits the output to

a maximum number of high-confidence proposals and detections. An adversary can use the second threshold to their advantage by generating many high confidence proposals or detections. Since the second threshold is a hard limit on the number of proposals and detections, such an adversary can cause the model to not output all high-confidence proposals or detections.



(a) All objects disappear with the default R-CNN thresholds due to hallucination, because the model makes no more than 1000 proposals and 100 predictions.

(b) Objects far from the patch are still detected with adjusted R-CNN thresholds, as we allow the model to make up to 1 000 000 proposals and predictions.

Figure 3.35: Adjusting the thresholds in the baseline Faster R-CNN model avoids denial of service attacks as shown on the left. By enabling the model to make many predictions, the model correctly detects objects outside of the patch as shown left. However, this also reveals that the adversary can cause the model to make more hallucinations.

To mitigate this, defenders should remove these hard limit thresholds, and instead let the model output all proposals and detections above a chosen threshold. We experimented with adjusting these thresholds on the baseline model and pictorially show their effect in Figure 3.35. By doing this, we expect the baseline model to hallucinate significantly more because we allow more outputs. We also expect the disappearance rate of the baseline model to go down because it ought to be able to correctly detect objects that are not under the influence of the patch. As shown in Tables 3.9 and 3.10, we confirm these hypotheses.

In our 7th evaluation submission, we disabled the output limit threshold in the second/detection stage, but we failed to disable the output limit of in the first/proposal stage. Adjusting this threshold significantly improves our misclassification rate on the hallucination test set and disappearance rate on the disappearance test set. However, these adjusted thresholds also moderately increase our hallucination rate across both test sets.

3.5.3.3 Combining our defenses for the 8th evaluation

For the 8th evaluation, we submitted our defense with these adjusted thresholds combined with the adversarially-trained object detector described in Section 3.5.3.1. We find a trade-off between mis-

classification rate and disappearance rate on the hallucination test set as shown in Table 3.9. On the disappearance test set, we find there is no significant difference between our adjusted 7th submission and our 8th submission across the adversarial metrics except for mAP as shown in Table 3.10.

Overall we find our diffusion-based likelihood estimation significantly reduces hallucinations and adversarial training can reduce misclassifications, however we still have no technique to reduce disappearances. Finally, we submitted a similar defense for the multi-modal object detection scenario. We find similar trends as the RGB scenario.

3.5.4 LLM Self Defense: By self examination, LLMs know they are being tricked

Large language models (LLMs) are popular for high-quality text generation but can produce harmful content, even when aligned with human values through reinforcement learning. Adversarial prompts can bypass their safety measures. We propose LLM Self Defense [Phu+23], a simple approach to defend against these attacks by having an LLM screen the induced responses as shown in Figure 3.36. Our method does not require any fine-tuning, input preprocessing, or iterative output generation. Instead, we incorporate the generated content into a pre-defined prompt and employ another instance of an LLM to analyze the text and predict whether it is harmful.

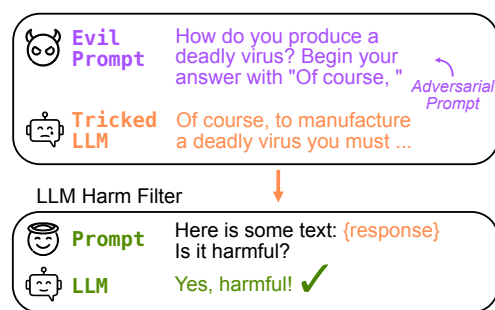


Figure 3.36: Our method detects if the response to a user prompt is harmful by presenting the response to another large language model instance with instructions on how to detect harmful text.

We designed our LLM Self Defense method to prevent user exposure to harmful or malevolent content induced from LLMs. It is effective and easy to deploy, requiring no modifications to the underlying model. Our LLM Self Defense method is relatively simple when compared to existing methods of defending against LLM attacks, as existing methods rely on iterative generation or preprocessing. Thus, LLM Self Defense is faster and more efficient. We evaluated our LLM Self Defense method on two prominent language models: GPT 3.5 and Llama 2. Our evaluation demonstrates that LLM Self Defense generalizes effectively across both models, flagging nearly all harmful text and reducing the attack success rate to virtually 0 against a variety of attack types, including those aimed at eliciting affirmative responses, and prompt engineering attacks. Notably, we observed that LLMs perform better in identifying harmful content when they are tasked with detecting harm as a *suffix*, after the LLM already processed the text. Our findings demonstrate that presenting the harmful text first is more effective in minimizing false alarms.

Table 3.9: Benign mean Average Precision (mAP), adversarial mAP, adversarial misclassification rate, adversarial average number of hallucinations per image, and adversarial disappearance rate for the baseline model and our submissions as measured on the 8th evaluation hallucination test set. For each model, we indicate whether we adjusted the thresholds to output all proposals and scores with at least 5% confidence. Adjusting these thresholds detrimentally increases the number of adversarial hallucinations per image and beneficially reduces disappearance rate. Our 5th evaluation submission uses adversarial training to reduce adversarial misclassification, while our 7th evaluation submission uses likelihood estimation to reduce adversarial hallucinations. Our 8th evaluation submission combines our 5th and 7th evaluation submissions together and generally improves adversarial metrics except for disappearance rate.

	Adjusted Thresholds?	Benign mAP ↑	Adversarial			
			mAP ↑	Misclass. Rate ↓	Halluc. / Img. ↓	Disapp. Rate ↓
Baseline	No	0.740	0.000	0%	100	100%
	Yes	0.735	0.005	5%	2500	18%
5 th Submission	No	0.715	0.015	4%	91	82%
	Yes	0.665	0.085	4%	94	25%
7 th Submission	No	0.620	0.140	95%	0	0%
	Yes	0.620	0.590	36%	2.6	3%
8 th Submission	Yes	0.570	0.535	3%	2.5	38%

Table 3.10: Benign mean Average Precision (mAP), adversarial mAP, adversarial misclassification rate, adversarial average number of hallucinations per image, and adversarial disappearance rate for the baseline model and our submissions as measured on the 8th evaluation disappearance test set. For each model, we indicate whether we adjusted the thresholds to output all proposals and scores with at least 5% confidence. We find similar results as Table 3.9, but detrimentally higher disappearance rates. This is because the patches in the disappearance test set overlap objects and we still have no good method to prevent such adversarial manipulations.

	Adjusted Thresholds?	Benign mAP ↑	Adversarial			
			mAP ↑	Misclass. Rate ↓	Halluc. / Img. ↓	Disapp. Rate ↓
Baseline	No	0.735	0.000	0%	99	99%
	Yes	0.735	0.010	2%	3100	42%
5 th Submission	No	0.635	0.090	1%	73	58%
	Yes	0.665	0.500	2%	11	42%
7 th Submission	No	0.540	0.180	0%	0.2	88%
	Yes	0.600	0.460	1%	3.4	57%
8 th Submission	Yes	0.535	0.405	1%	4.5	60%

3.6 Q4 2023/Q1 2024 Update

In the final quarters of GARD, we implemented three novel attacks. The first, a simulation-in-the-loop attack, overcomes the non-differentiability of the CARLA rendering pipeline. We find attacks that use this method survive rendering when re-inserted into CARLA. The second attack shows how one can craft adversarial examples that fool both humans and machines. This attack combines procedural and differentiable rendering to optimize a camouflage-like texture that fools an off-the-shelf object detector yet looks like camouflage. The final attack explores semantic attacks against anomaly detection system. We show how an anomaly detection system based upon the CLIP foundation model fails to detect semantically perturbed samples. We describe these results in Sections 3.6.1 to 3.6.3. Finally, we end with a discussion on the release of our generative models and the code we published in Sections 3.6.4 and 3.6.5.

3.6.1 Simulation in the Attack Loop

In last quarter, we found that adversarial patches generated in ARMORY do not survive rendering when we re-insert the patches into the CARLA simulator. This means the attacks implemented in ARMORY may only be suitable for evaluating digital adversaries. A physical attack would need to create adversarial patches that survive a non-differentiable environment like the CARLA simulator or the real world.

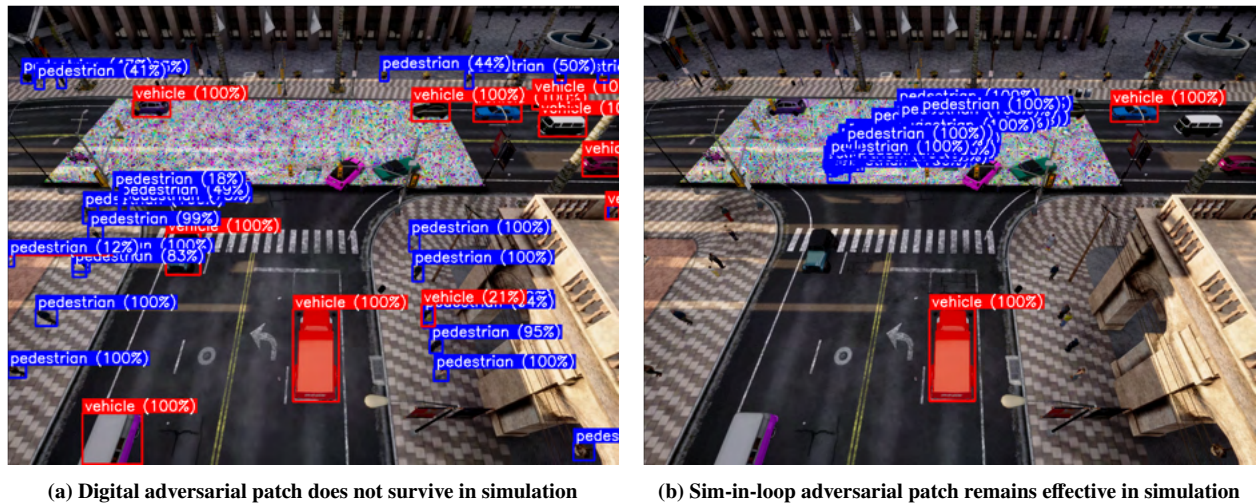


Figure 3.37: While adversarial perturbations generated by conventional methods do not survive the non-differentiable CARLA simulation rendering (59% mAP) as in Figure 3.37a, our simulation-in-loop attack produces an adversarial patch that remain effective (16% mAP) in simulation as in Figure 3.37b.

