

Fakultät für Informatik

Evaluating the Robustness of Image Classifiers With Adaptive Black-Box Adversarial Attacks

Thomas Brunner

Vollständiger Abdruck der von der

Fakultät für Informatik

der Technischen Universität München zur Erlangung des akademischen Grades

eines Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzende/-r: Prof. Dr. Tobias Nipkow

Prüfende/-r der Dissertation:

1. Prof. Dr.-Ing. habil. Alois Knoll

2. Prof. Dr. Guang Chen

Die Dissertation wurde am 02.11.2021 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 29.05.2022 angenommen.





# Abstract

Ever since the term was first coined, adversarial examples have enjoyed much attention from machine learning and computer vision researchers. The fact that tiny perturbations can lead otherwise robust-seeming models to misclassify an input could pose a major problem for safety and security. However, adversarial examples have so far been considered mostly in laboratory settings that do not apply to real-world systems as they are in use today.

This thesis investigates and evaluates the robustness of image classifiers in an adaptive black-box adversarial setting, where an attacker has no knowledge about the inner workings of a classifier but is allowed to query it multiple times. In theory, it would be possible to conduct such attacks against virtually any image processing system, but existing methods are brittle and suffer from low efficiency. This is a major roadblock that has so far prevented a systematic side-by-side robustness evaluation of classifiers and defense methods.

The thesis addresses this problem by proposing a new black-box adversarial attack that is both reliable and more efficient than prior work. The method exploits several sources of knowledge about the domain of natural images to speed up the search for adversarial examples. This greatly reduces the required number of queries to the black box, leading the attack to outperform previously proposed methods.

With this efficient attack, a comprehensive evaluation of image classifiers is conducted. The empirical robustness of state-of-the-art neural network architectures is compared and contrasted with traditional computer vision pipelines. Recently proposed defenses against adversarial attack are investigated and empirically evaluated, with the result that many approaches do in fact not increase robustness against a strong adaptive attack such as the one developed in this thesis. Finally, the attack is successfully applied to popular cloud-based image processing services, demonstrating the threat to commercial applications in the real world.

The results show that, while some approaches are slightly more robust than others, none of the investigated image classifiers are completely robust in the evaluation setting. This may pose a problem for the safety and security of computer vision systems in the future, since attacks such as the one developed in this thesis can be performed relatively easily. Nevertheless, the experiments show that a select few defense methods do offer tangible robustness benefits. Based on these results, several recommendations for realistic adversarial defense are given.



# Zusammenfassung

In den vergangenen Jahren haben Adversarial Examples viel Aufmerksamkeit von Forschern im Bereich des maschinellen Lernens und des maschinellen Sehens erfahren. Die Tatsache, dass winzige Manipulationen dazu führen können, dass ansonsten robust erscheinende Modelle eine Eingabe falsch klassifizieren, könnte ein großes Sicherheitsproblem darstellen. Allerdings wurden Adversarial Examples bisher meist in Laborumgebungen betrachtet, welche nicht direkt auf real existierende Systeme übertragbar sind.

Diese Dissertation untersucht die Robustheit von Bildklassifikatoren in einem adaptiven Black-Box-Szenario, in dem ein Angreifer keinerlei Wissen über das Innenleben eines Klassifikators hat, jedoch mehrere Anfragen an diesen stellen kann. Theoretisch wäre es möglich, solche Angriffe gegen beinahe jedes Bildverarbeitungssystem durchzuführen, jedoch sind bisherige Verfahren unzuverlässig und leiden unter einer geringen Effizienz. Dies ist ein großes Hindernis, welches bisher einen direkten Vergleich der Robustheit vieler Klassifikatoren und Abwehrmethoden verhindert hat.

Im Rahmen dieser Arbeit wird zunächst ein neuartiger Black-Box-Angriff vorgestellt, welcher sowohl zuverlässig als auch effizient ist. Die Methode nutzt mehrere Arten von Vorwissen über die Domäne der natürlichen Bilder, um die Suche nach Adversarial Examples zu beschleunigen. Dies reduziert die erforderliche Anzahl von Abfragen erheblich, wodurch der Angriff deutlich schneller erfolgen kann als bei etablierten Verfahren.

Dieser effiziente Angriff wird für eine umfassende Robustheitsmessung von Bildklassifikatoren genutzt. Die Robustheit moderner neuronaler Netze wird verglichen und in Relation zu traditionellen Methoden der Bildverarbeitung betrachtet. Danach wird eine Reihe von Abwehrmaßnahmen aus der Literatur evaluiert. Auch hier zeigt sich, dass viele vorgeschlagene Ansätze tatsächlich kaum Schutz bieten. Abschließend wird ein Angriff auf die Bildverarbeitungssysteme mehrerer beliebiger Cloud-Dienstleister demonstriert, was die potenzielle Bedrohung für kommerzielle Anwendungen in der echten Welt aufzeigt.

Die Ergebnisse zeigen, dass einige Methoden zwar etwas robuster sind als andere, allerdings existiert derzeit kein Bildklassifikator, der im untersuchten Szenario vollständig robust ist. Dies könnte ein erhebliches Problem für die Sicherheit bildverarbeitender Systemen darstellen, da solche Angriffe verhältnismäßig leicht durchgeführt werden können. Nichtsdestotrotz zeigen die Experimente auch, dass einige ausgewählte Verteidigungsmethoden die Robustheit eines Klassifikators deutlich erhöhen können. Basierend auf diesen Ergebnissen werden Empfehlungen für den praktischen Einsatz gegeben.



# Contents

<b>Abstract</b>	<b>iii</b>
<b>Zusammenfassung</b>	<b>v</b>
<b>List of Figures</b>	<b>xi</b>
<b>List of Tables</b>	<b>xiii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Adversarial examples . . . . .	1
1.2 Robustness in computer vision . . . . .	2
1.2.1 Image perturbations . . . . .	2
1.2.2 The challenge of generalization . . . . .	5
1.3 The need for robustness . . . . .	5
1.3.1 Safety and security of real-world systems . . . . .	5
1.3.2 Implications for machine learning theory . . . . .	6
1.4 Towards realistic black-box evaluation . . . . .	7
1.5 Thesis outline and contributions . . . . .	8
1.6 List of publications . . . . .	9
<b>2 Adversarial robustness: State of the art</b>	<b>11</b>
2.1 Image classification as a benchmark task . . . . .	11
2.2 Threat setting . . . . .	11
2.2.1 White-box versus black-box . . . . .	11
2.2.2 Advantages of black-box evaluation . . . . .	12
2.2.3 Restricted access . . . . .	13
2.2.4 Discussion . . . . .	14
2.3 Measuring robustness . . . . .	14
2.3.1 Terms and definitions . . . . .	14
2.3.2 Evaluation metrics . . . . .	15
2.3.2.1 Misclassification . . . . .	16
2.3.2.2 Similarity to real examples . . . . .	17
2.3.3 Geometric intuition on adversarial examples . . . . .	20
2.4 Adversarial attacks and defenses . . . . .	22
2.4.1 White-box attacks . . . . .	22
2.4.1.1 Direct optimization attack . . . . .	22
2.4.1.2 Fast Gradient Method (FGM) . . . . .	23
2.4.1.3 Projected Gradient Descent (PGD) . . . . .	24

2.4.1.4	Stronger optimization attacks . . . . .	25
2.4.2	Black-box attacks . . . . .	26
2.4.2.1	Transfer attacks . . . . .	26
2.4.2.2	Gradient estimation . . . . .	28
2.4.2.3	Boundary attacks . . . . .	29
2.4.3	Defense mechanisms . . . . .	31
2.4.3.1	Gradient obfuscation . . . . .	31
2.4.3.2	Ensembling and redundancy . . . . .	31
2.4.3.3	Randomization . . . . .	33
2.4.3.4	Anomaly detection . . . . .	33
2.4.3.5	Pre-processing . . . . .	35
2.4.3.6	Adversarial training . . . . .	35
2.5	Summary . . . . .	36
<b>3</b>	<b>A new and efficient black-box attack</b>	<b>37</b>
3.1	Problem statement and approach . . . . .	38
3.2	Biased sampling with prior knowledge . . . . .	39
3.2.1	Low-frequency perturbations . . . . .	40
3.2.2	Regional masking . . . . .	44
3.2.3	Surrogate gradients . . . . .	45
3.3	Efficient initialization . . . . .	47
3.3.1	Criteria for suitable starting points . . . . .	48
3.3.2	Initialization with salient patches . . . . .	48
3.3.2.1	Segmentation by saliency . . . . .	49
3.3.2.2	Placing salient patches . . . . .	50
3.4	Summary . . . . .	51
<b>4</b>	<b>Evaluation</b>	<b>53</b>
4.1	Setup . . . . .	53
4.1.1	Data set . . . . .	53
4.1.2	Evaluation procedure . . . . .	54
4.1.3	Success criterion . . . . .	55
4.1.4	Choice of classifier under attack . . . . .	55
4.1.5	Choice of surrogate model . . . . .	56
4.1.6	Enforcing the black-box setting . . . . .	57
4.1.7	Software and hardware setup . . . . .	58
4.2	Ablation study of biases . . . . .	59
4.2.1	Setup . . . . .	59
4.2.2	Hyperparameters . . . . .	59
4.2.3	Results . . . . .	59
4.3	Low-frequency bias . . . . .	63
4.3.1	Comparison of frequencies . . . . .	63
4.3.2	Randomized combination of frequencies . . . . .	63
4.3.3	Hyperparameters . . . . .	63

4.3.4	Results . . . . .	63
4.4	Regional masking bias . . . . .	67
4.4.1	Mask strength . . . . .	67
4.4.2	Color channels . . . . .	67
4.4.3	Hyperparameters . . . . .	68
4.4.4	Results . . . . .	68
4.5	Surrogate gradient bias . . . . .	71
4.5.1	Surrogate bias strength . . . . .	71
4.5.2	Surrogate models . . . . .	71
4.5.3	Hyperparameters . . . . .	72
4.5.4	Results . . . . .	72
4.6	Initialization with synthetic starting points . . . . .	76
4.6.1	Creating salient patches of the target class . . . . .	76
4.6.2	Hyperparameters . . . . .	76
4.6.3	Results . . . . .	76
4.7	Comparison with other state-of-the-art attacks . . . . .	80
4.7.1	Gradient estimation attacks . . . . .	80
4.7.2	Boundary attacks . . . . .	82
4.7.3	Transfer attacks (simple surrogate) . . . . .	82
4.7.4	Transfer attacks (strong surrogate) . . . . .	83
4.8	NeurIPS 2018 Adversarial Vision Challenge . . . . .	85
4.9	Summary . . . . .	85
<b>5</b>	<b>Benchmarking the robustness of image classifiers</b>	<b>87</b>
5.1	Setup . . . . .	87
5.1.1	Data set . . . . .	87
5.1.2	Image pre-processing . . . . .	88
5.1.3	Evaluation procedure . . . . .	88
5.1.4	Choice of attacks . . . . .	89
5.1.5	Software and hardware setup . . . . .	90
5.2	Deep learning architectures . . . . .	91
5.2.1	Selection of classifiers . . . . .	91
5.2.1.1	Trained normally . . . . .	91
5.2.1.2	With self-supervised pre-training . . . . .	92
5.2.2	Results . . . . .	93
5.3	Classic computer vision methods . . . . .	96
5.3.1	Simplifying the task: a 10-class subset of ImageNet . . . . .	96
5.3.2	Building ImageNet classifiers with traditional CV methods . . . . .	97
5.3.2.1	Feature extractors . . . . .	97
5.3.2.2	Encodings . . . . .	98
5.3.2.3	Classification methods . . . . .	98
5.3.2.4	Comparison to CNNs . . . . .	100
5.3.3	Results . . . . .	100
5.3.3.1	Accuracy . . . . .	100

## Contents

5.3.3.2	Adversarial robustness . . . . .	102
5.3.3.3	A second look at CNNs: Increasing robustness with more data . . . . .	105
5.4	Defenses against adversarial attack . . . . .	107
5.4.1	Selecting suitable methods . . . . .	107
5.4.2	Description of defenses . . . . .	107
5.4.2.1	Ensembling . . . . .	107
5.4.2.2	Randomization . . . . .	108
5.4.2.3	Anomaly detection . . . . .	108
5.4.2.4	Pre-processing . . . . .	109
5.4.2.5	Adversarial Training . . . . .	110
5.4.3	Evaluation setting . . . . .	111
5.4.4	Results . . . . .	112
5.4.5	Suitability of defenses for real-world application . . . . .	114
5.5	Real-world commercial services . . . . .	116
5.5.1	Setup . . . . .	116
5.5.2	Microsoft Azure . . . . .	117
5.5.3	Amazon Rekognition . . . . .	118
5.5.4	Clarifai . . . . .	119
5.5.5	Google Cloud Vision . . . . .	120
5.6	Summary . . . . .	120
<b>6</b>	<b>Conclusion</b>	<b>121</b>
6.1	Adversarial robustness is not easily achieved . . . . .	121
6.2	Recommendations for real-world systems . . . . .	121
6.3	Limitations and future work . . . . .	122
6.4	Summary of contributions . . . . .	123
	<b>Bibliography</b>	<b>125</b>
	<b>A Listing of evaluation images</b>	<b>137</b>
	<b>B NeurIPS 2018 Adversarial Vision Challenge</b>	<b>140</b>
B.1	Random guessing with low frequency . . . . .	141
B.2	Biased Boundary Attack . . . . .	141



# List of Figures

1.1	Typical adversarial examples . . . . .	2
1.2	Natural perturbations on images of plants . . . . .	4
1.3	Natural perturbations on images of a driving scenario . . . . .	4
1.4	Intuition behind max-margin classification . . . . .	7
2.1	Targeted and untargeted attacks . . . . .	16
2.2	Comparison of similarity measures . . . . .	18
2.3	Comparison of similarity measures (2) . . . . .	19
2.4	Decision boundaries of binary classifiers . . . . .	21
2.5	Three levels of $\epsilon$ -robustness . . . . .	21
2.6	FGM attack example . . . . .	24
2.7	Transferability between multiple classifiers . . . . .	27
2.8	Geometric illustration of a boundary attack . . . . .	29
2.9	Increasing robustness by ensembling . . . . .	32
2.10	Data flow in an ensemble of classifiers . . . . .	33
2.11	Data flow in an anomaly detector . . . . .	34
2.12	Data flow of pre-processing defenses . . . . .	35
3.1	Targeted black-box adversarial examples for an ImageNet classifier . . . . .	37
3.2	Correlations between neighboring in natural images . . . . .	39
3.3	Inspiration for low-frequency perturbations . . . . .	41
3.4	Close-up toy example of low-frequency adversarial perturbation . . . . .	41
3.5	Sampling directions for the low-frequency bias. . . . .	42
3.6	Obtaining colored low-frequency noise . . . . .	43
3.7	Perlin noise with different frequencies . . . . .	43
3.8	Masking based on per-pixel image difference . . . . .	44
3.9	Sampling directions for the surrogate gradient bias. . . . .	46
3.10	Segmentation of starting points . . . . .	49
3.11	Placing salient patches . . . . .	50
3.12	Number of queries versus perturbation magnitude for a Biased Boundary Attack . . . . .	51
4.1	Adversarial perturbations of varying magnitude . . . . .	56
4.2	Ablation study of biases: Success rate against number of queries . . . . .	60
4.3	Ablation study of biases: Perturbation magnitude against number of queries . . . . .	61
4.4	Ablation study of biases: Example images . . . . .	62
4.5	Probability distributions for randomized choice of frequency . . . . .	64
4.6	Noise frequency: Success rate against number of queries . . . . .	65

