

Designing and Evaluating Physical Adversarial Attacks and Defenses for Machine Learning Algorithms

by

Kevin Eykholt

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Computer Science and Engineering)
in The University of Michigan
2019

Doctoral Committee:

Professor Atul Prakash, Chair
Professor Vineet Kamat
Assistant Professor Bo Li
Professor Z. Morley Mao

Kevin Eykholt

keykholt@umich.edu

ORCID iD: 0000-0002-7040-1657

© Kevin Eykholt 2019

ACKNOWLEDGEMENTS

I thank my advisor, Atul Prakash, who acted as my graduate compass. He helped point me in a research direction, but gave me the freedom to explore it and allowing me to grow into the researcher I am now.

Earlence Fernandes and Amir Rahmati helped guide me through the ins and outs of being a graduate student. Their deep research and writing experience was invaluable to me and our collaborations were thoroughly enjoyable. I am lucky to have met such supportive friends and collaborators.

I thank my family: Mark, Jessica, and Brian for inspiring me to attend graduate school and supporting me along the way. Though it was hard work, having surprises like a Whole Foods Thanksgiving dinner and a Thanksgiving Skype call when I could not go home due to research definitely were welcome reminders that I have people looking out for me.

Ayesha Sundaram, my Fiancée and soon-to-be wife, went out of her way to attend school in Michigan to be by my side. She listened to my worries and helped me weather through the storm that is graduate research. Thank you for being by my side.

I am grateful to my graduate friends and colleagues Nikita Bhutani, Abhilash Dighe, Zhongjun Jin, Daniel LeJeune, Niharika Maheshwarim, Jie Song, and Will Sullivan.

I thank the members of my thesis committee Vineet Kamat, Bo Li, and Z. Morely Mao for their insightful suggestions and comments on this dissertation.

I thank my other collaborators Ivan Etimov, Tadayoshi Kohno, Dawn Song, Florian

Tramèr, and Chaowei Xiao for their work with developing and testing the RPP attack. With so many different research paths to explore, develop, and test, it would have been impossible to finish the work without their support.

I am grateful for the knowledge and assistance of Swati Gupta at Georgia Tech. Her unique insights and formal mathematical expertise greatly improved my originally messy formalization of robust feature augmentation.

Finally, thanks to the past and current members of Atul’s team: Alex Crowell and Haizhong Zheng. My first project as I began graduate research was inspired by Alex Crowell’s work on database systems and my last project before leaving was inspired in part by Haizhong Zheng’s work on adversarial machine learning.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS	ii
LIST OF FIGURES	vii
LIST OF TABLES	ix
LIST OF APPENDICES	xi
LIST OF ABBREVIATIONS	xii
ABSTRACT	xiii
CHAPTER	
I. Introduction	1
1.1 Contributions of This Dissertation	3
1.1.1 Developing Physical Adversarial Attacks for Com- puter Vision Systems	3
1.1.2 Robust Feature Augmentation	4
II. Related Works	6
2.1 Adversarial Attacks	6
2.1.1 White-Box Attacks	7
2.1.2 Black-Box Attacks	10
2.2 Adversarial Defenses	11
III. Robust Physical-World Attacks on Deep Learning Visual Clas- sification	15
3.1 Introduction	15
3.2 Related Work	19
3.3 Adversarial Examples for Physical Objects	21
3.3.1 Physical World Challenges	21

3.3.2	Robust Physical Perturbation	22
3.4	Experiments	25
3.4.1	Dataset and Classifiers	26
3.4.2	Experimental Design	26
3.4.3	Results for LISA-CNN	28
3.4.4	Results for GTSRB-CNN	31
3.4.5	Results for Inception-v3	32
3.5	Discussion	33
3.6	Conclusion	35
 IV. Physical Adversarial Examples for Object Detectors		36
4.1	Introduction	36
4.2	Related Work	40
4.3	Background on Object Detectors	42
4.4	Physical Adversarial Examples for Object Detectors	44
4.4.1	The RPP Attack	44
4.4.2	Extensions to the RPP attack for Object Detectors	46
4.5	Evaluation	49
4.5.1	Experimental Setup	50
4.5.2	Experimental Results	51
4.6	Future Work	53
4.7	Conclusion	55
 V. Robust Feature Augmentation		58
5.1	Introduction	58
5.2	Related Work	61
5.3	Preliminaries	63
5.3.1	Notation and Definitions	63
5.3.2	Adversarial Training	65
5.4	Robust Feature Augmentation	65
5.4.1	Binarization	67
5.4.2	Group Feature Extraction	70
5.5	Binarization Augmentation on MNIST Results	73
5.6	Group Feature Extraction Results	76
5.6.1	Dataset Description	76
5.6.2	Model Details	77
5.6.3	Experiment Results	78
5.7	Conclusion	79
 VI. Future Work & Conclusion		81
6.1	Conclusion	81
6.2	Future Work	82

6.2.1	Redefinition of Adversarial Constraints	82
6.2.2	Automatic Robust Feature Extraction	83
6.2.3	Adversarial Defense through Foreground Extraction	84
APPENDICES		85
A.1	Model Description	86
C.1	Sign Colors	91
C.2	Color Extraction Algorithm	92
C.2.1	Sign Localization	92
C.2.2	Color Classification	95
BIBLIOGRAPHY		96

LIST OF FIGURES

Figure

2.1	An image of a panda and its corresponding adversarial example. Adversarial examples are the results of small, imperceptible changes to a correctly classified input. Image credit to Goodfellow <i>et al.</i> [26].	7
3.1	The left image shows real graffiti on a Stop sign, something that most humans would not think is suspicious. The right image shows our a physical perturbation applied to a Stop sign. We design our perturbations to mimic graffiti, and thus “hide in the human psyche.”	17
3.2	RPP attack pipeline overview. The input is the target Stop sign. The RPP attack samples from a distribution that models physical dynamics (in this case, varying distances and angles), and uses a mask to project computed perturbations to a shape that resembles graffiti. The adversary prints out the resulting perturbations and sticks them to the target Stop sign.	18
4.1	For an input scene, the YOLO v2 CNN outputs a $19 \times 19 \times 425$ tensor. To generate this tensor, YOLO divides the input image into a square grid of S^2 cells ($S = 19$). For each grid cell, there are B bounding boxes ($B = 5$). Each bounding box predicts 5 values: probability of an object in the cell, co-ordinates of the bounding box (center x, center y, width, height). Additionally, for each bounding box the model predicts a probability distribution over all 80 output classes.	43
4.2	An example of an adversarial perturbation overlaid on a synthetic background. The Stop sign in the image is printed such that it is the same size as a U.S. Stop sign. Then, we cut out the two rectangle bars, and use the original print as a stencil to position the cutouts on a real Stop sign.	45
4.3	The image in (a) shows the image as it is stored digitally. The result of printing and taking a picture of the image in (a) is shown in (b).	45
4.4	Output of the extended RPP algorithm to attack YOLO v2 using poster and sticker attacks.	50
4.5	Patch created by the Creation Attack, aimed at fooling YOLO v2 into detecting nonexistent Stop signs.	50

4.6	Sample frame from our creation attack video after being processed by YOLO v2. The scene includes 4 adversarial stickers reliably recognized as Stop signs.	53
4.7	Sample frames from our attack videos after being processed by YOLO v2. In the majority of frames, the detector fails to recognize the Stop sign.	57
5.1	The MNIST image in (a) is correctly classified as a “4”, however image in (b) is misclassified as an “8”, despite only minor visual distortions in the image. Similarly, the image in (c) is correctly classified as a STOP sign, but the image in (d) is misclassified as German KEEP LEFT sign.	66
5.2	(a) Max-margin linear classifier, trained over pure data points \mathcal{P} , results in large adversarial input space. Binarizing test data to the nearest-neighbor in \mathcal{P} before classification removes these adversarial inputs completely. (b) When \mathcal{P} is not known, binarization to the nearest lattice point reduces adversarial input space.	67
5.3	The MNIST model with a binarization function b and classifier L	69
5.4	Basic architecture of a robust classification network using group classifiers.	71
5.5	The adversarial performance during testing (left) and training (right). Not shown in the figure: <i>MAT and BAT take approximately 10x more time per training iteration than BIN.</i>	75
5.6	The robustness of the color classifier for STOP when changing to blue or yellow signs as L_∞ bound increases.	79
5.7	Some examples of inputs the color classifier is not robust on. Often, this occurs due to either the image being too dark (which tends to shift colors to blue) or the image being too blurry (which causes errors during sign localization).	79
C.1	Examples images of signs for the three color classes we evaluated.	92
C.2	The color extractor pipeline. We show the step-by-step process for a STOP image.	94

LIST OF TABLES

Table

3.1	Sample of physical adversarial examples against LISA-CNN and GTSRB-CNN.	29
3.2	Targeted physical perturbation experiment results on LISA-CNN using a poster-printed Stop sign (subtle attacks) and a real Stop sign (camouflage graffiti attacks, camouflage art attacks). For each image, the top two labels and their associated confidence values are shown. The misclassification target was Speed Limit 45. See Table 3.1 for example images of each attack. Legend: SL45 = Speed Limit 45, STP = Stop, YLD = Yield, ADL = Added Lane, SA = Signal Ahead, LE = Lane Ends.	30
3.3	Drive-by testing summary for LISA-CNN. In our baseline test, all frames were correctly classified as a Stop sign. We have manually added the yellow boxes as a visual guide.	31
3.4	A camouflage art attack on GTSRB-CNN. See example images in Table 3.1. The targeted-attack success rate is 80% (true class label: Stop, target: Speed Limit 80).	32
3.5	Sticker perturbation attack on the Inception-v3 classifier. The original classification is microwave and the attacker’s target is phone. See example images in Appendix B Table B.1. Our targeted-attack success rate is 90%.	33
3.6	Sticker perturbation attack on the Inception-v3 classifier. The original classification is coffee mug and the attacker’s target is cash machine. See example images in Appendix B Table B.2. Our targeted-attack success rate is 71.4%.	34
4.1	Attack success rate for the disappearance attack on YOLO v2. We tested a poster perturbation, where a true-sized print is overlaid on a real Stop sign, and a sticker attack, where the perturbation is two rectangles stuck to the surface of the sign. The table cells show the ratio: number of frames in which a Stop sign was <i>not</i> detected / total number of frames, and a success rate, which is the result of this ratio.	51

4.2	Attack success rate for the disappearance attack on Faster R-CNN. We tested a poster perturbation, where the entire Stop sign is replaced with a true-sized print, and a sticker attack, where the perturbation is two rectangles stuck to the surface of the sign. The table cells show the ratio: number of frames in which a Stop sign was <i>not</i> detected / total number of frames, and a success rate, which is the result of this ratio.	52
5.1	The accuracy of each model evaluated against the MNIST test set and L_∞ perturbations within $\epsilon = 0.3$	74
5.2	# Adv. image is the number of adversarial images ($\epsilon = 8$) in which the predicted label matched the adversarial target. The correction rate is the percentage of adversarial examples for which the color extractor outputs red.	78
A.1	Traffic sign classifier architecture. The model expects $32 \times 32 \times 3$ images as input with values in the range $[-0.5, 0.5]$	87
B.1	Uncropped images of the microwave with an adversarial sticker designed for Inception-v3.	89
B.2	Cropped Images of the coffee mug with an adversarial sticker designed for Inception-v3.	90
C.1	Class labels of the LISA-GTSRB traffic sign dataset used in the experiments.	93
C.2	Red and blue sign groupings. Yellow is not included as they have been grouped into a single label with respect to classification.	93

LIST OF APPENDICES

Appendix

A. Traffic Sign Classifier Details 86

B. Inception-v3 Physical Adversarial Images 88

C. Color Extractor 91

LIST OF ABBREVIATIONS

DNN Deep Neural Network

CNN Convolutional Neural Network

RPP Robust Physical Perturbations

YOLO You Only Look Once

JSMA Jacobian-based Saliency Map Attack

FGSM Fast Gradient Sign Method

GTSRB German Traffic Sign Recognition Benchmark

ABSTRACT

Studies show that state-of-the-art deep neural networks (DNNs) are vulnerable to adversarial examples, resulting from small-magnitude perturbations added to the input in a calculated fashion. These perturbations induce mistakes in the network’s output. However, despite the large interest and numerous works, there have only been limited studies on the impact of adversarial attacks in the physical world. Furthermore, these studies lack well-developed, robust methodologies for attacking real physical systems.

In this dissertation, we first explore the technical requirements for generating physical adversarial inputs through the manipulation of physical objects. Based on our analysis, we design a new adversarial attack algorithm, Robust Physical Perturbations (RPP) that consistently computes the necessary modifications to ensure the modified object remains adversarial across numerous varied viewpoints. We show that the RPP attack results in physical adversarial inputs for classification tasks as well as object detection tasks, which, prior to our work, were considered to be resistant.

We, then, develop a defensive technique, robust feature augmentation, to mitigate the effect of adversarial inputs, both digitally and physically. We hypothesize the input to a machine learning algorithm contains predictive feature information that a bounded adversary is unable to manipulate in order to cause classification errors. By identifying and extracting adversarially robust feature information, we can obtain evidence of the possible set of correct output labels and adjust the classification decision accordingly. As adversarial inputs are a human-defined phenomenon, we leverage human-recognizable features to identify adversarially robust, predictive feature information for a given problem domain. Due to the safety-critical nature of

autonomous driving, we focus our study on traffic sign classification and localization tasks.

CHAPTER I

Introduction

Deep Neural Networks (DNNs) are being used with great success in a variety of domains ranging from speech processing [32] to medical diagnostics [18] and are increasingly being used to control physical objects such as cars [44], UAVs [7], and robots [92]. These DNNs used for the control of physical objects rely on sensory input to inform decisions. However, many works have demonstrated that DNNs are vulnerable to adversarial inputs [26, 40, 41, 81, 64, 12, 70, 35]. These maliciously crafted inputs specifically modify key parts of the input to a DNN in order to cause the systems they control to misbehave in unexpected and potentially dangerous ways.

This thesis examines adversarial inputs from both the perspective of an adversary and a defender. As an adversary, most adversarial attack techniques assume digital access to machine learning systems in order to craft adversarial examples, which is not always possible. Thus, in an effort to develop a more realistic threat model, we analyze the challenges of creating robust, physical adversarial inputs for computer vision systems. Based on our analysis, we develop a new adversarial attack, in which, we can add small, colored stickers to objects in the environment and cause computer vision systems to misidentify or fail to detect the modified objects.

Due to the pervasive use of machine learning technologies, the existence of adversarial inputs is a potential concern wherever machine learning is used, especially in safety

or security critical tasks. Thus, it is of critical importance to also develop techniques, which prevent the use of or mitigate the risks of adversarial inputs. Informed by our work in designing adversarial attacks, we identify a fundamental problem in current machine learning algorithms: the inability to separate and recognize adversarially robust, predictive features. By definition, adversarial inputs are, limited to small, sometimes undetectable distortions of correctly recognized inputs. With respect to computer vision, distinct features such as the color, shape, or size of identifiable objects would be robust to small adversarial modifications. In the second part of the dissertation, we propose a new machine learning technique, which augments the standard machine learning pipeline with robust, predictive information derived from known unmodifiable features in the task domain. We also describe an adversarially robust machine learning architecture with theoretical guarantees and experimental results.

In this dissertation, we **systematically identify the challenges of both attacking and defending supervised machine learning algorithms and, based our analysis, provide novel solutions to overcome those challenges**. We first define a realistic threat model for adversarial attacks on real systems and develop an attack algorithm using this new threat model. Based on our understanding of adversarial attacks, we identify a major limitation in current machine learning training techniques and propose a new technique with theoretical guarantees to address this limitation. Finally, we test both our adversarial attack and defensive techniques in a real machine learning problem domain, traffic sign classification and localization.

1.1 Contributions of This Dissertation

1.1.1 Developing Physical Adversarial Attacks for Computer Vision Systems

Most adversarial attacks assume digital access to the machine learning system, such that fine-grained input manipulations are possible. With respect to computer vision systems, these manipulations manifest themselves as floating point changes to the pixels in an image. However, obtaining digital access to the system could prove hard or impossible. Even if such access was obtained, the adversary is likely able to perform other, more destructive actions such as disabling the vision sensors entirely. Although analyzing adversarial attacks using digital manipulations can prove useful when designing defenses with respect to a strong adversary, we believe that such a threat model is theoretical. Rather, a more realistic threat model is one in which the adversary physically modifies the operational environment of a machine learning system so as to cause predictable errors.

In Chapter III, we detail the process of developing a physical adversarial attack for computer vision classifiers. With autonomous driving as our target domain, we identify several technical challenges an attacker must consider when generating robust, physical adversarial examples. Informed by these challenges, we design the Robust Physical Perturbations (RPP) attack. The attack creates robust, physical adversarial perturbations by sampling a distribution of images containing the target object under varying physical conditions such as different distances, lightning, and camera angles and applying synthetic transformations to the proposed physical adversarial perturbation to roughly match the physical condition of the samples. Our experimental results demonstrate that the adversarial modification proposed by the attack can reliably fool visual classification systems at least 80% of the time.

Computer vision systems are not only limited to classification tasks. Object

detection is another computer vision task in which two sub-tasks are performed: localization and classification. First, given an image of a scene, an object detector must identify zero or more areas of interest in the scene that may contain objects. Then, it attempts to label each object in the proposed areas. In Chapter IV, we analyze and overcome the additional challenges introduced when attacking object detection frameworks. Namely, the localization task introduces an additional challenge for physical adversarial examples: the object’s relative size and position in a scene is variable. To overcome this challenge, we modify the RPP attack to apply synthetic transformations to an image of the target object as well as the adversarial perturbation. These transformations include both the sources of noise considered in Chapter III as well as the new sources of noise due to variability in an object’s size and position in a scene.

We introduce two proof-of-concept adversarial attacks on object detectors, which target the object detector’s localization and classification components separately. Using a well-known object detection framework, You Only Look Once (YOLO), we show that our modified attack is able to create robust, physical adversarial modifications for a target object and fool YOLO in at least 60% of the evaluation images of the target object.

1.1.2 Robust Feature Augmentation

Given a data set of representative examples, a machine learning algorithm is typically trained to maximize the algorithm’s predictive accuracy on any other example from the same distribution. Towards this goal, machine learning algorithms learn to use any predictive feature in the input. Some of these features such as the shape, color, or size of an object are predictive features recognizable to a human. However, machine learning algorithms also learn other equally predictive features, which are imperceptible to humans, and thus used in adversarial attacks.

In Chapter V, we describe a new machine learning technique, *robust feature augmentation*, to improve the adversarial robustness of machine learning classifiers, thus mitigating the effect of adversarial examples. Robust feature augmentation relies on the assumption that there exists a set of predictive features that cannot be modified due to constraints on the adversary. We denote such features as robust, and can use these features as supporting evidence for the final prediction. Under the assumption that robust features exist and can be reliably extracted, we discuss two possible architectures, which have theoretical guarantees to mitigate or prevent the effect of adversarial examples. Furthermore, we deploy robust feature augmentation for digit recognition and traffic sign classification. In both cases, robust feature augmentation improves the adversarial robustness of machine learning classifiers by at least 75%. We also show that combining robust feature augmentation with another adversarial mitigation technique, adversarial training, can further improve adversarial robustness.

CHAPTER II

Related Works

We survey the related work in regarding digital adversarial attacks on classifiers and adversarial defenses. For work related to physical adversarial attacks on classifiers refer to Chapter III. For work related to both digital and physical adversarial attacks on object detectors, refer to Chapter IV.

2.1 Adversarial Attacks

Given a classifier $f_{\theta}(\cdot)$ with parameters θ and an input x with ground truth label y , an adversarial example x' is generated such it is “close” to x measured by a certain distance metric, such as the L_p norm distance. This closeness ensures that the adversarial example is perceptibly identical to the original input. x' is also designed to cause the classifier to make an incorrect prediction. If x' is generated such that $f_{\theta}(x') = y^*$ for a specific $y^* \neq y$, then x' is a targeted adversarial input. Otherwise, if x' is generated such that $f_{\theta}(x') \neq y$, then x' is an untargeted adversarial input. See Figure 2.1 for an example of an input and its adversarial counterpart [26].

Existing work in generating adversarial examples for machine learning algorithms can be divided into either *white-box* or *black-box* adversarial attack algorithms. In a white-box setting, the attacker is assumed to have full access to the classifier including the model weights and parameters. In a black-box setting, the adversary can typically

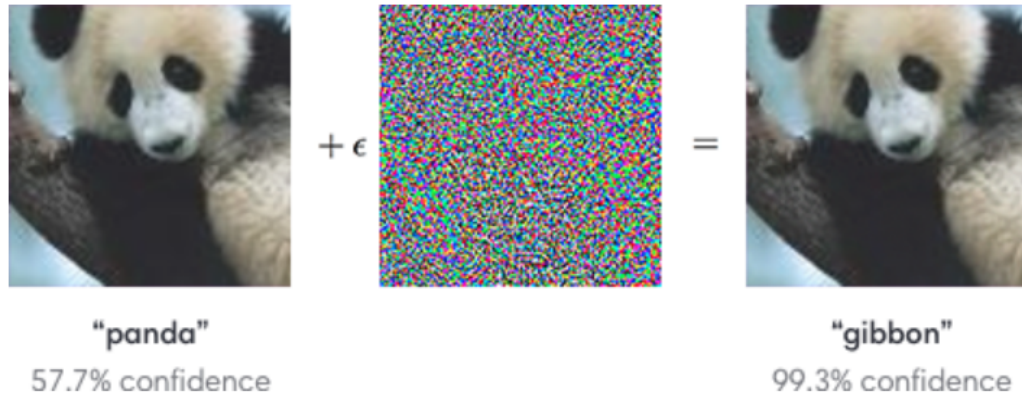


Figure 2.1: An image of a panda and its corresponding adversarial example. Adversarial examples are the results of small, imperceptible changes to a correctly classified input. Image credit to Goodfellow *et al.* [26].

only access the softmax output or the final classification decision. In some cases, the number of queries the adversary can make is also limited.