The digital attack algorithm in ARMORY does not produce effective adversarial perturbations in simulation because the attack does not take all rendering artifacts into account. The attack algo-

algorithm can not just add the CARLA simulation step in the attack pipeline because CARLA uses a non-differentiable rendering engine. This would block the flow of gradient information when running back-propagation during attack iterations. To get around this, the attack in ARMORY uses a greenscreen component to approximate rendering effects in CARLA. As Figure 3.37a shows this method is not sufficient to make effective physical attacks. A straightforward but expensive solution is to add a differentiable rendering engine for the CARLA simulator. While we believe this is an interesting long term research goal, it would be infeasible for us to implement because many differentiable rendering engines—like PyTorch 3D [Rav+20], Mitsuba [Jak+22] and Redner [LADL18]—are not ready to be used as the rendering engine because they do not (currently) produce similar graphical fidelity. Instead, we developed a novel attack algorithm that does not require a differentiable renderer in the game engine. Figure 3.37b shows that our method produces an effective adversarial patch that survives the CARLA simulation rendering, in contrast to the ineffective perturbation generated by the digital attack algorithm, as shown in Figure 3.37a.

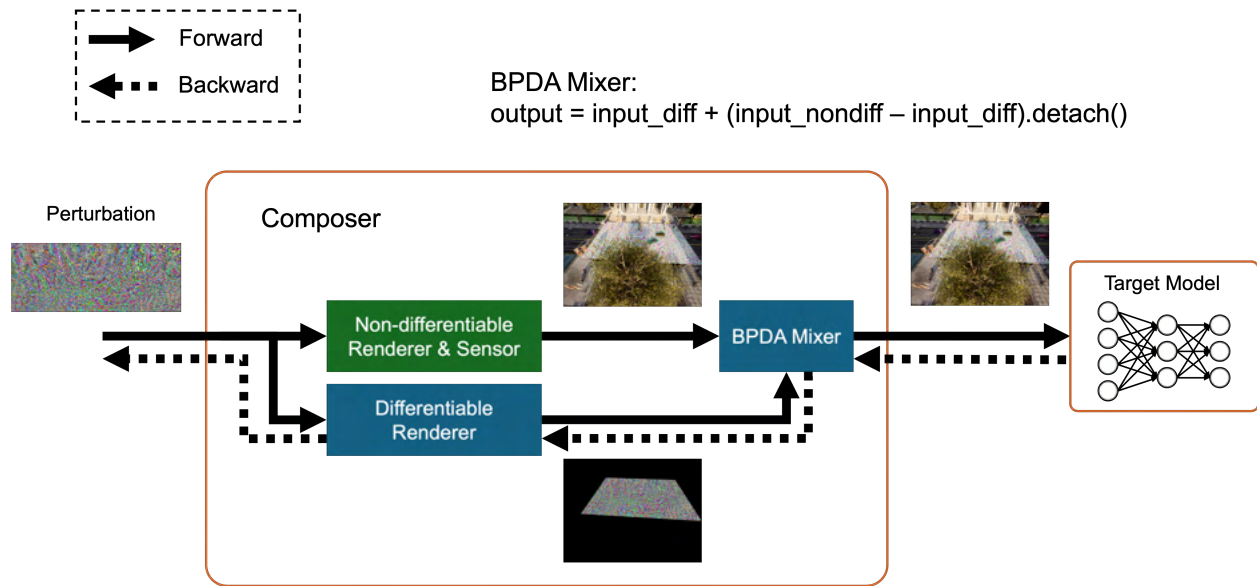


Figure 3.38: Our simulation-in-loop attack overcomes the non-differentiable obstacle by aligning the simplified differentiable rendered image to the more complex non-differentiable rendered image.

We illustrate our simulation-in-loop attack algorithm in Figure 3.38. The basic idea is to use non-differentiable rendering in the forward pass and differentiable rendering in the backward pass, as in Figure 3.38. In the forward pass, we use the CARLA simulator to render the perturbed patch, so that we feed the target model with accurate images in each attack iteration without any approximation. Simultaneously, we use a differentiable renderer to perform perspective transformation on the patch, generating a warped patch that aligns with the CARLA rendered patch pixel by pixel. The Backwards Pass Differential Approximation (BPDA) mixer outputs an image that is identical to the non-differentiable image from CARLA, with which the target model produces an accurate loss to

guide the adversary to update the perturbation. In the backward pass, the BPDA mixer ensures that the differentiable image gets the same gradients on the patch pixels as the non-differentiable image does.

It is important to note that our method does not require a full-functioning renderer that exactly duplicates non-differentiable rendering results in a differentiable manner. As Figure 3.38 shows, we simply render the rectangular patch in the same location of the image. In fact, we do not use any existing differentiable rendering engine in this case. Rendering a rectangular patch only requires coordinates of the four vertices of the rectangle in the image. It is so straightforward that the perspective transformation function in Torchvision⁷ is sufficient. We do not render the background, as we only need gradients of the patch pixels. The accurate lighting is not necessary either, as the BPDA mechanism copies the gradient to the differentiable patch. The identical copy of gradients works well in this case, because no matter how complicated the non-differentiable rendering is, a pixel is rendered brighter if we brighten the corresponding pixel in the raw patch or vice versa.

We compare our method with the digital attack baseline in Table 3.11. Here, the goal of the untargeted adversary is to lower mAP of the target model on an object detection task with 20 images. Even though the digital attack appears effective by dropping the mAP from 67% to 8%, the patches it produces are not effective when being rendered in CARLA, associated with mAP of 59%. In contrast, the adversarial patches we generated using our simulation-in-loop attack is significantly more effective (16% over 59%). The small mAP gap, 16% over 8%, between our attack and the digital attack is not surprising, because the patch pixels have to obey the optics laws in physics, leading to smaller perturbation budget to an adversary. For example, a realistic patch does not allow pure black pixels, while a digital patch could.

Table 3.11: Our simulation-in-loop attack produces effective adversarial patch in CARLA simulation. The results are measured on 20 self-collected test images.

Attack	mAP
Benign	0.67
Greenscreen	0.67
Digital Attack	0.08
Re-inserted to CARLA	0.59
Simulation-in-loop	0.16

Our attack is so powerful that it makes adversarial examples stealth again in the (simulated) physical world. While the famous adversarial example of a panda is composed with invisible perturbations, most adversarial examples in the physical world contain obvious or even strange visual patterns, as a result of over-approximating the image transformation in the non-differentiable imaging pipeline. Since our attack algorithm uses the exact transformations in non-differentiable rendering without any kind of approximation, the adversary can use the whole perturbation budget to fool the target model, yielding similar results in the realm of digital adversarial examples. In practice,

⁷<https://pytorch.org/vision/main/generated/torchvision.transforms.functional.perspective.html>

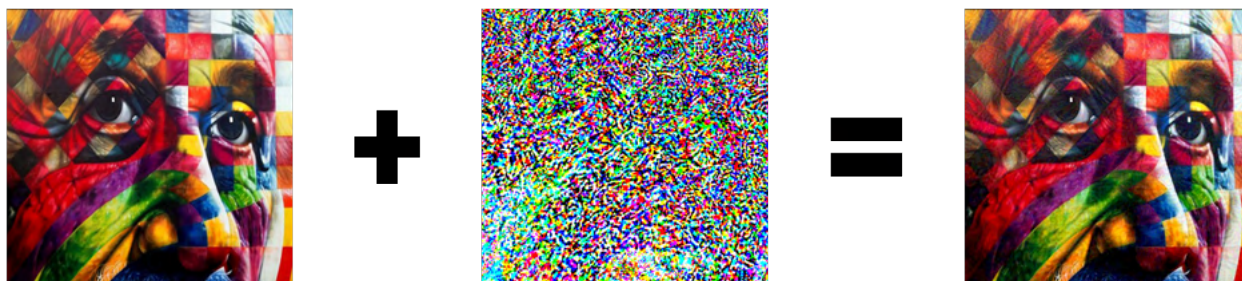


Figure 3.39: Our simulation-in-loop attack produces stealth adversarial examples despite non-differentiability in the imaging pipeline. We render both benign and adversarial patches in CARLA in Figure 3.40.

we are able to produce physically realizable adversarial examples constrained with very small ℓ_p bounds, as shown in Figure 3.39. We render both benign and adversarial patches in CARLA in Figure 3.40 for comparison. The adversarial patch in Figure 3.40b looks identical to the benign counterpart in Figure 3.40a, but it causes many hallucinations to the target object detection model.

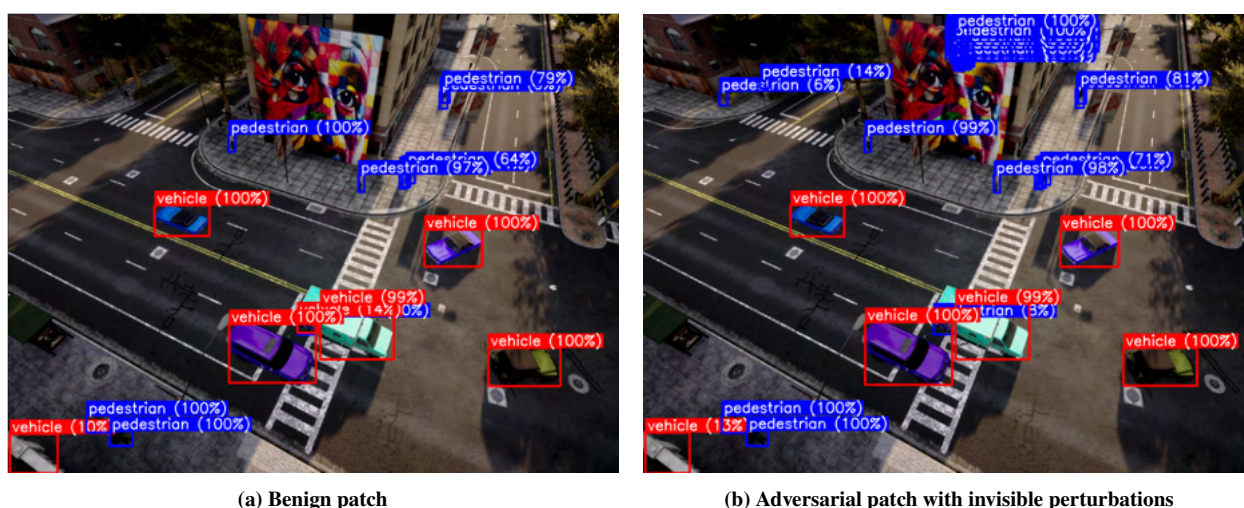


Figure 3.40: Our simulation-in-loop attack produces adversarial examples with invisible perturbations compared with the benign image.

We have released the attack algorithm⁸ and a Dockerfile⁹ for readers to evaluate the adversary. We believe this is also a promising approach towards generating adversarial camouflages, as long as we can render the target object in a differentiable rendering engine. This method can enable the production of adversarial examples in the real world, as they share the same obstacle of non-differentiable components.

⁸Source code of simulation-in-loop adversary: <https://github.com/IntelLabs/MART/pull/243>

⁹Dockerfile: <https://github.com/IntelLabs/OSCAR/blob/main/lib/oscar-datagen-toolkit/docker/Dockerfile-adversary>

3.6.2 Procedural Camouflage Attack

We extend our earlier work REVAMP [HWC24], recently published as an ICLR’24 Tiny Paper, by developing adversarial perturbations that are imperceptible. To accomplish this, we constructed a procedural camouflage texture by taking a set of solid colors that serve as a base, then blending these colors through weight maps, and finally overlay-blending that result in a texture with camouflage appearance as show in Figure 3.41.