*List of Figures*

4.7	Noise frequency: Example images . . . . .	66
4.8	Mask strength: Success rate against number of queries . . . . .	69
4.9	Mask strength: Examples of color and grayscale masks . . . . .	70
4.10	Surrogate strength: Success rate against number of queries . . . . .	73
4.11	Surrogate models: Success rate against number of queries . . . . .	74
4.12	Surrogate strength: Example images . . . . .	75
4.13	Starting points generated by salient patches . . . . .	77
4.14	Initialization methods: Success rate against number of queries . . . . .	78
4.15	Initialization methods: Example images . . . . .	79
4.16	Comparison with other attacks . . . . .	81
4.17	Comparison with other attacks, using a stronger surrogate model . . . . .	84
5.1	Robustness of deep learning classifiers (model-free attack) . . . . .	95
5.2	Robustness of deep learning classifiers (strong attack) . . . . .	95
5.3	Max-margin classification in Support Vector Machines . . . . .	99
5.4	Classification accuracy of traditional computer vision pipelines . . . . .	101
5.5	Robustness of traditional computer vision pipelines (model-free attack) . .	104
5.6	Robustness of traditional computer vision pipelines (strong attack) . . . .	104
5.7	Robustness of adversarial defense methods (strong attack) . . . . .	113
5.8	Making pedestrians disappear for Microsoft Azure . . . . .	117
5.9	Making pedestrians disappear for Amazon Rekognition . . . . .	118
5.10	Making pedestrians disappear for Clarifai . . . . .	119
5.11	Making pedestrians disappear for Google Cloud Vision . . . . .	120
B.1	Adversarial examples generated with different biases in the targeted at- tack submission to the NeurIPS 2018 Adversarial Vision Challenge . . . .	142

# List of Tables

4.1	Popular computer vision data sets for machine learning . . . . .	54
4.2	Ablation study of biases . . . . .	60
4.3	Comparison of different noise frequencies . . . . .	65
4.4	Comparison of different mask strengths . . . . .	69
4.5	Comparison of color and grayscale masks . . . . .	70
4.6	Comparison of different surrogate strengths . . . . .	73
4.7	Comparison of surrogate models . . . . .	74
4.8	Comparison of initialization methods . . . . .	78
4.9	Comparison with recently proposed label-only black-box attacks . . . . .	81
4.10	Comparison with transfer attacks when using a stronger surrogate . . . . .	84
5.1	Robustness of deep learning classifiers (model-free attack) . . . . .	94
5.2	Robustness of deep learning classifiers (strong attack) . . . . .	94
5.3	Classification accuracy of traditional computer vision pipelines . . . . .	101
5.4	Robustness of traditional computer vision pipelines (model-free attack) . . . . .	103
5.5	Robustness of traditional computer vision pipelines (strong attack) . . . . .	103
5.6	Improving CNN robustness with more data (model-free attack) . . . . .	105
5.7	Improving CNN robustness with more data (strong attack) . . . . .	105
5.8	Robustness of adversarial defense methods (strong attack) . . . . .	113
5.9	Comparison of the positive and negative aspects of defense methods . . . . .	115
A.1	The evaluation data set used in Chapters 4 and 5. . . . .	139
B.1	Random guessing in the NeurIPS 2018 Adversarial Vision Challenge . . . . .	141
B.2	Configuration of the Biased Boundary Attack in the NeurIPS 2018 Adversarial Vision Challenge . . . . .	141



# 1 Introduction

Machine learning and especially deep learning techniques have been wildly successful across a broad range of computer vision tasks. Yet, they are known to produce significant errors when faced with inputs that are unexpected or maliciously crafted by an attacker. This Chapter introduces adversarial examples, the notion of robustness to such inputs, and discusses why it is required to make machine learning systems safe and secure. The need for a comprehensive black-box evaluation of classifiers is motivated, which is a gap in the current literature. Finally, the main contributions are summed up, together with an outline of the thesis.

## 1.1 Adversarial examples

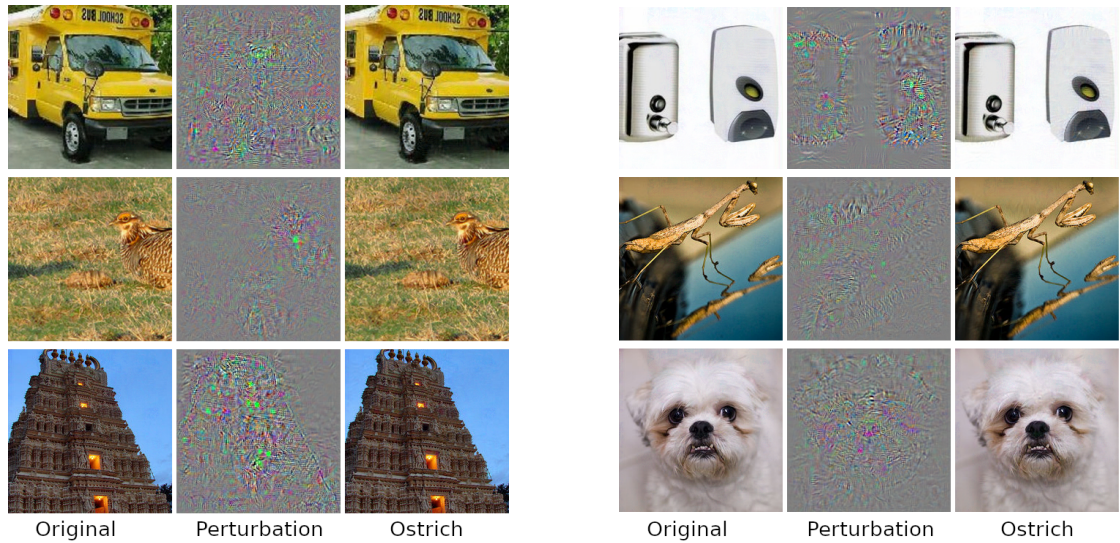
Ever since the term was coined by Szegedy et al. in 2014 [2], the startling phenomenon of adversarial examples has been the subject of heated discussion in machine learning research. The fact that tiny perturbations can lead otherwise robust-seeming models to make drastic mistakes could pose a major problem for safety and security. But how real is this threat? Machine learning is enjoying widespread adoption. How robust are systems that are in use today? Is it truly a danger that needs special attention from engineers, or is it merely a curious phenomenon that is limited to the laboratory?

In general, an adversarial example is an input to a system, specifically designed by an attacker to maximize an error function [2]. This confronts the system with a worst-case situation, causing it to make catastrophic and often surprising mistakes. While adversarial examples are a general phenomenon of machine learning, it can be helpful to focus on a specific domain for purposes of illustration.

**Adversarial examples in computer vision.** This thesis focuses on computer vision, where adversarial examples are images whose pixels have been slightly perturbed, causing the system to misbehave. At the same time, the perturbation is small enough that a human observer either does not notice it or otherwise interprets it as meaningless noise. Consider Figure 1.1 (top left): Here, an image of a school bus is modified so that an image classifier reports "ostrich" with high confidence. All images in the Figure are manipulated so that a machine learning classifier sees nothing but ostriches. This may seem surprising to humans – not only is the image difference hardly noticeable, but the image classifier (a Convolutional Neural Network) is otherwise known for high accuracy! Clearly, something is wrong.

It should be noted that adversarial examples exist in other domains as well, such as speech-to-text audio processing [3], text understanding [4] or even reinforcement learning [5]. They are a general problem of machine learning, and most of the findings presented in this thesis are applicable to those domains as well. However, computer vision is the

## 1 Introduction



**Figure 1.1:** Typical adversarial examples. (Left): Original images, which are predicted correctly by a Convolutional Neural Network. (Center): Difference between original and adversarial examples, magnified  $\times 10$ . Since image differences can be negative, they are displayed against a gray background. (Right): Adversarial examples. The perturbation is almost invisible to the human eye, yet all images are strongly misclassified as ostrich. The classifier being fooled is AlexNet [1]. Taken from Szegedy et al. [2].

most representative domain for adversarial examples, as it is arguably one of the largest fields where machine learning has been deployed so far [6]. There exists a broad range of commercial applications, not only in cloud services and social networks but also in robotics. Therefore, it serves as an ideal domain in which to study adversarial examples.

## 1.2 Robustness in computer vision

Robustness is typically defined as "the degree to which a system or component can function correctly in the presence of invalid inputs or stressful environmental conditions" [7]. Being somewhat ambiguous, the term carries different meanings depending on the area of research. For example, robustness exists not only in computer vision and machine learning [8], but also in motion planning and control theory [9]. To avoid any misunderstandings, it is therefore necessary to first describe robustness in the context of this work.

### 1.2.1 Image perturbations

In the field of computer vision, robustness is typically investigated in the presence of *perturbation* – the modification of a known test image in order to obtain a new one that

is more difficult to process [10, 11]. The system is then considered robust if it performs correctly on the new, difficult, input.

Ideally, this is achieved by *invariance*: The system should yield the same output for both the original and the perturbed input. Consider, for example, Figure 1.1: An image classifier is certainly expected to be invariant to such tiny perturbations. In more difficult cases, it can also be acceptable to perform *detection*: If a system cannot perform the intended functionality on a perturbed input, it should at least be able to detect the perturbation. In this case, an error can be reported and the system might switch to a different mode of operation – or perhaps initiate a safe shutdown.

Typically, researchers distinguish between *natural* and *adversarial* perturbations:

- **Natural:** This can be any change applied to an image, as long as it occurs naturally. This could be simple image transforms, such as translation and rotation, but also shifts in color and brightness (see Figure 1.2), or even more semantic shifts such as rain or snow (see Figure 1.3).

Crucially, this form of perturbation is known to humans and is therefore *expected* to some degree. A human engineer might expect rainy weather, but a machine learning system might not – due to limited training data. Therefore, it will not be automatically robust to this perturbation, even though a human might expect it. For this reason, machine learning engineers choose models that are naturally invariant to some perturbations (Convolutional neural networks (CNN), for example, are partially invariant to translation [13]), and choose to synthetically augment the training data with others.

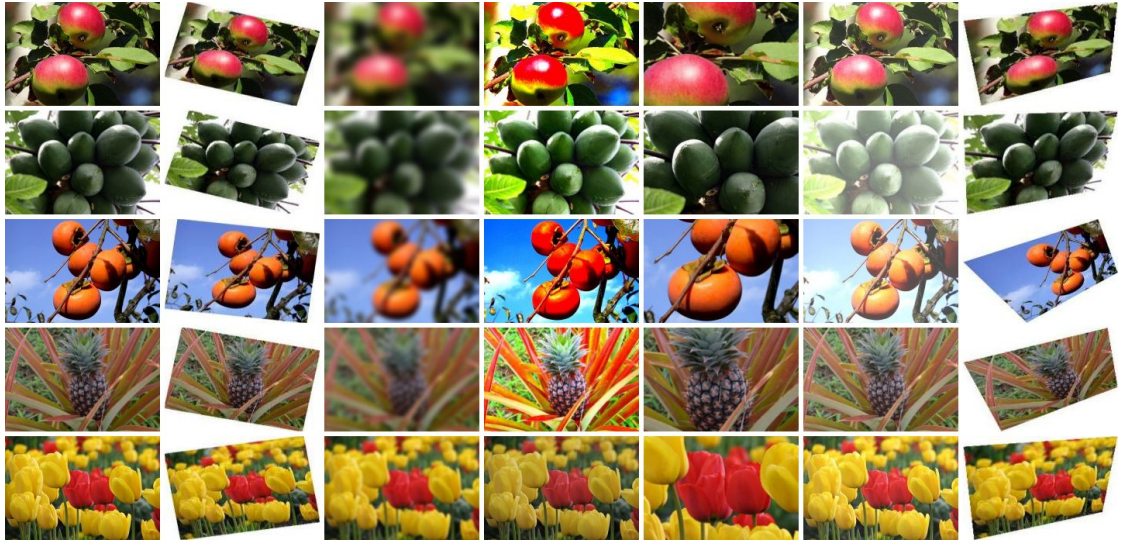
Natural perturbations are chiefly a concern of *safety*. They are expected to appear often, but seldom with malicious intent.

- **Adversarial:** Adversarial perturbations, in contrast, are specifically engineered by an attacker to cause malfunction. They are worst-case inputs which are not expected to appear in a natural environment, and it is therefore difficult to guard against them. However, if they do appear, they have the potential to cause significant damage. Compare Figure 1.1, where an extremely small perturbation causes the system to output a prediction that is completely wrong.

Robustness against adversarial perturbation means that an attacker cannot successfully craft such inputs. Chapter 2 discusses the nature of adversarial examples, shows how to measure them, and compares recent literature on proposed attacks and defenses.

Adversarial perturbations are the focus of this work. They are a concern of *security*, as they are typically crafted by a malicious attacker. However, as they have the potential to cause large amounts of damage, they are by extension also a concern of *safety*.

## 1 Introduction



**Figure 1.2:** Natural perturbations on images of plants. A machine learning system is expected to be robust against changes such as rotation or translation, as well as shifts in brightness and contrast or noise and blurring effects. These image transforms can be expected to occur in a natural environment and are often used in data augmentation schemes [12]. Being robust to these changes, a classifier should report the same result for all images in each row. Taken from Pawara et al. [12].



**Figure 1.3:** Natural perturbations on images of a driving scenario. All images have similar contents (same road) but differ significantly in environmental conditions (weather, day/night). Even though the images look extremely different on a per-pixel level, their semantics are very similar. An automated driving system would be expected to perform the same function, i.e. be robust to these changes. In some cases, there could be a difference – perhaps more careful behavior in darkness – but the system should be largely invariant to cloudy or sunny weather. As a result, such environmental changes are expected and fall under the category of natural perturbations. Taken from Cheng et al. [11].



### 1.2.2 The challenge of generalization

It should be noted that both types of robustness deal with the issue of generalization. Modern computer vision systems are fed with large amounts of training images and extract statistical patterns from them. These statistics then generalize, which means that they can be used to perform accurate predictions on new, unseen data. This is the core premise of machine learning [8].

However, in practice, this generalization has been limited. Currently, automated driving systems are not inherently robust to natural perturbations like rain, snow, and darkness [11]. CNNs are relatively robust to translation, but not to other affine transforms such as rotation [14]. But more importantly still, all neural network architectures known today are vulnerable to adversarial perturbation [15, 16, 17, 18].

Robustness issues arise whenever a predictor generalizes in an unintended fashion – adversarial examples are testament to this. To remedy the problem, it would therefore suffice to simply “solve” generalization. Certainly, a machine learning system that perfectly mimics human vision and understanding would achieve sufficient robustness against both natural and adversarial perturbation.

As it stands today, solving the problem of generalization seems to be out of reach for many years to come [19]. In the meantime, it is reasonable to concentrate on the machine learning systems of today and to try and improve their robustness step by step.

## 1.3 The need for robustness

This work is motivated from two directions. Not only is robustness an important property of safe and secure cyber-physical systems, but it also holds a special place in machine learning theory, where advances in robustness allow new insights into the nature of generalization.