Most of the work in this field assumes that the attacker has “digital-level” access to an input. For example, the attacker can make arbitrary pixel-level changes to the input image of a classifier. When considering the use of deep learning in cyber-physical systems (*e.g.*, in an autonomous vehicle), these attacks thus implicitly assume that the adversary controls a DNN’s input system (*e.g.*, a camera). A stronger and more realistic threat model would assume that the attacker only controls the physical layer, such as the environment or objects that the system interacts with, but not the internal sensors and data pipelines of the system. We discuss such a threat model in Chapters III and IV.

2.1.1 White-Box Attacks

Numerous different methods have been proposed to generate adversarial examples in a white-box setting, where the adversary has full access to the classifier and its data [81, 26, 12, 57, 64, 6, 37]. In this section, we will discuss a few of the more well-known or widely used white-box attacks.

In the initial work by Szegedy *et al.*, given an input x they generated adversarial

examples by finding the closest image x' , based on L_2 distance, with target label l [81]. Formally, they solved the following box-constrained optimization problem:

$$\min c * \|x - x'\|_2 + \text{loss}_{f,l}(x') \quad (2.1)$$

where $\text{loss}_{f,l}$ is a continuous loss function and c is a regularization constant. In their work, they used line search to find the smallest $c > 0$ for which the adversarial example worked.

Papernot *et al.* proposed an alternative approach, the Jacobian-based Saliency Map Attack (JSMA), for crafting adversarial examples constrained by the L_0 norm [64]. Given an input x and an adversarial label y^* , they use the forward derivative $\Delta f(x)$ to construct an *adversarial saliency map* $S(x, y^*, i)$. For each input feature i :

$$S(x, y^*)[i] = \begin{cases} 0 & \frac{\Delta f_{y^*}(x)}{\Delta x_i} < 0 \text{ or } \sum_{j \neq y^*} \frac{\Delta f_j(x)}{\Delta x_i} > 0 \\ \left(\frac{\Delta f_{y^*}(x)}{\Delta x_i} \right) \left| \sum_{j \neq y^*} \frac{\Delta f_j(x)}{\Delta x_i} \right| & \text{otherwise} \end{cases}$$

This map measures the impact each input feature (*e.g.*, pixel) has on the overall classification. Changing pixels with a high value on the saliency map increases the likelihood of the input being misclassified as y^* . In each iteration, the attack chooses the best pixel based on the saliency map until the input is misclassified as y^* or until the attack is considered detectable based on a maximum number of allowable pixel changes.

Carlini and Wagner improved upon the work done by Szegedy *et al.*, by experimenting with different box-constraint encodings and different objective functions [12]. In their work, they demonstrated that although the box-constraint encoding does not matter, the choice of loss function greatly affects the quality of the adversarial examples. In fact, cross-entropy loss was the worst performing loss function in their

experiments. Based on their analysis, they extended the original box-constrained problem to work with L_0 , L_2 , and L_∞ distance measures and generated higher quality adversarial examples compared to previous work.

Optimization based attacks like the ones just discussed are able to generate high quality adversarial examples, but suffer from poor performance. In an effort to generate adversarial examples quickly, Goodfellow *et al.* proposed a non-optimization based attack, the Fast Gradient Sign Method (FGSM) [26], for generating adversarial examples measured by the L_∞ distance. Given an image x , the adversarial example x' is given by:

$$x' = x + \epsilon * \text{sign}(\nabla \text{loss}_{f,l}(x)) \quad (2.2)$$

where ϵ is very small. In essence, this attack uses the gradient of the loss function to determine the direction each pixel in the input should be tweaked and then modifies them all at the same time by ϵ . Although this first-order attack arrives at an adversarial example very quickly, there is no guarantee that the adversarial example is optimal. An iterative FGSM attack was proposed by Kurakin *et al.*, which was more likely to generate successful adversarial inputs [37]. In each iteration of the attack, the adversarial example is updated by a small amount α :

$$x'_i = \text{clip}_\epsilon(x'_{i-1} + \alpha * \text{sign}(\nabla \text{loss}_{f,l}(x'_{i-1}))) \quad (2.3)$$

where clip_ϵ is a function that ensures the adversarial example is within the ϵ - L_∞ ball from the original input and has valid values for an image. Additional improvements have been made by others, resulting in higher adversarial success rates [93, 86].

2.1.2 Black-Box Attacks

Black-box adversarial attacks study how an adversary can generate adversarial examples without perfect knowledge of the model. Often, the adversary is assumed to have access to either the output probabilities or the output label for a given input. One common attack strategy, proposed by Papernot *et al.*, is to exploit the transferability of adversarial examples [63]. An input which is adversarial to a classifier f' has the possibility to also be adversarial to a different classifier f assuming they are training on similar input data [26, 81]. The attack strategy is divided into two steps. First, the adversary, using the target classifier f as an oracle, trains a substitute classifier f' that is a good enough approximation of the target classifier. Starting from an initial set of inputs, Papernot *et al.* iteratively train a network and use Jacobian-based Dataset Augmentation to create new inputs to query and train on. Once trained, they perform a white-box attack on f' with the expectation that most of the adversarial inputs generated for f' will transfer to f . In their original work, they used neural networks for both the substitute and target models. In a later work, Papernot *et al.* studied the transferability of adversarial examples for five different classifier techniques: neural networks, logistic regression, support vector machines, decision trees, nearest neighbors, and ensemble networks [62]. They concluded that with their black-box attack strategy, a neural network classifier can approximate most other machine learning classifiers and generate highly transferable adversarial examples.

Liu *et al.* also developed a black-box attack that exploits the transferability of adversarial examples through the use of ensemble networks [49]. Unlike the work by Papernot *et al.*, Liu *et al.* assumed the attacker cannot query the target classifier. To generate transferable adversarial examples, they trained an ensemble of classifiers and then generated adversarial examples for the ensemble model. Using the Imagenet dataset, they demonstrated that their technique greatly improves the transferability of untargeted adversarial examples as well as generating targeted transferable adversarial

examples.

Another common attack strategy is query-based adversarial input searches or gradient estimations. Narodytska and Kasiviswanathan describe a greedy local search algorithm, which uses oracle queries to identify a small set of vulnerable pixels in the image that cause large shifts in the output probability [59]. Chen *et al.* and Bhagoji *et al.* both use oracle queries to compute finite differences and estimate the gradient of the target classifier [15, 5]. The estimated gradient is then used to craft an adversarial example. For instance, Bhagoji *et al.* use the estimated gradient to enable the fast gradient sign attack. Often, these query-based black-box attacks rely on access to the output probabilities, but attacks which only use the output label to estimate the gradient have been recently proposed [17].

2.2 Adversarial Defenses

In an effort to mitigate the effect of adversarial examples, there is a push to design defensive measures against adversarial examples. Research in this area can be divided into defensive methods [26, 65, 83, 78], which improve a model’s resistance to adversarial examples, and detection solutions [31, 42, 25], which use statistical properties or other information to identify adversarial examples from natural examples.

The main issue with many of the proposed strategies is that many of them do not provide provable security guarantees under a certain threat model. Rather, they implement a technique, such as gradient masking, PCA analysis, or input normalization, and show experimentally the proposed strategy is resistant to many adversarial attacks. However, when exposed to an adaptive adversary that modifies the attack based on the defense, the defense fails [12, 2, 11, 3].

Currently, the most reliable technique to improve a model’s adversarial robustness is augment the training dataset with adversarial examples [81]. This technique, known as adversarial training, has been shown to significantly improve a model’s robustness

to white-box adversarial attacks. In their work, Madry *et al.* generated adversarial inputs that maximized the model’s loss during training. This approach resulted in significant improvements in adversarial robustness on the MNIST and CIFAR datasets [51]. Further work by Kurakin *et al.*, demonstrated how to scale adversarial training to larger datasets and models by using a single-step adversarial attack rather than an iterative one [38]. Tramèr *et al.* proposed implementing adversarial training in an ensemble fashion to reduce adversarial training overhead and further improve adversarial robustness for large scale datasets [84]. During training, they pre-computed adversarial inputs for an ensemble of models, which were used to train a new robust model, that was not part of the ensemble. They demonstrated that this ensemble approach both improved the trained model’s robustness to white-box attacks and black-box transferability attacks.

As adversarial training is a data augmentation technique, it is possible to combine it with other techniques to create new adversarial defenses. One such approach is to combine adversarial training with denoising algorithms as adversarial examples are typically generated by adding noise to a correctly labelled input. Gu and Rigazio first explored this approach by adding a denoising autoencoder to the input of a Deep Neural Network (DNN) [29]. Through adversarial training, they created an autoencoder that learned to minimize the pixel-based reconstruction loss between an adversarial input and the original input. However, they discovered that an adaptive adversary can bypass the defense by computing adversarial examples for the full pipeline. The issue lies in the fact that adversarial noise added at the input layer is amplified as it passes through successive layers of a classifier eventually resulting in misclassification. Liao *et al.* improved upon the denoising autoencoder approach by using an objective function designed to mitigate the error amplification effect, though they still did not evaluate their approach with respect to an adaptive adversary [43]. Rather than train the autoencoder to minimize the pixel-based reconstruction loss

between an adversarial input and the original input, they instead train the autoencoder to minimize the following:

$$\|f_i(x) - f_i(x^*)\|_1 \tag{2.4}$$

where f is the classifier’s output at the i -th layer for a given input and x^* is the adversarial input after being pre-processed by the autoencoder. A similar objective function is used by Xie *et al.*, but instead of adding a denoising step to the input of a network, they choose to add denoising steps in between the hidden layers[90].

As an alternative approach to completely negating the effect of adversarial attacks, some recent work has proposed methods that provides provable security guarantees against adversarial attacks under certain assumptions. Hein and Andriushchenko provided the first formal guarantees on the adversarial robustness of a classifier [30]. More specifically, given a specific instance, they demonstrated a lower-bound on the L_2 norm of the input manipulation required to change the output of the classifier. Similarly, Raghunathan *et al.* also established lower bounds on adversarial perturbations, but do so for the L_∞ norm [66].

Concurrently, Wong and Kolter also provided an adversarial robust training methodology for arbitrarily deep ReLU networks given a certain norm-bound on adversarial perturbations [87]. In their work, they established a convex space around each input and ensure that the classification decision for correctly labeled inputs does not change within the space. Then, they optimized the convex space on the training point with the highest training loss. Finally, Sinha *et al.* also provided methods to guarantee adversarial robustness, but their method uses defined distributional Wasserstein distances rather than the norm of the adversarial perturbations [77]. Similar to these works, our robust feature augmentation approach also establishes provable security guarantees against adversarial examples, under the assumption that

robust, predictive feature information exists and can be reliably extracted.

CHAPTER III

Robust Physical-World Attacks on Deep Learning Visual Classification

In this chapter, we evaluate the challenges of deploying adversarial attacks in the physical world. From our analysis, we design a new, robust attack algorithm against deep learning visual classifiers that creates physical adversarial examples through modifications of existing physical object.

3.1 Introduction

Deep Neural Networks (DNNs) have achieved state-of-the-art, and sometimes human-competitive, performance on many computer vision tasks [36, 82, 39]. Based on these successes, they are increasingly being used as part of control pipelines in physical systems such as cars [44, 21], UAVs [7, 58], and robots [92]. Recent work, however, has demonstrated that DNNs are vulnerable to adversarial perturbations [26, 40, 41, 81, 64, 12, 70, 35, 56, 60]. These carefully crafted modifications to the (visual) input of DNNs can cause the systems they control to misbehave in unexpected and potentially dangerous ways.

This threat has gained recent attention, and work in computer vision has made great progress in understanding the space of adversarial examples, beginning in the

digital domain (*e.g.*, by modifying images corresponding to a scene) [56, 60, 81, 26], and more recently in the physical domain [75, 37, 1, 4]. Along similar lines, our work contributes to the understanding of adversarial examples when perturbations are physically added to the *objects themselves*. We choose road sign classification as our target domain for several reasons: (1) The relative visual simplicity of road signs make it challenging to hide perturbations. (2) Road signs exist in a noisy unconstrained environment with changing physical conditions such as the distance and angle of the viewing camera, implying that physical adversarial perturbations should be robust against considerable environmental instability. (3) Road signs play an important role in transportation safety. (4) A reasonable threat model for transportation is that an attacker might not have control over a vehicle’s systems, but is able to modify the objects in the physical world that a vehicle might depend on to make crucial safety decisions.

The main challenge with generating robust physical perturbations is environmental variability. Cyber-physical systems operate in noisy physical environments that can destroy perturbations created using current digital-only algorithms [50]. For our chosen application area, the most dynamic environmental change is the distance and angle of the viewing camera. Additionally, other practicality challenges exist: (1) Perturbations in the digital world can be so small in magnitude that it is likely that a camera will not be able to perceive them due to sensor imperfections. (2) Current algorithms produce perturbations that occupy the background imagery of an object. It is extremely difficult to create a robust attack with background modifications because a real object can have varying backgrounds depending on the viewpoint. (3) The fabrication process (*e.g.*, printing of perturbations) is imperfect.

Informed by the challenges above, we design the *RPP* attack, which can generate perturbations robust to widely changing distances and angles of the viewing camera. The RPP attack creates a visible, but inconspicuous perturbation that only perturbs



Figure 3.1: The left image shows real graffiti on a Stop sign, something that most humans would not think is suspicious. The right image shows our a physical perturbation applied to a Stop sign. We design our perturbations to mimic graffiti, and thus “hide in the human psyche.”

the object (*e.g.*, a road sign) and not the object’s environment. To create robust perturbations, the algorithm draws samples from a distribution that models physical dynamics (*e.g.*, varying distances and angles) using experimental data and synthetic transformations (Figure 3.2).

Using the proposed algorithm, we evaluate the effectiveness of perturbations on physical objects, and show that adversaries can physically modify objects using low-cost techniques to reliably cause classification errors in DNN-based classifiers under widely varying distances and angles. For example, our attacks cause a classifier to interpret a subtly-modified physical Stop sign as a Speed Limit 45 sign. Specifically, our final form of perturbation is a set of black and white stickers that an adversary can attach to a physical road sign (Stop sign). We designed our perturbations to resemble graffiti, a relatively common form of vandalism. It is common to see road signs with random graffiti or color alterations in the real world as shown in Figure 3.1 (the left image is of a real sign in a city). If these random patterns were adversarial perturbations (right side of Figure 3.1 shows our example perturbation), they could lead to severe consequences for autonomous driving systems, without arousing suspicion in human operators.

Given the lack of a standardized method for evaluating physical attacks, we draw

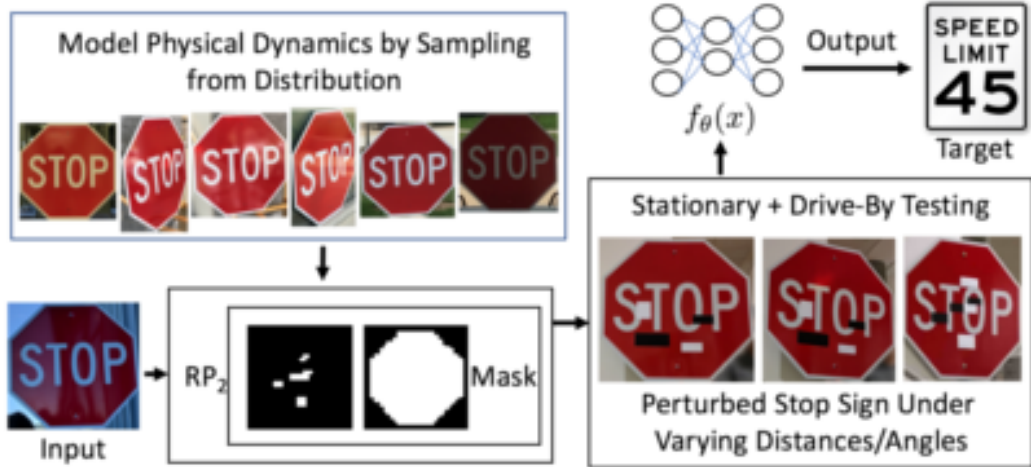


Figure 3.2: RPP attack pipeline overview. The input is the target Stop sign. The RPP attack samples from a distribution that models physical dynamics (in this case, varying distances and angles), and uses a mask to project computed perturbations to a shape that resembles graffiti. The adversary prints out the resulting perturbations and sticks them to the target Stop sign.

on standard techniques from the physical sciences and propose a two-stage experiment design: (1) A lab test where the viewing camera is kept at various distance/angle configurations; and (2) A field test where we drive a car towards an intersection in uncontrolled conditions to simulate an autonomous vehicle. We test our attack algorithm using this evaluation pipeline and find that the perturbations are robust to a variety of distances and angles. Figure 3.2 shows an overview of our pipeline to generate and evaluate robust physical adversarial perturbations.

Our Contributions:

- We introduce the RPP attack, which is the first attack that generates physical perturbations for *physical-world* objects that can consistently cause misclassification in a DNN-based classifier under a range of dynamic physical conditions, including different viewpoint angles and distances (Section 3.3).
- Given the lack of a standardized methodology in evaluating physical adversarial perturbations, we propose an evaluation methodology to study the effectiveness

of physical perturbations in real world scenarios (Section 3.4.2).

- We evaluate our attacks against two standard-architecture classifiers that we built: LISA-CNN with 91% accuracy on the LISA test set and GTSRB-CNN with 95.7% accuracy on the GTSRB test set. Using two types of attacks (object-constrained poster and sticker attacks) that we introduce, we show that the RPP attack produces robust perturbations for real road signs. For example, poster attacks are successful in 100% of stationary and drive-by tests against LISA-CNN, and sticker attacks are successful in 80% of stationary testing conditions and in 87.5% of the extracted video frames against GTSRB-CNN.
- To show the generality of our approach, we generate robust physical adversarial examples for general image recognition tasks and modify physical objects. We show that by adding a single sticker to an object, we can achieve a targeted misclassification success rate of at least 71% on a pre-trained Inception-v3 classifier.

Our work, thus, contributes to understanding the susceptibility of image classifiers to robust adversarial modifications of *physical objects*. These results provide a case for the potential consequences of adversarial examples on deep learning models that interact with the physical world through vision. Our overarching goal with this work is to inform research in building robust vision models and to raise awareness on the risks that future physical learning systems might face.

3.2 Related Work

We survey the related work on physical adversarial attacks on classifiers. For related work on digital adversarial attacks on classifiers, please refer to Chapter II.

Physical adversarial attacks were first proposed by Kurakin *et al.* They generated physical adversarial examples by printing digital adversarial examples on paper [37].

In their work, they found that a significant portion of the printed adversarial examples fooled an image classifier. However, their experiments were done without any variation in the physical conditions such as different viewing angles or distances.

Athalye improved upon the work of Kurakin *et al.* by creating physical adversarial examples for classifiers that are robust to a set of two-dimensional synthetic transformations [1]. Concurrent with our work, Athalye *et al.* later improved upon the original attack creating 3D-printed replicas of perturbed objects [4]. The main intellectual differences include: (1) Athalye *et al.* *only* use a set of synthetic transformations during optimization, which can miss subtle physical effects, while our work samples from a distribution modeling both physical *and* synthetic transformations. (2) Our work modifies *existing* true-sized objects whereas Athalye *et al.* 3D-print small-scale replicas. (3) Our work generates adversarial examples for both classifiers and object detectors.

Sharif *et al.* attacked face recognition systems by printing adversarial perturbations on the frames of eyeglasses [75]. Their most recent work demonstrated successful physical attacks against a variety of classifiers using Generative Adversarial Networks (GAN) to produce the adversarial eyeglasses [76]. In order to account for small variations in the environment conditions (*e.g.*, pose, distance/angle from the camera, and facial expression changes), they would pre-collect training images of the attacker wearing a set of eyeglasses to align the attack. However, the environmental conditions in facial recognition systems have little variance compared classification and object detector tasks on the road. In our work, we explicitly design our perturbations to be effective in the presence of very diverse physical-world conditions (specifically, large distances/angles and resolution changes).

Finally, Brown *et al.* use “adversarial patches” to attack image classifiers [8]. In their work, they relax the imperceptibility constraint on adversarial manipulations and creates multi-colored stickers to add to any scene. Their attack works by adding

a new object, the adversarial patch, to the scene, in addition to the original object. As image inputs are typically expected to contain a single object class, the additional object serves as a strong distraction for the classifier. Although they demonstrate the success of their attack using an adversarial toaster patch, they remark that in a physical setting with varied physical-world conditions, the adversarial patch must occupy a significant portion of the image in order to cause misclassification. In our work, however, it is not necessary for our adversarial perturbations to occupy a large portion of the input image in order to remain adversarial.