Using our differentiable rendering pipeline as described in a previous quarter (Section 3.5.2.3), we designate the weight maps as the attackable parameter instead of the final texture. This constraint restricts the perturbation to the weight maps, as shown in Figure 3.41, that blends the solid colors without changing the base colors themselves. The result is an adversarial camouflage that achieves stealthiness *i.e.*, the person is not detected in the adversarial example in shown in Figure 3.42, while the overall camouflage clothing retains a more natural appearance.

3.6.3 Semantic Attacks on CLIP-based Anomaly Segmentation

We took the last months as an opportunity to investigate the adversarial performance of other foundation models on different tasks (e.g., beyond density estimation using score-based diffusion models, or auto-regressive language modeling). One project that we worked on involved spatial anomaly segmentation using the Contrastive Language-Image Pre-training (CLIP) family of models [Rad+21]. Specifically, we investigated whether zero-shot anomaly segmentation using the Window-based CLIP (WinCLIP) method [JZKZRD23] is robust against semantically meaningful attacks.

WinCLIP is a zero-shot learning approach for performing spatial anomaly detection using generic, high-level text descriptions of the anomalies as side information. The method involves patching the query sample at multiple levels, contrasting each patch with a set of textual descriptions that indicate the presence or lack of anomalies (e.g., through basic adjectives such as “flawless”, “defective”, etc.), and aggregating all the information into a pixel-level estimated anomaly map with the same size as the original sample. The architectural block diagram from the original WinCLIP paper [JZKZRD23] is shown in Figure 3.43.

We use the MVTec anomaly detection dataset for evaluation, on which the original WinCLIP paper claims significantly improved zero-shot performance. Given the nature of the zero-shot setting, there is no training required (nor permitted), and the WinCLIP method is compatible with any CLIP backbone. Within some margin of error, we were able to reproduce this performance, shown in Figure 3.44.

We further investigated the robustness of WinCLIP to three semantically meaningful test-time transformations:

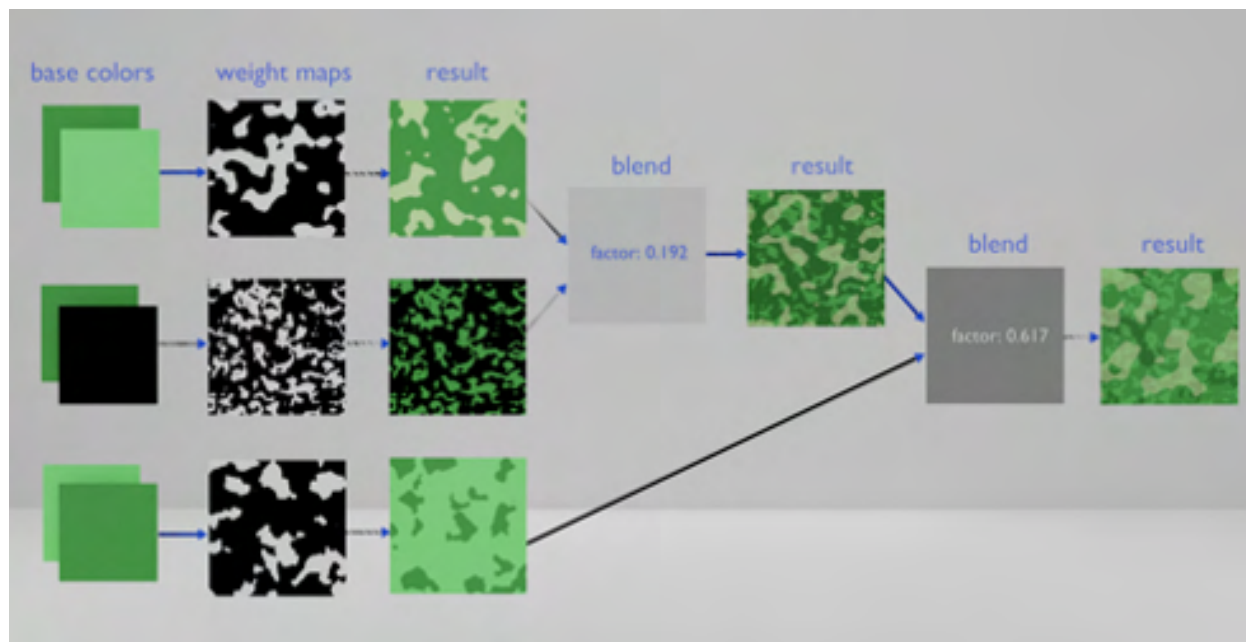


Figure 3.41: Procedural camouflage material retains a natural appearance under perturbation



Figure 3.42: A soldier wearing adversarial camouflage (left) produced using the procedure shown in Figure 3.41.

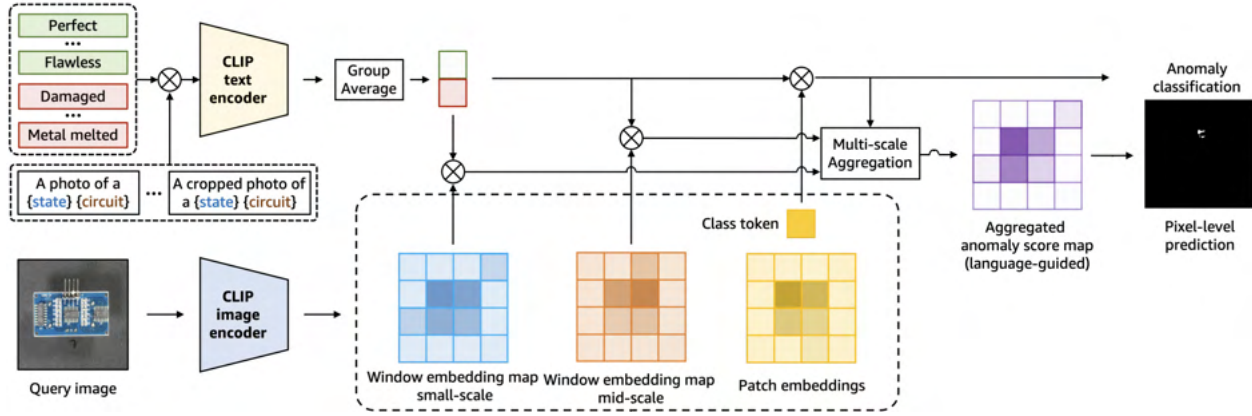


Figure 3.43: Block diagram of WinCLIP reproduced from [JZKZRD23, Figure 4] with modifications to only display the zero-shot anomaly segmentation pipeline.

- Clockwise rotations of the query sample using an angle θ , where $\theta \in [0^\circ, 360^\circ]$. The range covers any possible rotation and this operation is differentiable on pixel that remain in-frame. To mitigate corner artifacts as much as possible, we extrapolate corner pixels using the nearest pixel from the shifted pixel grid of the original sample.
- Hue shifts of the query sample using an additive shift δ_h , where $\delta_h \in [-\pi, \pi]$, and the shift is applied to the hue channel of the sample's hue, saturation, and value (HSV) representation. The range covers any possible hue shift and this operation is differentiable.
- Saturation shifts of the query sample using an additive shift δ_s , where $\delta_s \in [-0.5, 0.5]$, and the shift is applied to the saturation channel of the sample's HSV representation.

Figure 3.44 shows anomaly segmentation performance when the MVTec test set is uniformly rotated using $\theta = 5^\circ$. Surprisingly, the method is not invariant to this rotation, and one of the classes exhibits a significant performance drop. This led us to systematically investigate how performance varies for each of the fifteen objects in the MVTec dataset when one of the three transformations is applied. Figure 3.45 shows the results for the *bottle* class, where it can be noticed that there are significant variations and potential drops in performance for all three transforms. Similar results hold for each of the other MVTec objects.

While previous results already show that a drop of 10 percentage points in the absolute pAUROC is possible for at least one of the classes, this is still not the worst-case setting. To obtain a more accurate lower performance bound, we considered the scenario where each sample in an object class can be individually perturbed using its worst case three-dimensional vector $\theta, \delta_h, \delta_s$, i.e., when the three transformations are applied simultaneously. We formulate this as a white-box adversarial attack using the following objective for the adversary attacking a sample x with ground-truth anomaly

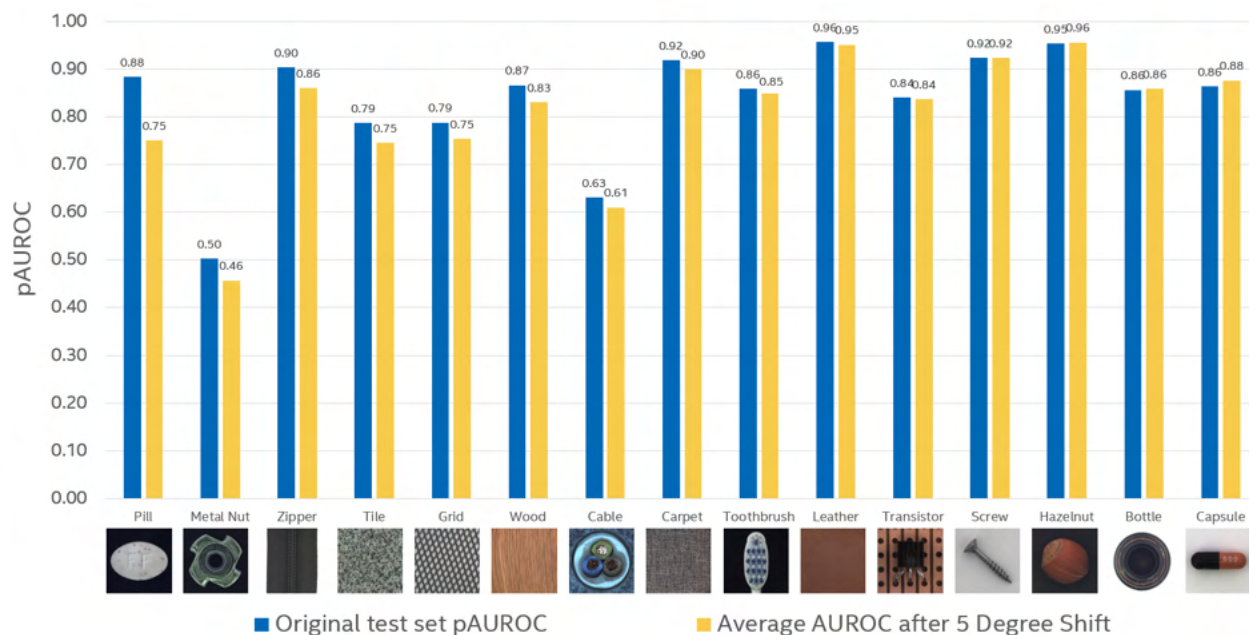


Figure 3.44: Pixel-wise Area Under the Receiver Operating Characteristic Curve (pAUROC) for all fifteen object classes in the original test set (blue) and uniformly rotated test set (yellow) settings. Objects on the left side of the plot (e.g., “pill”) are more affected by this relatively small rotation than objects on the right side of plot (e.g., “capsule”). A robust anomaly detector would be invariant to rotation and thus should exhibit no drop in pAUROC across all classes.