### 1.3.1 Safety and security of real-world systems

For some time, adversarial examples were considered a mostly theoretical problem. They were seen as difficult to create and early techniques (such as [2, 20]) were only suitable in white-box scenarios: Full knowledge of the system under attack was required, as well as full control over all inputs to the system. For reasons such as this, it was often considered impossible to perform practical attacks on real-world robotic systems [21].

**Case study: Attacks on automated driving systems.** As a motivating example, consider the domain of autonomous driving. Already today, many driver assistance systems employ computer vision systems to a certain extent [22], and companies such as Waymo and Tesla are known to rely heavily on machine learning solutions [23]. While it is known that such technology is vulnerable to adversarial examples, these companies have at times stated that they do not consider it a serious problem, as an attacker would be unlikely to gain enough information to mount a successful attack against systems employing high levels of redundancy [24].

## 1 Introduction

This approach is commonly called "security through obscurity" [25]. In short, it describes the usage of a system that has known vulnerabilities, but it is assumed that an attacker would never gain enough information to actually find them. This strategy is often used, but it is considered brittle: Attackers can be very creative, and under certain circumstances manage to do the seemingly impossible.

Consider the case of Tesla: In 2019, a group of security researchers from Tencent Keen Security Lab demonstrated an attack on the Tesla Autopilot driver assistance system, which performs lane detection mainly through computer vision [26]. By placing specifically crafted markers on the road, they were able to fool the lane detection system, prompting the car to change lanes into potentially oncoming traffic.

It should be noted that the researchers had to go to great lengths to identify vulnerabilities in the computer vision system. They used a number of zero-day exploits to gain access to the vehicle's operating system and then reverse-engineered its perception pipeline until they found the lane detection component. Only then were they able to perform an adversarial attack with methods from contemporary literature [27].

However, with the vulnerabilities of the computer vision system now identified, it is within reason to assume that similar adversarial markers could have fooled other vehicles with the same software as well. In this way, a seemingly unlikely attack may quickly scale to a worldwide problem.

**Towards robust real-world computer vision.** In the public debate on robustness, opinions range from "adversarial examples are no problem" to "machine learning can never be safe". Quite likely, the answer is somewhere in between. This work aims to uncover some of the details: By performing experiments with novel strong attacks in a realistic threat setting, it evaluates the robustness of image classifiers and defense strategies. From these insights, it becomes possible to develop recommendations and best practices for designing robust real-world systems.

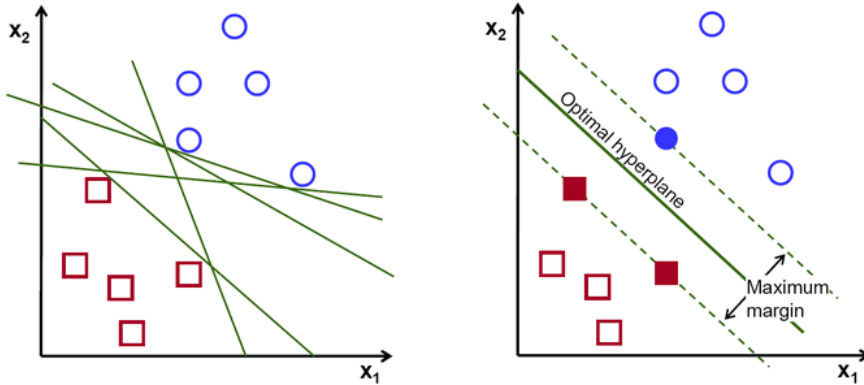
### 1.3.2 Implications for machine learning theory

In addition to practical considerations, the existence of adversarial examples also poses a dilemma for machine learning theory. Consider the example of Support Vector Machines (SVMs): Before the current wave of deep learning, they were considered not only the state of the art in machine learning but also superior to deep neural networks because of their solid foundations in statistical learning theory [28, 29].

In order to achieve good generalization, SVMs work with the concept of max-margin separation: Considering a classifier that predicts one of two classes for each data point, there exists a decision boundary that separates the two classes of data points. Figure 1.4 illustrates this: In order to achieve good generalization, this boundary is picked in a way that maximizes its distance to the nearest data points. This automatically improves robustness, as an adversarial perturbation must be very large to push a data point across the decision boundary.<sup>1</sup>

---

<sup>1</sup>Section 2.3.3 of Chapter 2 further discusses the relationship between decision boundaries and robustness.



**Figure 1.4:** Intuition behind max-margin classification. Two classes (blue circles and red squares) are linearly separated with a hyperplane. (Left): There exist many ways to perform classification with perfect accuracy, but not all of them have a large margin. (Right): An SVM picks the hyperplane that maximizes the distance (margin) between data points of different classes. Motivated by statistical learning theory, this principle of max-margin classification is thought to guarantee good generalization and robustness [28]. Such theoretical guarantees currently do not exist for deep neural network classifiers, which adds to the popular belief that classic methods could be more robust. Figures taken from Garcia et al. [30].

As a result, one would expect such classifiers to display significant robustness to adversarial attack. But is this true in practice? For any claim of robustness, adversarial examples are counter-examples. For any classifier, they are empirical evidence that the margin is small. Surely, theoretically-grounded methods such as the SVM would then be robust by design?

An experiment conducted in Chapter 5 will show that this assumption does not hold, at least for high-dimensional problems of computer vision. Here, SVMs and other tried-and-true machine learning methods may be just as vulnerable to adversarial attack as deep neural networks.

## 1.4 Towards realistic black-box evaluation

So far, the majority of research into adversarial examples has been conducted in unrealistic settings. While there exists a large body of literature on robustness, at this time it is lacking a comprehensive side-by-side evaluation of state-of-the-art methods in a fair but realistic threat setting.

Most works evaluate adversarial examples in white-box settings, where an attacker has full knowledge of the system under attack. That line of research focuses on the theoretical existence of a vulnerability, giving little regard to its practicability. In the real world, however, attackers typically do not have access to the exact specifications (and parameters) of a system and therefore operate under a black-box scenario. Chapter 2 discusses these threat settings in detail, as well as the state of the art in literature.

## 1 Introduction

Black-box attacks are not only realistic but also flexible. Since white-box methods are typically handcrafted with knowledge of the target system, black-box methods cannot afford this luxury. As a result, they need to work on *any* system, regardless of its inner workings. This means that black-box attacks can not only be applied easily to many systems, but also allow for a direct side-by-side comparison of heterogeneous systems and defenses.

With all these advantages, it seems surprising that black-box attacks are still relatively obscure. The reason is simple: Current methods either have a low success rate or, alternatively, succeed but require a high amount of computational power (and many queries to the black box). This makes evaluations tedious, if not completely infeasible.

This work aims to fill the gap by proposing a novel adaptive black-box attack, which is significantly more efficient than the previous state of the art. With this new method, a fair comparison of state-of-the-art approaches and systems becomes possible.

### 1.5 Thesis outline and contributions

The thesis consists of 6 chapters:

- Chapter 1 has outlined the problem of adversarial examples and motivated the need for systematic black-box evaluations of adversarial robustness in computer vision.
- Chapter 2 introduces the necessary terms and definitions for an investigation of adversarial robustness. Relevant threat models are discussed, together with the current state of the art in attacks and defenses. This equips the reader with the necessary background to understand recent developments and the limitations of existing approaches.
- Chapter 3 introduces a novel approach for generating black-box adversarial examples, the Biased Boundary Attack. This method can utilize prior knowledge about the domain of natural images to perform efficient attacks. A number of such biases are proposed, together with an efficient strategy for initialization.
- In Chapter 4, an experimental evaluation of the new attack method is performed. This includes ablations on the various biases and their parameters, as well as a comparison with recent works on black-box adversarial attack. It is found that the Biased Boundary Attack significantly outperforms the previous state of the art both in terms of efficiency and reliability.
- Chapter 5 uses the new attack to benchmark the adversarial robustness of a broad range of image classifiers. First, state-of-the-art deep learning architectures are evaluated. Next, classic computer vision pipelines are investigated, and it is found that they share the same vulnerabilities. A range of proposed defenses against adversarial attacks is compared side-by-side, with the result that most do not increase robustness as significantly as claimed. Finally, the attack is also performed

on popular commercial image processing services from Microsoft, Amazon, Google and Clarifai. In all cases, it is possible to successfully create adversarial examples.

- Chapter 6 provides a summary of all results and discusses their implications. Finally, practical recommendations for increasing the robustness of computer vision systems are given, together with directions for future work.

## 1.6 List of publications

The following publications resulted from research performed during the pursuit of the doctoral degree. Out of these, [1-4] address the content of this thesis:

1. Thomas Brunner and Alois C. Knoll. Comparing the adversarial robustness of image classifiers and defenses against adaptive black-box attacks. *Under Review*, 2021.
2. Wieland Brendel, Jonas Rauber, Alexey Kurakin, Nicolas Papernot, Behar Velicki, Sharada P. Mohanty, Florian Laurent, Marcel Salathé, Matthias Bethge, Yaodong Yu, Hongyang Zhang, Susu Xu, Hongbao Zhang, Pengtao Xie, Eric P. Xing, Thomas Brunner, Frederik Diehl, Jérôme Rony, Luiz Gustavo Hafemann, Shuyu Cheng, Yinpeng Dong, Xuefei Ning, Wenshuo Li, and Yu Wang. Adversarial Vision Challenge. In *The NeurIPS '18 Competition*, pages 129–153. Springer, Cham, 2020.
3. Thomas Brunner, Frederik Diehl, Michael Truong Le, and Alois C. Knoll. Guessing smart: Biased sampling for efficient black-box adversarial attacks. In *IEEE International Conference on Computer Vision, ICCV*, pages 4957–4965. IEEE, 2019.
4. Thomas Brunner, Frederik Diehl, and Alois C. Knoll. Copy and paste: A simple but effective initialization method for black-box adversarial attacks. *arXiv preprint arXiv:1906.06086*, 2019.
5. Chih-Hong Cheng, Chung-Hao Huang, Thomas Brunner, and Vahid Hashemi. Towards safety verification of direct perception neural networks. In *Design, Automation & Test in Europe Conference & Exhibition, DATE*, pages 1640–1643. IEEE, 2020.
6. Thomas Brunner, Frederik Diehl, Michael Truong Le, and Alois C. Knoll. Leveraging semantic embeddings for safety-critical applications. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPRW*, pages 1389–1394. IEEE, 2019.
7. Frederik Diehl, Thomas Brunner, Michael Truong Le, and Alois C. Knoll. Graph neural networks for modelling traffic participant interaction. In *IEEE Intelligent Vehicles Symposium, IV*, pages 695–701. IEEE, 2019.

## 1 Introduction

8. Tobias Kessler, Julian Bernhard, Martin Büchel, Klemens Esterle, Patrick Hart, Daniel Malovetz, Michael Truong Le, Frederik Diehl, Thomas Brunner, and Alois C. Knoll. Bridging the gap between open source software and vehicle hardware for autonomous driving. In *IEEE Intelligent Vehicles Symposium, IV*, pages 1612–1619. IEEE, 2019.
9. Michael Truong Le, Frederik Diehl, Thomas Brunner, and Alois C. Knoll. Uncertainty estimation for deep neural object detectors in safety-critical applications. In *IEEE International Conference on Intelligent Transportation Systems, ITSC*, pages 3873–3878. IEEE, 2018.

## 2 Adversarial robustness: State of the art

This chapter outlines the state of the art in robustness research. First, ways to define and measure robustness are discussed, together with an appropriate threat setting. Then, prior work on adversarial attacks is presented. This includes seminal works as well as recent developments in adversarial attacks, some of which form the basis for the efficient black-box attack developed in Chapter 3. Finally, a number of defense mechanisms and mitigation strategies are presented, along with a discussion of their strengths and weaknesses.

### 2.1 Image classification as a benchmark task

Attacks on computer vision systems can take many forms and are highly dependent on the task which the system is performing. To conduct a systematic analysis, it is necessary to pick a task that can be isolated and performed in a controlled environment, and which is at the same time relevant for real-world applications.

The natural choice here is image classification, in which a system receives a single input image and returns as output a class label. This problem is conceptually simple and has been a staple of computer vision and machine learning research for decades [8]. As a consequence, most recent analyses of machine learning robustness have been performed on this task, yielding a large body of prior work [2, 20, 31, 32, 33, 34, 15]. Finally, the image classification task is also directly relevant for real-world applications, as it forms the basis of more advanced computer vision techniques such as object detection and semantic segmentation [35], which are active areas of research for modern computer vision applications [36].

### 2.2 Threat setting

When evaluating the security of a system, one must consider the scenario in which attacks are performed. An adversary typically does not have unrestricted access to real-world systems – if they did, an elaborate adversarial attack would hardly be necessary! Depending on the restrictions, the efficient creation of adversarial examples can be very difficult. It is therefore important to choose a setting that is as realistic as possible, while at the same time not completely prohibitive for an informative evaluation.

#### 2.2.1 White-box versus black-box

In general, most attack scenarios are categorized as either white-box or black-box:

- In a **white-box** scenario, the attacker has full knowledge of the system that is under attack. For machine learning methods such as neural networks, this includes the architecture and parameters (weights) of the model, as well as knowledge of input (resolution) and output (class labels) formats.

White-box attacks (and defenses) typically aim to investigate robustness from a fundamental perspective: A successful white-box defense would entail "perfect" security, as the setting assumes that – even with full knowledge of the system – an attacker can never succeed [37].

Naturally, proving this is very difficult for defenders. One must verify that there exists *no* perturbation of a certain magnitude which can change the output of the classifier. Such formal proofs of robustness are very hard and currently suffer from scalability issues [38].

For attackers, it is much easier: If only a single adversarial example can be produced, then robustness is disproven. Indeed, current developments in adversarial examples are a cat-and-mouse game between attackers and defenders, and newly proposed defenses are often broken within months of being published [39, 16, 40].

- Adversarial attacks in a **black-box** scenario are considerably more difficult: Here, it is assumed that the inner workings of a model are entirely unknown and that an attacker can merely query its output after providing an input. In reality, some small amount of information is typically known even in this case, such as the task which the model performs, some possible outputs, or even correlations in the data on which the model was trained<sup>1</sup>.

The black-box setting does seem easier for defenders: As the attacker has very limited knowledge, it suffices to simply hide vulnerabilities in a clever way. Naturally, this amounts to "security by obscurity", which is of doubtful use as it hardly gives any real guarantees. As described in Section 2.4.3, most recently proposed defense mechanisms rely on obscurity in one way or another. The question is – are they really safe, or could they be broken by stronger attacks?

*Note on terminology:* Strictly speaking, a black box would imply that no knowledge is available at all. Some authors have argued that it would be more formally correct to call the setting "gray-box" instead [41]. Perhaps unsurprisingly, this ambiguous terminology has not caught on in the field. For all practical purposes, the term "white-box" is used when a classifier's architecture and parameters are known, and "black-box" when they are not.

## 2.2.2 Advantages of black-box evaluation

Although both settings are interesting in their own regard, this work focuses on the black-box scenario. This has the following reasons:

---

<sup>1</sup>This observation forms the basis of the efficient black-box attack developed in Chapter 3: Even small amounts of knowledge can drastically improve the success chance of an attack.