3.3 Adversarial Examples for Physical Objects

Our goal is to examine whether it is possible to create robust physical perturbations for real-world objects that mislead classifiers to make incorrect predictions even when images are taken in a range of varying physical conditions. We first present an analysis of environmental conditions that physical learning systems might encounter, and then present our algorithm to generate physical adversarial perturbations taking these challenges into account.

3.3.1 Physical World Challenges

Physical attacks on an object must be able to survive changing conditions and remain effective at fooling the classifier. We structure our discussion of these conditions around the chosen example of road sign classification, which could be potentially applied in autonomous vehicles and other safety sensitive domains. A subset of these conditions can also be applied to other types of physical learning systems such as drones, and robots.

Environmental Conditions. The distance and angle of a camera in an autonomous vehicle with respect to a road sign varies continuously. The resulting images that are fed into a classifier are taken at different distances and angles. Therefore, any

perturbation that an attacker physically adds to a road sign must be able to survive these transformations of the image. Other environmental factors include changes in lighting/weather conditions, and the presence of debris on the camera or on the road sign.

Spatial Constraints. Current algorithms focusing on digital images add adversarial perturbations to all parts of the image, including background imagery. However, for a physical road sign, the attacker cannot manipulate background imagery. Furthermore, the attacker cannot count on there being a fixed background imagery as it will change depending on the distance and angle of the viewing camera.

Physical Limits on Imperceptibility. An attractive feature of current adversarial deep learning algorithms is that their perturbations to a digital image are often so small in magnitude that they are almost imperceptible to the casual observer. However, when transferring such minute perturbations to the real world, we must ensure that a camera is able to perceive the perturbations. Therefore, there are physical limits on how imperceptible perturbations can be, and is dependent on the sensing hardware.

Fabrication Error. To fabricate the computed perturbation, all perturbation values must be valid colors that can be reproduced in the real world. Furthermore, even if a fabrication device, such as a printer, can produce certain colors, there will be some reproduction error [75].

In order to successfully physically attack deep learning classifiers, an attacker should account for the above categories of physical world variations that can reduce the effectiveness of perturbations.

3.3.2 Robust Physical Perturbation

We derive our algorithm starting with the optimization method that generates a perturbation for a single image x , without considering other physical conditions; then, we describe how to update the algorithm taking the physical challenges above into

account. This single-image optimization problem searches for perturbation δ to be added to the input x , such that the perturbed instance $x' = x + \delta$ is misclassified by the target classifier $f_\theta(\cdot)$:

$$\min H(x + \delta, x), \quad \text{s.t.} \quad f_\theta(x + \delta) = y^*$$

where H is a chosen distance function, and y^* is the target class.¹ To solve the above constrained optimization problem efficiently, we reformulate it in the Lagrangian-relaxed form similar to prior work [49, 12].

$$\operatorname{argmin}_\delta \lambda \|\delta\|_p + J(f_\theta(x + \delta), y^*) \tag{3.1}$$

Here $J(\cdot, \cdot)$ is the loss function, which measures the difference between the model’s prediction and the target label y^* . λ is a hyper-parameter that controls the regularization of the distortion. We specify the distance function H as $\|\delta\|_p$, denoting the ℓ_p norm of δ .

Next, we will discuss how the objective function can be modified to account for the *environmental conditions*. We model the distribution of images containing object o under both physical and digital transformations X^V . We sample different instances x_i drawn from X^V . A physical perturbation can only be added to a specific object o within x_i . In the example of road sign classification, o is the stop sign that we target to manipulate.

Given images taken in the physical world, we need to make sure that a single perturbation δ , which is added to o , can fool the classifier under different physical conditions. Concurrent work [4] only applies a set of transformation functions to synthetically sample such a distribution. However, modeling physical phenomena is

¹For untargeted attacks, we can modify the objective function to maximize the distance between the model prediction and the true class. We focus on targeted attacks in the rest of the chapter.

complex and such synthetic transformations may miss physical effects. Therefore, to better capture the effects of changing physical conditions, we sample instance x_i from X^V by both generating experimental data that contains actual physical condition variability as well as synthetic transformations. For road sign physical conditions, this involves taking images of road signs under various conditions, such as changing distances, angles, and lightning. This approach aims to approximate physical world dynamics more closely. For synthetic variations, we randomly crop the object within the image, change the brightness, and add spatial transformations to simulate other possible conditions.

To ensure that the perturbations are only applied to the surface area of the target object o (considering the *spatial constraints* and *physical limits on imperceptibility*), we introduce a mask. This mask serves to project the computed perturbations to a physical region on the surface of the object (*i.e.*, road sign). In addition to providing spatial locality, the mask also helps generate perturbations that are visible but inconspicuous to human observers. To do this, an attacker can shape the mask to look like graffiti—commonplace vandalism on the street that most humans expect and ignore, therefore hiding the perturbations “in the human psyche.” Formally, the perturbation mask is a matrix M_x whose dimensions are the same as the size of input to the road sign classifier. M_x contains zeroes in regions where no perturbation is added, and ones in regions where the perturbation is added during optimization.

In the course of our experiments, we empirically observed that the position of the mask has an impact on the effectiveness of an attack. We therefore hypothesize that objects have strong and weak physical features from a classification perspective, and we position masks to attack the weak areas. Specifically, we use the following pipeline to discover mask positions: (1) Compute perturbations using the L_1 regularization and with a mask that occupies the entire surface area of the sign. L_1 makes the optimizer favor a sparse perturbation vector, therefore concentrating the perturbations

on regions that are most vulnerable. Visualizing the resulting perturbation provides guidance on mask placement. (2) Recompute perturbations using L_2 with a mask positioned on the vulnerable regions identified from the earlier step.

To account for *fabrication error*, we add an additional term to our objective function that models printer color reproduction errors. This term is based upon the Non-Printability Score (NPS) by Sharif *et al.* [75]. See the supplemental materials for a formal definition of NPS.

Based on the above discussion, our final robust spatially-constrained perturbation is thus optimized as:

$$\operatorname{argmin}_{\delta} \lambda \|M_x \cdot \delta\|_p + NPS + \mathbb{E}_{x_i \sim X^v} J(f_{\theta}(x_i + T_i(M_x \cdot \delta)), y^*) \quad (3.2)$$

Here we use function $T_i(\cdot)$ to denote the alignment function that maps transformations on the object to transformations on the perturbation (*e.g.*, if the object is rotated, the perturbation is rotated as well).

Finally, an attacker will print out the optimization result on paper, cut out the perturbation (M_x), and put it onto the target object o . As our experiments demonstrate in the next section, this kind of perturbation fools the classifier in a variety of viewpoints.²

3.4 Experiments

In this section, we will empirically evaluate the proposed RPP attack. We first evaluate a safety sensitive example, Stop sign recognition, to demonstrate the robustness of the proposed physical perturbation. To demonstrate the generality of our approach, we then attack Inception-v3 to misclassify a microwave as a phone.

²For our attacks, we use the ADAM optimizer with the following parameters: $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, $\eta \in [10^{-4}, 10^0]$

3.4.1 Dataset and Classifiers

We built two Convolutional Neural Network (CNN) classifiers based on a standard crop-resize-then-classify pipeline for road sign classification as described in [73, 63]. Our LISA-CNN uses LISA, a U.S. traffic sign dataset containing 47 different road signs [55]. However, the dataset is not well-balanced, resulting in large disparities in representation for different signs. To alleviate this problem, we chose the 17 most common signs based on the number of training examples.

LISA-CNN’s architecture is defined in the Cleverhans library [61] and consists of three convolutional layers and an FC layer. It has an accuracy of 91% on the test set.

Our second classifier is GTSRB-CNN, that is trained on the German Traffic Sign Recognition Benchmark (GTSRB) [28, 79]. We use a publicly available implementation [91] of a multi-scale CNN architecture that has been known to perform well on road sign recognition [73]. Further details about the model architecture can be found in Appendix A.1. Because we did not have access to German Stop signs for our physical experiments, we replaced the German Stop signs in the training, validation, and test sets of GTSRB with the U.S. Stop sign images in LISA. GTSRB-CNN achieves 95.7% accuracy on the test set. When evaluating GTSRB-CNN on our own 181 stop sign images, it achieves 99.4% accuracy.

3.4.2 Experimental Design

To the best of our knowledge, there is currently no standardized methodology of evaluating physical adversarial perturbations. Based on our discussion from Section 3.3.1, we focus on angles and distances because they are the most rapidly changing elements for our use case. A camera in a vehicle approaching a sign will take a series of images at regular intervals. These images will be taken at different angles and distances, therefore changing the amount of detail present in any given image. Any successful physical perturbation must cause targeted misclassification in a range of

distances and angles because a vehicle will likely perform voting on a set of frames (images) from a video before issuing a controller action. Our current experiments do not explicitly control ambient light, and as is evident from experimental data (Section 3.4), lighting varied from indoor lighting to outdoor lighting.

Drawing on standard practice in the physical sciences, our experimental design encapsulates the above physical factors into a two-stage evaluation consisting of controlled lab tests and field tests.

Stationary (Lab) Tests. This involves classifying images of objects from stationary, fixed positions.

1. Obtain a set of clean images C and a set of adversarially perturbed images ($\{\mathcal{A}(c)\}, \forall c \in C$) at varying distances $d \in D$, and varying angles $g \in G$. We use $c^{d,g}$ here to denote the image taken from distance d and angle g . The camera’s vertical elevation should be kept approximately constant. Changes in the camera angle relative the the sign will normally occur when the car is turning, changing lanes, or following a curved road.

2. Compute the attack success rate of the physical perturbation using the following formula:

$$\frac{\sum_{c \in C} \mathbb{1}_{\{f_{\theta}(\mathcal{A}(c^{d,g}))=y^* \wedge f_{\theta}(c^{d,g})=y\}}}{\sum_{c \in C} \mathbb{1}_{\{f_{\theta}(c^{d,g})=y\}}} \quad (3.3)$$

where d and g denote the camera distance and angle for the image, y is the ground truth, and y^* is the targeted attacking class.³

Note that an image $\mathcal{A}(c)$ that causes misclassification is considered as a successful attack only if the original image c with the same camera distance and angle is correctly classified, which ensures that the misclassification is caused by the added perturbation instead of other factors.

³For untargeted adversarial perturbations, change $f_{\theta}(e^{d,g}) = y^*$ to $f_{\theta}(e^{d,g}) \neq y$.

Drive-By (Field) Tests. We place a camera on a moving platform, and obtain data at realistic driving speeds. For our experiments, we use a smartphone camera mounted on a car.






















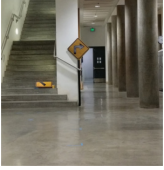

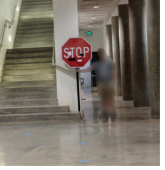
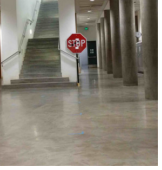
1. Begin recording video at approximately 250 ft away from the sign. Our driving track was straight without curves. Drive toward the sign at normal driving speeds and stop recording once the vehicle passes the sign. In our experiments, our speed varied between 0 mph and 20 mph. This simulates a human driver approaching a sign in a large city.
2. Perform video recording as above for a “clean” sign and for a sign with perturbations applied, and then apply similar formula as Eq. 3.3 to calculate the attack success rate, where C here represents the sampled frames.

An autonomous vehicle will likely not run classification on every frame due to performance constraints, but rather, would classify every j -th frame, and then perform simple majority voting. Hence, an open question is to determine whether the choice of frame (j) affects attack accuracy. In our experiments, we use $j = 10$. We also tried $j = 15$ and did not observe any significant change in the attack success rates. If both types of tests produce high success rates, the attack is likely to be successful in commonly experienced physical conditions for cars.

3.4.3 Results for LISA-CNN

We evaluate the effectiveness of our algorithm by generating three types of adversarial examples on LISA-CNN (91% accuracy on test-set). For all types, we observe high attack success rates with high confidence. Table 3.1 summarizes a sampling of stationary attack images. In all testing conditions, our baseline of unperturbed road signs achieves a 100% classification rate into the true class.

Table 3.1: Sample of physical adversarial examples against LISA-CNN and GTSRB-CNN.

Distance/Angle	Subtle Poster	Subtle Poster Right Turn	Camouflage Graffiti	Camouflage Art (LISA-CNN)	Camouflage Art (GTSRB-CNN)
5' 0°					
5' 15°					
10' 0°					
10' 30°					
40' 0°					
Targeted-Attack Success	100%	73.33%	66.67%	100%	80%

Object-Constrained Poster-Printing Attacks. This involves reproducing the attack of Kurakin *et al.* [37]. The crucial difference is that in our attack, the perturbations are confined to the surface area of the sign excluding the background, and are robust against large angle and distance variations. The Stop sign is misclassified into the attack’s target class of Speed Limit 45 in 100% of the images taken according to our evaluation methodology. The average confidence of predicting the manipulated sign as the target class is 80.51% (second column of Table 3.2).

For the Right Turn warning sign, we choose a mask that covers only the arrow since we intend to generate subtle perturbations. In order to achieve this goal, we increase

Table 3.2: Targeted physical perturbation experiment results on LISA-CNN using a poster-printed Stop sign (subtle attacks) and a real Stop sign (camouflage graffiti attacks, camouflage art attacks). For each image, the top two labels and their associated confidence values are shown. The misclassification target was Speed Limit 45. See Table 3.1 for example images of each attack. Legend: SL45 = Speed Limit 45, STP = Stop, YLD = Yield, ADL = Added Lane, SA = Signal Ahead, LE = Lane Ends.

Distance & Angle	Poster-Printing			Sticker		
	Subtle		Camouflage-Graffiti	Camouflage-Art		
5' 0°	SL45 (0.86)	ADL (0.03)	STP (0.40)	SL45 (0.27)	SL45 (0.64)	LE (0.11)
5' 15°	SL45 (0.86)	ADL (0.02)	STP (0.40)	YLD (0.26)	SL45 (0.39)	STP (0.30)
5' 30°	SL45 (0.57)	STP (0.18)	SL45 (0.25)	SA (0.18)	SL45 (0.43)	STP (0.29)
5' 45°	SL45 (0.80)	STP (0.09)	YLD (0.21)	STP (0.20)	SL45 (0.37)	STP (0.31)
5' 60°	SL45 (0.61)	STP (0.19)	STP (0.39)	YLD (0.19)	SL45 (0.53)	STP (0.16)
10' 0°	SL45 (0.86)	ADL (0.02)	SL45 (0.48)	STP (0.23)	SL45 (0.77)	LE (0.04)
10' 15°	SL45 (0.90)	STP (0.02)	SL45 (0.58)	STP (0.21)	SL45 (0.71)	STP (0.08)
10' 30°	SL45 (0.93)	STP (0.01)	STP (0.34)	SL45 (0.26)	SL45 (0.47)	STP (0.30)
15' 0°	SL45 (0.81)	LE (0.05)	SL45 (0.54)	STP (0.22)	SL45 (0.79)	STP (0.05)
15' 15°	SL45 (0.92)	ADL (0.01)	SL45 (0.67)	STP (0.15)	SL45 (0.79)	STP (0.06)
20' 0°	SL45 (0.83)	ADL (0.03)	SL45 (0.62)	STP (0.18)	SL45 (0.68)	STP (0.12)
20' 15°	SL45 (0.88)	STP (0.02)	SL45 (0.70)	STP (0.08)	SL45 (0.67)	STP (0.11)
25' 0°	SL45 (0.76)	STP (0.04)	SL45 (0.58)	STP (0.17)	SL45 (0.67)	STP (0.08)
30' 0°	SL45 (0.71)	STP (0.07)	SL45 (0.60)	STP (0.19)	SL45 (0.76)	STP (0.10)
40' 0°	SL45 (0.78)	LE (0.04)	SL45 (0.54)	STP (0.21)	SL45 (0.68)	STP (0.14)



the regularization parameter λ in equation (3.2) to demonstrate small magnitude perturbations. We achieve a 73.33% targeted-attack success rate (Table 3.1). Out of 15 distance/angle configurations, four instances were not classified into the target. However, they were still misclassified into other classes that were not the true label (Yield, Added Lane). Three of these four instances were an Added Lane sign—a different type of warning. We hypothesize that given the similar appearance of warning signs, small perturbations are sufficient to confuse the classifier.

Sticker Attacks. Next, we demonstrate how effective it is to generate physical perturbations in the form of stickers, by constraining the modifications to a region resembling graffiti or art. The fourth and fifth columns of Table 3.1 show a sample of images, and Table 3.2 (columns 4 and 6) shows detailed success rates with confidences. In the stationary setting, we achieve a 66.67% targeted-attack success rate for the

graffiti sticker attack and a 100% targeted-attack success rate for the sticker camouflage art attack. Some region mismatches may lead to the lower performance of the LOVE-HATE graffiti.

Drive-By Testing. Per our evaluation methodology, we conduct drive-by testing for the perturbation of a Stop sign. In our baseline test we record two consecutive videos of a clean Stop sign from a moving vehicle, perform frame grabs at $k = 10$, and crop the sign. We observe that the Stop sign is correctly classified in all frames. We similarly test subtle and abstract art perturbations for LISA-CNN using $k = 10$. Our attack achieves a targeted-attack success rate of 100% for the subtle poster attack, and a targeted-attack success rate of 84.8% for the camouflage abstract art attack. See Table 3.3 for sample frames from the drive-by video.

Table 3.3: Drive-by testing summary for LISA-CNN. In our baseline test, all frames were correctly classified as a Stop sign. We have manually added the yellow boxes as a visual guide.

Perturbation	Attack Success	A Subset of Sampled Frames $k = 10$
Subtle poster	100%	
Camouflage abstract art	84.8%	

3.4.4 Results for GTSRB-CNN

To show the versatility of our attack algorithms, we create and test attacks for GTSRB-CNN (95.7% accuracy on test-set). Based on our high success rates with the camouflage-art attacks, we create similar abstract art sticker perturbations. The last column of Table 3.1 shows a subset of experimental images. Table 3.4 summarizes our attack results—our attack fools the classifier into believing that a Stop sign is a Speed Limit 80 sign in 80% of the stationary testing conditions. Per our evaluation

Table 3.4: A camouflage art attack on GTSRB-CNN. See example images in Table 3.1. The targeted-attack success rate is 80% (true class label: Stop, target: Speed Limit 80).

Distance & Angle	Top Class (Confid.)	Second Class (Confid.)
5' 0°	Speed Limit 80 (0.88)	Speed Limit 70 (0.07)
5' 15°	Speed Limit 80 (0.94)	Stop (0.03)
5' 30°	Speed Limit 80 (0.86)	Keep Right (0.03)
5' 45°	Keep Right (0.82)	Speed Limit 80 (0.12)
5' 60°	Speed Limit 80 (0.55)	Stop (0.31)
10' 0°	Speed Limit 80 (0.98)	Speed Limit 100 (0.006)
10' 15°	Stop (0.75)	Speed Limit 80 (0.20)
10' 30°	Speed Limit 80 (0.77)	Speed Limit 100 (0.11)
15' 0°	Speed Limit 80 (0.98)	Speed Limit 100 (0.01)
15' 15°	Stop (0.90)	Speed Limit 80 (0.06)
20' 0°	Speed Limit 80 (0.95)	Speed Limit 100 (0.03)
20' 15°	Speed Limit 80 (0.97)	Speed Limit 100 (0.01)
25' 0°	Speed Limit 80 (0.99)	Speed Limit 70 (0.0008)
30' 0°	Speed Limit 80 (0.99)	Speed Limit 100 (0.002)
40' 0°	Speed Limit 80 (0.99)	Speed Limit 100 (0.002)

methodology, we also conduct a drive-by test ($k = 10$, two consecutive video recordings).

The attack fools the classifier 87.5% of the time.

3.4.5 Results for Inception-v3

To demonstrate generality of the RPP attack, we computed physical perturbations for the standard Inception-v3 classifier [80, 36] using two different objects, a microwave and a coffee mug. Due to a different task domain, we chose a sticker attack since poster printing an entirely new surface for the objects may raise suspicions upon casual inspection of the objects. Furthermore, we have reduced the range of distances used for evaluation due to the smaller size of the cup and microwave compared to a road sign (*e.g.*, Coffee Mug height- 11.2cm, Microwave height- 24cm, Right Turn sign

Table 3.5: Sticker perturbation attack on the Inception-v3 classifier. The original classification is microwave and the attacker’s target is phone. See example images in Appendix B Table B.1. Our targeted-attack success rate is 90%

Distance & Angle	Top Class (Confid.)	Second Class (Confid.)
2' 0°	Phone (0.78)	Microwave (0.03)
2' 15°	Phone (0.60)	Microwave (0.11)
5' 0°	Phone (0.71)	Microwave (0.07)
5' 15°	Phone (0.53)	Microwave (0.25)
7' 0°	Phone (0.47)	Microwave (0.26)
7' 15°	Phone (0.59)	Microwave (0.18)
10' 0°	Phone (0.70)	Microwave (0.09)
10' 15°	Phone (0.43)	Microwave (0.28)
15' 0°	Microwave (0.36)	Phone (0.20)
20' 0°	Phone (0.31)	Microwave (0.10)

height- 45cm, Stop Sign- 76cm). For the microwave, our adversarial sticker causes the classifier to misclassify it as our target class, “phone,” in 90% of the tests. For the coffee mug, our adversarial sticker causes the classifier to misclassify it as our target class, “cash machine”, in 71.4% of the tests. These results are shown in Tables 3.5 and 3.6 respectively. Images of the microwave and the mug can be found in Appendix B.