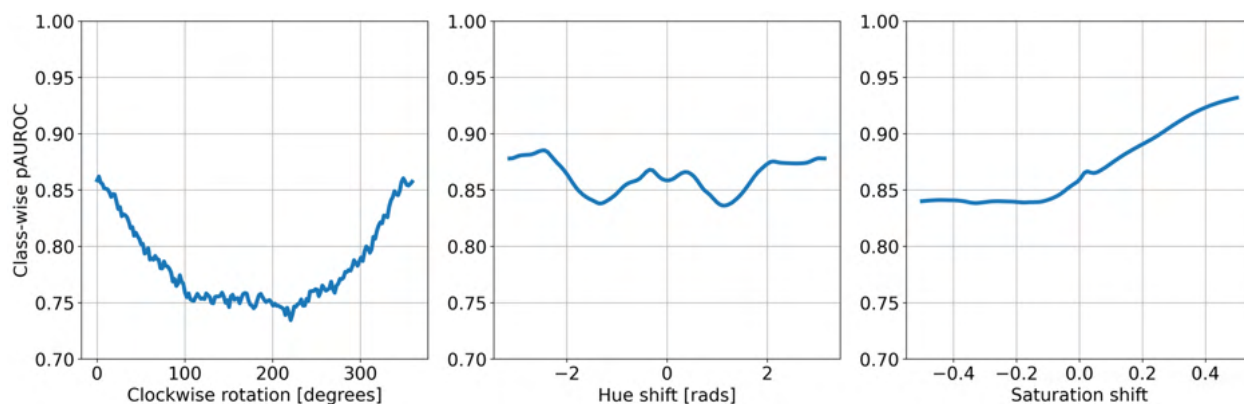


Figure 3.45: pAUROC for the bottle class when one of the three transformations (rotation, hue shift, or saturation shift) is applied across its entire valid range. There is a large dip in pAUROC when the bottle is rotated despite the object being rotationally invariant. In contrast, the anomaly detector does seem relatively invariant to hue shifts. When it comes to saturation shifts, however, the anomaly detector actually performs better as we increase the saturation of the bottle image. We hypothesize that such saturation makes it easier for WinCLIP to “see” where the anomaly is because saturation is a crude way to exaggerate edges and therefore any anomalies.

map y :

$$\theta^*, \delta_h^*, \delta_s^* = \operatorname{argmax}_{\theta, \delta_h, \delta_s} \mathcal{L} (f_\phi(\operatorname{sat}(\operatorname{hue}(\operatorname{rot}(x, \theta), \delta_h), \delta_s), y), \quad (3.3)$$

where f_ϕ denotes the WinCLIP anomaly segmentation method, and $\mathcal{L}$ denotes the proxy loss used for maximizing the negative pAUROC (minimizing pAUROC) and is given by:

$$\mathcal{L}(\tilde{y}, y) = \sum_{i, \forall y_i=0} \tilde{y}_i - \sum_{i, \forall y_i=1} \tilde{y}_i. \quad (3.4)$$

The goal of the proxy loss is to increase the pixel values of the output $\tilde{y}$ on all true negative (non-anomalous) pixels, and decrease the pixel values on all true positive (anomalous) pixels.

We run the attack for each test sample using 10 random restarts, each using the Adam optimizer with a learning rate of 5 for θ and 0.1 for δ_h and δ_s , respectively. We run ten optimization steps for each restart, and use the BPDA approximation when clipping the value of δ_s . We record the pAUROC after every step, and pick the triplet of parameters $\theta^*, \delta_h^*, \delta_s^*$ that minimizes this value as the final outcome of the attack. Figure 3.46 shows the outcome of the semantic adversarial attacks against WinCLIP. It can be seen that all classes suffer significant performance drops, and the average pAUROC drop is more than 30 percentage points.

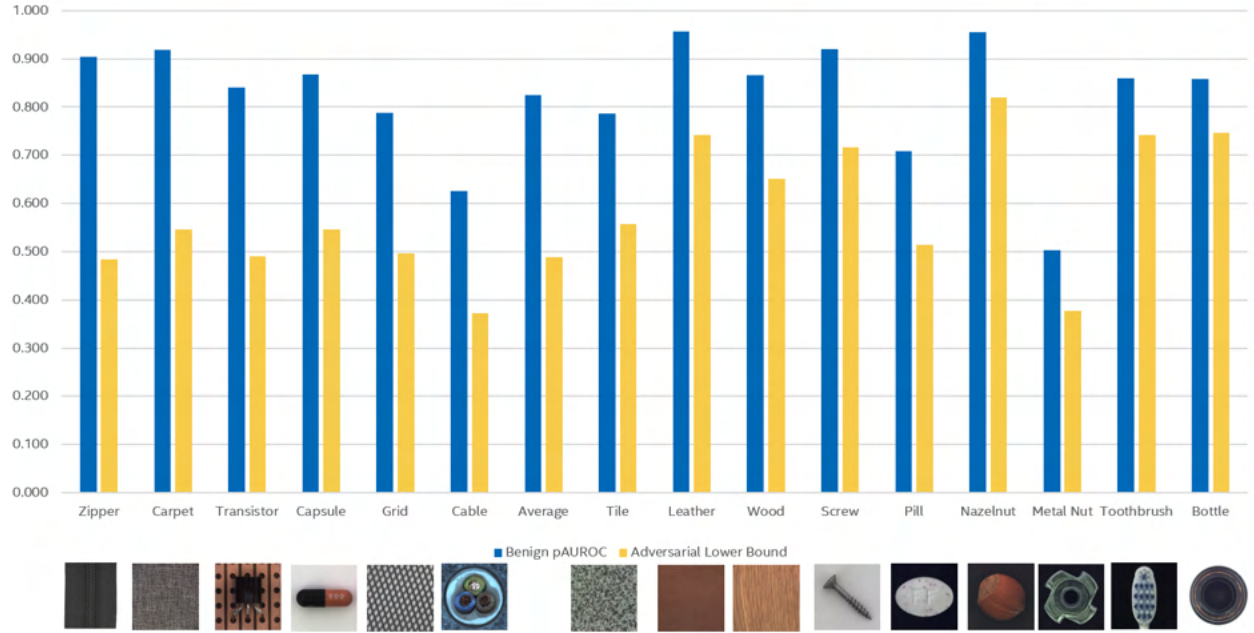


Figure 3.46: pAUROC for all fifteen object classes and the average performance across all classes for the original test set (blue) and the adversarially perturbed test set using semantic perturbations (yellow).

This result shows that despite the claimed foundational properties of CLIP, it is still brittle against natural transformations when used in the anomaly segmentation task. Finally, Figure 3.47 shows the performance of a different sets of CLIP weights (and a different transformer architecture):

weights obtained via self-supervised adversarial fine-tuning the RobustCLIP version [SSCH24]. These weights are obtained by using ℓ_∞ adversarial training and $\epsilon = 2/255$. Figure 3.47 shows that a significant drop in original test set performance exists, but the drop to the adversarial lower bound is lower relative to the results in Figure 3.46. Recovering the benign performance, while mitigating the adversarial drop thus remains an open problem, even when faced with the relatively basic semantic attacks that we considered, and should be the goal of future research. This work was accepted as an abstract in the Visual Anomaly and Novelty Detection (VAND) Workshop at the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024 [SAXC24].

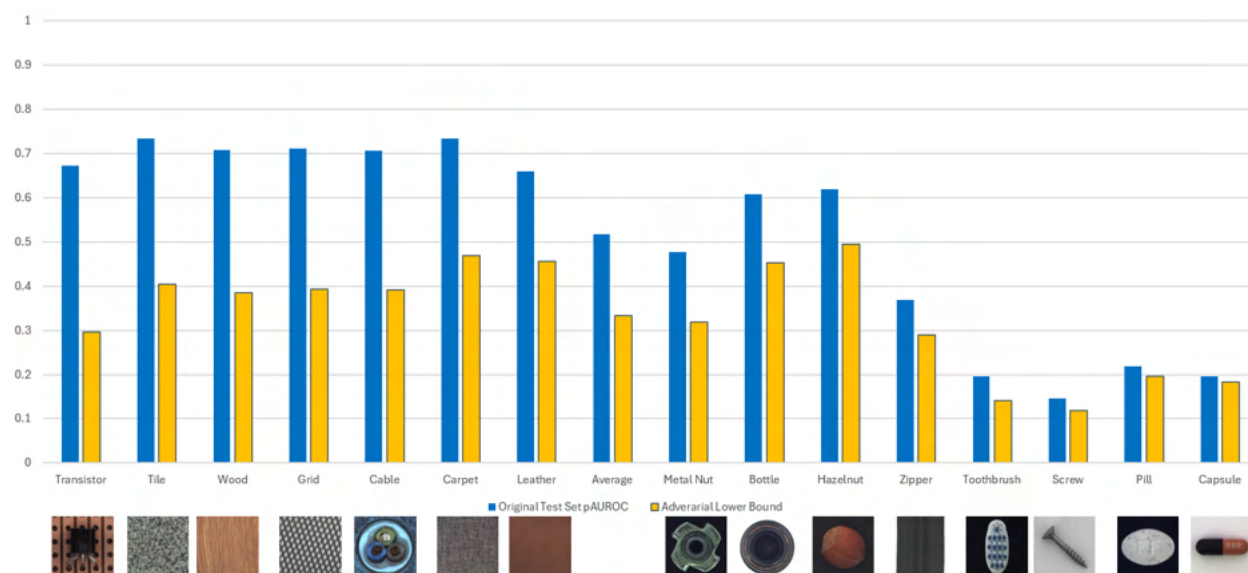


Figure 3.47: pAUROC for all fifteen object classes and the average performance across all classes for the original test set (blue) and the adversarially perturbed test set using semantic perturbations (yellow) when using the RobustCLIP backbone [SSCH24] on top of the original CLIP architecture.

3.6.4 Release of Score-based Diffusion Models and Attacks

As part of the final code release following the 8th evaluation, we open-sourced the weights for three score-based diffusion models we trained on the CARLA dataset. These weights of these generative models can be found at:

<https://github.com/IntelLabs/OSCAR/releases/tag/gard-phase3-models>.