- **White-box defenses are difficult to compare:** Since an attacker has complete knowledge of the system, any defense is therefore also known to them. This typically prompts the creation of specialized attacks, which are custom-tailored to penetrate one particular defense but fail against others [16]. As a result, the current landscape of white-box methods is very fragmented and it is difficult to evaluate attacks and defenses in a unified benchmark.
- **Black-box attacks are versatile:** An effective black-box attack can be applied to virtually any system. Since white-box information is not available, these attacks cannot afford specialization, and instead focus on being applicable to as many systems as possible. This allows for a side-by-side evaluation of many different systems – including traditional computer vision pipelines, for which white-box attacks would otherwise be difficult to design.
- **Real-world systems are black boxes:** Finally, black-box attacks can evaluate real-world systems that are in use today. At the time of writing and to the best of this author’s knowledge, no white-box defense exists that is provably robust and that can be scaled to real-world applications (such as high-resolution image classification). Therefore, it is unlikely that large-scale commercial applications could be completely robust. It seems promising to test black-box attacks on them and see how robust they really are.

In short, black-box attacks are the tool of choice for a unified evaluation of a broad range of systems. However, there is one significant drawback: Current black-box attacks are known to be either slow or unreliable, rendering attacks on real-world systems expensive and, in many cases, impossible. This weakness is addressed by the efficient black-box attack developed in Chapter 3, paving the way for the evaluations conducted in Chapter 5.

### 2.2.3 Restricted access

In a black-box setting, attackers can access the inputs and outputs of a system. To make things even harder, this access is often restricted further:

- **Limited queries:** Searching for adversarial examples is an optimization problem, which typically requires many queries to the classifier under attack. State-of-the-art black-box attacks (see Section 2.4) require a large number of such queries to find even a single adversarial example. Naturally, the defender can simply limit the number of queries available to the attacker.

This can be very effective: State-of-the-art attacks that require millions of queries against commercial classifiers [42] are quickly rendered infeasible when access is restricted, for example, to one query per second. Query efficiency therefore is a major requirement for black-box attacks.

- **Decision-based / label-only classification:** Consider a typical machine learning classifier, which outputs a real-valued vector of class probabilities (e.g. normalized by a softmax function). A small change in the input is then reflected by a small change in the output, allowing attackers to perform gradient estimation with finite-difference methods. Gradient information allows for efficient optimization, and some of the most prominent attacks exploit this to quickly find adversarial examples [27, 44, 42].

The remedy is simple: Instead of delivering confidence scores to an attacker, a defender might return only the class label of the final decision. Since the output is now discrete, it greatly diminishes the efficiency of gradient estimation [42], up to the point where such attacks become infeasible. This setting is commonly named "decision-based" [15], "label-only" [42], or "hard-label" [45].

The decision-based setting is very difficult to attack, but it also confers a significant advantage: Considering machine learning systems in this setting allows for the evaluation of any classifier or system that outputs a single, discrete decision. Attacks that operate in this scenario are therefore applicable not only to deep learning, but also to traditional machine learning classifiers, feature extractors, and even decision trees.

#### 2.2.4 Discussion

This work focuses on the query-limited decision-based black-box scenario. It holds great promise for a direct and fair comparison of a large range of heterogeneous classifiers. However, attacks in this scenario must not only demonstrate high success rates but also succeed with a relatively low amount of queries.

### 2.3 Measuring robustness

With the threat setting now defined, a principled way to measure robustness can be established. This section introduces the terminology and general setup of classification and adversarial robustness. Some intuition about decision boundaries is given and, finally, the state of the art in measuring adversarial perturbation is discussed.

#### 2.3.1 Terms and definitions

The following is a brief review of common terminology for machine learning classifiers:

- **Classifiers and models.** In machine learning, a *classifier* function  $f(x)$  maps from a data vector  $x \in \mathbb{R}^k$  to a class label  $y \in \{C_1, C_2, \dots, C_n\}$ . Here,  $k$  is the number of dimensions (or features) of the data point, and  $n$  is the number of classes. This is a common definition of classification in machine learning [8]. For each data point  $x$ , the classifier outputs a prediction  $\hat{y}$ :

$$\hat{y} = f(x) \tag{2.1}$$

Since a classifier is a *model* that is used for prediction, the term *model* is often used informally when referring to  $f$ .

- **Data points, inputs and images.**  $x \in \mathbb{R}^k$  is the input to the classifier and is a vector of dimensionality  $k$ , where each dimension describes a specific feature. It is often called a *data point* or an *example* that is obtained from measurements in the real world (e.g. using a camera).

In computer vision,  $x \in \mathbb{R}^{h \times w \times c}$  is an *image* that is represented not as a flat vector, but as a matrix or tensor.  $h$ ,  $w$  and  $c$  are height, width and the number of color channels. A simple way to use images with machine learning notation is to flatten the image into a vector of  $k = h \cdot w \cdot c$  dimensions.

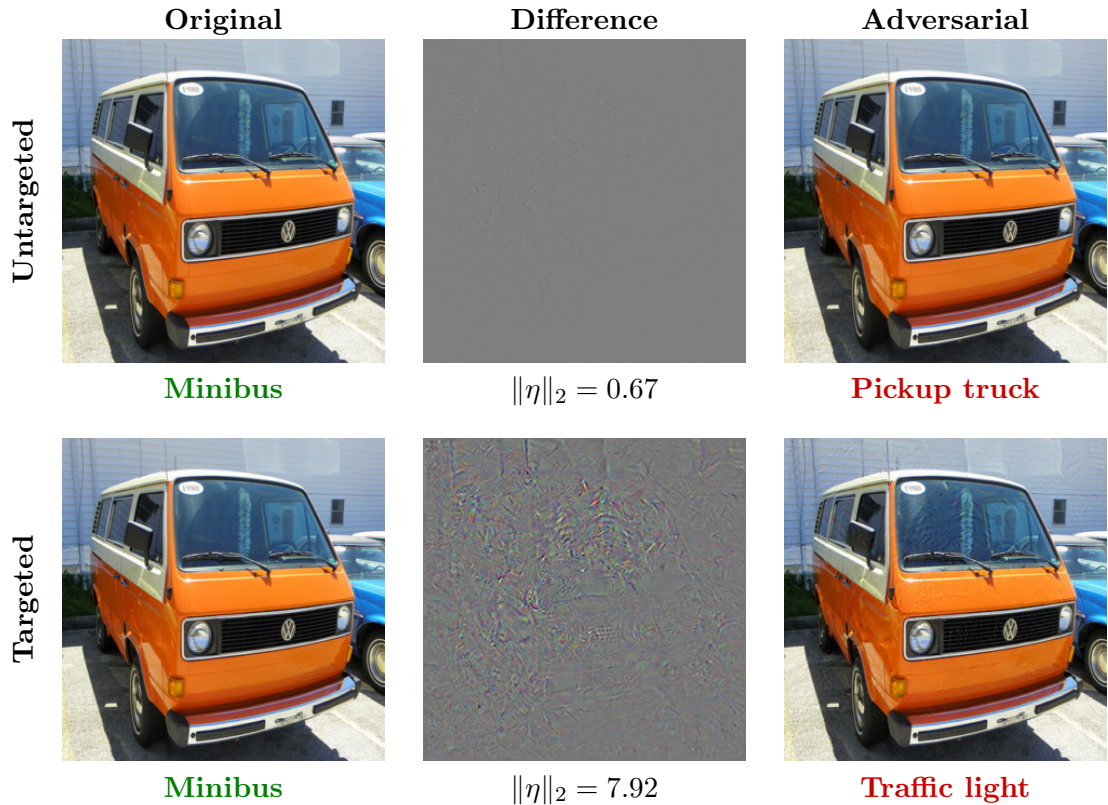
Naturally, this is somewhat oversimplified, as contemporary computer vision classifiers (both CNNs and classic feature extractors) do in fact make use of locality information instead of removing it by flattening. However, the description of inputs as vectors  $x \in \mathbb{R}^k$  allows us to continue with concise machine learning notation. This simplifies many of the aspects of adversarial examples, such as describing decision boundaries or measuring the magnitude of adversarial perturbations. For this reason,  $x$  is simply described as an *input* of dimensionality  $k$ .

- **Input space.** The space spanned by all possible inputs. For inputs  $x \in \mathbb{R}^k$ , the input space is  $\mathbb{R}^k$ .
- **Decision boundary.** In the input space, the decision boundary is a surface that divides it into multiple regions, each of which then contains data points of a different class. See Section 2.3.3 for an illustration.
- **Data set.** In this work, a data set contains pairs  $(x, y_{gt})$  of data points and the corresponding ground truth label. Here, data sets for *supervised learning* are considered exclusively, i.e. that always contain ground truth annotations.
- **Ground truth.** The correct class label of an image which is provided in the data set. The prediction of a classifier is correct only if it matches the *ground truth* class label:  $\hat{y} = y_{gt}$ . Ground truth annotations are not directly required for adversarial attacks, but they are useful for evaluation purposes.

### 2.3.2 Evaluation metrics

Classifiers are most commonly evaluated by prediction accuracy on predefined data sets [8]. For each example  $x$ , the data set contains a ground truth label  $y_{gt}$ . The classifier  $f$  predicts the label  $\hat{y} = f(x)$ , and success is registered if  $f(x) = y_{gt}$ . The overall accuracy then is the percentage of correct predictions out of all examples.

Robustness, in contrast, can be considered the accuracy of a classifier on a set of perturbed data. When benchmarking the robustness of a classifier, an attacker typically chooses a data set, creates adversarial copies of its examples, and then observes the change in accuracy [2, 20, 33]. For simple data such as MNIST [46], the entire test set



**Figure 2.1:** Comparison of untargeted and targeted misclassification. (Top): An untargeted attack, which is considered successful whenever the predicted class label changes. This attack is easier and the result is often of little concern – the predicted class is still an automobile. (Bottom): A targeted attack, with  $y_{target}$  manually chosen to be “traffic light”. This attack is more difficult, and a larger perturbation is required. Targeted attacks give the attacker fine-grained control and have the potential to cause significant damage. The classifier being fooled is InceptionV3 [47], trained on the ImageNet dataset [48].

can be evaluated [39]. For larger and more complex data sets such as ImageNet [77] this can be computationally infeasible – it is therefore common to evaluate only a small subset of the validation data [42]. But what exactly are the criteria that make an image “adversarial”?

### 2.3.2.1 Misclassification

The first and obvious criterion is misclassification – this means that the classifier must produce incorrect output. In the following, the original example is named  $x_{orig}$  and the adversarially perturbed copy is named  $x_{adv}$ . Research on adversarial examples has so far focused on two forms of misclassification:

- **Untargeted:** Formally, misclassification is achieved when  $f(x_{adv}) \neq y_{gt}$ . In other words, any prediction which is not the ground truth label satisfies the criterion.

Untargeted attacks have a major drawback: Depending on the distribution of class labels, there might be significant overlaps. Consider, for example, the popular ImageNet [48] data set: It contains 1000 different classes, among which are many similar types of road vehicles. Here, an untargeted attack typically changes the prediction to another class that is only marginally different. Compare Figure 2.1 (top): Causing a classifier to mistake a minibus for a pickup truck satisfies the misclassification criterion, but in reality it is a minor mistake at best.

This effect can be seen in recent works on efficient black-box attacks [49, 50], where evaluations are conducted in untargeted settings on data sets with ambiguous class labels – this is not a strong demonstration of vulnerability.

- **Targeted:** To remedy this problem, the attacker can instead choose a specific target label  $y_{target}$ , and adversarial misclassification is achieved only if  $f(x_{adv}) = y_{target}$  (and  $y_{target} \neq y_{gt}$ ).

Ideally, this target label is far removed from the true label and implies a dangerous mistake. See Figure 2.1 (bottom): Confusing a minibus with traffic lights is a serious concern for safety. Naturally, this makes targeted attacks more difficult to perform, but also more critical: If successful, then the attacker has very fine-grained control over the outcome.

Targeted adversarial examples are both more dangerous and difficult to create. Since robustness is typically considered in worst-case scenarios, targeted attacks should be the method of choice – with  $y_{target}$  aptly chosen to demonstrate the adversarial effect. This work therefore concentrates on targeted attacks exclusively.

### 2.3.2.2 Similarity to real examples

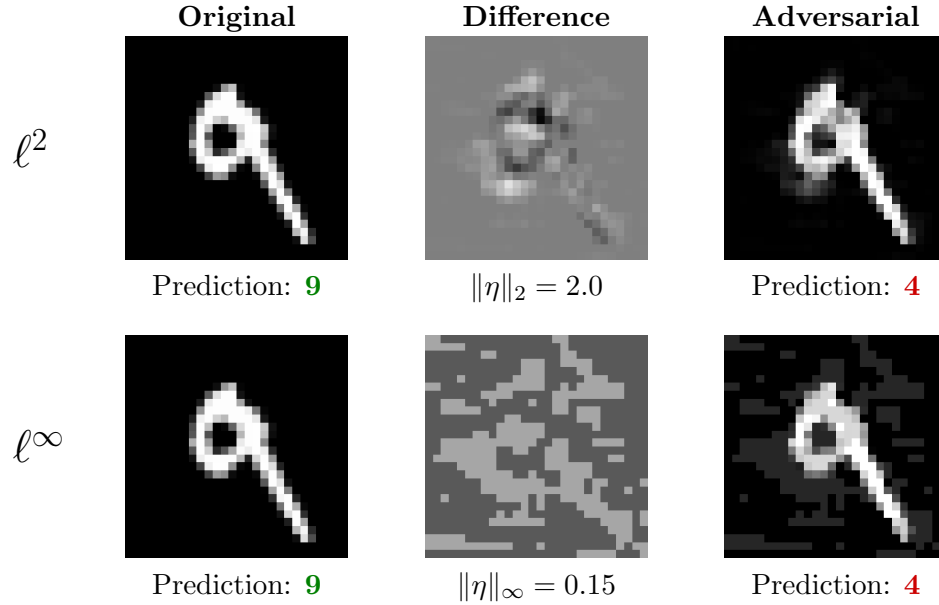
The second criterion is similarity: The adversarial image should be as close to the original as possible. This results in perturbations that not only cause a machine learning system to malfunction, but that are often imperceptible to humans [2]. How can this similarity be measured in a principled way?

- **Euclidean distance** ( $\ell^2$ -norm of the perturbation):

An adversarial example is created by perturbing a clean example:  $x_{adv} = x_{orig} + \eta$ . Then,  $\|\eta\|_2$  is the  $\ell^2$ -norm of the perturbation, or the Euclidean distance between  $x_{orig}$  and  $x_{adv}$ .

Effectively embedding all data points into the Euclidean space, this simple way of measuring similarity has long been a staple in machine learning and statistics [8]. Being a sum of squares, the  $\ell^2$  distance is also directly equivalent to the mean squared error (MSE) of individual features:

$$MSE = \frac{\|\eta\|_2^2}{k} \quad (2.2)$$



**Figure 2.2:** Adversarial examples that minimize different distance measures. (Top): A small number of pixels is perturbed strongly. (Bottom): Many pixels are perturbed, but none by more than exactly 0.15. While  $\ell^\infty$  seems convenient as a measurement,  $\ell^2$ -perturbations have a more organic look and can express patterns that are harder to spot by humans.

Here,  $k$  is the dimensionality of the data points. If the input is an image, then  $k = h \cdot w \cdot c$ , where  $h$  is height,  $w$  width, and  $c$  the number of color channels. MSE is a popular way to measure image corruption in computer vision and image processing [51], which makes the  $\ell^2$ -distance a natural fit for measuring adversarial perturbations.