3.5 Discussion

Black-Box Attacks. In developing the RPP attack, we design the attack for a white-box setting for two reasons. First, to develop a foundation for future defenses, we must assess the abilities of powerful adversaries, and this can be done in a white-box setting. Through studying a white-box attack like RPP, we can analyze the requirements for a successful attack using the strongest attacker model and better inform future defenses. Second, in our chosen autonomous vehicle domain, the machine learning system is obtainable by the attacker, and thus is able to act as an oracle. Based on this, we expect that the RPP attack can be further augmented with the black-box attack

Table 3.6: Sticker perturbation attack on the Inception-v3 classifier. The original classification is coffee mug and the attacker’s target is cash machine. See example images in Appendix B Table B.2. Our targeted-attack success rate is 71.4%.

Distance & Angle	Top Class (Confid.)	Second Class (Confid.)
8” 0°	Cash Machine (0.53)	Pitcher (0.33)
8” 15°	Cash Machine (0.94)	Vase (0.04)
12” 0°	Cash Machine (0.66)	Pitcher (0.25)
12” 15°	Cash Machine (0.99)	Vase (0.01)
16” 0°	Cash Machine (0.62)	Pitcher (0.28)
16” 15°	Cash Machine (0.94)	Vase (0.01)
20” 0°	Cash Machine (0.84)	Pitcher (0.09)
20” 15°	Cash Machine (0.42)	Pitcher (0.38)
24” 0°	Cash Machine (0.70)	Pitcher (0.20)
24” 15°	Pitcher (0.38)	Water Jug (0.18)
28” 0°	Pitcher (0.59)	Cash Machine (0.09)
28” 15°	Cash Machine (0.23)	Pitcher (0.20)
32” 0°	Pitcher (0.50)	Cash Machine (0.15)
32” 15°	Pitcher (0.27)	Mug (0.14)

techniques discussed in Chapter II to generate successful physical adversarial attacks.

Image Cropping and Attacking Detectors. When evaluating the RPP attack, we manually controlled the cropping of each image every time before classification. This was done so the adversarial images would match the clean sign images provided to the attack. Later, we evaluated the camouflage art attack using a pseudo-random crop with the guarantee that at least most of the sign was in the image. Against LISA-CNN, we observed an average targeted attack rate of 70% and untargeted attack rate of 90%. Against GTSRB-CNN, we observed an average targeted attack rate of 60% and untargeted attack rate of 100%. We include the untargeted attack success rates because causing the classifier to not output the correct traffic sign label is still a safety risk. Although image cropping has some effect on the targeted attack success rate, Chapter IV demonstrates that an improved version of the RPP attack can successfully attack object detectors, where cropping is not needed.

3.6 Conclusion

We introduced the RPP attack, which generates robust, physically realizable adversarial perturbations. Using our attack, and a two-stage experimental design consisting of lab and drive-by tests, we contribute to understanding the space of physical adversarial examples when the *objects themselves* are physically perturbed. We target road-sign classification because of its importance in safety, and the naturally noisy environment of road signs. Our work shows that it is possible to generate physical adversarial examples robust to widely varying distances/angles. This implies that future defenses should not rely on physical sources of noise as protection against physical adversarial examples.

CHAPTER IV

Physical Adversarial Examples for Object Detectors

In this chapter, we improve upon the attack algorithm introduced in Chapter III by extending the RPP attack to object detection frameworks. We evaluate the attack against a well-known object detection framework and, furthermore, demonstrate that our attack is transferable, which is an important property necessary to perform black-box evaluations.

4.1 Introduction

Deep neural networks (DNNs) are widely applied in computer vision, natural language, and robotics, especially in safety-critical tasks such as autonomous driving [44]. At the same time, DNNs have been shown to be vulnerable to *adversarial examples* [26, 64, 12, 70, 35], maliciously perturbed inputs that cause DNNs to produce incorrect predictions. These attacks pose a risk to the use of deep learning in safety- and security-critical decisions. For example, an attacker can add perturbations, which are negligible to humans, to a Stop sign and cause a DNN embedded in an autonomous vehicle to misclassify or ignore the sign.

Early works studied adversarial examples in the digital space only. However, it has

recently been shown that it is also possible to create perturbations that survive under various physical conditions (*e.g.*, object distance, pose, lighting, etc.) [75, 37, 4, 9]. These works focus on attacking *classification networks*, *i.e.*, models that produce a single prediction on a static input image. In this chapter, we explore physical adversarial examples for *object detection networks*, a richer class of deep learning algorithms that can detect and label multiple objects in a scene. Object detection networks are a popular tool for tasks that require real-time and dynamic recognition of surrounding objects, autonomous driving being a canonical application. Object detectors are known to be vulnerable to digital attacks [89], but their vulnerability to physical attacks remains an open question.

Compared to classifiers, object detection networks are more challenging to attack: 1) Detectors process an entire scene instead of a single localized object. This allows detectors to use contextual information (*e.g.*, the orientation and relative position of objects in the scene) to generate predictions. 2) Detectors are not limited to producing a single prediction. Instead, they label every recognized object in a scene, usually by combining predictions of the *location* of objects in a scene, and of the labeling of these objects. Attacks on object detectors need to take both types of predictions (presence/absence of an object and nature of the object) into account, whereas attacks on classifiers only focus on modifying the label of a single (presumably present) object.

To create proof-of-concept attacks for object detectors, we start from the existing RPP attack we designed to produce robust physical perturbations for image classifiers. In the original algorithm, we sampled from a distribution that mimics physical perturbations of an object (*e.g.*, view distance and angle), and find a perturbation that maximizes the probability of mis-classification under this distribution. We find that the physical perturbations originally considered for attacking classifiers are insufficient to also work for object detectors.

Indeed, when working with image classifiers, prior works considered target objects

that make up a large portion of the image and whose relative position in the image varies little. Yet, when performing object detection in a dynamic environment such as a moving vehicle, the relative size and position of the multiple objects in a scene can change drastically. These changes produce additional constraints that have to be taken into account to produce successful robust physical attacks. Many object detectors, for instance, split a scene into a grid or use a sliding window to identify regions of interest, and produce separate object predictions for each region of interest. As the relative position of an object changes, the grid cells the object is contained in (and the corresponding network weights) change as well. Robust perturbations, thus, have to be adversarial for multiple grid cells simultaneously. We show that robustness to these new factors can be attained by extending the distribution of inputs considered in Chapter III to include these additional object transformations within the scene (*e.g.*, changes in perspective, size, and position).

Again, we consider physical adversarial attacks on the localization and classification of Stop signs, an illustrative example for the safety implications of a successful attack. The perturbations, while large enough to be visible to the human eye, are constrained to resemble human-made graffiti or subtle lighting artifacts that could be considered benign. We consider an untargeted attack specific to object detectors, which we refer to as a *Disappearance Attack*. In a Disappearance Attack, we create either an adversarial poster or physical stickers applied to a real Stop sign (see Figure 4.2), which causes the sign to be ignored by an object detector in different scenes with varying object distance, location, and perspective. This attack is analogous attacking image classifiers, but targets a richer class of deep neural networks.

We further introduce a new *Creation Attack*, wherein physical stickers that humans would ignore as being inconspicuous can cause an object detector into recognizing nonexistent Stop signs. This attack differs from prior attacks that attempt to fool a network into misclassifying one object into another, in that it creates an entirely

new object classification. Specifically, we experiment with creating adversarial stickers (similar to the ones considered in [9]). Such stickers could for instance be used to mount *Denial of Service* attacks on road sign detectors.

For our experiments, we target the state-of-the-art YOLO v2 (You Only Look Once) object detector [68]. YOLO v2 is a deep convolutional neural network that performs real-time object detection for 80 object classes. Our indoor (laboratory) and outdoor experiments show that up to distances of 30 feet from the target object, detectors can be tricked into *not* perceiving the attacker’s target object using poster and sticker perturbations.

Our Contributions:

- We improve the RPP attack to create one of the first proof-of-concept attack algorithms for object detection networks, a richer class of DNNs than image classifiers.
- Using our attack algorithm, we propose a new physical attack on object detection networks: the Disappearance Attack that causes physical objects to be ignored by a detector.
- We evaluate our attacks on the YOLO v2 object detector in an indoor laboratory setting and an outdoor setting. Our results show that our adversarial poster perturbation fools YOLO v2 in 85.6% of the video frames recorded in an indoor lab environment and in 72.5% of the video frames recorded in an outdoor environment. Our adversarial stickers fool YOLO v2 in 85% of the video frames recorded in a laboratory environment and in 63.5% of the video frames recorded in an outdoor environment.
- We evaluate the transferability of our attack using the Faster R-CNN object detector in laboratory and outdoor environments. Our results show that our

attack fools Faster R-CNN in 85.9% of the video frames recorded in a laboratory environment and in 40.2% of the video frames recorded in an outdoor environment.

- We propose and experiment with a new *Creation* attack, that aims at fooling a detector into recognizing adversarial stickers as non-existing objects. Our results with this attack type are preliminary, yet encouraging.

Our work demonstrates that physical perturbations are effective against object detectors, and leaves open some future questions: 1) Generalization to other physical settings (*e.g.*, moving vehicles, or even real autonomous vehicles). 2) Further exploration of other classes of attacks: Our work introduces the disappearance and creation attacks which use posters or stickers, yet there are other plausible attack types (*e.g.*, manufacturing physical objects that are not recognizable to humans, but are recognized by DNNs). 3) Physical attacks on segmentation networks. We envision that future work will build on the findings presented here, and will create attacks that generalize across physical settings (*e.g.*, real autonomous vehicles), and across classes of object detection networks (*e.g.*, semantic segmentation [89]).

4.2 Related Work

We survey the related work on adversarial attacks on object detection and image segmentation frameworks. For related work on digital adversarial attacks on classifiers, please refer to Chapter II.

Xie *et al.* proposed the Dense Adversary Generation (DAG) algorithm, which generates adversarial examples for object detection and segmentation frameworks [89]. Given a set of targets $\{t_1, t_2, \dots, t_n\} \in T$, DAG seeks to optimize the adversarial

example x' such that the label assigned to each target is incorrect. In other words,

$$\forall n, \max f(x', t_n) \neq l_n \tag{4.1}$$

where l_n is the correct label for t_n . For segmentation, T is all of the pixels in the image. For object detection, they increase the number of region proposals and target the most probable bounding boxes.

Metzen *et al.* introduced the concept of universal attacks for an image segmentation network in which the attack produces a single image [53]. This image, when added to an input, either causes the network to output the same, incorrect, segmentation result or causes the network to ignore certain objects in the image. Finally, Cisse *et al.* proposed Houdini, which enables targeted attacks against image segmentation networks [19].

In an effort to study the threat of physical adversarial, Lu *et al.* performed experiments using adversarial road signs printed on paper with the YOLO object detector [50]. Their results suggested it is very challenging to fool YOLO with physical adversarial examples generated using existing digital attack algorithms. In our evaluation, however, our proposed attacks can generate physical adversarial examples that reliably fool YOLO.

Concurrent to our work attacking YOLO, Chen *et al.* attacked the Faster R-CNN object detector [16]. Their attack fixes the region proposals and optimizes over a set of random transformations of the target object. They generate adversarial poster perturbations that completely replace the road sign to fool Faster R-CNN and cause it to mislabel the object. Our work differs in that we introduce two different attacks on object detectors, the disappearance and creation attacks, which use either adversarial poster or sticker perturbations. We also demonstrate limited black-box transferability from YOLO to the Faster-RCNN detector.

4.3 Background on Object Detectors

Object classification is a standard task in computer vision. Given an input image and a set of class labels, the classification algorithm outputs the most probable label (or a probability distribution over all labels) for the image. Object classifiers are limited to categorizing a single object per image. If an image contains multiple objects, the classifier only outputs the class of the most dominant object in the scene. In contrast, object detectors both locate and classify multiple objects in a given scene.

The first proposed deep neural network for object detection was Overfeat [72], which combined a sliding window algorithm and convolution neural networks. A more recent proposal, Regions with Convolutional Neural Networks (R-CNN) uses a search algorithm to generate region proposals, and a CNN to label each region. A downside of R-CNN is that the region proposal algorithm is too slow to be run in real-time. Subsequent works—Fast R-CNN [24] and Faster R-CNN [69]—replace this inefficient algorithm with a more efficient CNN.

The above algorithms treat object detection as a two-stage problem consisting of region proposals followed by classifications for each of these regions. In contrast, so-called “single shot detectors” such as YOLO [67] (and the subsequent YOLO v2 [68]) or SSD [48] run a single CNN over the input image to jointly produce confidence scores for object localization and classification. As a result, these networks can achieve the same accuracy while processing images much faster. In this work, we focus on YOLO v2, a state-of-the-art object detector with real-time detection capabilities and high accuracy.

The classification approach of YOLO v2 is illustrated in Figure 4.1. A single CNN is run on the full input image and predicts object location (bounding boxes) and label confidences for 361 separate grid cells (organized into a 19×19 square over the original image). For each cell, YOLO v2 makes a prediction for 5 different boxes. For each box, the prediction contains the box confidence (the probability that this box

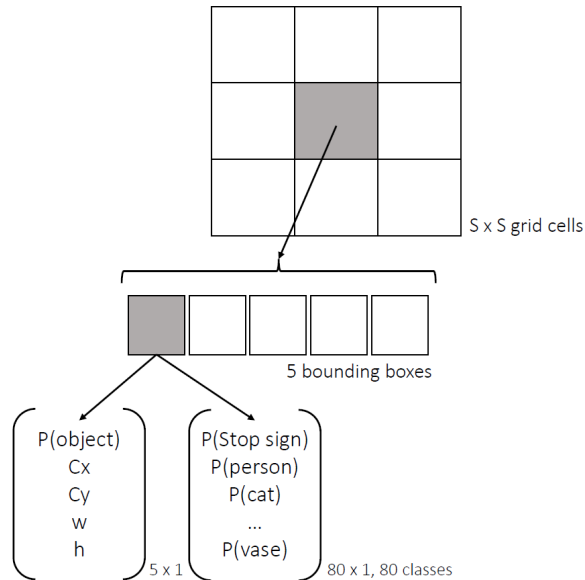


Figure 4.1: For an input scene, the YOLO v2 CNN outputs a $19 \times 19 \times 425$ tensor. To generate this tensor, YOLO divides the input image into a square grid of S^2 cells ($S = 19$). For each grid cell, there are B bounding boxes ($B = 5$). Each bounding box predicts 5 values: probability of an object in the cell, co-ordinates of the bounding box (center x, center y, width, height). Additionally, for each bounding box the model predicts a probability distribution over all 80 output classes.

contains an object), its location and the probability of each class label for that box. A box is discarded if the product of the box confidence and the probability of the most likely class is below some threshold (this threshold is set to 0.1 in our experiments). Finally, the *non-max suppression* algorithm is applied in a post-processing phase to discard redundant boxes with high overlap [68].

Such an object detection pipeline introduces several new challenges regarding physical adversarial examples: First, unlike classification where an object is always assumed present and the attack only needs to modify the class probabilities, attacks on a detector network need to control a combination of box confidences and class probabilities for all boxes in all grid cells of the input scene. Second, classifiers assume the object of interest is centered in the input image, whereas detectors can find objects at arbitrary positions in a scene. Finally, the object’s size in the detector’s input is not fixed. In classification, the image is usually cropped and resized to focus on

the object being classified. Object detectors are meant to reliably detect objects at multiple scales, distances and angles in a scene.

These challenges mainly stem from object detectors being much more flexible and broadly applicable than standard image classifiers. Thus, albeit harder to attack, object detectors also represent a far more interesting attack target than image classifiers, as their extra flexibility makes them a far better candidate for use in reliable cyber-physical systems.

4.4 Physical Adversarial Examples for Object Detectors

We will first review the RPP attack presented in Chapter III, before discussing the modifications necessary to adapt the algorithm to attack object detectors.

4.4.1 The RPP Attack

The RPP attack optimizes the following objective function:

$$\operatorname{argmin}_{\delta} \lambda \|M_x \cdot \delta\|_p + NPS(M_x \cdot \delta) + \mathbb{E}_{x_i \sim X^V} J(f_{\theta}(x_i + T_i(M_x \cdot \delta)), y^*) \quad (4.2)$$

The first term of the objective function is the ℓ_p norm (with scaling factor λ) of the perturbation δ masked by M_x . The mask is responsible for spatially constraining the perturbation δ to the surface of the target object. For example, in Figure 4.2, the mask shape is two horizontal bars on the sign.

The second term of the objective function measures the printability of an adversarial perturbation. This term is borrowed from prior work by Sharif *et al.* [75]. The printability of a perturbation is affected by two factors. First, the colors the computed perturbation must reproduce. Modern printers have a limited color gamut, thus certain colors that appear digitally may not be printable. Second, a printer may not



Figure 4.2: An example of an adversarial perturbation overlaid on a synthetic background. The Stop sign in the image is printed such that it is the same size as a U.S. Stop sign. Then, we cut out the two rectangle bars, and use the original print as a stencil to position the cutouts on a real Stop sign.

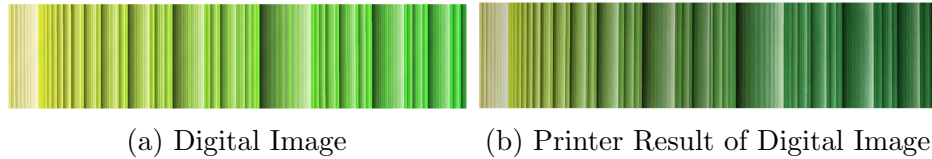


Figure 4.3: The image in (a) shows the image as it is stored digitally. The result of printing and taking a picture of the image in (a) is shown in (b).

faithfully reproduce a color as it is shown digitally (see Figure 4.3).

The last term of the objective function is the value of the loss function, $J(\cdot, \cdot)$ averaged across all of the images sampled from X^V . In practice, this is a set of victim images, which is composed of multiple images of the object taken under a variety of physical conditions such as changes in viewing angle, viewing distance and lighting. T_i is an “alignment function” that applies a digital transformation that mimics the physical conditions of victim object x_i . For example, if the victim object x_i is a rotated version of the “canonical” target object, then the perturbation $M_x \cdot \delta$ should also be rotated appropriately. Thus, in Chapter III, to simulate physical consistency of the perturbed object, we apply the alignment function T_i to the masked perturbation. $f_\theta(\cdot)$ is the output of the classifier network, and y^* is the adversarial target class.

4.4.2 Extensions to the RPP attack for Object Detectors

The modified version of the RPP attack contains three key differences from the original attack algorithm proposed. First, due to differences in the output behavior of classifiers and object detectors, we make modifications to the adversarial loss function. Second, we observed additional constraints that an adversarial perturbation must be robust to and model these constraints synthetically. Finally, we introduce a smoothness constraint into the objective, rather than using the ℓ_p norm. In the following, we discuss each of these changes in detail.

4.4.2.1 Modified Adversarial Loss Function

An object detector outputs a set of bounding boxes and the likelihood of the most probable object contained within that box given a certain confidence threshold. See Figure 4.1 for a visualization of this output. By contrast, a classifier outputs a single vector where each entry represents the probability that the object in the image is of that type. Attacks on image classifiers typically make use of the cross-entropy loss between this output vector, and a one-hot representation of the adversarial target. However, this loss function is not applicable to object detectors due to their richer output structure. Thus, we introduce a new adversarial loss function suitable for use with detectors. This loss function is tailored to the specific attacks we introduce in this work.

Disappearance Attack Loss. The goal of the attacker is to prevent the object detector from detecting the target object. To achieve this, the adversarial perturbation must ensure that the likelihood of the target object in any bounding box is less than the detection threshold (the default is 25% for YOLO v2). In our implementation of the attack, we used the following loss function:

$$J_d(x, y) = \max_{s \in S^2, b \in B} P(s, b, y, f_\theta(x)) \quad (4.3)$$

Where $f_\theta(x)$ represents the output of the object detector (for YOLO v2, this is a $19 \times 19 \times 425$ tensor). $P(\cdot)$ is a function that extracts the probability of an object class from this tensor, with label y (in our case, this is a Stop sign) in grid cell s and bounding box b . We denote x as the input scene containing our perturbed target object.

Therefore, the loss function outputs the maximum probability of a Stop sign if it occurs within the scene. Using this loss function, the goal of the adversary is to directly minimize that probability until it falls below the detection threshold of the network.

Creation Attack Loss. We propose a new type of *Creation Attack*, wherein the goal is to fool the model into recognizing nonexistent objects. Similar to the “adversarial patch” approach of [9], our goal is to create a physical sticker that can be added to any existing scene. Contrary to prior work, rather than causing a mis-classification our aim is to create a new classification (*i.e.*, a new object detection) where non existed before.