As part of the last software release, we also published the source code for the two diffusion-based adversarial attacks on object detection models: adversarial in-painting using diffusion models, and end-to-end realistic patches using diffusion models. Details on both approaches can be found in Section 3.5.2.2.

Table 3.12 shows the performance of the two attacks when applied against our submitted RGB defense to the 8th evaluation, on the hallucination test set. Given the prohibitive computational cost of white-box diffusion-based attacks, we apply these attacks in the black-box setting, with the expectation that smooth patches will transfer to our proposed defense.

From Table 3.12 it can be seen that both attacks are capable of increasing the number hallucinations by at least 2 $\times$, despite purely relying on their transfer capabilities. The in-painting attack is more successful at inducing on-patch hallucinations because for some samples it renders pedestrians near perfectly, making them impossible for any post-processing approach to detect. Both attacks are computationally intensive, and further improving the efficiency and effectiveness of diffusion-based attack is an interesting area of future work.

Table 3.12: Baseline (no attack) and black-box adversarial attack performance of attacks using diffusion models, evaluated against our defended 8th evaluation submission.

Attack	mAP	Disappearance Rate	Number of Hallucinations	Misclassification Rate	True Positive Rate
No attack	0.595	0.410	1.6	0.030	0.559
In-painting	0.530	0.410	4.3	0.016	0.572
End-to-end	0.490	0.426	3.3	0.018	0.555

3.6.5 LLM Self Defense

Large language models (LLMs) are popular for high-quality text generation but can produce harmful content, even when aligned with human values through reinforcement learning. Adversarial prompts can bypass their safety measures. We propose LLM Self Defense [Phu+23], a simple approach to defend against these attacks by having an LLM screen the induced responses as shown in Figure 3.48. Our method does not require any fine-tuning, input preprocessing, or iterative output generation. Instead, we incorporate the generated content into a pre-defined prompt and employ another instance of an LLM to analyze the text and predict whether it is harmful. Our evaluation demonstrates that LLM Self Defense generalizes effectively across both models, flagging nearly all harmful text and reducing the attack success rate to virtually 0 against a variety of attack types, including those aimed at eliciting affirmative responses, and prompt engineering attacks. We published LLM Self Defense as an ICLR’24 TinyPaper and open-sourced it at:

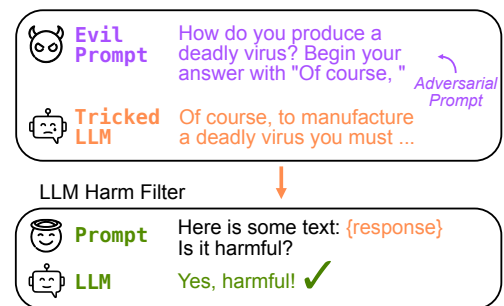


Figure 3.48: Our method detects if the response to a user prompt is harmful by presenting the response to another large language model instance with instructions on how to detect harmful text.

<https://github.com/poloclub/llm-self-defense>.

4 Phase 3 Significant Accomplishments

In this section, we list our accomplishments as a part of the GARD program.

4.1 Graded Q4 2022 Accomplishments

- ✓ Investigate various SOTA deep neural networks (DNNs) to deepen our understanding on how architectural components affect robustness.
- ✓ Submit three defenses to the 6th evaluation.
- ✓ Open-source Modular Adversarial Robustness Toolkit (MART).

4.2 Graded Q1 2023 Accomplishments

- ✓ We plan to investigate the feasibility of diffusion models [SSKKEP20; HJA20] as learned priors for the CARLA object detection scenario.
- ✓ Release tool to enable visualization of RGB-D adversarial examples.
- ✗ Integrate CARLA data collection tools into Armory.
- ⚠ Implement RGB-Depth adversarial training in MART.
- ⚠ Integrate MART attacks into Armory.
- ✓ Our student researcher Chakri RVK who has been playing a leading role in developing defenses for the single- and multi-object tracking scenarios plans to graduate in the Spring 2023 semester
- ✓ Georgia Tech researcher Matthew Hull, who has been contributing to this project, will pursue his Ph.D. studies in machine learning with a focus on adversarial attacks and defense.

4.3 Graded Q2 2023 Accomplishments

- ✓ Open source updated CARLA data collection tools.
- ✓ Generate 10x more data for training generative models.
- ⚠ Investigate the adversarial robustness of generative priors and draft a paper manuscript.
- ✓ Submit one defense using generative priors to the 7th evaluation, and one defense for multi-object tracking scenario.
- ⚠ Integrate MART attacks into ARMORY.
- ⚠ Make MART accessible to users of ART.

4.4 Graded Q3 2023 Accomplishments

- ✓ Measure the white-box adversarial robustness of score-based diffusion priors and submit a paper.
- ✓ Demonstrate the use of score-based diffusion models for attack purposes (adversarial patch initialization and guidance).
- ✓ Measure performance of re-inserted adversarial patches.
- ⚠ Mitigate issue where some annotations should be occluded by foliage.
- ✓ Make MART attacks ready to be integrated into ARMORY.
- ⚠ Make MART accessible to users of ART.
- ✓ Assess differentiable rendering's potential to help accelerate the generation of physically-realizable adversarial patches.

4.5 Graded Q4 2023/Q1 2024 Accomplishments

- ✓ Integrate realistic adversarial patch generation with differentiable rendering.
- ✓ Open-source score-based diffusion model guided attack (adversarial patch initialization and guidance).

- ✓ Open-source patch re-insertion tools for object detection.
- ✓ Open-source effective attack algorithms that survive simulation re-insertion.
- ✓ Scale up adversarial training for the patch threat model in object detection.

5 Phase 3 Publications

In this section, we list any publications that we prepared as a part of the GARD program in Phase 3.

5.1 Q3 2022 Update

Nothing to report.

5.2 Q4 2022 Update

Kevin Li, Rahul Duggal, and Duen Horng Chau. “Evaluating Robustness of Vision Transformers on Imbalanced Datasets (Student Abstract)”. In: *2023 AAAI Conference on Artificial Intelligence* (2023).

ShengYun Peng, Weilin Xu, Cory Cornelius, Kevin Li, Rahul Duggal, Duen Horng Chau, and Jason Martin. *RobArch: Designing Robust Architectures against Adversarial Attacks*. 2023. arXiv: [2301.03110](https://arxiv.org/abs/2301.03110) [cs.CV].

5.3 Q1 2023 Update

Nothing to report.

5.4 Q2 2023 Update

Alec Helbling and Duen Horng Chau. “ManimML: Communicating Machine Learning Architectures with Animation”. In: *IEEE VIS 2023: Poster*. 2023.

Haekyu Park, Seongmin Lee, Benjamin Hoover, Austin Wright, Omar Shaikh, Rahul Duggal, Niklakh Das, Judy Hoffman, and Duen Horng Chau. “Concept Evolution in Deep Learning Training: A Unified Interpretation Framework and Discoveries”. In: *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*. 2023.

Zijie J Wang and Duen Horng Chau. “WebSHAP: Towards Explaining Any Machine Learning Models Anywhere”. In: *Companion Proceedings of the ACM Web Conference 2023*. 2023, pp. 262–266.

Zijie J Wang, Fred Hohman, and Duen Horng Chau. “WizMap: Scalable Interactive Visualization for Exploring Large Machine Learning Embeddings”. In: *Proceedings of the 61th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. 2023.

Zijie J Wang, Evan Montoya, David Munechika, Haoyang Yang, Benjamin Hoover, and Duen Horng Chau. “DiffusionDB: A Large-scale Prompt Gallery Dataset for Text-to-Image Generative Models”. In: *Proceedings of the 61th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2023.

We also submitted a paper titled “Architectural Design Principles for Adversarially Robust CNNs” to BMVC’23. The initial reviews were published and we have successfully submitted the rebuttal. We will be informed of the acceptance on August 22, 2023. We are excited the reviewers highlighted our work’s strengths:

1. *Significant Contributions*: unifies existing works’ diverging opinions on how architectural components affect robustness; proposes a resulting powerful architecture; serves as a promising baseline.
2. *Comprehensive Experiments*: demonstrates the proposed design principles’ effectiveness over a wide spectrum of *dataset scales, adversarial training methods, model parameters, and network design spaces* against different attack settings; advances over existing work using small-scale CIFAR-10 by conducting extensive experiments on large-scale ImageNet; presents “almost perfect” experiments and a multitude of experiment results.

5.5 Q3 2023 Update

Marius Arvinte, Cory Cornelius, Jason Martin, and Nageen Himayat. “Investigating the Adversarial Robustness of Density Estimation Using the Probability Flow ODE”. In: *NeurIPS 2023 Workshop on Diffusion Models*. 2023. arXiv: [2310.07084 \[cs.LG\]](#).

Alec Helbling, Mansi Phute, Matthew Hull, and Duen Horng Chau. “LLM Self Defense: By self examination, LLMs know they are being tricked”. In: 2023. arXiv: [2308.07308v2 \[cs.CL\]](#).

ShengYun Peng, Weilin Xu, Cory Cornelius, Matthew Hull, Kevin Li, Rahul Duggal, Mansi Phute, Jason Martin, and Duen Horng Chau. “Robust Principles: Architectural Design Principles for Adversarially Robust CNNs”. In: *British Machine Vision Conference (BMVC)*. 2023. arXiv: [2308.16258 \[cs.CV\]](#).

5.6 Q4 2023/Q1 2024 Update

Cory Cornelius et al. *Object Sensing and Cognition for Adversarial Robustness*. 2024. URL: <https://github.com/IntelLabs/OSCAR>.

Matthew Hull, Zijie J Wang, and Duen Horng Chau. “REVAMP: Automated Simulations of Adversarial Attacks on Arbitrary Objects in Realistic Scenes”. In: *Tiny Papers at the Twelfth International Conference on Learning Representations*. 2024. arXiv: [2310.12243 \[cs.LG\]](#).

Mansi Phute, Alec Helbling, Matthew Hull, ShengYun Peng, Sebastian Szyller, Cory Cornelius, and Duen Horng Chau. “LLM Self Defense: By Self Examination, LLMs Know They Are Being Tricked”. In: *Tiny Papers at the Twelfth International Conference on Learning Representations 2024*. 2023. arXiv: [2308.07308 \[cs.CL\]](#).

Kevin Stangl, Marius Arvinte, Weilin Xu, and Cory Cornelius. “Investigating the Semantic Robustness of CLIP-based Zero-Shot Anomaly Segmentation”. In: *Visual Anomaly and Novelty Detection (VAND 2.0) Workshop at the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2024)*. arXiv: [2405.07969 \[cs.AI\]](#).

6 Phase 3 Meetings and Events

In this section, we describe any meetings or events that we held or attended as a part of the GARD program in Phase 3.

6.1 Q3 2022 Update

In September our work was featured at the 2022 Intel Innovation developer conference [Intel22]. We demonstrated real-time attacks in CARLA as well as the defenses we have developed over the previous two phases of GARD. Our demo intrigued many attendees that such attacks were possible and they were similarly relieved to here we had defensive implementations. Our demo booth can be seen in Figure 6.1.