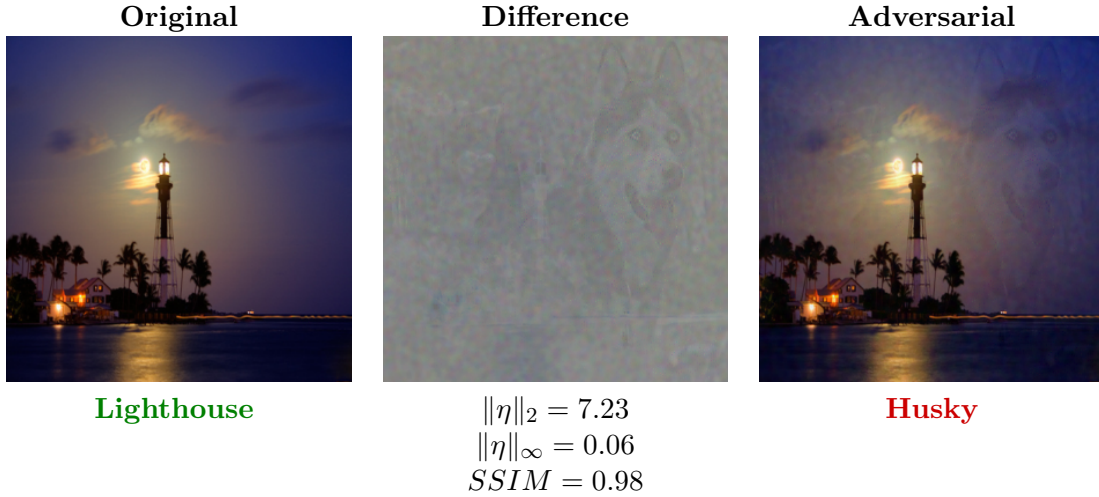
- **Chebyshev distance** ( $\ell^\infty$ -norm of the perturbation):

The  $\ell^\infty$ -norm returns the maximum element of a vector, like so:

$$\|\eta\|_\infty = \max\{|\eta_1|, \dots, |\eta_k|\} \quad (2.3)$$

This measure is used to limit the maximum perturbation that an attacker can perform on any feature. When comparing images,  $\|\eta\|_\infty = 0.1$  would indicate that no pixel in the image has been modified by more than  $\pm 0.1$  (or 26 out of 255 in an 8-bit RGB image).

The  $\ell^\infty$ -norm has been popular in the early days of adversarial example research [20, 52]. However, it has an important drawback: Attacks that minimize it typically result in equal perturbation of most pixels. Compare Figure 2.2 (bottom), where a weak perturbation equally covers the entire image. The  $\ell^2$ -based attack in Figure 2.2 (top) is more flexible: It can choose to focus on fewer pixels and perturb them more strongly, while still appearing very similar to the original image.



**Figure 2.3:** Similarity measures are not always aligned with human perception. This ”adversarial” perturbation is clearly visible to humans – an image of a husky dog has been blended into the original image – but all measures report high similarity. This is not surprising in the case of  $\ell^p$ -distance, but more so in the case of SSIM: The two images have an almost perfect SSIM score (1.00 is the maximum). Unfortunately, this means that established measures for perception-based similarity often do not apply to adversarial examples, and therefore do not automatically offer advantages over the simpler  $\ell^p$ -based measures.

- **Number of perturbed pixels** ( $\ell^0$ -pseudonorm of the perturbation):

While formally not a norm, the  $\ell^0$ -measure simply counts the number of nonzero elements in a vector. In simplified terms, it is the number of pixels that an attacker needs to change in order to fool the classifier:

$$\|\eta\|_0 = |\{i : |\eta_i| > 0\}| \quad (2.4)$$

Here, the actual strength of the perturbation is not taken into account. This makes the  $\ell^0$ -distance the direct opposite of the  $\ell^\infty$ -distance, where the number of perturbed pixels is irrelevant.

This measure is rather exotic and not in widespread use. It has been used for spectacular demonstrations such as a one-pixel-attack [53], but this form of attack is rather difficult [39] and the criterion (perturb as few pixels as possible) can also be satisfied by the more flexible  $\ell^2$ -distance.

- **Perception-based measures:**

The overarching goal of an adversarial attack is to perform manipulations that can not be detected by a human. It is well known that  $\ell^p$ -norms, while mathematically convenient, do not align too closely with human perception [54]. This drawback is evident in Figure 2.3, where a perturbation with small  $\ell^2$ - and  $\ell^\infty$ -norms is clearly visible.

It is tempting to turn towards the field of image processing, where perception-based similarity measures have a rich history. Scores such as Structural Similarity (SSIM) [54] and Visual Information Fidelity (VIF) [55] are commonly used to measure corruption artifacts caused by image and video compression. While these seem very promising, they are of doubtful use for measuring adversarial examples: See again Figure 2.3, where SSIM does not perform any better than  $\ell^p$ -distances at capturing the perturbation.

Another approach towards perception-based measures is the Fréchet Inception Distance (FID) [56]. It is used for generative models such as Generative Adversarial Networks (GANs) [57], judging similarity between synthetic and real images. While this seems promising, it should be noted that FID is calculated from features extracted by a neural network which then, naturally, must be vulnerable to adversarial examples as well. An attacker would likely not only fool the classifier but also the distance measure with it. This would make such a measure very unreliable.

Which measure should one choose?  $\ell^p$ -norms are mathematically convenient, but their weakness is limited alignment to human perception. Nevertheless, clearly any perceptual difference is minimized when  $\ell^p$ -norms approach zero, so they are indeed useful when measuring very small perturbations – which is precisely the setup of adversarial examples. For this reason, they remain relevant despite their flaws and are to this day the main way of measuring adversarial robustness [34, 33, 15, 18, 27, 44]. Here, the  $\ell^2$ -distance is the most flexible, as it incorporates both the number of perturbed pixels and the perturbation strength. Therefore, the rest of this work focuses on  $\ell^2$ -distance exclusively.

### 2.3.3 Geometric intuition on adversarial examples

To gain intuition about why adversarial examples exist, it can be helpful to consider a toy example with two input features only ( $x \in \mathbb{R}^2$ ). This allows us to create a two-dimensional plot of the entire input space, visualizing the decision boundary that divides it into regions of different classes.

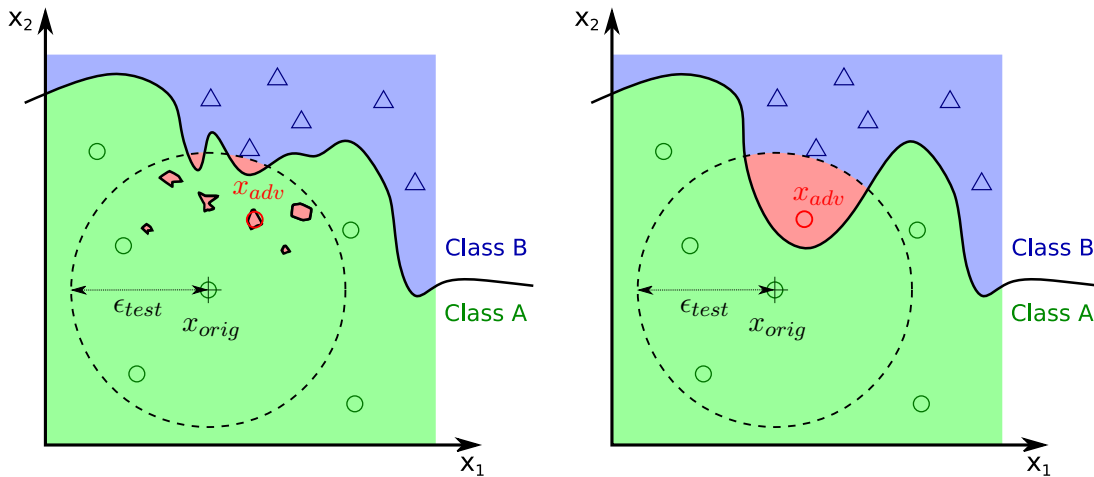
Figure 2.4 shows the input space of binary classifiers that process two features  $x_1$  and  $x_2$  (e.g. tiny images with only two pixels). Their robustness is evaluated in a testing scenario: If a classifier is robust to all perturbations of magnitude  $\|\eta\|_2 \leq \epsilon_{test}$ , then there exists a circle of radius  $\epsilon_{test}$  around each data point and all points within this circle are of the same class. Correspondingly, in higher dimensions, this is the *robustness hypersphere* around a data point<sup>2</sup>. Adversarial examples therefore exist wherever the decision boundary intersects the robustness hypersphere.

**Adversarial regions are not small.** It was long thought that adversarial examples were "low-probability pockets on the data manifold" [2], which lends credibility to the interpretation in Figure 2.4 (left). Since these pockets are so small, it would be easy

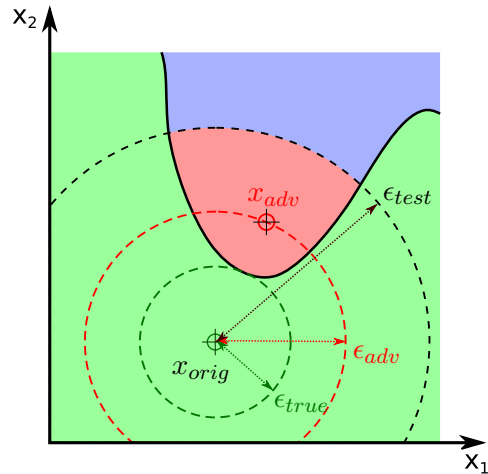
---

<sup>2</sup>Since we use the  $\ell^2$ -distance for robustness, this is simply a Euclidean (hyper-)sphere. In the case of  $\ell^\infty$ , it would be a (hyper-)cube. Other measures, not being true metrics, do not allow for this simple intuition.





**Figure 2.4:** Decision boundaries of binary classifiers. The red regions contain adversarial examples, which arise wherever the decision boundary is inside the robustness hypersphere. This means that a classifier is not robust to perturbations of magnitude  $\epsilon_{test}$ . (Left): A common misconception, according to which adversarial examples would reside in tiny, isolated pockets. These would be difficult to find by an attacker and easy to remove by a defender. (Right): According to recent research [34], adversarial regions are often large and contiguous. This makes them rather easy to find for an attacker, and difficult to remove without hurting classification accuracy.



**Figure 2.5:** Three levels of  $\epsilon$ -robustness. The true robustness  $\epsilon_{true}$  is the distance from  $x_{orig}$  to the decision boundary. However, calculating  $\epsilon_{true}$  is extremely expensive and often intractable for complex classifiers [38]. Instead, robustness can be determined empirically by testing: A robustness target  $\epsilon_{test}$  is fixed, and if an adversarial example  $x_{adv}$  with  $\epsilon_{adv} \leq \epsilon_{test}$  can be found, then the classifier is proven to be not robust. As an approximation,  $\epsilon_{adv}$  is an empirical *upper bound* for the true robustness  $\epsilon_{true}$  of the classifier.

to get rid of them by filtering, or perhaps by adding small amounts of random noise. However, over time it became more and more clear that the problem was much bigger than initially thought. Finally, Tramèr et al. [58] found that adversarial regions often form the shape of a cone – a large contiguous region that can be traversed from start to finish. This is illustrated in Figure 2.4 (right).

This counter-intuitive fact may be caused by high dimensionality. In high-dimensional spaces, such an *adversarial cone* might be of small volume when compared to the entire input space, but still extremely large when considered locally. Once a point inside the adversarial cone has been found, most points in the vicinity are also adversarial. As a result, an attacker can traverse the cone from start to finish, quickly arriving at a powerful adversarial example. This is the main intuition behind boundary attacks (see Section 2.4.2).

**Empirical evaluation of robustness.** Exact calculation of a classifier’s robustness is often intractable [38]. Instead, it can be approximated empirically by creating adversarial examples: If  $\epsilon_{adv} = \|\eta\|_2$  is the magnitude of a successful adversarial perturbation, then  $\epsilon_{adv}$  is an upper bound for the true robustness of the classifier. Figure 2.5 illustrates this.

This geometric interpretation is a handy tool for understanding the motivation behind most state-of-the-art attack and defense methods. The rest of this chapter discusses them in detail.

## 2.4 Adversarial attacks and defenses

This section discusses the state of the art in adversarial attacks and defenses. First, a historical perspective on white-box attacks is given, which serves as an introduction. Then, several recent approaches to black-box attack are presented. Finally, common defenses against adversarial attacks are described, together with a discussion of their usefulness.

### 2.4.1 White-box attacks

To understand modern black-box attacks, it can be helpful to start with the white-box setting. Most of the early works on white-box robustness are conceptually simple and straightforward, which makes them a good introduction to the techniques employed in adversarial attacks. Black-box attacks often directly build upon these methods.

#### 2.4.1.1 Direct optimization attack

Although adversarial classification had been explored in the context of email spam detection as early as 2004 [59], it only started gaining traction in 2014 when Szegedy et al. [2] presented a working adversarial attack for deep neural networks. At the time, this demonstration caught many deep learning enthusiasts by surprise, as it directly challenged the then-common assumption that deep learning was bound to achieve human-like performance on computer vision tasks. Deep neural networks were already

surpassing humans in large-scale image classification [60], and it was thought that they learned to see the same features in images as humans do. Szegedy et al. provided strong counterexamples to this – adversarial examples.

Conceptually, their formulation of adversarial attack is straightforward. They choose targeted misclassification with minimal  $\ell^2$ -distance and directly optimize for the adversarial criterion:

$$\begin{aligned} x_{adv} &= x_{orig} + \eta \\ \eta &= \arg \min_{\eta} c \|\eta\|_2 + L_f(x_{orig} + \eta, y_{target}) \end{aligned} \quad (2.5)$$

Here,  $\eta$  is the adversarial perturbation and  $L_f$  is the cross-entropy classification loss of the original neural network. Since this is a multi-objective optimization problem (both classification loss and perturbation size are minimized), the constant  $c$  is used to balance the two objectives. Additionally, care must be taken that  $x_{adv}$  results in a valid image:  $x_{orig} + \eta \in [0, 1]^k$ .

In the white-box scenario, the classifier  $f$  is known and therefore the gradient  $\nabla L_f$  can be computed with respect to the perturbation  $\eta$ :

$$\nabla L_f = \frac{\partial L_f(x_{orig} + \eta, y_{target})}{\partial \eta} \quad (2.6)$$

This naturally lends itself to efficient higher-order optimizers. Szegedy et al. use a box-constrained L-BFGS optimizer [61] to find  $x_{adv}$ . L-BFGS is a quasi-Newton method but, as the problem is non-convex, the solution can only be approximated [2].

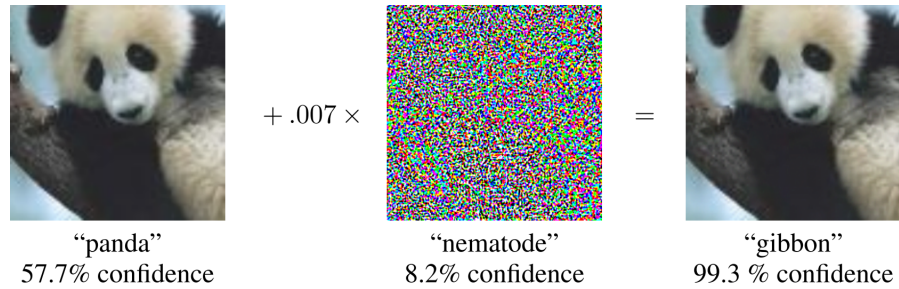
It should be noted that this optimization procedure is very similar to standard neural network training, and later works have abandoned the use of L-BFGS for the more common stochastic gradient descent (SGD) and its derivatives [39]. Nevertheless, the work of Szegedy et al. can be considered the prototypical formulation of adversarial attacks in the white-box scenario. The images shown in the introduction – Chapter 1, Figure 1.1 – were created with this method.