For this, we use a composite loss function, that first aims at creating a new object localization, followed by a targeted “mis-classification.” The mask M_x is sampled randomly so that the adversarial patch is applied to an arbitrary location in the scene. As above, let $f_\theta(x)$ represent the full output tensor of YOLO v2 on input scene x , and let $P(s, b, y, f_\theta(x))$ represent the probability assigned to class y in box b of grid cell s . Further let $P_{\text{box}}(s, b, f_\theta(x))$ represent the probability of the box only, *i.e.*, the model’s

confidence that the box contains *any* object. Our loss is then

$$\begin{aligned} \text{object} &= P_{\text{box}}(s, b, f_{\theta}(x)) > \tau \\ J_c(x, y) &= \text{object} + (1 - \text{object}) \cdot P(s, b, y, f_{\theta}(x)) \end{aligned} \tag{4.4}$$

Here, τ is a threshold on the box confidence (set to 0.2 in our experiments), after which we stop optimizing the box confidence and focus on increasing the probability of the targeted class. As our YOLO v2 implementation uses a threshold of 0.1 on the product of the box confidence and class probability, any box with a confidence above 0.2 and a target class probability above 50% is retained.

4.4.2.2 Synthetic Representation of New Physical Constraints

Generating physical adversarial examples for detectors requires simulating a larger set of varying physical conditions than what is needed to trick classifiers. In our initial experiments, we observed that the generated perturbations would fail if the object was moved from its original position in the image. This is likely because a detector has access to more contextual information when generating predictions. As an object’s position and size can vary greatly depending on the viewer’s location, perturbations must account for these additional constraints.

To generate physical adversarial perturbations that are positionally invariant, we chose to synthetically model two environmental conditions: object rotation (in the Z plane) and position (in the X-Y plane). In each epoch of the optimization, we randomly place and rotate the object. Our approach differs from the original approach for attack image classifiers, in we previously modeled an object’s rotation physically using a diverse dataset. We avoided this approach because of the added complexity necessary for the alignment function, T_i , to properly position the adversarial perturbation on the sign. Since these transformations are done synthetically, the alignment function,

T_i , simply needs to use the same process to transform the adversarial perturbation.

4.4.2.3 Noise Smoothing using Total Variation

The unmodified RPP attack uses the ℓ_p norm to smooth the perturbation. However, in our initial experiments, we observed that the ℓ_p norm results in very pixelated perturbations. The pixelation hurts the success rate of the attack, especially as the distance between the viewer and the object increases. We found that using the total variation norm in place of the ℓ_p norm gave smoother perturbations, thus increasing the effective range of the attack. Given a mask, M_x , and noise δ , the total variation norm of the adversarial perturbation, $M_x \cdot \delta$, is:

$$TV(M_x \cdot \delta) = \sum_{i,j} |(M_x \cdot \delta)_{i+1,j} - (M_x \cdot \delta)_{i,j}| + |(M_x \cdot \delta)_{i,j+1} - (M_x \cdot \delta)_{i,j}| \quad (4.5)$$

where i, j are the row and column indices for the adversarial perturbation. Thus our final modified objective function is:

$$\operatorname{argmin}_{\delta} \lambda TV(M_x \cdot \delta) + NPS + \mathbb{E}_{x_i \sim X^V} J_d(x_i + T_i(M_x \cdot \delta), y^*) \quad (4.6)$$

where $J_d(\cdot, y^*)$ is the loss function (discussed earlier) that measures the maximum probability of an object with the label y^* contained in the image. In our attack, y^* is a Stop sign.

4.5 Evaluation

We first discuss our experimental method, where we evaluate attacks in a whitebox manner using YOLO v2, and in a blackbox manner using Faster-RCNN. Then, we discuss our results, showing that state-of-the-art object detectors can be attacked



Figure 4.4: Output of the extended RPP algorithm to attack YOLO v2 using poster and sticker attacks.



Figure 4.5: Patch created by the Creation Attack, aimed at fooling YOLO v2 into detecting nonexistent Stop signs.

using physical posters and stickers. Figure 4.4 shows the digital versions of posters and stickers used for disappearance attacks, while Figure 4.5 shows a digital version of the sticker used in a creation attack.

4.5.1 Experimental Setup

We evaluated our disappearance attack in a mix of lab and outdoor settings. For both the poster and sticker attacks, we generated adversarial perturbations and recorded several seconds of video. In each experiment, recording began 30 feet from the sign and ended when no part of the sign was in the camera’s field of view. Then,

YOLO v2	Poster	Sticker
Indoors	202/236 (85.6%)	210/247 (85.0%)
Outdoors	156/215 (72.5%)	146/230 (63.5%)

Table 4.1: Attack success rate for the disappearance attack on YOLO v2. We tested a poster perturbation, where a true-sized print is overlaid on a real Stop sign, and a sticker attack, where the perturbation is two rectangles stuck to the surface of the sign. The table cells show the ratio: number of frames in which a Stop sign was *not* detected / total number of frames, and a success rate, which is the result of this ratio.

we fed the video into the object detection network for analysis. We used the YOLO v2 object detector as a white-box attack. We also ran the same videos through the Faster-RCNN network to measure black-box transferability of our attack.

For the creation attack, we experimented with placing stickers on large flat objects (*e.g.*, a wall or cupboard), and recording videos within 10 feet of the sticker.

4.5.2 Experimental Results

We evaluated the perturbations for a disappearance attack using two different masks and attacked a Stop sign. First, we tested a poster perturbation, which used an octagonal mask to allow adversarial noise to be added anywhere on the surface of the Stop sign. Next, we tested a sticker perturbation. We used the mask to create two rectangular stickers positioned at the top and bottom of the sign. The results of our attack are shown in Table 4.1.

In indoor lab settings, where the environment is relatively stable, both the poster and sticker perturbation demonstrate a high success rate in which at least 85% of the total video frames do not contain a Stop sign bounding box. When we evaluated our perturbations in an outdoor environment, we notice a drop in success rate for both attacks. The sticker perturbation also appears to be slightly weaker. We noticed that the sticker perturbation did especially poorly when only a portion of the sign was in the camera’s field of view. Namely, when the sticker perturbation began to

FR-CNN	Poster	Sticker
Indoors	189/220 (85.9%)	146/248 (58.9%)
Outdoors	84/209 (40.2%)	47/249 (18.9%)

Table 4.2: Attack success rate for the disappearance attack on Faster R-CNN. We tested a poster perturbation, where the entire Stop sign is replaced with a true-sized print, and a sticker attack, where the perturbation is two rectangles stuck to the surface of the sign. The table cells show the ratio: number of frames in which a Stop sign was *not* detected / total number of frames, and a success rate, which is the result of this ratio.

leave the camera’s field of view, the Stop sign bounding boxes appear very frequently. In contrast, this behavior was not observed in the poster perturbation experiments, likely because some part of the adversarial noise is always present in the video due to the mask’s shape. Figure 4.7 shows some frame captures of our adversarial Stop sign videos.

To measure the transferability of our attack, we also evaluated the recorded videos using the Faster R-CNN object detection network.¹ The results for these experiments are shown in Table 4.2.

We see from these results that both perturbations transfer with a relatively high success rate in indoor lab settings where the environment conditions are stable. However, once outdoors, the success rate for both perturbations decreases significantly, but both perturbations retain moderate success rates. We observe that our improved attack algorithm can generate an adversarial poster perturbation, which transfers to other object detection frameworks, especially in stable environments.

Finally, we report on some preliminary results for creation attacks (the results are considered preliminary in that we have spent considerably less time optimizing these attacks compared to the disappearance attacks—it is thus likely that they can be further improved). When applying multiple copies of the sticker in Figure 4.5 to a

¹We used the Tensorflow-Python implementation of Faster R-CNN found at <https://github.com/endernewton/tf-faster-rcnn>. It has a default detection threshold of 80%

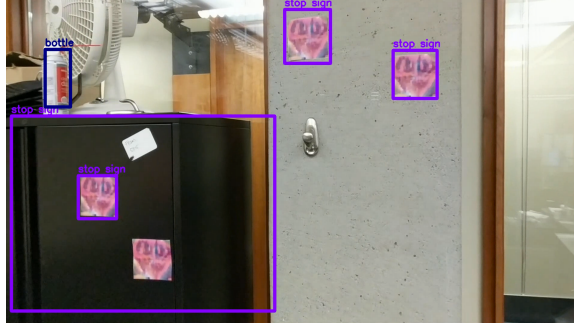


Figure 4.6: Sample frame from our creation attack video after being processed by YOLO v2. The scene includes 4 adversarial stickers reliably recognized as Stop signs.

cupboard and office wall, YOLO v2 detects stop signs in 25%–79% of the frames over multiple independent videos. A sample video frame is shown in Figure 4.6. Compared to the disappearance attack, the creation attack is more sensitive to the sticker’s size, surroundings, and camera movement in the video. This results in highly variable success rates and is presumably because (due to resource constraints) we applied fewer physical and digital transformations when generating the attack. Enhancing the reliability and robustness of our creation attack is an interesting avenue for future work, as it presents a novel attack vector (*e.g.*, DOS style attacks) for adversarial examples.

4.6 Future Work

In the process of generating physical adversarial examples for object detectors, we note several open research questions that we leave to future work.

Lack of detail due to environmental conditions. We noticed physical conditions (*e.g.*, poor lighting, far distance, sharp angles), which only allowed macro features of the sign (*i.e.*, shape, general color, lettering) to be observed clearly. Due to such conditions, the details of the perturbations were lost, causing it to fail. This is expected as our attack relies on the camera being able to perceive the adversarial perturbations

somewhat accurately. When extreme environmental conditions prevent the camera from observing finer details of the perturbation on the sign, the adversarial noise is lost. We theorize that in order to successfully fool object detectors under these extreme conditions, the macro features of the sign need to be attacked. For example, we could create attachments on the outside edges of the sign in order to change its perceived shape.

Alternative attacks on object detectors. In this work, we explored attacking the object detector such that it fails to locate an object, or that it detects non-existent objects. There are several alternative forms of attack we could consider. One alternative is to attempt to generate physical perturbations that preserve the bounding box of an object, but alter its label (this is similar to targeted attacks for classifiers). Another option is to generate further 2D or even 3D objects that appear nonsensical to a human, but are detected and labeled by the object detector. The success of either of these attacks, which have been shown to work digitally [89, 60], would have major safety implications.

Extensions to semantic segmentation. A broader task than object detection is semantic segmentation—where the network labels every pixel in a scene as belonging to an object. Recent work has shown digital attacks against semantic segmentation [89]. An important future work question is how to extend current attack techniques for classifiers, and detectors (as this work shows) to create physical attacks on segmentation networks.

Impact on Real Systems. Existing cyber-physical systems such as cars and drones integrate object detectors into a control pipeline that consists of pre- and post-processing steps. The attacks we show only target the object detection component in isolation (specifically YOLO v2). Understanding whether these attacks are capable of

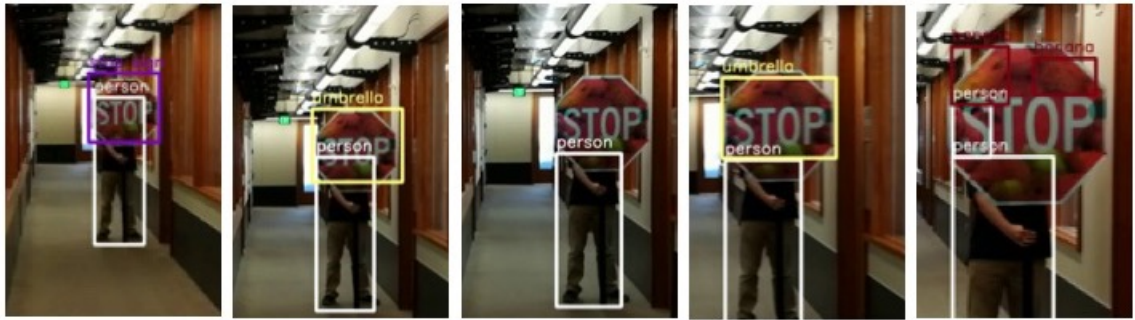
compromising a full control pipeline in an end-to-end manner is an important open question. Although YOLO v2 does recognize a Stop sign in some frames from our attack videos, a real system would generally base its control decisions on a majority of predictions, rather than a few frames. Our attack manages to trick the detector into not seeing a Stop sign in a majority of the tested video frames.

Despite these observations, we stress that a key step towards understanding the vulnerability of the broad class of object detection models to physical adversarial examples is to create algorithms that can attack state-of-the-art object detectors. In this work, we have shown how to can extend the existing RPP attack to attack object detectors in relatively controlled settings, but we have noticed increased interest in regards to adversarial attacks on real complex systems. Of note is work that utilizes differential rendering engines, which better approximates the effect the physical environment has on adversarial noise, thus resulting in more robust adversarial inputs [47, 88, 10].

4.7 Conclusion

Starting from an algorithm to generate robust physical perturbations for *classifiers*, we extend it with positional and rotational invariance to generate physical perturbations for state-of-the-art object *detectors*—a broader class of deep neural networks that are used in dynamic settings to detect and label objects within scenes. Object detectors are popular in cyber-physical systems such as autonomous vehicles. We experiment with the YOLO v2 object detector, showing that it is possible to physically perturb a Stop sign such that the detector ignores it. When presented with a video of the adversarial poster perturbation, YOLO failed to recognize the sign in 85.6% of the video frames in a controlled lab environment, and in 72.5% of the video frames in an outdoor environment. When presented with a video of the adversarial sticker perturbation, YOLO failed to recognize the sign in 85% of the video frames in a controlled lab

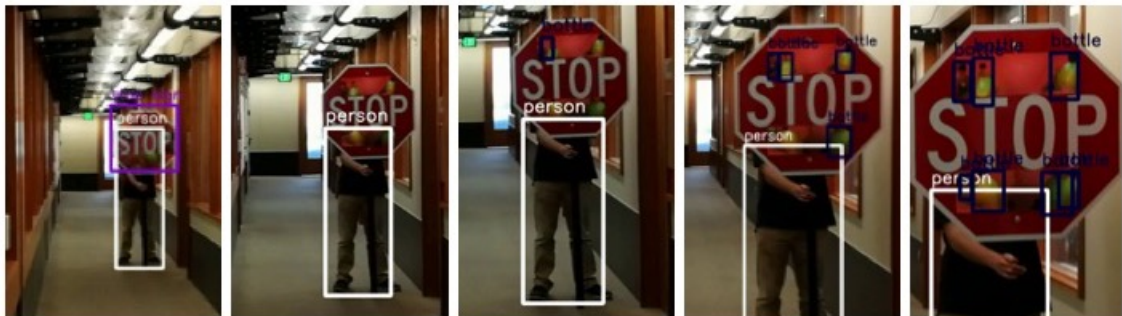
environment, and in 63.5% of the video frames in an outdoor environment. We also observed limited blackbox transferability to the Faster-RCNN detector. The poster perturbation fooled Faster R-CNN in 85.9% of the video frames in a controlled lab environment, and in 40.2% of the video frames in an outdoor environment. Our work, thus, takes steps towards developing a more informed understanding of the vulnerability of object detectors to physical adversarial examples.



(a) The poster attack inside



(b) The poster attack outside



(c) The sticker attack inside



(d) The sticker attack outside

Figure 4.7: Sample frames from our attack videos after being processed by YOLO v2. In the majority of frames, the detector fails to recognize the Stop sign.

CHAPTER V

Robust Feature Augmentation

In this chapter, we introduce a technique to improve adversarial robustness by removing the effect non-robust features have on the final classification decision. We first present a theoretical framework to understand the notion of a robust feature and then demonstrate how clever usage of robust features can improve the adversarial robustness of a classifier.

5.1 Introduction

Deep neural networks (DNNs) are used for various tasks, including image classification with applications to character recognition, traffic sign classification, and autonomous driving. However, the pervasive use of DNNs has also raised concerns as to their robustness, and thus trustworthiness. Namely, existing DNNs have been shown to be vulnerable to *adversarial inputs* [81]. These are inputs that, to a human, appear similar to each other, but are assigned different labels by the DNN.

Currently, there is an interest in designing networks that are robust to adversarial examples. Shafai *et al.* argue that adversarial robustness is limited based on the dimensionality of the input space [74]. Schmidt *et al.* suggest that accurate, but not robust models are a result of an insufficient number of training samples [71]. Under a theoretical model in which it is possible to learn an accurate classifier from a single

sample, they demonstrate that learning a robust classifier requires at least $O(\sqrt{d})$ samples. This problem manifests itself during training as the classifier learns to rely on predictive, but non-robust features. For example, Malhotra *et al.* added pixel noise to training inputs based on the true label of the input and found that the classifier learned to value the position of the noise pixel over any other feature when classifying the data [52]. Other works make similar findings, showing that the traditional training of classifiers results in classifiers learning highly predictive, but non-robust features, thus making them exploitable [33, 22, 85, 13].

In this chapter, we propose a new approach, *robust feature augmentation*, as a component of standard machine learning techniques. In this approach, we augment a classification pipeline with *robust features* that mitigate the effect of most adversarial perturbations, thus improving the overall adversarial robustness of the classifier. Under a theoretical model, we provide results and characterizations that help explain as to why this approach improves adversarial robustness. Our work is also interesting in the light of recent works on certifiable robustness, for *e.g.*, Cohen *et al.* [20] mention that “*it is typically impossible to tell whether a prediction by an empirically robust classifier is truly robust to adversarial perturbations*” however, with robust feature augmentation in the classification pipeline itself, one can expect robustness to bounded adversarial perturbations, by construction.

Adversarial training, popularized by Madry *et al.*, is the current standard approach for designing robust machine learning models, in which L_∞ -bounded adversarial examples are generated during training (see Section 5.3.2 for details). However, adversarial training is costly. As an alternative approach, we suggest augmenting a classification pipeline with robust features. Compared to Madry *et al.* [51], robust feature augmentation without adversarial training achieved 80% adversarial accuracy 33x faster on MNIST. Additionally, combining robust feature augmentation with adversarial training achieved a 14x training time speedup for achieving 90% adversarial

accuracy. Since robust feature augmentation works well with any DNN, we can get higher accuracy compared to recent attempts at certifiable robustness [66] on the MNIST dataset.

A concurrently developed approach is to create a dataset that only contains robust features [33]. Previously, this approach was shown to improve the robustness of a trained model, but required precise manipulations of the dataset [22]. Using adversarial training on CIFAR, Ilyas *et al.* created an adversarially robust model, from which they identified robust and non-robust features. They removed the non-robust features from the dataset and showed that standard training on the robust dataset improved adversarial performance by about 45% while decreasing test accuracy by 10%. However, this approach improvement still failed to outperform the model adversarially trained on the original dataset. In our approach, we choose to improve robustness by identifying robust features that can be added directly to the classification pipeline, thus preserving standard training techniques. Our intuition is that since adversarial examples are a human-defined phenomenon, robust features can also be similarly defined.

Our Contributions:

- We define the notion of a robust feature, computed from an input. Informally, a robust feature is a feature that does not change as the input is adversarially perturbed. Typically, we intend these features to be meaningful attributes such as color and shape of an object being classified. But, it can also be a coarser categorization of the input that is expected to be stable under permitted adversarial perturbations (Section 5.4).
- We show that a function computed on a set of robust features is also robust. In other words, we can use robust features as an input for robust classification decisions (Section 5.4).

- We show theoretical connections between (1) adversarially-trained classifiers that attempt to discover a non-linear separation boundary to maximize the separation between natural inputs and (2) using robust functions to map natural inputs to "pure" natural inputs and then using a linear classifier to separate the points (Section 5.4).
- On MNIST, we use a binarization function as a robust feature and show that it improves the robustness of a standard classifier from 0% to 74.64% without any adversarial training. For an adversarially trained classifier, binarization reduces the training time by 14x for comparable adversarial accuracy and retains better accuracy as attack radius increases, e.g., 87.13% adversarial accuracy vs 34.88% for $\epsilon = 0.35$ as compared to [51] (Section 5.5).
- On a traffic sign dataset, we design a robust color extractor to augment a standard traffic sign classifier. Our augmented classifier prevents more than 90% of adversarial attacks between signs of different colors (Section 5.6).

5.2 Related Work

We survey the related work on robust and non-robust feature information. For related work on adversarial defenses, please refer to Chapter II.

In robust feature augmentation, we hypothesize that the input contains non-robust, predictive feature information. Thus, the weakness of current machine learning algorithm to adversarial attacks is due to a heavy bias on such information. Malhoutra and Bowers observed such an effect in their experiments in which they trained networks on poisoned datasets [52]. For each image in the training data, they added noise dependent on the underlying image labels. They found that standard training techniques created classifiers, which were heavily biased towards the presence or absence of the label-dependent noise. Thus, removing the network's bias on such

features information can theoretically improve the network’s adversarial robustness.

In their work on an Imagenet classifier, Geirhos *et al.* observed that traditional training techniques resulted in classifiers using drastically different classification strategies compared to their human counterparts [22]. While a human observer learns to inspect shape or color information when labeling an object, the researchers found that machine learning classifiers were instead heavily biased towards examining local texture information to determine the output label. To correct the classifiers’ bias, Geirhos *et al.* created Stylized-ImageNet, which is an altered version of Imagenet where local texture information is removed from the images. After training classifiers on this new dataset, they observed that shape based representations were learned, which resulted in improved performance and robustness. From their work, it appears that directly embedding human priors (*i.e.*, shape) into the dataset can improve the general robustness of models.