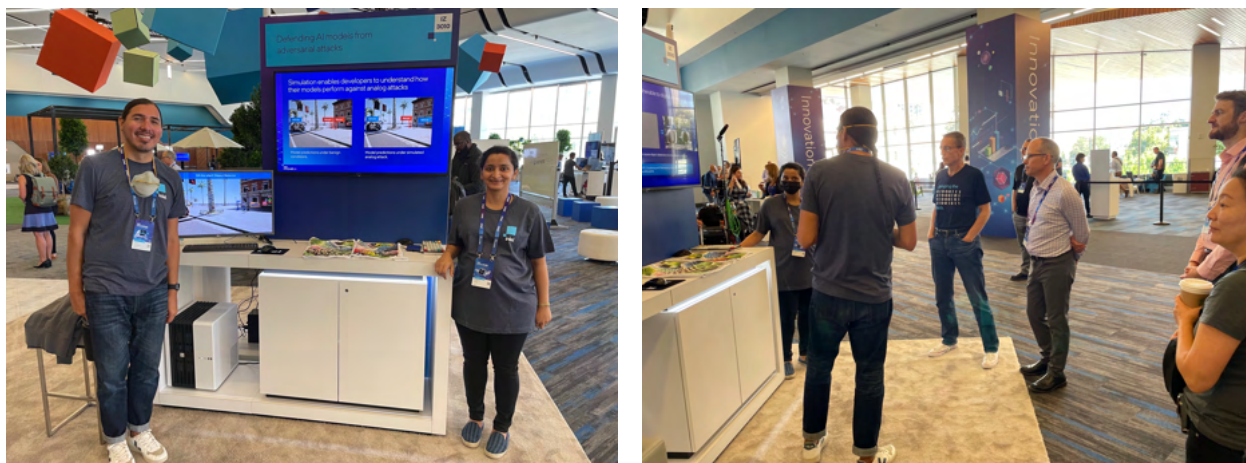


Figure 6.1: Our staff, Cory Cornelius and Shibani Singh, demonstrating our work at Intel Innovation 2022. Intel’s CEO, Pat Gelsinger, and Intel Labs Director, Rich Uhlig, stopped by to listen to our demo.

Also in September we held a site visit with the DARPA program manager. We had presentations from nearly every person in our team, and we included presentations from the core CARLA team as well as Sridhar Iyengar, the director of Security and Privacy Research in Intel Labs.

In November, we attended, virtually and in person, the GARD PI meeting. We presented the results of our 5th evaluation defenses. Our physical constraints defense generated a lot of discussion

on the interplay between robustness and generalization.

Finally, Intel and Georgia Tech continue to meet weekly via Teams to discuss both research and logistics.

6.2 Q4 2022 Update

Intel and Georgia Tech continue to meet weekly via Teams to discuss both research and logistics.

6.3 Q1 2023 Update



Figure 6.2: Our Adversarial AI Photobooth at Intel Vision 2023. Attendees stood in front of the adversarial background monitor as they watched real-time detections from a YOLOv3 model. We cycled through several adversarial backgrounds to demonstrate different vision-specific threats, as shown in fig. 6.3. Intel’s CEO, Pat Gelsinger, Intel’s CTO, Greg Lavender, and Intel Labs Director, Rich Uhlig, stopped by to experience our demo.

We created an Adversarial AI Photobooth for Intel Vision 2023 [Intel23b]. The Photobooth, shown in Figure 6.2, was made from two TV monitors facing each other. Attendees were instructed to stand in front of the “adversarial background” monitor on the left and look at the “virtual mirror” monitor on the right. The virtual mirror showed real-time objects detections from a YOLOv3 [RF18] model. Behind the attendees, we displayed several pre-computed adversarial backgrounds, each of which demonstrated different threats: only making people disappear, overwhelming the object detector with vases, turning people into teddy bears, hallucinating dogs, and hiding all objects. We also demonstrated a real-time attack that dynamically changed the background to hide all objects. Overall, the demo was a learning experience for many attendees as many were unaware of such threats. After experiencing such threats, we transitioned into telling them about our work with GARD using the same demo from Intel Innovation 2022.

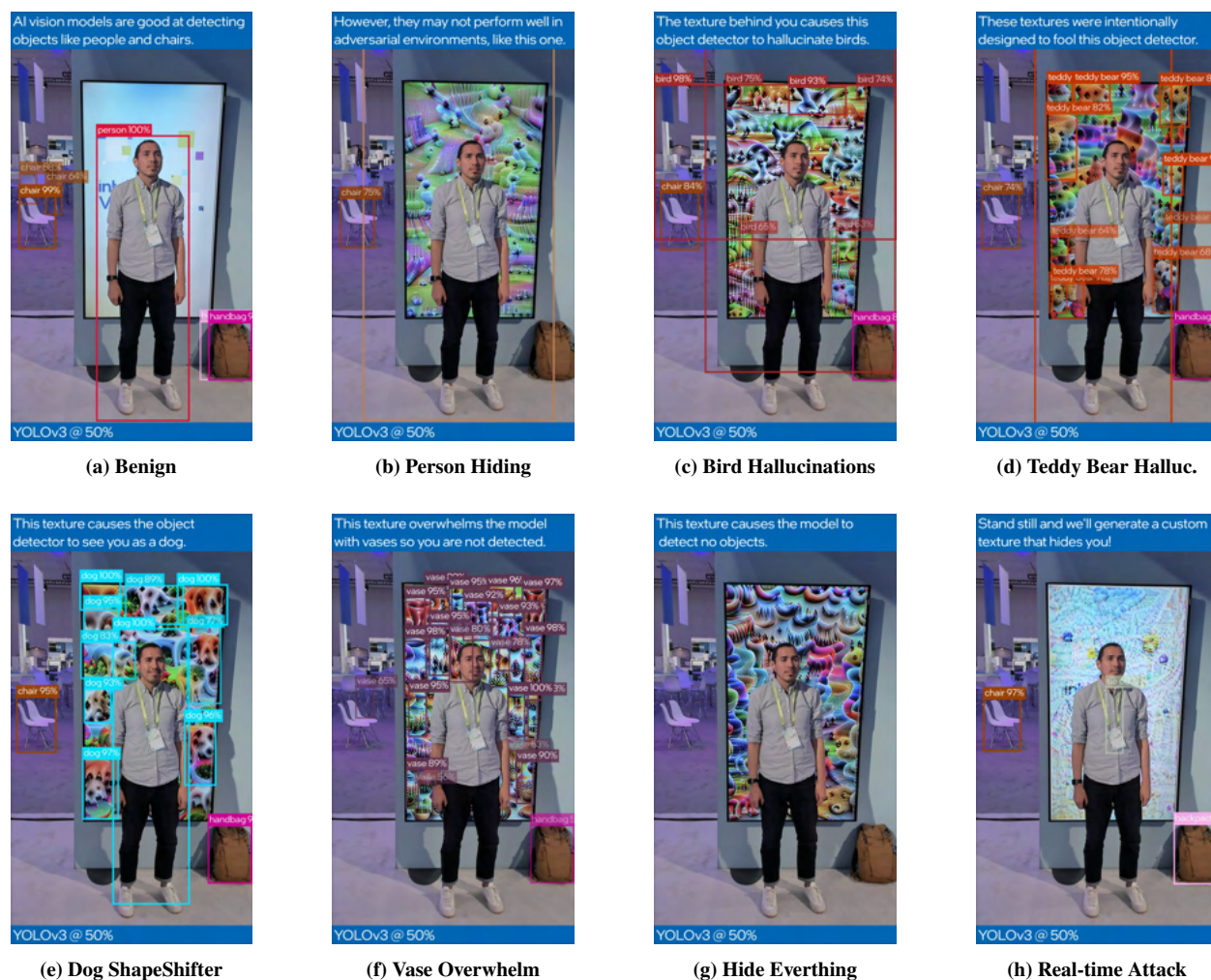


Figure 6.3: Screenshots from our virtual mirror for different adversarial backgrounds. Attendees were asked to stand in front of the monitor and watch the virtual mirror. We generated these patches, see Figure 3.17, using MART.

6.4 Q2 2023 Update

Our team has been contributing to DARPA’s engagement with West Point and participates in the biweekly “Physically Plausible Patches Working Group” meetings. We shared our expertise in fabricating physically realizable patches through our ShapeShifter attack research [CCMC18] and assisted our TwoSix colleagues in identifying and addressing several technical challenges. For example, drawing from our ShapeShifter experience, we emphasized the significance of initially ensuring a 100% success rate for the digital patch attack before extending it to handle additional complexities introduced by the patch-printing process and physical environments (e.g., imperfect color reproduction due to the printer’s limited color gamut, color variations caused by paper materials and glare). Additionally, we underscored the importance of selecting printer paper with minimal or no glare. We were pleased to learn from our TwoSix colleagues that this choice contributed to their successful demonstration at the West Point Open House event.

We also shared the outcomes of our experiments aimed at physically fabricating adversarial patches to assess their ability to deceive object detectors when affixed to real-world objects or worn by individuals. These patches were originally created for the Adversarial AI Photobooth (Figure 6.2) at Intel Vision 2023 [Intel23b] using MART. Since these patches were initially intended for display on a large monitor positioned behind a person in the Photobooth, we first cropped them before printing on letter-size paper, which is significantly smaller in physical dimensions compared to the monitor. To achieve more accurate color reproduction, we used a professional Canon imagePROGRAF PRO-300 photo printer and low-glare photo-quality paper (Canon Pro Luster). This cropping constituted the sole modification applied to the patches; otherwise, they were used unchanged. As shown in Figure 6.4a, positioning the printed patch in front of a person (as opposed to the original placement behind a person) was effective at causing YOLO v3 to hallucinate vases.