### 2.4.1.2 Fast Gradient Method (FGM)

While direct optimization of the entire adversarial objective (both misclassification and similarity) is conceptually straightforward, it takes a long time to converge [2]. Experimenting with the attack formulation, Goodfellow et al. [20] discovered that, in many cases, it can be sufficient to optimize for the misclassification criterion alone, ignoring the similarity criterion. Naturally, this greatly simplifies the formulation. Capitalizing on this insight, they proposed a simple one-step attack called the Fast Gradient Sign Method (FGSM):

$$\begin{aligned} x_{adv} &= x_{orig} + \eta \\ \eta &= -\epsilon \cdot \text{sgn}(\nabla L_f(x_{orig}, y_{target})) \end{aligned} \quad (2.7)$$

This equation requires only a single gradient evaluation, which makes the attack extremely fast. The perturbation  $\eta$  is then equivalent to a single update step of gradient



**Figure 2.6:** The well-known ”adversarial panda” was created with an untargeted one-step FGM attack. (Left): Original image. (Center): Difference image, strongly magnified. (Right): Adversarial example. It is evident that all pixels are perturbed with equal strength, as is customary for  $\ell^\infty$  attacks. The classifier being fooled is GoogLeNet [62]. Taken from Goodfellow et al. [20].

descent on the classification loss  $\nabla L_f$ , with a step size of  $\epsilon > 0$ . Consequently,  $\epsilon$  directly controls the magnitude of the perturbation – in this case,  $\|\eta\|_\infty = \epsilon$ .

The FGSM attack uses the sign function to normalize the adversarial gradient for  $\ell^\infty$ -based perturbations. It is trivial to extend the formulation to other norms, forming the more general family of Fast Gradient Method (FGM) attacks. For example, a FGM attack that creates  $\ell^2$ -minimal perturbations normalizes the gradient as follows:

$$\eta = -\epsilon \frac{\nabla L_f(x_{orig}, y_{target})}{\|\nabla L_f(x_{orig}, y_{target})\|_2} \quad (2.8)$$

It may seem surprising that a single, large, step of gradient descent would lead to good results. And indeed, this method typically requires the perturbation magnitude  $\epsilon$  to be relatively large – which makes it slightly less effective when compared with more expensive optimization attacks [39]. However, in untargeted settings, the perturbation can still be small enough to fool a human observer. This can be seen in Figure 2.6, which shows the now-famous panda image that popularized adversarial examples in 2015 [20].

The FGM attack delivers acceptable results at very low computational cost. For this reason, it has become the tool of choice for quick evaluations of robustness and still enjoys great popularity to this day [33, 34].

#### 2.4.1.3 Projected Gradient Descent (PGD)

While the FGM attack is very fast, it is not always successful. Kurakin et al. [31] found that it does perform well in untargeted scenarios, where the only goal is to reduce the accuracy of classification, but not so much in a targeted setting, where the attacker wants to force classification of a specific label and requires a certain minimum amount of precision. The reason for this is the large step size - since the FGM method performs one-step gradient descent, it can be expected to be very imprecise.

As an obvious extension of the FGM attack, Kurakin et al. proposed to simply apply it multiple times, using a smaller step size  $\alpha$ :

$$\begin{aligned} x_{adv,0} &= x_{orig} \\ x_{adv,n} &= \text{Clip}_{x_{orig},\epsilon} \left( x_{adv,n-1} - \alpha \frac{\nabla L_f(x_{adv,n-1}, y_{target})}{\|\nabla L_f(x_{adv,n-1}, y_{target})\|_2} \right) \end{aligned} \quad (2.9)$$

This attack now performs  $n$  steps of FGM gradient descent. Being somewhat of a middle ground, this attack has a significantly higher success rate than one-step FGM attacks, while still being faster than the original optimization attack [31].

To make sure that the adversarial example stays similar to the original image, the function  $\text{Clip}_{x_{orig},\epsilon}(x)$  clamps  $x$  to the  $\epsilon$ -neighborhood of  $x_{orig}$ . This *projection* guarantees that  $x_{adv}$  is always inside a hypersphere<sup>3</sup> with radius  $\epsilon$ . This matches the formulation of *robustness hyperspheres* in Section 2.3.3: If the attack finds an adversarial example, then the classifier is not robust within this  $\epsilon$ .

**Naming.** In their 2016 work, Kurakin et al. [31] introduced this attack as the "Basic Iterative Method" (BIM), however subsequent work by Madry et al. [33] established that the  $\text{Clip}(\cdot)$  function effectively performs a projection onto a hypersphere around  $x_{orig}$ . Therefore, the BIM is simply an implementation of Projected Gradient Descent (PGD), a very common general-purpose technique for optimization with constraints [63]. As a result, today this family of attacks is simply called PGD, and the term has become the dominant name in the field [15, 42, 18, 58].

#### 2.4.1.4 Stronger optimization attacks

By 2016, the first techniques had been proposed to defend against adversarial attack, and authors were quick to claim that their methods would make classifiers immune [64, 65]. However, most of these new defenses only showed robustness against fast attacks such as FGM or few-step PGD. Far from a true demonstration of robustness, this completely ignored the possibility that those attacks were specifically geared towards fast computation, and that a slower (but stronger) optimization attack might still find adversarial examples.

Realizing this, Carlini and Wagner [39] set out to create an extremely strong (if slow) attack with the goal of breaking established defenses. They modified the original optimization objective of Szegedy et al. (see Section 2.4.1.1) and introduced several improvements to allow for efficient optimization via Stochastic Gradient Descent (SGD).

The original formulation is considered to perform poorly in gradient descent, as extreme points and flat regions on the loss surface make optimization very hard [39]. To combat this, Carlini and Wagner substituted  $\tanh(\eta)$  for  $\eta$ , which allows for smooth optimization of the objective. Along with other improvements, this allows advanced SGD methods such as the popular Adam optimizer [66] to converge efficiently on adversarial examples with extremely small  $\eta$ .

<sup>3</sup>This also generalizes to other metrics than  $\ell^2$ . For example, an  $\ell^\infty$ -hypersphere is an axis-aligned hypercube. Clamping can then be implemented on a per-pixel/per-axis basis.

In their 2017 paper, Carlini and Wagner showed that their attack can find adversarial perturbations of smaller magnitude than previous methods before them. They also found that proposed defenses such as distillation [65] failed when confronted with their new, strong, attack. This observation would foreshadow the development of adversarial attacks and defenses in the years to come: It became a game of cat and mouse in which defense authors would evaluate their methods against fast/weak attacks, only for others to show that the defenses actually failed against stronger attacks [16, 67].

Over the years, many strong white-box attacks have been proposed. They often use advanced optimization techniques such as Elastic Nets [68] or the Decoupled Direction and Norm approach [69]. All of them have in common that they produce strong adversarial examples while often requiring a high computational budget. In the rest of this work, these methods are simply referred to as the family of strong white-box attacks (as opposed to fast white-box attacks such as FGM and PGD).

### 2.4.2 Black-box attacks

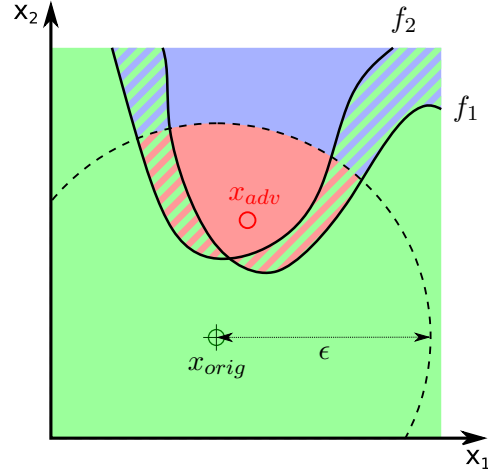
Now that the major families of white-box attacks have been described, it is time to consider the main focus of this work: Black-box attacks. Here, gradient information is not directly available. As a result, attacks are often slower and/or weaker than their white-box counterparts. This section describes the three major approaches to black-box attack, along with their individual strengths and weaknesses.

#### 2.4.2.1 Transfer attacks

Transferability is one of the many surprising properties of adversarial examples. Perturbations created against one classifier  $f_1$  can often fool another classifier  $f_2$ , even though they are tailor-made with the knowledge of only  $f_1$ . Figure 2.7 illustrates this: Transferability exists whenever there is an overlap between the adversarial regions of both classifiers.

The phenomenon was already discovered by Szegedy et al. in their initial work [2]. Later, it was shown that transferability is surprisingly high even between very different classifier architectures. For example, adversarial examples created on VGG-16 [70] have been shown [71] to transfer easily to other major architectures such as ResNet-50 [72] and MobileNet [73].

This phenomenon can be exploited for attacks such as described by Papernot et al. [32]: An attacker creates and trains a *surrogate model*, to which they have white-box access. Ideally, it is trained with the same data as the black box, however this is not a strict requirement [74]. Then, they perform a white-box attack on the surrogate model and feed the resulting adversarial example to the black box. Due to the transferability effect, there is a moderate possibility of success on the first try [32, 74]. This is the one-step formulation originally proposed by Papernot et al., who perform a simple FGM attack on their surrogate model. However, modern attacks have found more success with multi-step PGD methods, which are now the prevalent form of transfer-based attacks



**Figure 2.7:** Transferability between multiple classifiers. This figure shows the decision boundaries of two classifiers  $f_1$  and  $f_2$ . Each separates the input space into two classes (green or blue). It can be seen here that the adversarial regions (red) of both classifiers overlap. Therefore, many adversarial examples created for  $f_1$  will also fool  $f_2$ .

[75, 33, 69]:

$$x_{adv,0} = x_{orig} \quad (2.10)$$

$$x_{adv,n} = \text{Clip}_{x_{orig},\epsilon} \left( x_{adv,n-1} - \alpha \frac{\nabla L_{f_{surr}}(x_{adv,n-1}, y_{target})}{\|\nabla L_{f_{surr}}(x_{adv,n-1}, y_{target})\|_2} \right)$$

Finally,

$$\hat{y} = f_{BB}(x_{adv,n}) \quad (2.11)$$

Here,  $f_{BB}$  is the black-box model under attack, and  $f_{surr}$  is the surrogate model. The formulation differs from the original PGD attack only in that gradients are calculated on the surrogate, but predictions are tested on the black box. Once  $\hat{y}$  fulfills the adversarial criterion, the attack is successful.

At the time of writing, transfer attacks were considered the most powerful approach. They consistently achieved winning scores in open competitions on adversarial examples, such as the NeurIPS 2017 and 2018 Adversarial Vision Challenges [17, 18]. However, it must be noted that their dependence on surrogate models is also a great liability:

- **Limited adaptivity.** Typically, these attacks are of a hit-and-miss nature. If the adversarial gradient of the surrogate does not closely align with the (unknown) gradient of the black box, then a transfer attack takes a shot in the dark. If an adversarial example is not found quickly, then subsequent updates on a badly approximated gradient will only worsen the situation. Therefore, transfer attacks are considered strong but extremely brittle [15].

- **Expense of training a surrogate.** Because of this brittleness, great care must be taken when crafting the surrogate. An attacker must train the model on similar data, try to extract labels from the black box [32], try data augmentation techniques like adversarial training (see Section 2.4.3.6), compare different architectures, or even train an entire ensemble of them. These are the steps that are typically attempted in order to improve the chance of transfer [34, 32, 18].
- **Computational cost and memory requirements.** The attacker must store at least one model with similar size and complexity as the black box. Often, an entire ensemble of surrogates is used [18], requiring many gigabytes of RAM (typically on GPUs) and considerable computing power for calculating gradients.

### 2.4.2.2 Gradient estimation

Because of these drawbacks, it seems promising to try and conduct black-box attacks in a model-free manner – that is, without using a surrogate model.

So far, all attacks described in this chapter have relied on gradient information to calculate adversarial examples, which means they perform *first-order* optimization. In the white-box setting, gradients are directly available, and in the black-box setting, a transfer attack can approximate them through surrogates. However, when no gradients are available at all, the problem essentially becomes one of *zeroth-order* optimization.

Zeroth-order optimization has a long history and consists of many search techniques. Methods such as simulated annealing, evolutionary strategies, or even random search are considered staples of optimization, and could – in theory – be used to search for adversarial examples. However, most of these techniques are not considered tractable as the dimensionality of the search space is extremely high. For example, a typical image from the ImageNet dataset [48] has a dimensionality of  $k = 299 \times 299 \times 3 = 268.203$ .

It turns out that some carefully-selected zeroth-order methods can indeed be used to find adversarial examples even for high-resolution image classifiers. One such technique is stochastic gradient estimation with SPSA [43, 76]. Other methods of gradient estimation have also found success in black-box attacks [42, 27, 44, 50], but the general principle can be explained with the example of SPSA.

SPSA is a stochastic version of the finite difference method: An input  $x$  is randomly perturbed and the corresponding change in the classifier’s output  $\hat{y} = f(x)$  is recorded. When multiple such measurements are taken, the derivative  $\frac{\partial f(x)}{\partial x}$  can be estimated and used as a substitute for the true gradient. Crucially, it is not necessary to perturb each dimension individually – the stochastic nature of SPSA allows us to take a random sample that is significantly smaller, thereby retaining reasonable efficiency even in high dimensions. This results in a rough approximation of the true gradient, which is often sufficient for gradient descent. Finally, this gradient can then be plugged into white-box methods such as PGD.

Gradient estimation attacks are adaptive, i.e. no matter which defense is inside the black box, they always approximate its true gradient – given that enough queries to the model are performed. And indeed, early black-box attacks using gradient estimation were



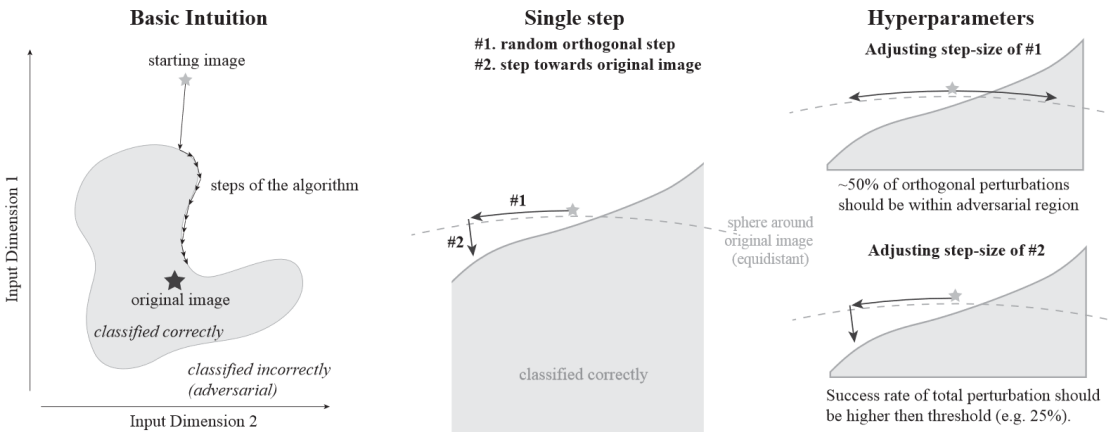
inefficient: One of the first such attacks by Chen et al. [27] needed several million queries to create good adversarial examples for an ImageNet classifier. But soon, optimizations were proposed that brought this number down to below 15,000 [44, 42].