Ilyas improved upon this dataset transformation approach through the use of adversarial training [33]. Given a dataset, an adversarially robust model is created using adversarial training. From the adversarially robust model, they derived a set of robust features and then transformed the original dataset into a dataset that is only comprised of robust features. They define features as the activation values in the penultimate layer of a classifier for a given input. Through standard training on the robust dataset, compared to the baseline model, they obtained a 45% increase in adversarial robustness, while suffering a 10% decrease in natural test accuracy. Their approach is interesting as it allows for the use of standard training techniques given a robust dataset. However, in order to create the robust dataset, adversarial training is still required and, compared to the model trained on the robust dataset, an adversarially trained model has both increased natural and adversarial accuracy, thus negating the usefulness of the robust dataset. Their work, though, does suggest one method to automatically discover robust features for any dataset. As such, their

technique may improve the applicability of robust feature augmentation.

5.3 Preliminaries

Before delving into the details of the main contributions of our work, we first give an overview and some useful notation and definitions, as well as a more thorough explanation of adversarial training

5.3.1 Notation and Definitions

In this section, we establish some notation and definitions that will be useful in the exposition of the remaining chapter. We often refer to the set $\{1, \dots, m\}$ as $[m]$ for the ease of notation. We assume there is an underlying data distribution \mathcal{D} which the input set $\mathcal{X} \subseteq \mathbb{R}^n$ belongs to and each $x \in \mathcal{X}$ has a corresponding ground truth label $y \in [k]$. In particular, one can think of \mathcal{X} as the set of inputs that a human (or an oracle) is able to classify. We follow the supervised machine learning setup where the basic goal is given a training dataset and corresponding labels $(x_i, y_i)_{i \in [N]}$, learn a function $F : \mathcal{X} \mapsto [k]$, that is a good approximation for the unknown function $f : \mathcal{X} \mapsto [k]$. We will assume that f is such that $f(x_i) = y_i$ in the given data. More specifically, the goal is to seek to minimize the loss over a random sample over the input space $\mathbb{P}_{x \sim \mathcal{D}}(F(x) \neq f(x))$, which is often approximated by minimizing an empirical loss over a random training sample.

It has been shown that although highly accurate approximations of $f(\cdot)$ can be learned, these approximations are not robust for with respect to perturbations of a majority of inputs. Let $d(\cdot, \cdot)$ be a distance function that measures the distance between inputs in \mathbb{R}^n and let us denote an ϵ -neighborhood of x , $B(x, \epsilon)$, as the set of points in \mathcal{X} at a distance at most ϵ away from x , i.e., $B(x, \epsilon) = \{z \in \mathcal{X} \mid d(x, z) \leq \epsilon\}$ for some given $\epsilon > 0$. We call a function *robust* if it does not change its output over small neighborhoods around a subset of desired inputs $\mathcal{P} \subseteq \mathcal{X}$.

Definition 1. A function $F : \mathcal{X} \rightarrow [k]$ is said to be **robust** over a subset $\mathcal{P} \subseteq \mathcal{X}$ with respect to $\epsilon > 0$ if for all $x \in \mathcal{P}$: $F(x) = F(z)$ for all $z \in B(x, \epsilon)$.

We refer to \mathcal{P} as “pure” inputs. Since the set of all possible inputs, \mathcal{X} , encountered in practice is assumed classifiable by a human (or an oracle), we can assume¹ that $\mathcal{X} = \cup_{x \in \mathcal{P}} B(x, \delta)$ for some $\delta > 0$. As an example, a constant function is robust over all inputs, by definition, however it may not be accurate. For some large enough ϵ , the ground truth $f(\cdot)$ may itself not be robust, although it is accurate. Combining accuracy and robustness, we can define an adversarial input as follows:

Definition 2. Given a ground truth function $f : \mathcal{X} \rightarrow [k]$ and a learned classification function $F : \mathcal{X} \rightarrow [k]$, suppose $F(x) = f(x) = f(z)$ for some $z \in B(x, \epsilon)$ and $F(x) \neq F(z)$, then z is an **adversarial example** for the classification function $F(\cdot)$.

Suppose a function F is robust² on a subset $R_{F, \epsilon} \subseteq \mathcal{X}$ with respect to ϵ (i.e., $F(x) = F(z)$ for all $z \in B(x, \epsilon)$, $x \in R_{F, \epsilon}$). By definition, any input in $R_{F, \epsilon}$ cannot be an adversarial example for the function F with respect to ϵ and any arbitrary ground truth function f .

Ideally, we would like a robust classifier that is also **accurate** on this input space, i.e., a classifier that minimizes the loss on pure inputs as well as the percentage of pure inputs that have adversarial inputs. Increases in robustness may result in a loss of accuracy, and the goal is to find a feasible trade-off. For the rest of the chapter, we will assume that ϵ is chosen small enough such that perturbing inputs in \mathcal{P} within an ϵ -neighborhoods does not change the ground truth classification, and we would like to compute classifiers that are robust over \mathcal{P} .

¹Note that by this definition, the classifiable set of inputs is not convex since convex combination of two points in different neighborhoods may not lie in the neighborhood of any pure data point.

²A related notion is that of certifiable robustness that deals with the user being able to certify robustness of a given classifier, in the sense of property testing [66].

5.3.2 Adversarial Training

Madry et al. [51] proposed the use of adversarial training in which they solve

$$\min_{\theta} \rho(\theta), \text{ where: } \rho(\theta) = \mathbb{E}_{(x,y) \sim \mathcal{D}} \left[\max_{\delta \in \mathcal{S}} L(\theta, x + \delta, y) \right]$$

In their formulation, \mathcal{S} is the set of allowed perturbations. The loss function L quantifies the loss relative to the perturbed input $x + \delta$ and the original label y . The inner maximization problem seeks to find a perturbation δ that maximizes the loss for a given input x . The outer minimization problem aims to find the model parameters θ such that the expected adversarial loss in the inner maximization problem is minimized.

In practice, Madry *et al.* use projected gradient descent (PGD) to generate adversarial examples. At each iteration t , the input x^t is modified based on the negative gradient of the loss function:

$$x^{t+1} = \Pi_{x+\mathcal{S}}(x^t + \alpha \text{sgn}(\nabla_x L(\theta, x^t, y)))$$

\mathcal{S} is the set of allowed perturbations as defined previously. $\Pi_{x+\mathcal{S}}$ is a clip function, which ensures the perturbed input x^{t+1} is within the allowable range. Their approach was only demonstrated to work well on small to medium datasets such as MNIST and CIFAR, but others have modified the approach to work on larger datasets such as Imagenet [38].

5.4 Robust Feature Augmentation

We propose two general techniques of developing robust classifiers: binarization (Section 5.4.1) and group feature extraction (Section 5.4.2).

First, we propose that if the first stage of a deep learning pipeline is robust to

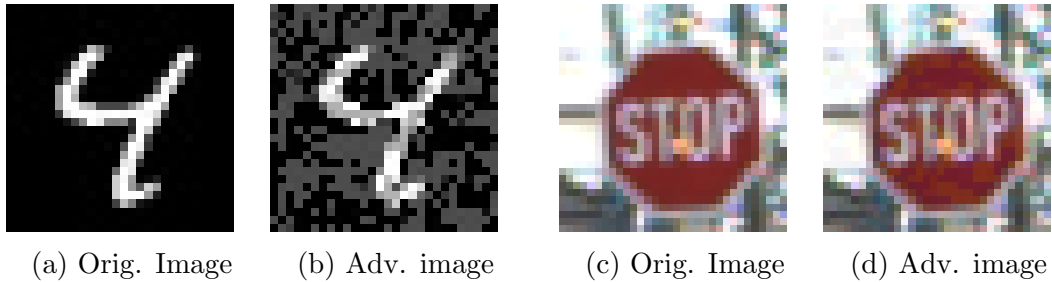


Figure 5.1: The MNIST image in (a) is correctly classified as a “4”, however image in (b) is misclassified as an “8”, despite only minor visual distortions in the image. Similarly, the image in (c) is correctly classified as a STOP sign, but the image in (d) is misclassified as German KEEP LEFT sign.

a class of perturbations, then the overall pipeline will also be robust against those perturbations. An example of binarization is a simple rounding filter that, when applied to an image, can remove perturbations on most pixels. Such a function is useful in images where there is a notion of a static background and only the presence of a single type of pixel defines the object. Previous works have demonstrated that, for MNIST, binarization is remarkably effective in improving adversarial robustness with respect to small pixel perturbations, thus we also use this technique [14, 27]. We will present theoretical reasons in Section 5.4.1 on why the use of a binarizer improves robustness even without requiring adversarial training for the special case of a linear classifier, as well as present experimental results on MNIST in Section 5.5 that show improved adversarial accuracy with this simple, yet powerful idea.

Our second proposal is a generalization of binarization: to use one or more simpler image features (*e.g.*, color and shape for objects) that are expected to be robust to adversarial perturbations. Consider the domain of traffic sign images in the US: a standard STOP traffic sign is known to be predominantly red and with octagonal shape. Traditional adversarial attacks on images change neither feature as there is a constraint to maintain the visual appearance of the original input (*e.g.*, Figure 5.1). Therefore, it is apparent that standard classifiers do not learn to prioritize these features, shape and color, for labeling the sign. Rather, other predictive, non-robust

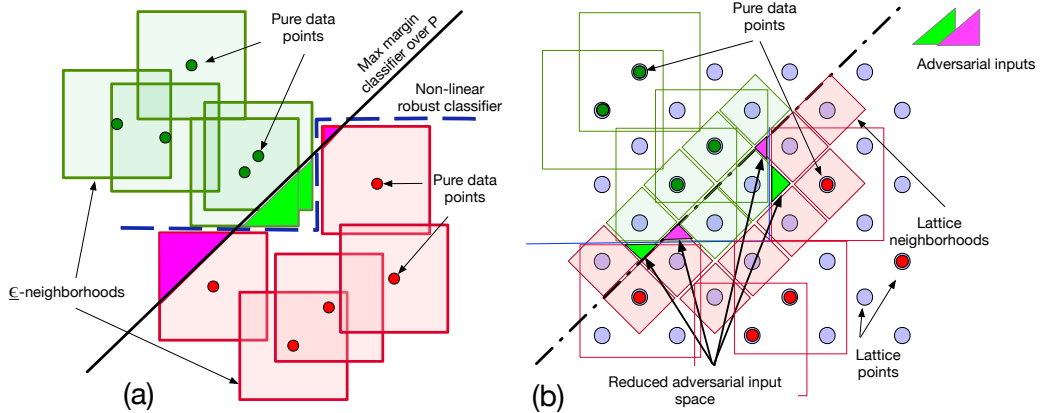


Figure 5.2: (a) Max-margin linear classifier, trained over pure data points \mathcal{P} , results in large adversarial input space. Binarizing test data to the nearest-neighbor in \mathcal{P} before classification removes these adversarial inputs completely. (b) When \mathcal{P} is not known, binarization to the nearest lattice point reduces adversarial input space.

features, are learned, which are then exploited by the adversary so as to maintain the visual appearance of the STOP sign, while causing the predicted label to change. Our goal, then, is to make classifiers more robust by explicitly factoring in any known discriminating features that are robust to perturbations on a large subset of the input space.

5.4.1 Binarization

In order to remove spurious noise learned by a DNN, we propose *binarization* or a *snapping* of input data to desired intervals. Experimentally, we found that about 82% of the pixels in MNIST images are concentrated near 0 and 8% are concentrated near 1. The remaining pixels are somewhat evenly distributed between 0.1 and 0.9. We observed that adversarial attacks often changed background pixels, and if the changes were removed, the classifier would correctly label the example. Previous work suggests that a binarization function, which rounds all the pixels to $\{0,1\}$ based on a threshold, can improve robustness of the resulting classifier [14, 27]. Although its name suggests rounding values in $[0, 1]$ to $\{0, 1\}$, we define binarization more generally:

Definition 3. Consider a set $\mathcal{S} \subseteq \mathbb{R}^n$. Any function $b(\cdot)$ that maps each data point

in the input space $\mathcal{X} \subseteq \mathbb{R}^n$ to elements in \mathcal{S} is called a binarizer, and $b(x)$ is referred to as the binarization of $x \in \mathcal{X}$.

Typically, \mathcal{S} is chosen to be much smaller in cardinality compared to \mathcal{X} . Suppose the binarizer b is defined with respect to a distance $d(\cdot)$ such that $b(x)$ is the nearest neighbor of x in \mathcal{S} , i.e. $b(x) \in \arg \min_{z \in \mathcal{S}} d(x, z)$. If $\mathcal{S} = \mathcal{P}$, we get a binarizer to map any data point to the nearest neighbor from the pure set of points \mathcal{P} . If $\mathcal{S} = \{0, 1\}$ and $d(x, y) = \|x - y\|_\infty$ we get the vanilla form of binarization where every pixel is rounded to 0 or 1. One could also define a binarizer with respect to a threshold³, for example $b(x) = \mathbb{I}_{x_i \geq \tau}$, which rounds each element to $\mathcal{S} = \{0, 1\}$ based on whether the coordinate-wise value is less than threshold or not.

We verify in Section 5.5 that binarization has a minimal effect on the standard accuracy of the classifier, but greatly improves the adversarial robustness. Furthermore, binarization can be combined with adversarial training. The combination achieved both an order of magnitude faster training time and higher adversarial accuracy as compared to Madry *et al.* [51], with similar test accuracy.

Why does binarization help? To see why binarization works in practice, consider the example of a support vector machine that computes a max-margin linear classifier. In Figure 5.2(a), suppose the set of “pure” data points \mathcal{P} are the green and red dots, and their ϵ -neighborhoods in the L_∞ norm are the colored squares enclosing them. In this example, the pure data points are linearly separable, although the ϵ -neighborhoods are not. We depict the max-margin linear classifier with a solid line that separates the green points from the red points. Clearly, this example has a large adversarial instance space (pink, bright green regions in Figure 5.2(a)) which belongs to an ϵ -neighborhood of some pure data point, however, these would be misclassified by the linear classifier. On the other hand, suppose a data point was first binarized

³Here, \mathbb{I}_A is simply an indicator vector for whether A is true or not.

to nearest-neighbor in \mathcal{P} , this would *completely remove* adversarial instances and we could obtain a perfect classifier even with the underlying classification technique being a support-vector machine. This point is important enough to be stated again:

Augmenting the classification pipeline with a nearest-neighbor mapping increases the power of linear classification to allow non-linear separability (blue decision boundary in Figure 5.2(a)).

Note that the resultant model from augmenting binarization and linear classification (see Figure 5.3) is not only powerful in removing adversarial samples but also does so in an interpretable way. We formalize this example in the theorem below.

Theorem 1. *Consider a max-margin classifier that is trained on $\{(x_i, y_i)\}_{i=1}^N = \mathcal{P}$ that is linearly separable. Consider a distance function $d : \mathcal{P} \times \mathcal{P} \rightarrow \mathbb{R}$ and a parameter $\epsilon > 0$. For any two data points $x_i, x_j \in \mathcal{P}$, suppose that the $d(x_i, x_j) > 2\epsilon$ whenever the ground truth labels $y_i \neq y_j$. Consider a nearest-neighbor binarizer $b(x) = \arg \min_{z \in \mathcal{P}} d(x, z)$, and the max-margin linear classifier L (trained over \mathcal{P}), then the augmented classifier $C(x) = L(b(x))$ is **robust** over \mathcal{P} with respect to ϵ and *exact*⁴ over the ϵ -neighborhood.*

The theorem holds because $b(x)$ is uniquely (and correctly) mapped to the original (unperturbed) data point using the nearest-neighbor binarizer, and these are perfectly classified using $C = L(b(\cdot))$ since \mathcal{P} is linearly separable (therefore $L(\cdot)$ did not introduce errors on data points in \mathcal{P}). Since the ϵ -neighborhoods of oppositely classified points do not overlap, we are able to perfectly classify the perturbed points using a linear classifier composed with nearest neighbor matching.

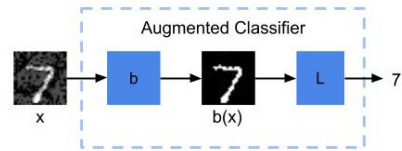


Figure 5.3: The MNIST model with a binarization function b and classifier L .

In the case when \mathcal{P} is not known, we use binarization to map training/testing data

⁴By *exact*, we mean no errors in classification.

points to the nearest points in a lattice, *e.g.*, the set of all 0/1 vectors $\{0, 1\}^n$. This binarizer naturally acts as a regularizer for the output function since outputs in the neighborhoods of lattice points cannot change with small perturbations. Classification boundary over 0/1 vectors is much simpler than over the original (non-binarized) adversarial data. In our experiments, we augment a DNN with a lattice binarizer, which already gives compelling experimental results without adversarial training. We depict the reduction in adversarial input space in Figure 5.2(b).

5.4.2 Group Feature Extraction

Although binarization improves adversarial robustness on MNIST, previous work concluded that the improvements are a special case [14]. On higher dimensional datasets, Chen *et al.* show that regardless of the granularity, binarization does not significantly improve adversarial performance. However, their work does not accurately characterize binarization for higher dimensional datasets. On MNIST, binarization should be viewed as a *robust feature extraction* for digit shape. As such, we extend our definitions to encompass a collection of features (such as color, shape, size) that are found to be robust to perturbations. We think of a data point as a member in a group defined by the value of such a feature (*e.g.*, STOP, DO NOT ENTER are members of the “red” color group). Given a predominantly red US traffic sign, it will require a large perturbation to change the majority of the sign to another sign color. However, unlike binarization to a lattice or \mathcal{P} , features like color or shape lie in a much smaller dimension, and lose the finer classification information. We propose two architectures for classification that can incorporate robust group features: (i) *intersection of multiple group features*, and (ii) *augmentation with original classifier*.

In the first architecture (Figure 5.4a), we propose to use multiple robust group feature extractors T_i each of which feeds the feature into G_i to get a subset of possible labels. For example, suppose T_1 extracts the dominant sign color (*e.g.*, red) then G_1

can map the color to a set of possible road signs with the color (*e.g.*, map "red" color to {STOP, DO NOT ENTER}). As this information may not be sufficient uniquely label the input, adding another group extractor T_2 (*e.g.*, for shape) would allow for more precise classification (*e.g.*, identify STOP or DO NOT ENTER). The classifier output is simply the intersection of the possible labels given the extracted robust group features. We show that if all T_i are robust, the resultant classifier formed by intersection is also robust.

Why do group features help improve

robustness?

Recall that in Section 5.3, we defined $F : \mathcal{X} \rightarrow [k]$ as robust over a subset $\mathcal{P} \subseteq \mathcal{X}$ with respect to $\epsilon > 0$ if for all $x \in \mathcal{P}$: $F(x) = F(z)$ for all $z \in B(x, \epsilon)$. For a given classification task that attempts to classify to labels in $[k]$, a *group feature* extractor can

be viewed as a function $T : \mathcal{X} \rightarrow [m]$ that is robust with respect to $\gamma \gg \epsilon$ and maps to

features in $[m]$ (typically, $m < k$). When referring to the architecture, we also refer to T as a group feature extractor. The intuition here is that, if a group feature is known, then designing a feature extractor T , which is robust and accurate, is an easier task than learning a robust and accurate function F . Further, let $G : [m] \rightarrow 2^{[k]}$ map to possible labels given a group feature in $[m]$. We next show that the robustness guarantees naturally follow under function composition of T and G :

Theorem 2. *Consider a group feature extractor $T : \mathcal{X} \rightarrow [k]$ that is robust on some subset of inputs R with respect to $\gamma > 0$, and a potential-label mapping $G : [m] \rightarrow 2^{[k]}$. Then the composition $G(T(\cdot)) : \mathcal{X} \rightarrow 2^{[k]}$ is also robust on R with respect to γ .*

Theorem 2 holds since the internal group feature extractor T acts as a *shock*

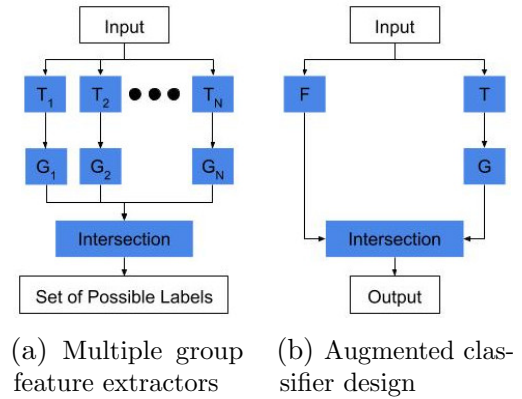


Figure 5.4: Basic architecture of a robust classification network using group classifiers.

absorber and the G function is oblivious to the noise. Indeed for any $z \in B(x, \gamma)$ for $x \in R$, $T(x) = T(z)$ (due to robustness of T), and therefore, $G(T(x)) = G(T(z))$, i.e., $G(T(\cdot))$ is robust on R with respect to γ . Robustness guarantees also hold in the case of intersection of multiple robust features:

Theorem 3. *Consider a set of robust feature extractors $T_i : \mathcal{X} \rightarrow [k]$ that are robust on some subset of inputs R_i with respect to γ_i , and a sequence of potential-labels mappings $G_i : [m] \rightarrow 2^{[k]}$ for $i = 1, \dots, p$. Then the classifier that results by intersecting these: $C(x) = \bigcap_{i=1}^p G_i(T_i(\cdot))$ is robust on $R = \bigcap_{i=1}^p R_i$ with respect to $\gamma = \min_{i=1, \dots, p} \gamma_i$.*

Theorem 3 holds trivially if $R = \emptyset$. Now consider $x \in R$ and γ as defined. Then, for any $z \in B(x, \gamma)$, we have $G_i(T_i(z)) = G_i(T_i(x))$ using Theorem 2. Therefore, $C(z) = \bigcap_{i=1}^p G_i(T_i(z)) = \bigcap_{i=1}^p G_i(T_i(x))$ for all $z \in B(x, \gamma)$ and $x \in R$.