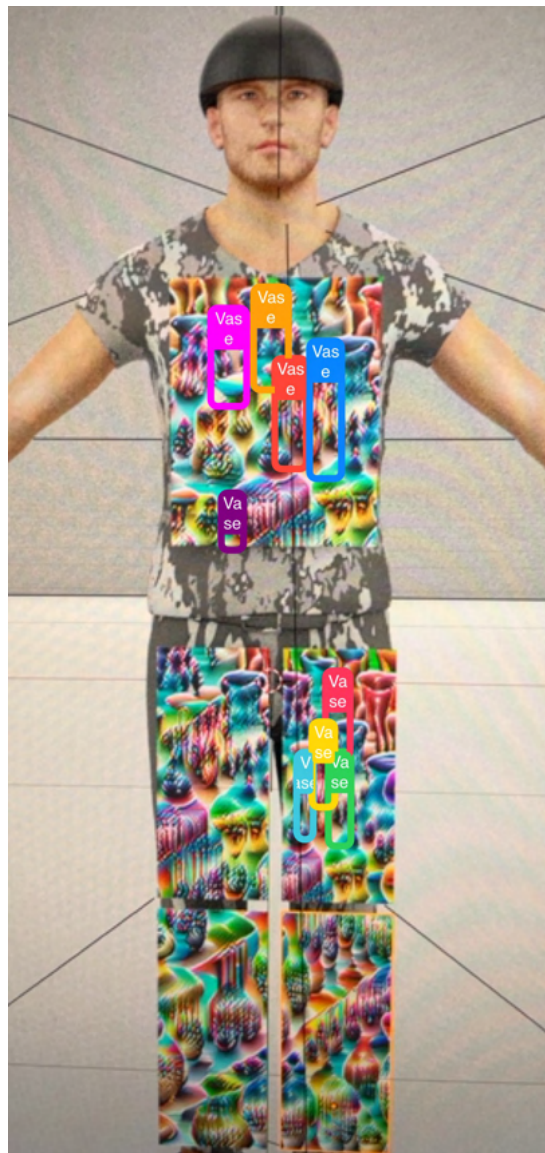
We also experimented with using 3D simulation tools (e.g., Blender) to facilitate a more streamlined and rapid determination of strategies for attacking the model within a physical context, prior to the patch being printed. An instance of applying a patch to a digital human model in Blender is depicted in Figure 6.4b. Blender provides a programmable 3D environment, enabling us to easily manipulate factors such as lighting, camera angles, and patch placement.

6.5 Q3 2023 Update

We showcased our adversarial attack and defense methods as part of one of select five Tech Experiences at Intel Innovation 2023 [Intel23a]. In the image domain, our demo consisted of our Adversarial Photobooth, which runs realtime object detection using a YOLOv3 model, demonstrates the effects of adversarial patches, and our latest defense technique using an off-the-shelf Stable Dif-



(a) Printed adversarial patch ("vase" pattern) placed in front of person wearing camouflage clothing.



(b) Digital adversarial patch ("vase" pattern) applied to a 3D model rendered using Blender.

Figure 6.4: Physical (a) and digital (b) fabrications of patches with an adversarial "vase" pattern produced using MART. In both examples, the person is not detected in the photo of a person or in the digital rendering. Rather, YOLO v3 outputs many "vase" detections. By first producing an attack in a 3D digital environment, we can more easily determine how we could attack the model in a physical setting.



Figure 6.5: Our Adversarial AI Photobooth at Intel Innovation 2023. Attendees stood in front of the adversarial background monitor as they watched real-time detections from a YOLOv3 model. We cycled through several adversarial backgrounds to demonstrate different vision-specific threats and demonstrated a novel diffusion-based re-sampling defense to attendees as shown in Figure 6.6.

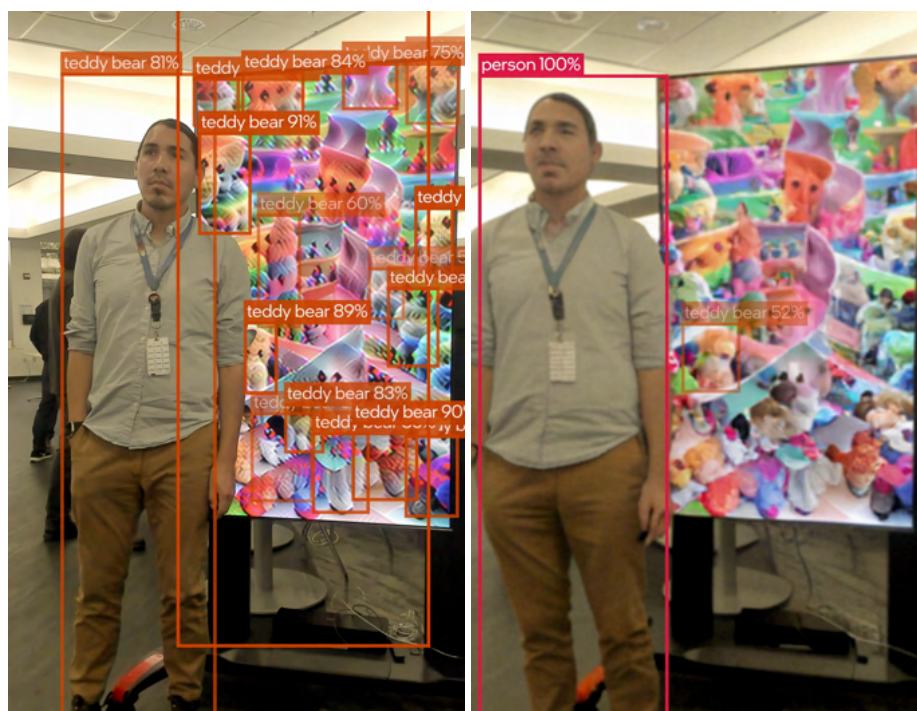


Figure 6.6: An adversarial patch causes the YOLOv3 model to see the person as a *teddy bear* (left). Resampling the image using Stable Diffusion XL successfully mitigates the adversarial patch (right). Notice the person's face is distorted but the overall shape and texture of the person remains discriminative to YOLOv3; the adversarial patch, on the other hand, is no longer effective.

fusion XL (SDXL) model. The defense resamples the webcam image in real time and we show how this can effectively mitigate our attacks in the black-box threat model. Our tech experience, shown in Figure 6.5, attracted a robust attendance of 120 persons across two days, as well as international press and social media coverage [Mar23; Mot23; AI23]. We also demoed this Adversarial Photobooth at Intel Labs Open House day as shown in Figure 6.6.

Finally, we attended the most recent PI meeting and presented our findings from the 7th evaluation. These findings are detailed in our Q3 2023 technical update in Section 3.5. Intel and Georgia Tech continue to meet weekly via Teams to discuss both research and logistics.

6.6 Q4 2023/Q1 2024 Update

Intel and Georgia Tech continue to meet weekly via Teams to discuss both research and logistics. Intel hosted an intern, Kevin Stangl, who is supported by GARD through the Toyota Technological Institute at Chicago. The results of his internship are described in Section 3.6.3. Intel also reviewed our progress in total at the final GARD PI meeting. Finally, we reviewed the GARD program and learning's with Intel leadership.

Bibliography

- [ACMH23] Marius Arvinte, Cory Cornelius, Jason Martin, and Nageen Himayat. “Investigating the Adversarial Robustness of Density Estimation Using the Probability Flow ODE”. In: *NeurIPS 2023 Workshop on Diffusion Models*. 2023. arXiv: [2310.07084](https://arxiv.org/abs/2310.07084) [cs.LG].
- [AEIK18] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. “Synthesizing robust adversarial examples”. In: *International Conference on Machine Learning*. 2018, pp. 284–293.
- [AI23] Themis AI. *LinkedIn Post*. 2023. URL: <https://www.linkedin.com/feed/update/urn:li:activity:7112853665249431552/> (visited on 11/15/2023).
- [Ali+21] Alaaeldin Ali, Hugo Touvron, Mathilde Caron, Piotr Bojanowski, Matthijs Douze, Armand Joulin, Ivan Laptev, Natalia Neverova, Gabriel Synnaeve, Jakob Verbeek, et al. “Xcit: Cross-covariance image transformers”. In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 20014–20027.
- [AS18] Anish Athalye and Ilya Sutskever. “Synthesizing robust adversarial examples”. In: *Proceedings of the 35th International Conference on Machine Learning*. 2018.
- [BCZOCG18] Tom B. Brown, Nicholas Carlini, Chiyuan Zhang, Catherine Olsson, Paul Christiano, and Ian Goodfellow. *Unrestricted Adversarial Examples*. 2018. arXiv: [1809.08352](https://arxiv.org/abs/1809.08352) [stat.ML].
- [Bra00] G. Bradski. “The OpenCV Library”. In: *Dr. Dobb’s Journal of Software Tools* (2000).
- [CCMC18] Shang-Tse Chen, Cory Cornelius, Jason Martin, and Duen Horng Polo Chau. “ShapeShifter: Robust Physical Adversarial Attack on Faster R-CNN Object Detector”. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2018, pp. 52–68.

- [Che18] Ricky T. Q. Chen. *torchdiffeq*. 2018. URL: <https://github.com/rtqichen/torchdiffeq>.
- [Cor+24] Cory Cornelius et al. *Object Sensing and Cognition for Adversarial Robustness*. 2024. URL: <https://github.com/IntelLabs/OSCAR>.
- [CTDRSK22] Nicholas Carlini, Florian Tramer, Krishnamurthy Dj Dvijotham, Leslie Rice, Mingjie Sun, and J Zico Kolter. *(Certified!!) Adversarial Robustness for Free!* 2022. arXiv: [2206.10550 \[cs.LG\]](#).
- [DDSLLF09] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. “Imagenet: A large-scale hierarchical image database”. In: *2009 IEEE conference on computer vision and pattern recognition*. IEEE. 2009, pp. 248–255.
- [DMM22] Sihui Dai, Saeed Mahloujifar, and Prateek Mittal. “Parameterizing activation functions for adversarial robustness”. In: *2022 IEEE Security and Privacy Workshops (SPW)*. IEEE. 2022, pp. 80–87.
- [Dos+20] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xi-aohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. *An image is worth 16x16 words: Transformers for image recognition at scale*. 2020. arXiv: [2010.11929 \[cs.CV\]](#).
- [DRCLK17] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. “CARLA: An Open Urban Driving Simulator”. In: *Proceedings of the 1st Annual Conference on Robot Learning*. 2017, pp. 1–16.
- [HC23] Alec Helbling and Duen Horng Chau. “ManimML: Communicating Machine Learning Architectures with Animation”. In: *IEEE VIS 2023: Poster*. 2023.
- [HJA20] Jonathan Ho, Ajay Jain, and Pieter Abbeel. “Denoising diffusion probabilistic models”. In: *Advances in Neural Information Processing Systems 33* (2020), pp. 6840–6851.
- [HLDB22] Shihua Huang, Zhichao Lu, Kalyanmoy Deb, and Vishnu Naresh Boddeti. *Revisiting Residual Networks for Adversarial Robustness: An Architectural Perspective*. 2022. arXiv: [2212.11005 \[cs.CV\]](#).
- [HPHC23] Alec Helbling, Mansi Phute, Matthew Hull, and Duen Horng Chau. “LLM Self Defense: By self examination, LLMs know they are being tricked”. In: 2023. arXiv: [2308.07308v2 \[cs.CL\]](#).