**Dependence on confidence scores.** Gradient estimation also has a significant weakness: Most techniques require the output  $\hat{y}$  to be real-valued [15]. In neural network classifiers, this is typically a softmax vector of class probabilities, where each value indicates the classifier’s confidence for a specific class. If a defender chooses to hide the confidence vector and instead returns only the class label of the final top-1 prediction, then gradient estimation becomes very difficult: A small change in  $x$  often produces no change in  $\hat{y}$  at all, as the label stays the same. Consequently, the loss landscape becomes flat as  $\frac{\partial f(x)}{\partial x} = 0$ . Some workarounds have been proposed, but they often increase the number of required queries to millions [42], negating earlier improvements in efficiency.

In short, gradient estimation attacks are very powerful when confidence scores are available – but very inefficient in a decision-based setting.

### 2.4.2.3 Boundary attacks

The boundary attack formulation was first introduced in 2018 by Brendel et al. [15]. It shows that it is possible to create adversarial examples in a black-box decision-based setting without training surrogate models *or* estimating gradients. Since much of the work in Chapter 3 is based on this method, it is worth understanding it in detail.



**Figure 2.8:** A geometric illustration of a boundary attack. The gray region is the set of inputs  $x$  which are classified as the correct label, i.e.  $f(x) = y_{gt}$ . The white region is the set of potential adversarial examples, i.e.  $f(x) = y_{target} \neq y_{gt}$ . The boundary between the regions is the decision boundary, along which the algorithm performs a greedy hill-climbing search. Taken from Brendel et al. [15].

In essence, the attack is a greedy hill-climbing search algorithm that performs rejection sampling from random perturbations. Figure 2.8 (left) describes the general concept. The boundary attack starts with an input  $x_{start}$ , which may be far away from the original image  $x_{orig}$  but satisfies the adversarial criterion  $f(x_{start}) = y_{target} \neq y_{gt}$ . This can simply be a real image of the target class.

## 2 Adversarial robustness: State of the art

The algorithm initializes  $x_{adv} := x_{start}$ . From here, it interpolates through the input space and tries to move *towards* the original image  $x_{orig}$ , refining  $x_{adv}$  in the process. This stands in contrast to other attacks presented so far, which always start at  $x_{orig}$  and move *away* from it. At some point during this interpolation process, the decision boundary is reached. As shown in Figure 2.8 (center), the attack must now choose a sideways direction (#1) before it can proceed further towards  $x_{orig}$  (#2).

**Orthogonal step (#1).** To do this, the boundary attack samples a random vector  $r \sim \mathcal{N}(0, 1)^k$  from a multidimensional normal distribution and projects it onto a hypersphere around the original image (dashed line in Figure 2.8). More precisely, if  $\delta = x_{orig} - x_{adv}$  is the vector pointing from the current image to the original image, then  $r$  can be projected orthogonally to it:

$$r \sim \mathcal{N}(0, 1)^k \tag{2.12}$$

$$\eta_{ortho} = \frac{r}{\|r\|_2} - \delta \frac{r \cdot \delta}{\|r\|_2 \cdot \|\delta\|_2}$$

$\eta_{ortho}$  is now a random unit vector on the hyperplane which is orthogonal to the direction of  $\delta$ . By construction, this hyperplane is tangential to the hypersphere with radius  $\|\delta\|_2$  around  $x_{orig}$ . From here, a slight correction can be performed so that  $\eta_{ortho}$  directly lies on the hypersphere. Formally, this is required to guarantee that the orthogonal step does not increase the distance to the original image. In practice, this is not needed as the tangential hyperplane approximates the hypersphere well enough<sup>4</sup>. Therefore, this and subsequent Chapters will omit spherical correction for brevity.

Performing the orthogonal step results in an intermediate position  $x_{ortho}$ :

$$x_{ortho} = x_{curr} + \epsilon_{ortho} \cdot \|\delta\|_2 \cdot \eta_{ortho} \tag{2.13}$$

The step size  $\epsilon_{ortho} \in [0, 1]$  is a hyperparameter and is scaled by the current distance to the original image,  $\|\delta\|_2$ .

**Source Step (#2).** Afterwards, the attack takes a step towards the original image (#2 in Figure 2.8):

$$\eta_{source} = \frac{(x_{orig} - x_{ortho})}{\|(x_{orig} - x_{ortho})\|_2} \tag{2.14}$$

$$x_{candidate} = x_{ortho} + \epsilon_{source} \cdot \|\delta\|_2 \cdot \eta_{source} \tag{2.15}$$

Again, the step size  $\epsilon_{source} \in [0, 1]$  is a hyperparameter which is scaled by  $\|\delta\|_2$ .

**Rejection test.** Reducing the distance to the original image,  $x_{candidate}$  is now a candidate for being a better adversarial example than the previous  $x_{adv}$ . If  $f(x_{candidate}) = y_{target}$ , then the decision boundary has not been crossed and  $x_{candidate}$  is an adversarial example. The current position is updated:  $x_{adv} := x_{candidate}$ . However, if  $f(x_{candidate}) \neq y_{target}$ , then the decision boundary has been crossed and  $x_{candidate}$  is

<sup>4</sup>Spherical correction reduces the distance to  $x_{orig}$  by a factor of  $\sqrt{1 + \epsilon_{ortho}^2}$ . Brendel et al. recommend a step size of  $\epsilon_{ortho} = 0.01$ , which results in a final correction factor of  $\approx 1.00005$ . As the algorithm does not have strong requirements for numerical stability, this correction is negligible in practice.

not adversarial. It must therefore be discarded, and the entire process is repeated with another random orthogonal step.

Experiments have shown that, given enough random samples and queries to the model, the boundary attack can produce strong adversarial examples with very small perturbation magnitude  $\|\eta\|_2 = \|x_{adv} - x_{orig}\|_2$  [15]. Typically, the attack is run until either this magnitude is small enough, or a certain budget of queries to the model has been exhausted.

In black-box evaluation settings, the latter is mostly the case: Often, hundreds of random directions must be tested before the algorithm can advance a single step. While otherwise promising, this deficiency renders the method infeasible for quick evaluations of black-box systems. In Chapter 3, this drawback is addressed and a new, efficient, family of boundary attacks is proposed.

### 2.4.3 Defense mechanisms

To understand the effectiveness of adversarial attacks, it is important to consider defense mechanisms as well. Here, the term "defense" is used loosely: A defender might want to thwart a specific malicious attack, but more generally they would aim to simply make a system more robust to anomalies. This section describes the most prominent approaches to defense that have been proposed in recent years.

#### 2.4.3.1 Gradient obfuscation

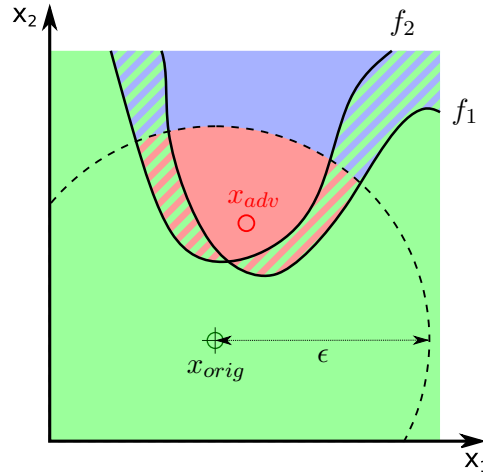
At first, research into defense mechanisms concentrated on directly countering white-box attacks. Since these mostly rely on gradient-based optimization in one form or another, early defenses aimed to diminish its effectiveness. Typically, this was achieved by modifying the neural network to have vanishing gradients, or by introducing non-differentiable operations into the classification process [16].

One of the first such defense methods was Defensive Distillation [65], which was used to successfully defend against fast attacks such as FGM. This directly prompted the rise of stronger attacks, which in turn were able to break that defense (see Section 2.4.1.4). In 2018, this cat-and-mouse game culminated in a seminal publication by Athalye et al. [16], who showed that a large portion of then state-of-the-art defenses worked solely because of gradient obfuscation, and that most of them did not work at all against stronger attacks.

Today, obfuscated gradient defenses are completely sidestepped by black-box attacks, which can be performed without gradients. As a result, this family of defenses has become largely obsolete.

#### 2.4.3.2 Ensembling and redundancy

Ensembling is a staple technique for increasing both the accuracy and generalization capability of machine learning models [8]. The motivation here is to produce a set of heterogenous classifiers and to combine their strengths while at the same time diminishing their joint vulnerabilities.



**Figure 2.9:** Ensembling can increase robustness. Here, the region of vulnerability is reduced to the intersection of adversarial regions of  $f_1$  and  $f_2$ . However, due to the transferability phenomenon, these regions often have significant overlap. This ultimately limits ensembling, and the combined adversarial region is only marginally smaller.

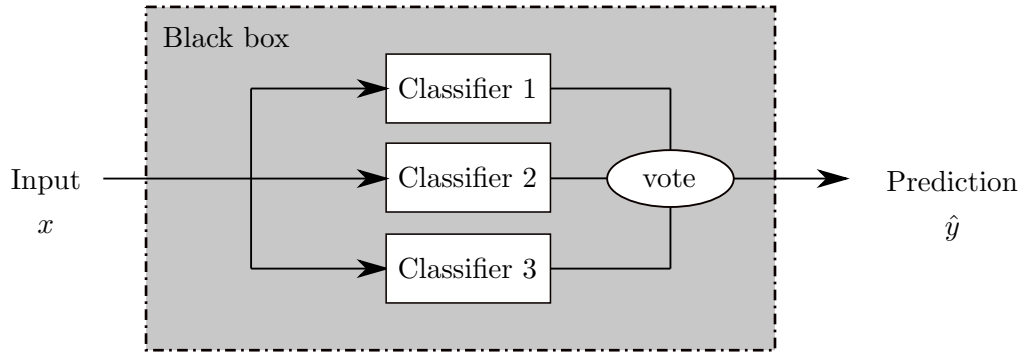
The same idea motivates ensemble adversarial defenses. Recall that, in a set of classifiers  $F$ , for each classifier  $f \in F$  there exists a region  $X_{adv,f} \subseteq X$  in the input space that contains adversarial examples  $x_{adv,f} \in X_{adv,f}$ . This is the *adversarial region* in the input space of classifier  $f$ .

Ensembling is then fueled by the hope that the adversarial regions of multiple classifiers do not overlap too strongly. As a result, the region of vulnerability of an ensemble would then be the intersection of those of the individuals:

$$X_{adv,F} = \bigcap_{i=1}^n X_{adv,f_i} \quad (2.16)$$

This works in principle, and it has been shown that ensembling does indeed increase robustness by a tangible amount [34].

However, ensembling does have its limits. Classifiers must not only be as diverse as possible but also have very high accuracy individually. This reveals a dilemma: On tasks such as image classification, deep learning methods are currently ahead of every other technology in terms of accuracy [77]. This realistically limits the choice to a small number of architectures which are mostly variations of Inception [47] and ResNets [72]. While these architectures seem very different, it turns out that they learn similar representations when trained on the same data – this is exactly the effect that is exploited by transfer attacks (see Section 2.4.2). As a result, their adversarial regions sometimes overlap so strongly that  $X_{adv,f_1} \approx X_{adv,f_2}$  for a pair of classifiers  $f_1$  and  $f_2$ . Figure 2.9 illustrates this phenomenon. In fact, the effect is often so pronounced that He et al. [78] found ensemble classifiers to be detrimental to accuracy *and* robustness when not carefully designed.



**Figure 2.10:** Data flow in an ensemble of classifiers. Here, multiple classifiers process the same input  $x$  and each produces a slightly different prediction. The individual predictions are then reduced, e.g. by majority vote, to a single output  $\hat{y}$ . This makes the ensemble a single, large, black box which can be attacked like any other classifier.

When attacking ensembles, it might be tempting to attack each classifier in the ensemble individually. Instead, it is much easier to simply treat the entire ensemble as a large black box – which can then be attacked by any black-box method. Figure 2.10 illustrates this.

Although their effectiveness is limited by the transferability phenomenon, ensembles can boost robustness by a nontrivial amount. Since they are easy to implement, they have become a staple in competitions on adversarial defenses [18] and are likely to be a cornerstone of real-world defense strategies in the future - at least whenever the computational budget allows it.

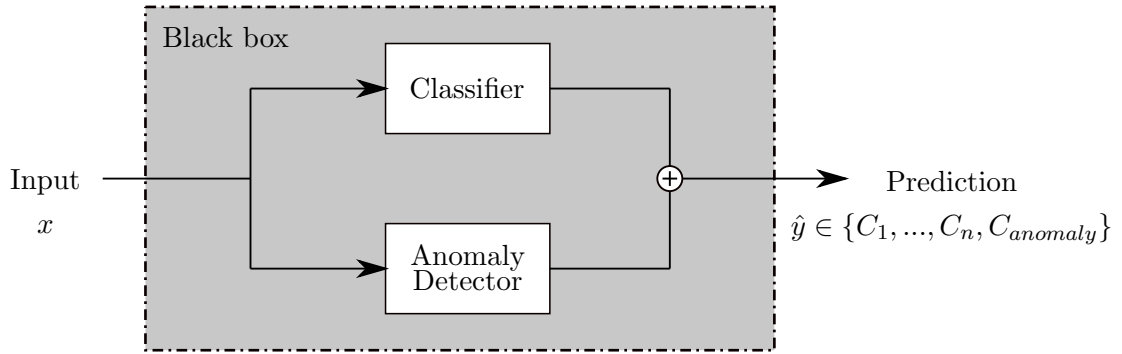
### 2.4.3.3 Randomization

Randomized (stochastic) defenses are a special form of ensembling. Using randomized predictions to combat adversarial examples is a popular idea, and many approaches such as Stochastic Activation Pruning [79], Monte-Carlo Dropout [80] or randomized pre-processing [81] have been proposed. It should be noted that the act of randomizing parts of a model – and then drawing multiple samples from it – is directly equivalent to ensembling. More precisely, it has been shown that a stochastic classifier approximates an infinite ensemble of classifiers as the number of random samples drawn approaches infinity [80].

Therefore, employing stochasticity results in the same benefits (slightly improved robustness) and the same drawbacks (high computational cost, often reduced accuracy) as explicit ensembling [16].

### 2.4.3.4 Anomaly detection

Another approach is to treat adversarial examples as anomalies – which they are per definition – and try to detect them with established techniques. Once detected, the



**Figure 2.11:** Anomaly detectors are a special form of ensembling: The system outputs either the classification or an error if an anomaly was detected. An attacker may simply concatenate these outputs and then treat the entire system as a single black box with an additional class (which is to be avoided).

adversarial input can then be discarded. Since adversarial perturbations often seem striking and colorful, surely they could be detected with the right method?