One limitation of the above architecture is that we may not know sufficient robust features to make an unambiguous classification. To address this, we propose the *augmented architecture* (Figure 5.4b). Specifically, we deploy two networks in parallel, a group feature extractor of Figure 5.4a operating in parallel with a standard classifier F . The output of group-based network will be classification possibilities and we require outputs of F and G to be consistent with each other. This prevents targeted attacks on x that change to a label $\notin G(T(x))$, e.g., changing a STOP label (red) to a TRAFFIC LIGHT AHEAD (yellow) label, thus reducing adversarial attack space.

This idea itself is quite powerful since it helps the DNN flag outputs where there might be an inconsistency: *Consider an augmented classifier $C(x) = F(x) \cap G(T(x))$. When $C(x) = \emptyset$ and $G(T(\cdot))$ is exact (i.e., no errors), then we know that $F(x)$ was definitely an example that was misclassified.* This can be very useful in practice, where a machine can flag a difficult instance of data, and let an oracle (or a human) take over in these cases until $F(\cdot)$ can be made more accurate. We formalize this in the following theorem:

Theorem 4. *Consider a classifier $F : \mathcal{X} \rightarrow [k]$ and suppose we have access to a group*

feature extractor $T : \mathcal{X} \rightarrow [m]$ as well as a labels mapping $G : [m] \rightarrow 2^{[k]}$. Consider the augmented classifier $C(x) = F(x) \cap G(T(x))$. If $T(\cdot)$ is robust over \mathcal{P} with respect to γ , then for all $z \in B(x, \gamma)$, $C(z)$ is non-empty if and only if $F(z) \in G(T(x))$.

The above theorem holds because robustness of $T(\cdot)$ implies robustness of $G(T(\cdot))$ from Theorem 2. Thus, the label of $C(\cdot)$ for both x and $z \in B(x, \gamma)$ must be in $G(T(x))$, ruling out targeted attacks that change label of $F(x)$ to a label not in $G(T(x))$.

As an example scenario of the above theorem, suppose $x \in \mathcal{P}$ is an image of a STOP sign. $T(x)$ is determined to be red. Then, $G(T(x))$ is the set of sign labels that can be red, e.g., a set including the STOP sign and DO NOT ENTER sign. Let's assume that normal case that F classifies the sign x correctly. Then, $C(x)$ will also give a correct classification. Furthermore, for an arbitrary input $z \in B(x, \gamma)$, since $G(T(z)) = G(T(x))$ due to robustness of T , label of $C(z)$ is restricted to be either \emptyset or in the set of red signs, $G(T(x))$. $C(z) = \emptyset$ implies an inconsistency between the two outputs of F and $G(T(\cdot))$ on input z , suggesting a problem, which may require human inspection or another intervention to resolve. A non-empty result implies that the two inputs are of the same color, though not necessarily the same label.

5.5 Binarization Augmentation on MNIST Results

We start with a simple classification task, digit classification on the MNIST dataset [54], and show that a binarization function both improves adversarial robustness and reduces training time compared to adversarial training to achieve a similar level of adversarial robustness. We use the pre-trained natural and adversarially trained MNIST classifiers used by Madry *et al.* [51]. For the attack, we use the PGD momentum attack code created by Zheng *et al.* [94]. Our experiments compare four models, two of which use proposed binary augmentation:

1. **Natural Model (NATURAL):** Madry *et al.*'s pre-trained natural classifier.
2. **Madry *et al.*'s Adv. Trained Model (MAT):** Madry *et al.*'s pre-trained robust classifier.
3. **Binarized Natural Model (BIN):** A natural classifier with a binarization function as the first processing step, trained on the natural training data (no adversarial training).
4. **Binarized Adv. Trained Model (BAT):** A classifier with a binarization function at the input, with the overall classifier trained on adversarially perturbed training data.

All models use the same model architecture (same as used in [51]). BIN and BAT include a binarization function, encoded as a step function centered at a threshold τ , at the input of the network. Any pixel which is below τ ($\tau = 0.5$ by default) is set to 0; else it is set to 1. For BAT and MAT, we generated adversarial examples in $B(x, 0.3)$ for any given x , we run 100 iterations

Table 5.1: The accuracy of each model evaluated against the MNIST test set and L_∞ perturbations within $\epsilon = 0.3$.

Model	Test Acc.	Adv. Acc.
NATURAL	99.17%	0%
BIN*	98.93%	74.64%
MAT	98.04%	89.72%
BAT*	99.29%	91%

of the PGD attack with a step size of 0.0075 and 20 random restarts. As in the original experiments done by Madry *et al.* [51], an adversarial attack on a particular input sample is considered successful if at least one of the 20 generated adversarial perturbations is successful in changing the predicted label. For BAT, since the step function is non-differentiable, we use the Backward Pass Differential Approximation (BPDA) technique to generate good adversarial examples, as suggested by Athalye *et al.* [3].

We first evaluated the test and adversarial accuracy of all 4 models for $\epsilon = 0.3$ (*i.e.*, using PGD to find adversarial examples for an input x within $B(x, 0.3)$, see Table 5.1).

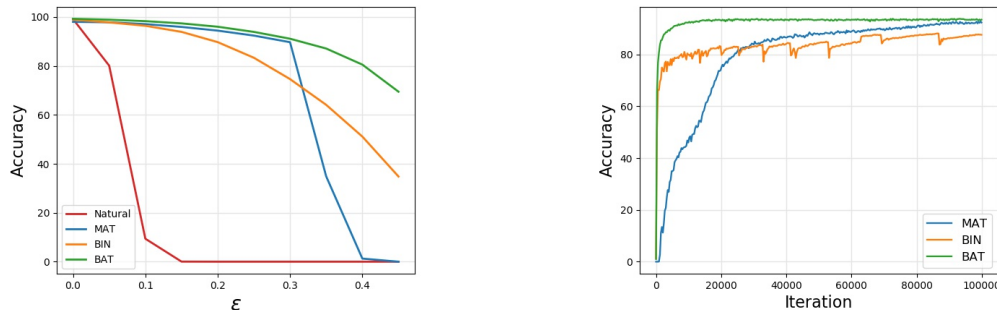


Figure 5.5: The adversarial performance during testing (left) and training (right). Not shown in the figure: MAT and BAT take approximately 10x more time per training iteration than BIN.

We observe that binarization greatly improves the adversarial accuracy of NATURAL from 0% to 74.64% despite no adversarial examples being used during training. We see that BAT, the binarized implementation of MAT, improved adversarial accuracy from 89.72% to 91.14%. Test accuracy was over 98% for all models.

We next measured the adversarial accuracy of all four models for different values of ϵ between 0 and 0.5. We emphasize that MAT and BAT are still trained for $\epsilon = 0.3$; only the attacker’s capabilities are changed. In Figure 5.5, we see that binarization is likely the reducing attack space for large ϵ (e.g., BIN outperforms MAT when $\epsilon = 0.35$ with adversarial accuracy of 64.11% versus 34.88%, respectively). Also, adversarial training used with binarization further improves the robustness of the classifier (e.g., BAT has an adversarial accuracy of 87.13% for $\epsilon = 0.35$, more than double that of MAT).

The above findings can be particularly important in settings where adversarial training is infeasible, say for learning on edge computing devices with smaller computational budget. BIN itself, with no adversarial training, results in a significant initial adversarial robustness. In MAT and BAT, each iteration of training is more expensive since a PGD attack is executed to create a set of adversarial training examples. To further analyze the training efficiency, we evaluated the adversarial accuracy every

300 training iterations for both binarized models and MAT⁵. Adversarial examples with $\epsilon = 0.3$ were generated using 100 iterations of the PGD attack with a step size of 0.0075 and no random restarts⁶. These results are shown in Figure 5.5. We observe that although MAT achieves a higher adversarial accuracy than BIN after about 30,000 iterations, each training iteration for MAT took 236 ms versus 22 ms for BIN. As a result, BIN achieved 80% adversarial accuracy after about 2.9 minutes of training versus 96 minutes of training for MAT. In BAT, where binarization is used during adversarial training, we see large reductions in training time required for comparable adversarial accuracy. BAT achieved 80% adversarial accuracy in about 3.6 minutes and 90% accuracy in about 19 minutes. MAT only achieved 90% after 273 minutes of training (14x slower than BAT).

5.6 Group Feature Extraction Results

We now move to a more complex task, traffic sign classification, and demonstrate how using a robust function to extract a robust feature, the dominant color of a sign, can help reduce the adversarial attack space, e.g., preventing attacks that would change a classification across colors (e.g., red STOP to a blue MINIMUM SPEED 30 sign in Germany).

5.6.1 Dataset Description

Traffic signs are fairly standard across countries (e.g., see <https://www.autoeurope.com/roadsigns/> for classes of traffic signs and examples). LISA [46, 55] and German Traffic Sign Recognition Benchmark (GTSRB) [28, 79] are two popular traffic sign datasets that have been extensively used in previous studies. We created a traffic sign dataset using images from both the LISA and GTSRB datasets. The LISA dataset

⁵All training was done on a 12GB Titan X Pascal GPU

⁶40 iterations with a step size of 0.01 is about twice as fast, but the adversarial accuracy of the model suffers

contains images of 47 different U.S. traffic signs. However, there are large class imbalances (*e.g.*, STOP has 1821 images and SPEED LIMIT 55 has 2 images). To address this problem, we first combine the LISA training dataset with the GTSRB training dataset, which has images for 43 German traffic signs classes. The image labelled as STOP in both datasets are combined as they have the same visual appearance. Similarly, the images labelled as DO NOT ENTER and STREETCLOSEDONEYWAY are combined.

The combined dataset still has low representation for some of the individual U.S. traffic signs. To address that, we created two super-classes composed of white rectangular U.S. traffic signs and yellow U.S. traffic signs. The first super-class contains U.S. Speed Limit signs and RIGHT LANE MUST TURN. The second super-class contains U.S. Warning signs and SCHOOL, which are yellow. The 45 class labels in the augmented dataset are provided in Table C.1.

5.6.2 Model Details

We use a publicly available implementation of a multi-scale DNN traffic sign classifier [91] and normally train a classifier on our traffic sign dataset. Our trained model has 97.51% test accuracy based on the GTSRB test dataset containing 12630 images. Based on the architecture shown in Figure 5.4b, we augment this classifier with a robust feature extraction pipeline, responsible for determining the dominant color of the sign and mapping the color to a set of possible traffic signs. Simply described, the color extractor first determines the sign’s position in the image. Once located, it assigns each pixel a label based on the closest color center in the hue color space, either “red”, “blue”, or “yellow”, then outputs the color based on a weighted majority vote. A more detailed description of the model architecture and color extractor can be found in Appendix Further details of the model architecture can be found in Appendix A.1 and Appendix C respectively..

Table 5.2: # Adv. image is the number of adversarial images ($\epsilon = 8$) in which the predicted label matched the adversarial target. The correction rate is the percentage of adversarial examples for which the color extractor outputs red.

Adversarial Target	# Adv. Images	Correction Rate
Blue Signs (GTSRB)	13633	93.53%
Yellow Signs (LISA)	2389	95.33%
Total # of Stop signs	3021	

5.6.3 Experiment Results

We perform 20 iterations of a targeted L_∞ -bounded PGD attack with $\epsilon = 8$ and step size of 2. The goal is to perturb a STOP sign into a target sign class that is either blue or yellow. The performance is evaluated on 9 target sign class (8 blue sign classes, 1 yellow sign class) and reported in Table 5.2.

Overall, the color extractor prevents over 93% of above adversarial attacks that change STOP to a blue or yellow sign (Table 5.2).

Of course, an attacker could attempt to adversarially attack the color extractor’s robustness assumption. Using the same set of STOP sign images, we explored the edges of the ϵ -neighborhood ($\epsilon = 8$) for each image and checked if the color extractor’s output changed at any point. From this, we found that the extractor is robust on approximately 75% of the STOP sign images.

In Figure 5.6, we further measure the robustness of the color extractor on STOP images for varying values of ϵ . We observe that the robustness of the color extractor is extremely high for small values of ϵ , and then steadily decreases. Upon closer examination, we find that many of the points the color extractor is non-robust on for small values of ϵ are points that are very close to a different color boundary, often due to noisy images. We provide a few examples in Figure 5.7. In some cases, like in Figures 5.7a and 5.7b, the sign has a blueish tint, often due to poor lighting. In other cases, like Figure 5.7c, the blurriness hinders correct sign localization. We attribute the differences between robustness for blue and yellow for higher ϵ values

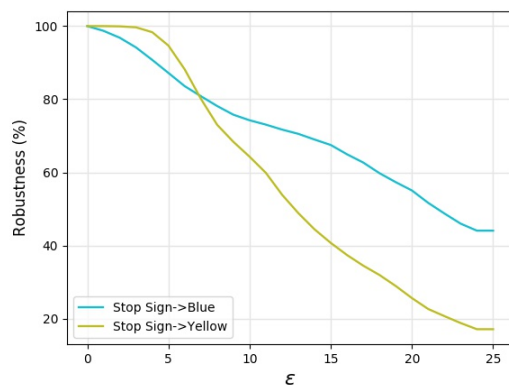
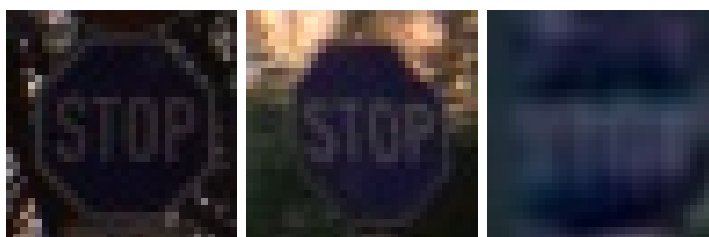


Figure 5.6: The robustness of the color classifier for STOP when changing to blue or yellow signs as L_∞ bound increases.



(a) Close to Blue (b) Close to Blue (c) Close to Yellow

Figure 5.7: Some examples of inputs the color classifier is not robust on. Often, this occurs due to either the image being too dark (which tends to shift colors to blue) or the image being too blurry (which causes errors during sign localization).

to the smaller hue distance between red and yellow as compared to between red and blue. For smaller values of ϵ , the difference is due to dataset artifacts – more STOP signs with very poor lighting in the dataset were closer to having a bluish hue than a yellowish hue (see Figure 5.7 for a few examples).

5.7 Conclusion

The existence of adversarial examples is attributed to a network’s reliance on predictive, but easily exploitable, features it learned during training. In this chapter, we introduced two methods of *robust feature augmentation* to mitigate this problem: *binarizers* and *robust group features*. Both map the input space to a smaller, more

robust, subspace (like a lattice or group labels) and we formally describe these two methods to improve DNN robustness. Experimentally, we demonstrated how these methods can improve the adversarial robustness of a digit classifier and a traffic sign classifier. Furthermore, when adversarial training is used in conjunction with these methods, we were able to train a more adversarially robust model for MNIST 14x faster than without these methods.

We recognize that human identification of robust features may not be applicable to all machine learning tasks, especially if non-interpretable, robust features exist. As such, it is important to develop techniques to identify such features, though doing so is for future work. However, concurrent work done by Ilyas *et al.* has already shown some progress in this area, through the use of adversarial training to remove non-robust features from training data [33]. We expect to see further research, in which robust feature augmentation can be *the* method for adversarial robustness.

CHAPTER VI

Future Work & Conclusion

6.1 Conclusion

Despite the expressive power of machine learning algorithms, adversarial examples are a large problem that potentially threatens their deployment, especially in safety or security critical systems. In recent years, previous work demonstrated that adversarial examples through subtle digital manipulation of the input could cause errors in both machine learning classification and object detection algorithms. However, these works failed to motivate it as a real threat since a high level of control over the system is necessary to digitally manipulate inputs.

This dissertation took the first step in developing a real adversarial attack. Rather than performing digital manipulations of the input, we proposed physically modifying the operational environment of the system as it is publicly accessible. Through careful analysis, we identified several key challenges, which reduce the success rate of physical adversarial attacks and developed the RPP attack to overcome these challenges. Our experiment with road sign detection and identification demonstrated that our attack is successful at creating robust, physical adversarial perturbations for target objects.

Next, we turned towards developing a technique to mitigate the effect of such attacks. From reviewing previous work on adversarial attacks as well as the experiences obtained from developing the RPP attack, we theorized that adversarial attacks succeed

due to subtle manipulations of imperceptible, predictive features. Therefore, in order to mitigate the effect of adversarial inputs, robust, predictive features need to be prioritized in the classification pipeline. Our technique, robust feature augmentation, introduced an adversarially robust classification pipeline, in which known robust features are extracted from the inputs. This extracted information is provided during classification and acts as evidence for the final classification decision. As robust features cannot be adversarially manipulated, the classification decision must be supported by the extracted robust, feature information.

Machine learning is a powerful tool and is seeing widespread use in both old and emerging technologies. However, although many may recognize numerous applications of machine learning, most fail to recognize or understand the risks associated with it. Thus, it is important to be able to evaluate these risks using real threat models so as to inform future development in mitigation techniques.

6.2 Future Work

We describe some additional future research directions stemming from our work in developing adversarial attacks and defenses.

6.2.1 Redefinition of Adversarial Constraints

In general, most adversarial attacks begin with the assumption that generated adversarial examples must have little to no perceptible difference between the adversarial example and the originally labeled input. Mathematically, the perceptible similarity between the two inputs is often represented by the L_p norm distance. In Chapters III and IV, we relaxed this constraint as maintaining the imperceptibility of the perturbation would make the attack impossible given that camera sensors would not be able to detect such subtleties. Furthermore, our attack is focused on creating an object that remains adversarial across any image of that object rather than a single

adversarial image. Thus, we choose to constrain our attack to preserve the *semantic information*, the general appearance of the sign.

Gilmer *et al.* have analyzed this problem and remark that there is little motivation to maintain imperceptible adversarial perturbations for many machine learning applications [23]. To evade a facial biometric system that scans for blacklisted persons, an adversary, presumably blacklisted, would seek to change their actual appearance as much as possible, while remaining inconspicuous. To upload objectionable content such as a lewd image to a social media platform, an adversary would seek to make large changes to the image, while maintaining the semantic content. To unlock a phone through facial verification, an adversary is interested in any input that unlocks the phone, rather than one that is indistinguishable from a verified face. The L_p norm distance, while the most widely used in adversarial attack research, only enforces a limited subset of non-suspicious inputs. In general, adversarial constraints need to be adapted based on the task domain and attack goals in order to create real adversarial attacks and accurately evaluate adversarial defenses.

6.2.2 Automatic Robust Feature Extraction

In Chapter V, we introduced a new technique, robust feature augmentation, as a method to mitigate or prevent the effects of adversarial inputs. However, in our experiments, we relied on domain knowledge to pre-select robust, predictive features such as the color of the road signs to augment the classification pipeline. In general, such easily identifiable, robust, predictive features may not always be available or easy to obtain, especially with non-image inputs. In order to fully utilize robust feature augmentation as a defensive technique, it is necessary to develop methods to automatically identify and extract robust, predictive features. Ilyas *et al.* provides a initial approach towards this goal as they adversarially train a robust model, and then use the model to identify robust features [33].

6.2.3 Adversarial Defense through Foreground Extraction

In Chapter V, we used binarization as a robust feature extraction function to improve the adversarial robustness of an MNIST classifier. Binarization, or more generally pixel discretization, of an input has not resulted in improved adversarial robustness of classifiers trained on larger datasets such as CIFAR or Imagenet [14]. As Chen *et al.* observed, simple pixel discretization as a general technique to improve adversarial robustness only works if the input features (*e.g.*, pixels for images) are separable. They showed that the MNIST dataset fits this criteria, but most complex datasets do not. However, they make the mistake of interpreting binarization in MNIST as pixel discretization for more complex datasets. While it is true that binarization is a form of pixel discretization, it is more accurately characterized as a shape or foreground extraction technique. In MNIST, pixels that are close to 1 represent foreground information (*i.e.*, the digit shape). Thus, binarization can be thought to improve adversarial robustness because it focuses the classifier on semantically informative input information. In work by Xie *et al.*, they noted that adversarial modifications causes a classifier’s feature maps to have high activation on semantically uninformative content. In other words, adversarial attacks cause a classifier to shift its focus from the foreground object or concept information to background information. Based on their analysis as well as our experimental results on MNIST, we expect that a classifier focused on semantically informative foreground information will possess high adversarial robustness.

APPENDICES

APPENDIX A

Traffic Sign Classifier Details

A.1 Model Description

We use a publicly available implementation of a multi-scale DNN architecture [91]. The architecture description is given in Table A.1. Before training, we triple the size of any class with less than 200 images through oversampling and randomly perturbing each image.

Table A.1: Traffic sign classifier architecture. The model expects $32 \times 32 \times 3$ images as input with values in the range $[-0.5, 0.5]$.