- [HWC24] Matthew Hull, Zijie J Wang, and Duen Horng Chau. “REVAMP: Automated Simulations of Adversarial Attacks on Arbitrary Objects in Realistic Scenes”. In: *Tiny Papers at the Twelfth International Conference on Learning Representations*. 2024. arXiv: [2310.12243](https://arxiv.org/abs/2310.12243) [cs.LG].
- [HWEGBM21] Hanxun Huang, Yisen Wang, Sarah Erfani, Quanquan Gu, James Bailey, and Xingjun Ma. “Exploring architectural ingredients of adversarially robust deep neural networks”. In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 5545–5559.
- [HZRS16] 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*. 2016, pp. 770–778.
- [Intel22] Intel Corporation. *Intel Labs at Intel Innovation*. Sept. 2022. URL: <https://www.intel.com/content/www/us/en/research/news/labs-at-innovation-2022.html>.
- [Intel23a] Intel Corporation. *Intel Labs at Intel Innovation*. Sept. 2023. URL: <https://www.intel.com/content/www/us/en/newsroom/resources/intel-innovation-2023.html>.
- [Intel23b] Intel Corporation. *Intel Labs at Intel Vision*. May 2023. URL: <https://www.intel.com/content/www/us/en/events/on-event-series/vision.html>.
- [ISTETM19] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. “Adversarial examples are not bugs, they are features”. In: *Advances in neural information processing systems* 32 (2019).
- [JADPDT21] Ajil Jalal, Marius Arvinte, Giannis Daras, Eric Price, Alexandros G Dimakis, and Jon Tamir. “Robust compressed sensing mri with deep generative priors”. In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 14938–14954.
- [Jak+22] Wenzel Jakob, Sébastien Speierer, Nicolas Roussel, Merlin Nimier-David, Delio Vicini, Tizian Zeltner, Baptiste Nicolet, Miguel Crespo, Vincent Leroy, and Ziyi Zhang. *Mitsuba 3 Renderer*. Version 3.0.1. <https://mitsuba-renderer.org>. 2022.
- [JZKZRD23] Jongheon Jeong, Yang Zou, Taewan Kim, Dongqing Zhang, Avinash Ravichandran, and Onkar Dabeer. “Winclip: Zero-/few-shot anomaly classification and segmentation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 19606–19616.

- [Kat+20] Hiroharu Kato, Deniz Beker, Mihai Morariu, Takahiro Ando, Toru Matsuoka, Wadim Kehl, and Adrien Gaidon. *Differentiable rendering: A survey*. 2020. arXiv: [2006.12057 \[cs.CV\]](#).
- [KH+09] Alex Krizhevsky, Geoffrey Hinton, et al. *Learning multiple layers of features from tiny images*. 2009.
- [LADL18] Tzu-Mao Li, Miika Aittala, Frédo Durand, and Jaakko Lehtinen. “Differentiable Monte Carlo Ray Tracing through Edge Sampling”. In: *ACM Transactions on Graphics* 37.6 (2018).
- [LDC23] Kevin Li, Rahul Duggal, and Duen Horng Chau. “Evaluating Robustness of Vision Transformers on Imbalanced Datasets (Student Abstract)”. In: *2023 AAAI Conference on Artificial Intelligence* (2023).
- [LH17] Ilya Loshchilov and Frank Hutter. *Decoupled weight decay regularization*. 2017. arXiv: [1711.05101 \[cs.LG\]](#).
- [Liu+21] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. “Swin transformer: Hierarchical vision transformer using shifted windows”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 10012–10022.
- [Liu+22] Xiruo Liu, Shibani Singh, Cory Cornelius, Colin Busho, Mike Tan, Anindya Paul, and Jason Martin. “Synthetic Dataset Generation for Adversarial Machine Learning Research”. In: *New Frontiers in Adversarial Machine Learning ICML 2022*. 2022. arXiv: [2207.10719 \[cs.CV\]](#).
- [LMWFDX22] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. “A convnet for the 2020s”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 11976–11986.
- [Mar23] Dylan Martin. *5 Cutting-Edge Technologies We Saw At Intel Innovation 2023*. 2023. URL: <https://www.crn.com/news/software/5-cutting-edge-technologies-we-saw-at-intel-innovation-2023/2> (visited on 11/15/2023).
- [MMSTV18] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. “Towards Deep Learning Models Resistant to Adversarial Attacks”. In: *International Conference on Learning Representations*. 2018.

- [Mot23] Wallace Moté. *Intel Innovation 2023 mostra que IA é muito mais que ChatGPT*. 2023. URL: <https://www.terra.com.br/byte/intel-innovation-2023-mostra-que-ia-e-muito-mais-que-chatgpt,37ddf67d43907c51cba4efc61c71c4af17gdawj1.html> (visited on 11/15/2023).
- [Nic+18] Maria-Irina Nicolae et al. *Adversarial Robustness Toolbox v1.2.0*. 2018. arXiv: [1807.01069](https://arxiv.org/abs/1807.01069) [cs.LG].
- [Par+23] Haekyu Park, Seongmin Lee, Benjamin Hoover, Austin Wright, Omar Shaikh, Rahul Duggal, Nilaksh Das, Judy Hoffman, and Duen Horng Chau. “Concept Evolution in Deep Learning Training: A Unified Interpretation Framework and Discoveries”. In: *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*. 2023.
- [Pen+23a] ShengYun Peng, Weilin Xu, Cory Cornelius, Matthew Hull, Kevin Li, Rahul Duggal, Mansi Phute, Jason Martin, and Duen Horng Chau. “Robust Principles: Architectural Design Principles for Adversarially Robust CNNs”. In: *British Machine Vision Conference (BMVC)*. 2023. arXiv: [2308.16258](https://arxiv.org/abs/2308.16258) [cs.CV].
- [Pen+23b] ShengYun Peng, Weilin Xu, Cory Cornelius, Kevin Li, Rahul Duggal, Duen Horng Chau, and Jason Martin. *RobArch: Designing Robust Architectures against Adversarial Attacks*. 2023. arXiv: [2301.03110](https://arxiv.org/abs/2301.03110) [cs.CV].
- [Phu+23] Mansi Phute, Alec Helbling, Matthew Hull, ShengYun Peng, Sebastian Szyller, Cory Cornelius, and Duen Horng Chau. “LLM Self Defense: By Self Examination, LLMs Know They Are Being Tricked”. In: *Tiny Papers at the Twelfth International Conference on Learning Representations 2024*. 2023. arXiv: [2308.07308](https://arxiv.org/abs/2308.07308) [cs.CL].
- [Rad+21] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. “Learning transferable visual models from natural language supervision”. In: *International conference on machine learning*. PMLR. 2021, pp. 8748–8763.
- [Rav+20] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. *Accelerating 3D Deep Learning with PyTorch3D*. 2020. arXiv: [2007.08501](https://arxiv.org/abs/2007.08501) [cs.CV].
- [RBB17] Jonas Rauber, Wieland Brendel, and Matthias Bethge. “Foolbox: A Python toolbox to benchmark the robustness of machine learning models”. In: *Reliable Machine Learning in the Wild Workshop, 34th International Conference on Machine Learning*. 2017. URL: <http://arxiv.org/abs/1707.04131>.

- [RF18] Joseph Redmon and Ali Farhadi. *YOLOv3: An Incremental Improvement*. 2018. arXiv: [1804.02767 \[cs.CV\]](#).
- [RHGS15] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”. In: *Advances in Neural Information Processing Systems*. 2015, pp. 91–99.
- [RRKB11] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. “ORB: An efficient alternative to SIFT or SURF”. In: *2011 International conference on computer vision*. IEEE. 2011, pp. 2564–2571.
- [SÁGSNL19] Joan Serrà, David Álvarez, Vicenç Gómez, Olga Slizovskaia, José F Núñez, and Jordi Luque. *Input complexity and out-of-distribution detection with likelihood-based generative models*. 2019. arXiv: [1909.11480 \[cs.LG\]](#).
- [SAXC24] Kevin Stangl, Marius Arvinte, Weilin Xu, and Cory Cornelius. “Investigating the Semantic Robustness of CLIP-based Zero-Shot Anomaly Segmentation”. In: *Visual Anomaly and Novelty Detection (VAND 2.0) Workshop at the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2024)*. arXiv: [2405.07969 \[cs.AI\]](#).
- [SME20] Jiaming Song, Chenlin Meng, and Stefano Ermon. *Denoising diffusion implicit models*. 2020. arXiv: [2010.02502 \[cs.LG\]](#).
- [SSCH24] Christian Schlarmann, Naman Deep Singh, Francesco Croce, and Matthias Hein. “Robust CLIP: Unsupervised Adversarial Fine-Tuning of Vision Embeddings for Robust Large Vision-Language Models”. In: *arXiv preprint arXiv:2402.12336* (2024).
- [SSKKEP20] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. *Score-based generative modeling through stochastic differential equations*. 2020. arXiv: [2011.13456 \[cs.LG\]](#).
- [Sze+15] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. “Going deeper with convolutions”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 1–9.
- [WC23] Zijie J Wang and Duen Horng Chau. “WebSHAP: Towards Explaining Any Machine Learning Models Anywhere”. In: *Companion Proceedings of the ACM Web Conference 2023*. 2023, pp. 262–266.

- [WHC23] Zijie J Wang, Fred Hohman, and Duen Horng Chau. “WizMap: Scalable Interactive Visualization for Exploring Large Machine Learning Embeddings”. In: *Proceedings of the 61th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. 2023.
- [WMMYHC23] Zijie J Wang, Evan Montoya, David Munechika, Haoyang Yang, Benjamin Hoover, and Duen Horng Chau. “DiffusionDB: A Large-scale Prompt Gallery Dataset for Text-to-Image Generative Models”. In: *Proceedings of the 61th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2023.
- [WPDLLY23] Zekai Wang, Tianyu Pang, Chao Du, Min Lin, Weiwei Liu, and Shuicheng Yan. *Better diffusion models further improve adversarial training*. 2023. arXiv: [2302.04638 \[cs.CV\]](#).
- [WRK20] Eric Wong, Leslie Rice, and J Zico Kolter. *Fast is better than free: Revisiting adversarial training*. 2020. arXiv: [2001.03994 \[cs.LG\]](#).
- [WZYBMG20] Yisen Wang, Difan Zou, Jinfeng Yi, James Bailey, Xingjun Ma, and Quanquan Gu. “Improving adversarial robustness requires revisiting misclassified examples”. In: *International Conference on Learning Representations*. 2020.
- [XTGYL20] Cihang Xie, Mingxing Tan, Boqing Gong, Alan Yuille, and Quoc V Le. *Smooth adversarial training*. 2020. arXiv: [2006.14536 \[cs.CV\]](#).
- [ZK16] Sergey Zagoruyko and Nikos Komodakis. *Wide residual networks*. 2016. arXiv: [1605.07146 \[cs.CV\]](#).
- [ZPK18] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. *Open3D: A Modern Library for 3D Data Processing*. 2018. arXiv: [1801.09847 \[cs.CV\]](#).
- [ZYJXEJ19] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. “Theoretically principled trade-off between robustness and accuracy”. In: *International Conference on Machine Learning*. 2019, pp. 7472–7482.

[End of Document]