Layer Type	Number of Channels	Filter Size	Stride	Activation
conv	3	1x1	1	ReLU
conv	32	5x5	1	ReLU
conv	32	5x5	1	ReLU
maxpool	32	2x2	2	-
conv	64	5x5	1	ReLU
conv	64	5x5	1	ReLU
maxpool	64	2x2	2	-
conv	128	5x5	1	ReLU
conv	128	5x5	1	ReLU
maxpool	128	2x2	2	-
FC	1024	-	-	ReLU
FC	1024	-	-	ReLU
FC	43	-	-	Softmax

APPENDIX B

Inception-v3 Physical Adversarial Images

We include the uncropped images of the adversarially modified microwave and mug along with the distance and viewing angle for the experiments performed in Chapter III.

Table B.1: Uncropped images of the microwave with an adversarial sticker designed for Inception-v3.







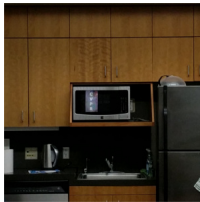
Distance/Angle	Image	Distance/Angle	Image
2' 0°		2' 15°	
5' 0°		5' 15°	
7' 0°		7' 15°	
10' 0°		10' 15°	
15' 0°		20' 0°	

Table B.2: Cropped Images of the coffee mug with an adversarial sticker designed for Inception-v3.

Distance/Angle	Image	Distance/Angle	Image
8" 0°		8" 15°	
12" 0°		12" 15°	
16" 0°		16" 15°	
20" 0°		20" 15°	
24" 0°		24" 15°	
28" 0°		28" 15°	
32" 0°		32" 15°	

APPENDIX C

Color Extractor

We designed a basic color extractor for traffic sign classification. First, we discuss the colors we designed the extractor to identify and the motivation for our choices. Then, we discuss the algorithm and provide additional information regarding the robustness of the algorithm.

C.1 Sign Colors

For a given image of a traffic sign, we designed the color extractor to identify one of three colors: red, yellow, and blue. U.S. red signs (See Figure C.1a) are generally regulatory in nature (e.g., STOP, DONOTENTER). U.S. yellow signs (see Figure C.1b) are used for cautioning a user (e.g., INTERSECTIONAHEAD, CURVERIGHT, CURVELEFT, SCHOOL ZONE). Blue signs (see Figure C.1c) are common in Germany and can be restrictive or mandatory (e.g., KEEPLEFT, MANDATORYLEFTTURN, TRAFFICCIRCLE, MANDATORYAHEAD). Table C.2 identifies the sign labels in the dataset and that are either red and blue. For the purpose of classification, yellow signs are grouped together in a single class due to low representation with respect to the original sign labels (e.g., INTERSECTION: 13 images, CURVELEFT: 24 images, TURNRIGHT: 24 images).



(a) Red sign examples



(b) Yellow sign examples



(c) Blue sign examples

Figure C.1: Examples images of signs for the three color classes we evaluated.

C.2 Color Extraction Algorithm

The color extraction process involves 2 steps:

1. Sign Localization - Determine the sign's location in the image
2. Color Classification - Determine the dominant color of the sign

The full pipeline is shown in Figure C.2.

C.2.1 Sign Localization

Before we can evaluate the dominant color of the sign, we must first identify the pixels in the image that compose the surface of the sign. Due to the presence of numerous noisy images in the dataset, like those shown in Figure 5.7, edge detection

Table C.1: Class labels of the LISA-GTSRB traffic sign dataset used in the experiments.

Class Label	Class Label	Class Label
speedLimit20	streetClosedBothWays	wildlifeWarning
speedLimit30	noTrucks	allRestrictionsEnd
speedLimit50	generalWarning	mandatoryRightTurn
speedLimit60	sharpLeftTurnAhead	mandatoryLeftTurn
speedLimit70	sharpRightTurnAhead	mandatoryAhead
speedLimit80	sequenceSharpTurnsAhead	mandatoryAheadOrRight
endSpeedLimit80	bumpsInRoad	mandatoryAheadOrLeft
speedLimit100	slipperyRoad	keepRight
speedLimit120	tighterRoadOnRight	keepLeft
noPassing	construction	trafficCircle
noPassingTrucks	trafficLight	endNoPassing
intersectionWarning	pedestrianCrossing	endNoPassingTrucks
rightOfWay	schoolCrossing	Yellow Signs
yield	bicycles	doNotEnter
stop	icyRoads	White Rectangles
Total # of Signs	44121	

Table C.2: Red and blue sign groupings. Yellow is not included as they have been grouped into a single label with respect to classification.

Red	Blue
Stop	mandatoryRightTurn
Do Not Enter	mandatoryLeftTurn
	mandatoryAhead
	mandatoryAheadOrRight
	mandatoryAheadOrLeft
	keepRight
	keepLeft
	TrafficCircle

and contour extraction algorithms perform poorly. Instead, given a three channel color image, (r,g,b) , we normalize each individual channel by the image intensity and

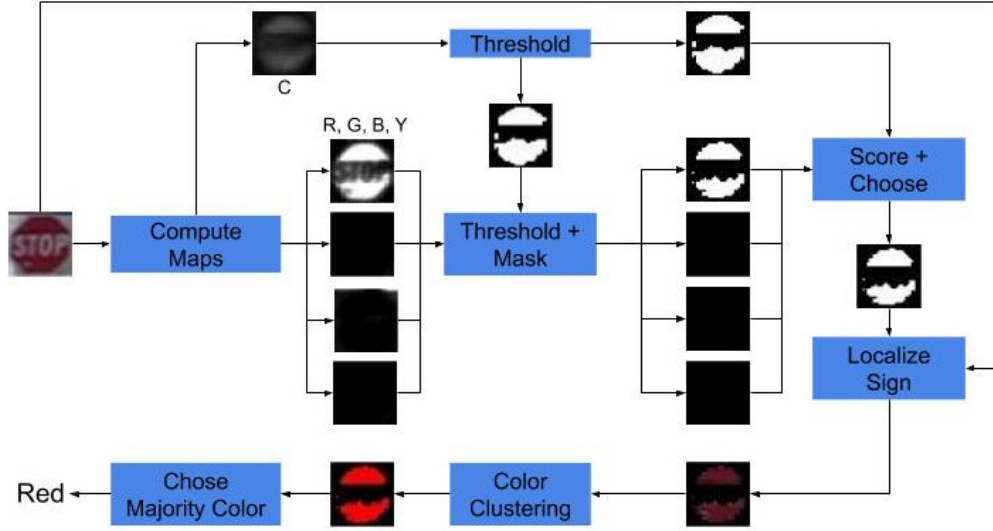


Figure C.2: The color extractor pipeline. We show the step-by-step process for a STOP image.

compute a chromaticity map (C) and 4 color maps (R, G, B, Y) [34, 45].

$$C = \max(r, g, b) - \min(r, g, b)$$

$$R = r - \frac{g + b}{2}$$

$$G = g - \frac{r + b}{2}$$

$$B = b - \frac{r + g}{2}$$

$$Y = \frac{r + g}{2} - \frac{|r - g|}{2} + b$$

Afterwards, all of the maps are converted to a binary image based on the mean of the non-zero values in each map. Then, we use the binary image of C to mask each of the binarized color maps and isolate the chromatic colors in each map. Finally, each channel is scored based on the number of non-zero pixels in the image. If less than 10% of the pixels in each of the four color channels are white, the inverted binary chromaticity map is output. Otherwise, the binarized color channel with the highest score is output. We make one optimization based on the fact that in most of the images, the traffic sign is centered in the image. As such, we restrict thresholding and

scoring to a small box around the center of the image. In our experiments, we used a 10 by 10 box.

C.2.2 Color Classification

The output of the sign localization step is a mask that is applied to the original color image, resulting in an image containing mostly foreground pixels. This image is converted to a hue-based representation (*e.g.*, HSV or HSL). Each non-zero pixel in the masked image is labelled based on the closest color center of three predefined color centers (red, yellow, and blue). Afterwards, a weighed majority vote is computed (*i.e.*, weight of a pixel’s vote increases the closer it is to the center) and the color with the most votes chosen.

For these proof-of-concept experiments, we choose to only detect red, yellow, and blue as these are the three most common colors in the dataset. We did not handle colors such as brown or green as there were no signs in the dataset with these colors. Traffic signs that are white do exist, but white is not characterized by hue. Instead, it is represented by other channel information such as value or lightness. As such, the color extractor is not robust for predominantly white signs, thus our analysis did not focus on such signs. This does not hurt the test accuracy of the augmented model, though, as we can include “white” sign labels in the group-labels for all three colors. When we augment the classifier with the color extractor, the test accuracy on the GTSRB test dataset is 97.51%. Extending the color extractor to extract other colors, or even multiple colors, for finer-grain color-based classification, remains future work.

BIBLIOGRAPHY

BIBLIOGRAPHY

- [1] Anish Athalye. Robust adversarial examples. <https://blog.openai.com/robust-adversarial-inputs/>, 2017.
- [2] Anish Athalye and Nicholas Carlini. On the robustness of the CVPR 2018 white-box adversarial example defenses. *CoRR*, abs/1804.03286, 2018.
- [3] Anish Athalye, Nicholas Carlini, and David A. Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International Conference on Learning Representations (ICLR)*, 2018.
- [4] Anish Athalye and Ilya Sutskever. Synthesizing robust adversarial examples. In *International Conference on Learning Representations (ICLR)*, 2017.
- [5] Arjun Nitin Bhagoji, Warren He, Bo Li, and Dawn Xiaodong Song. Practical black-box attacks on deep neural networks using efficient query mechanisms. In *European Conference on Computer Vision (ECCV)*, 2018.
- [6] Battista Biggio, Iginio Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (EMCLPKDD)*, 2013.
- [7] Haitham Bou-Ammar, Holger Voos, and Wolfgang Ertel. Controller design for quadrotor uavs using reinforcement learning. In *IEEE International Conference on Control Technology and Applications (CCTA)*, 2010.
- [8] Tom B. Brown, Dandelion Mané, Aurko Roy, Martín Abadi, and Justin Gilmer. Adversarial patch. *CoRR*, abs/1712.09665, 2017.
- [9] Tom B Brown, Dandelion Mané, Aurko Roy, Martín Abadi, and Justin Gilmer. Adversarial patch. *CoRR*, abs/1712.09665, 2017.
- [10] Yulong Cao, Chaowei Xiao, Dawei Yang, Jing Fang, Ruigang Yang, Mingyan Liu, and Bo Li. Adversarial objects against lidar-based autonomous driving systems. *CoRR*, abs/1907.05418, 2019.
- [11] Nicholas Carlini and David Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *ACM Workshop on Artificial Intelligence and Security (AISEC)*, 2017.

- [12] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *IEEE Symposium on Security and Privacy (S&P)*, 2017.
- [13] Prasad Chalasanani, Somesh Jha, Aravind Sadagopan, and Xi Wu. Adversarial learning and explainability in structured datasets. *CoRR*, abs/1810.06583, 2018.
- [14] Jiefeng Chen, Xi Wu, Vaibhav Rastogi, Yingyu Liang, and Somesh Jha. Towards understanding limitations of pixel discretization against adversarial attacks. 2019.
- [15] Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *ACM Workshop on Artificial Intelligence and Security (AiSec)*, 2017.
- [16] Shang-Tse Chen, Cory Cornelius, Jason Martin, and Duen Horng Chau. Shapeshifter: Robust physical adversarial attack on faster R-CNN object detector. *CoRR*, abs/1804.05810, 2018.
- [17] Minhao Cheng, Thong Le, Pin-Yu Chen, Jinfeng Yi, Huan Zhang, and Cho-Jui Hsieh. Query-efficient hard-label black-box attack: An optimization-based approach. *CoRR*, abs/1807.04457, 2018.
- [18] Dan Ciresan, Alessandro Giusti, Luca M Gambardella, and Jürgen Schmidhuber. Deep neural networks segment neuronal membranes in electron microscopy images. In *Neural Information Processing Systems (NIPS)*, 2012.
- [19] Moustapha Cisse, Yossi Adi, Natalia Neverova, and Joseph Keshet. Houdini: Fooling deep structured prediction models. *CoRR*, abs/1707.05373, 2017.
- [20] Jeremy M. Cohen, Elan Rosenfeld, and J. Zico Kolter. Certified adversarial robustness via randomized smoothing. *CoRR*, abs/1902.02918, 2019.
- [21] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.
- [22] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. In *International Conference on Learning Representations (ICLR)*, 2019.
- [23] Justin Gilmer, Ryan P. Adams, Ian J. Goodfellow, David Andersen, and George E. Dahl. Motivating the rules of the game for adversarial example research. *CoRR*, abs/1807.06732, 2018.
- [24] Ross Girshick. Fast R-CNN. In *IEEE International Conference on Computer Vision (ICCV)*, 2015.

- [25] Zhitao Gong, Wenlu Wang, and Wei-Shinn Ku. Adversarial and clean data are not twins. *CoRR*, abs/1704.04960, 2017.
- [26] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations (ICLR)*, 2014.
- [27] Abigail Graese, Andras Rozsa, and Terrance E. Boult. Assessing threat of adversarial examples on deep neural networks. 2016.
- [28] German Traffic Sign Recognition Benchmark. <http://benchmark.ini.rub.de/index.php?section=gtsrb&subsection=about>.
- [29] Shixiang Gu and Luca Rigazio. Towards deep neural network architectures robust to adversarial examples. In *International Conference on Learning Representations (ICLR), Workshop Track Proceedings*, 2015.
- [30] Matthias Hein and Maksym Andriushchenko. Formal guarantees on the robustness of a classifier against adversarial manipulation. 2017.
- [31] Dan Hendrycks and Kevin Gimpel. Visible progress on adversarial images and a new saliency map. In *International Conference on Learning Representations (ICLR)*, 2017.
- [32] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012.
- [33] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. *CoRR*, abs/1905.02175, 2019.
- [34] Laurent Itti, Christof Koch, and Ernst Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1254–1259, 1998.
- [35] Jernej Kos, Ian Fischer, and Dawn Song. Adversarial examples for generative models. In *IEEE Symposium on Security and Privacy (S&P)*, 2018.
- [36] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Conference on Neural Information Processing Systems (NIPS)*, 2012.
- [37] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *CoRR*, abs/1607.02533, 2016.

- [38] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial machine learning at scale. In *International Conference on Learning Representations (ICLR)*, 2017.
- [39] Q. V. Le, W. Y. Zou, S. Y. Yeung, and A. Y. Ng. Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.
- [40] Bo Li and Yevgeniy Vorobeychik. Feature cross-substitution in adversarial classification. In *Neural Information Processing Systems (NIPS)*, 2014.
- [41] Bo Li and Yevgeniy Vorobeychik. Scalable optimization of randomized operational decisions in adversarial classification settings. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2015.
- [42] Xin Li and Fuxin Li. Adversarial examples detection in deep networks with convolutional filter statistics. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.
- [43] Fangzhou Liao, Ming Liang, Yinpeng Dong, Tianyu Pang, Jun Zhu, and Xiaolin Hu. Defense against adversarial attacks using high-level representation guided denoiser. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
- [44] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2015.
- [45] King Hann Lim, Kah Phooi Seng, and Li Minn Ang. Intra color-shape classification for traffic sign recognition. In *International Computer Symposium (ICS)*, 2011.
- [46] LISA dataset. <http://cvrr.ucsd.edu/vivachallenge/index.php/signs/sign-detection/>.
- [47] Hsueh-Ti Derek Liu, Michael Tao, Chun-Liang Li, Derek Nowrouzezahrai, and Alec Michael Jacobson. Beyond pixel norm-balls: Parametric adversaries using an analytically differentiable renderer. In *International Conference on Learning Representation (ICLR)*, 2019.
- [48] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European Conference on Computer Vision (ECCV)*, 2016.
- [49] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. In *International Conference on Learning Representations (ICLR)*, 2016.

- [50] Jiajun Lu, Hussein Sibai, Evan Fabry, and David Forsyth. No need to worry about adversarial examples in object detection in autonomous vehicles. *CoRR*, abs/1707.03501, 2017.
- [51] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representation (ICLR)*, 2018.
- [52] Gaurav Malhotra and Jeffrey Bowers. What a difference a pixel makes: An empirical examination of features used by cnns for categorisation. <https://openreview.net/forum?id=ByePUo05K7>, 2019.
- [53] Jan Hendrik Metzen, Mummadi Chaithanya Kumar, Thomas Brox, and Volker Fischer. Universal adversarial perturbations against semantic image segmentation. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.
- [54] MNIST dataset. <http://yann.lecun.com/exdb/mnist/>. Obtained from within TensorFlow Library <https://www.tensorflow.org/tutorials>.
- [55] Andreas Møgelmoose, Mohan Manubhai Trivedi, and Thomas B. Moeslund. Vision-based traffic sign detection and analysis for intelligent driver assistance systems: Perspectives and survey. *IEEE Transactions on Intelligent Transportation Systems*, 13(4), 2012.
- [56] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [57] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [58] Christian Mostegel, Markus Rumpler, Friedrich Fraundorfer, and Horst Bischof. Uav-based autonomous image acquisition with multi-view stereo quality assurance by confidence prediction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [59] Nina Narodytska and Shiva Prasad Kasiviswanathan. Simple black-box adversarial perturbations for deep networks. *CoRR*, abs/1612.06299, 2016.
- [60] Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.
- [61] Nicolas Papernot, Ian Goodfellow, Ryan Sheatsley, Reuben Feinman, and Patrick McDaniel. cleverhans v1.0.0: an adversarial machine learning library. *CoRR*, abs/1610.00768, 2016.

- [62] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *CoRR*, abs/1605.07277, 2016.
- [63] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *ACM Asia Conference on Computer and Communications Security (AsiaCCS)*, 2017.
- [64] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *IEEE European Symposium on Security and Privacy (EuroS&P)*, 2016.
- [65] Nicolas Papernot, Patrick D. McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. 2016.
- [66] Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Certified defenses against adversarial examples. *CoRR*, abs/1801.09344, 2018.
- [67] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [68] Joseph Redmon and Ali Farhadi. YOLO9000: better, faster, stronger. 2016.
- [69] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39:1137–1149, 2015.
- [70] Sara Sabour, Yanshuai Cao, Fartash Faghri, and David J Fleet. Adversarial manipulation of deep representations. In *International Conference on Learning Representations (ICLR)*, 2015.
- [71] Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. Adversarially robust generalization requires more data. In *Conference on Neural Information Processing Systems (NIPS)*, 2018.
- [72] Pierre Sermanet, David Eigen, Xiang Zhang, Michael Mathieu, Rob Fergus, and Yann Lecun. Overfeat: Integrated recognition, localization and detection using convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2013.
- [73] Pierre Sermanet and Yann LeCun. Traffic sign recognition with multi-scale convolutional networks. In *IEEE International Joint Conference on Neural Networks (IJCNN)*, 2011.

- [74] Ali Shafahi, W. Ronny Huang, Christoph Studer, Soheil Feizi, and Tom Goldstein. Are adversarial examples inevitable? In *International Conference on Learning Representations (ICLR)*, 2019.
- [75] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K Reiter. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In *ACM Conference on Computer and Communications Security (CCS)*, 2016.
- [76] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K. Reiter. Adversarial generative nets: Neural network attacks on state-of-the-art face recognition. *CoRR*, abs/1801.00349, 2017.
- [77] Aman Sinha, Hongseok Namkoong, and John Duchi. Certifying some distributional robustness with principled adversarial training. In *International Conference on Learning Representations (ICLR)*, 2018.
- [78] Yang Song, Taesup Kim, Sebastian Nowozin, Stefano Ermon, and Nate Kushman. Pixeldefend: Leveraging generative models to understand and defend against adversarial examples. In *International Conference on Learning Representations (ICLR)*, 2018.
- [79] Johannes Stalkamp, Marc Schlipf, Jan Salmen, and Christian Igel. The German Traffic Sign Recognition Benchmark. IEEE, 2011.
- [80] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [81] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations (ICLR)*, 2014.
- [82] Yaniv Taigman, Ming Yang, Marc’Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.
- [83] Florian Tramr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. In *International Conference on Learning Representations (ICLR)*, 2018.
- [84] Florian Tramr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. In *International Conference on Learning Representations (ICLR)*, 2018.
- [85] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. In *International Conference on Learning Representations (ICLR)*, 2019.

- [86] Shiqi Wang, Yizheng Chen, Ahmed Abdou, and Suman Jana. Enhancing gradient-based attacks with symbolic intervals. *CoRR*, abs/1906.02282, 2019.
- [87] Eric Wong and J. Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. 2018.
- [88] Chaowei Xiao, Dawei Yang, Bo Li, Jianjun Deng, and Mingyan Liu. Meshadv: Adversarial meshes for visual recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [89] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Yuyin Zhou, Lingxi Xie, and Alan Yuille. Adversarial examples for semantic segmentation and object detection. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.
- [90] Cihang Xie, Yuxin Wu, Laurens van der Maaten, Alan L. Yuille, and Kaiming He. Feature denoising for improving adversarial robustness. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [91] Vivek Yadav. p2-traffic signs. <https://github.com/vxy10/p2-TrafficSigns>, 2016.
- [92] Fangyi Zhang, Jürgen Leitner, Michael Milford, Ben Upcroft, and Peter Corke. Towards vision-based deep reinforcement learning for robotic motion control. *CoRR*, abs/1511.03791, 2015.
- [93] Tianhang Zheng, Changyou Chen, and Kui Ren. Distributionally adversarial attack. *CoRR*, abs/1808.05537, 2018.
- [94] Tianhang Zheng, Changyou Chen, and Kui Ren. Distributionally adversarial attack. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2019.