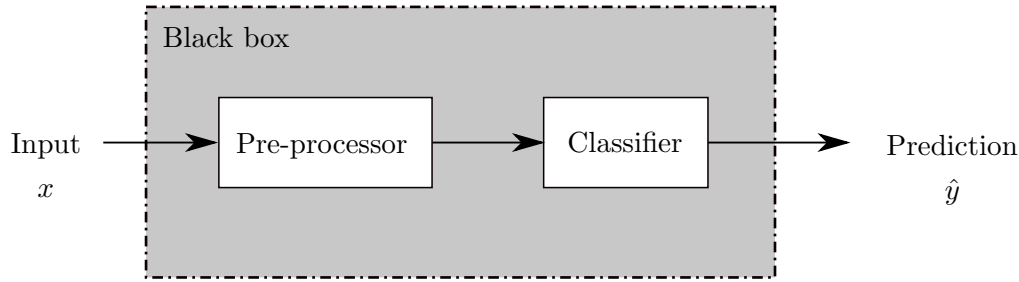
Several methods for detecting adversarial examples have been proposed. There are approaches to outlier detection via Principal Component Analysis (PCA) [40], or using the Mahalanobis distance as a measure of confidence [82]. Finally, it is also possible to use ensembling and uncertainty estimation techniques such as Monte-Carlo Dropout [80], with high uncertainty resulting in detection.

The issue with these approaches is that they do not solve the problem on a fundamental level. The authors of these methods claim that they reliably detect adversarial examples generated by white-box methods such as FGM or even strong optimization attacks. However, this argument is flawed: In the white-box scenario, the attacker knows everything, and this includes the detection technique. It is therefore trivial to design an attack on the detector as well. Carlini and Wagner [40] showed this in their seminal 2018 work, where they observed that all detection methods known at the time could be circumvented by an adaptive attacker, showing that previous claims of robustness were based largely on obscurity – again.

Perhaps ironically, attacking detectors in the black-box scenario is even easier. Since anomaly detectors are run in parallel with classifiers, they are simply a special form of ensembling (see Figure 2.11). The attacker can regard the entire system as a single black box with input  $x$  and output  $\hat{y} \in \{C_1, \dots, C_n, C_{anomaly}\}$ . With this formulation, the system can be attacked just like any other classifier<sup>5</sup>.

Defenses based on anomaly detection are therefore yet another flavor of classifier ensembles.

<sup>5</sup>This is true only if the detection result is available to the attacker. While this may seem a strong assumption, it is congruent with the threat setting in Section 2.2.



**Figure 2.12:** Data flow of pre-processing defenses. As before, the entire system can be treated as a single black box, which can then be attacked like any other classifier.

### 2.4.3.5 Pre-processing

Many forms of image pre-processing have been proposed to combat adversarial examples. The concept is simple: Adversarial patterns are very striking, therefore it should be possible to remove them with simple image filters.

It has been found that adversarial perturbations generated by attacks such as FGM and PGD are very brittle, and when an image is transformed only slightly, the pattern immediately loses its effect [83]. And indeed, many recently proposed defenses employ pre-processing such as image denoising [84], JPEG compression [83], simple median filters [17] or random transforms of the input image [81].

Pre-processors can increase robustness by a limited amount, as they project the space of all possible inputs  $X$  onto the space of pre-processed inputs  $X_{preproc}$ , and this function is typically not bijective. Therefore, the attacker has fewer options for perturbing the image as the adversarial region in  $X$  is effectively shrunk. However, a pre-processor must also preserve all features that are necessary for classification, or else accuracy will greatly suffer. In this way, pre-processors underlie the same robustness-accuracy trade-off as other defense techniques.

A pre-processor aims to separate adversarial from non-adversarial features and then forward the latter to the classifier. This is very similar to the formulation of an anomaly detector, only that the detector acts in parallel to the classifier, while pre-processing forms a serial pipeline (see Figure 2.12). This means that black-box attacks can attack pre-processing defenses in the same way as anomaly detectors or ensembles – by simply considering the defense to be part of the black box [16].

### 2.4.3.6 Adversarial training

Currently, the most effective [18] defense is also the most fundamental: Adversarial training. This method first creates adversarial examples and then adds them to the training dataset [2]. As a result, the model learns to classify these examples correctly. Adversarial training can then be considered a form of *data augmentation*: The training data is artificially inflated by creating perturbed copies of the original examples.

In classic data augmentation for computer vision, such perturbations typically consist of random rotation, changes in brightness and contrast, and the addition of small amounts of noise. As a result, the classifier gains invariance to these kinds of perturbations and generalizes better to unseen data. Data augmentation is considered highly effective and has been used by most award-winning image classifiers in recent years [77]. Adversarial training extends data augmentation to adversarial perturbation. The hope is for the classifier to generalize between clean and adversarial examples and therefore become invariant to adversarial perturbation as a whole.

Unfortunately, current methods fall short of this goal. Initial attempts at adversarial training using direct optimization attacks [2] were prohibitively expensive. This improved with fast methods such as FGM and PGD, which allow for non-stop creation of new adversarial examples at training time. These are then fed back into the training process, improving the model’s robustness before running the next attack<sup>6</sup>.

However, the perturbations created by these fast attacks are also considered relatively weak [39, 34]. While adversarial training greatly increases robustness against the specific attack methods utilized during training [33], it also tends to overfit on them. Madry et al. [33] even observed that adversarial training performed on examples obtained by minimizing  $\ell^\infty$ -distance can be ineffective against attacks that use the  $\ell^2$ -distance. It is therefore apparent that adversarial training overfits on training data, and does not achieve invariance to *all* adversarial perturbations.

Over the years, adversarial training has been continually improved to remedy this weakness. Tramèr et al. [34] suggest creating adversarial examples on an ensemble of models that is *different* from the model that is trained, therefore only using perturbation patterns that fool all models. These examples are then considered much stronger, and Tramèr et al. report a much larger increase in robustness as well as better generalization across attack methods.

Today, adversarial training performed with strong attacks on an ensemble of classifiers is considered the state of the art in defenses [69, 33, 34]. This is further supported by its strong performance in competitions such as the NeurIPS’18 Adversarial Vision Challenge [18], where all winning entries used adversarial training as part of their defenses.

## 2.5 Summary

There exist a plethora of adversarial attacks and defenses, and enumerating them all would be impossible. Having said that, most (if not all) recently devised methods follow the more general approaches which have been outlined in this Chapter.

Armed with knowledge about the landscape of attacks and defenses – and how to evaluate them – the next chapter presents the main contribution of this thesis: The Biased Boundary Attack.

---

<sup>6</sup>This formulation of adversarial training as a game between a classifier and a generator directly inspired the now-famous Generative Adversarial Networks [57], which learn to generate realistic-looking images through a game-theoretic formulation of the same objective.

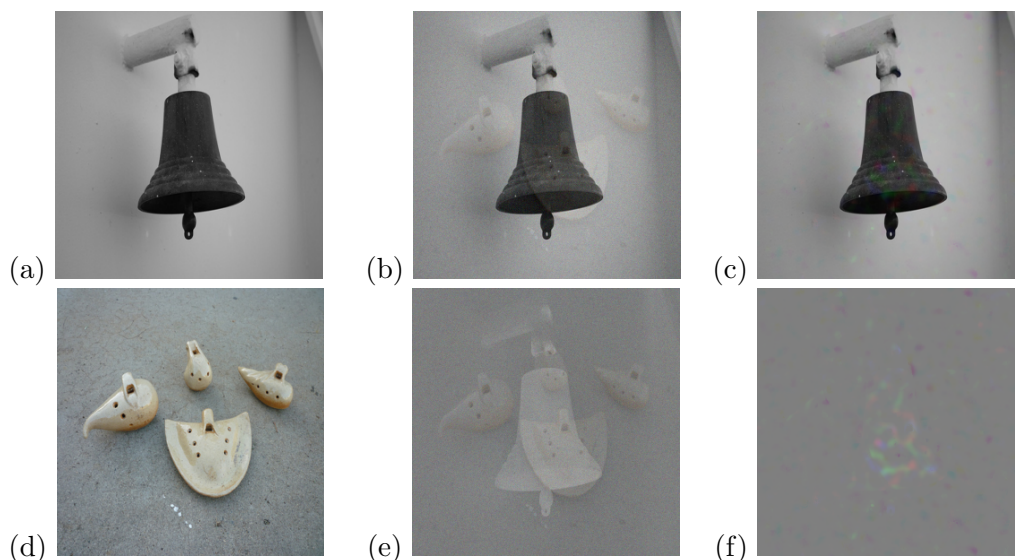


### 3 A new and efficient black-box attack

This chapter presents a new approach to finding black-box adversarial examples. Existing black-box attacks often use costly random search to "guess" vulnerabilities in image classifiers. This process is inefficient and can be accelerated by introducing prior knowledge about the domain of natural images.

One way to achieve this is to add specific biases to the attack's sampling procedure. In this chapter, three such biases are proposed: Low-frequency perturbations, regional masking, and surrogate gradients. Additionally, the question of initialization is discussed. Here, too, it is found that prior knowledge can be used to craft synthetic images that, when used as starting points for adversarial attack, greatly speed up the search procedure.

These strategies can be easily integrated into boundary attacks, an attack family that is reliable but suffers from low efficiency. The resulting attack, here called Biased Boundary Attack, tackles the efficiency problem and emerges as a new black-box attack that is both fast and reliable.



**Figure 3.1:** Targeted black-box adversarial examples for a high-resolution ImageNet classifier. (a) shows the original image (bell). (b) is an adversarial example generated by the original boundary attack ( $\|\eta\|_2 = 18.52$ ) and (c) shows the efficient attack developed in this work ( $\|\eta\|_2 = 4.78$ ). Both were obtained with the same number of model queries and are classified as "ocarina". (d) is an image of the target class, which is used as starting point, and (e) and (f) show the isolated perturbations.

### 3.1 Problem statement and approach

As discussed in Chapter 1, black-box adversarial attacks hold great promise for the empirical evaluation of robustness in machine learning systems. However, as outlined in Section 2.4.2 of Chapter 2, current methods have various drawbacks that limit their usefulness. To recall, these are the main performance criteria for black-box adversarial attacks:

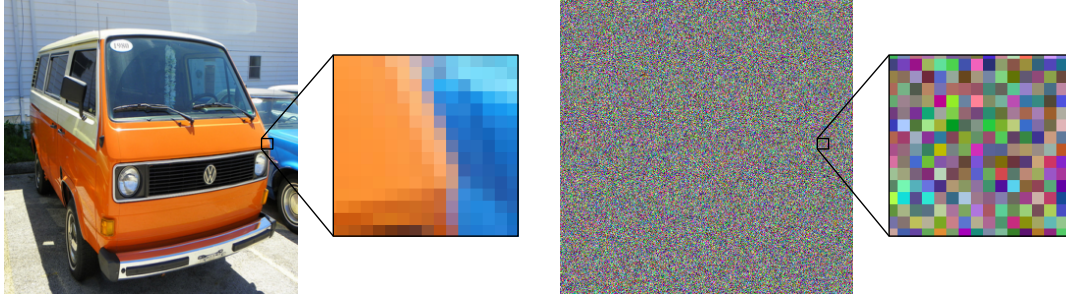
- **Reliability.** Model-free approaches, such as boundary attacks and gradient estimators, are considered very reliable. Given enough queries to the model, they manage to fool classifiers almost universally [15]. Transfer attacks, in contrast, are fast and efficient, but less reliable. If they are not carefully tuned, they often fail to produce an adversarial example altogether [34], no matter how many queries are performed.
- **Query efficiency.** Attacks are generally measured on the number of queries to the black-box model. Transfer attacks are considered very efficient: If transfer of an adversarial perturbation succeeds, then it typically does so within very few tries. This makes transfer attacks strong in terms of efficiency (as opposed to reliability). Model-free attacks such as boundary attacks, however, are considered inefficient, as they need a very large number of queries to the classifier under attack.
- **Ease of use.** Strong transfer attacks require large ensembles of carefully-trained surrogates [17], which results in a significant amount of preparation and tuning before an attack can be conducted. Boundary attacks and gradient estimation attacks, however, are usually model-free and can be applied out-of-the-box.
- **Applicability.** Attacks based on gradient estimation rely on real-valued output (such as confidence scores) and can not be applied to the more general label-only setting, where the black box outputs only a single discrete result<sup>1</sup>. Transfer and boundary attacks, however, can operate in any scenario as long as the final decision of the black box is obtained.

For reliable and efficient evaluation of black-box robustness, it is necessary to design an attack that satisfies all criteria. However, no such attack currently exists. Given the requirement of applicability to label-only settings, gradient estimation attacks cannot be considered at this time. This leaves two families of attacks: transfer attacks, which are efficient but not reliable, and boundary attacks, which are reliable but not efficient.

This work concentrates on the latter and addresses their main weakness: efficiency. In this chapter, the Biased Boundary Attack is proposed, a new form of boundary attack that addresses the efficiency problem by incorporating prior knowledge about the

---

<sup>1</sup>Even though some gradient estimation schemes are indeed applicable to the general label-only setting, their efficiency greatly suffers under it. This eliminates any advantage they may otherwise have over boundary attacks. See discussion in Section 2.4.2 of Chapter 2 and evaluation in Section 4.7 of Chapter 4.



**Figure 3.2:** In natural images, pixels strongly correlate with their neighbors. (Left): This image of a minibus has large regions of similarly-colored pixels. (Right): An image in which every pixel is sampled i.i.d from a normal distribution, as is customarily done for boundary attacks. Repeatedly adding such perturbations to an image results in high-frequency noise – the probability of obtaining more naturally-looking patterns is very low, and therefore a great number of attack steps are required.

computer vision domain. With this method, it becomes possible to conduct black-box adversarial attacks in a reliable *and* efficient way.

### 3.2 Biased sampling with prior knowledge

Let  $f(x)$  be a black-box classifier which takes as input an image  $x \in \mathbb{R}^k$ , with  $k$  being the number of input dimensions. In black-and-white images,  $k$  is the number of pixels and in color images,  $k$  is the number of pixels times channels. For example, a typical input from the ImageNet dataset is of dimensionality  $k = 299 \times 299 \times 3 = 268203$ .

As described in Section 2.4.2 of Chapter 2, the original boundary attack formulation draws random perturbation candidates from a multidimensional standard normal distribution:  $r \sim \mathcal{N}(0, 1)^k$ . This means that it performs unbiased sampling, perturbing each input feature independently of the others. While this is very flexible, it is also extremely inefficient.

This becomes clear in a simple thought experiment. Consider the distribution of natural images: Adjacent pixels are typically not independent of each other, but have similar color (see Figure 3.2). This alone is a strong indicator that drawing perturbations from independent and identically distributed (i.i.d.) random variables will lead to adversarial perturbations that are clearly from a different distribution than natural images. This, of course, makes it easy for a defender to separate the perturbation from the image signal, and the adversarial ”noise” is easily detected and filtered. By employing pre-processing defenses (see Chapter 2), state-of-the-art classifiers have become increasingly resilient against such patterns [17].

When a perturbation does not fool the classifier, the boundary attack must discard it and draw new samples until it finds success (rejection sampling). Only then can it take the next step, reducing the distance between  $x_{adv}$  and  $x_{orig}$ . When attacking a

robust classifier, this can take a very long time, requiring many queries to the black box. Therefore, unbiased sampling is likely to hinder the efficiency of boundary attacks.

Addressing this problem, it seems only logical to identify and remove unrealistic patterns from the distribution, and instead constrain the search space to perturbations that promise a higher chance of success. This adds a strong bias to the sampling procedure. In this way, the attack then utilizes domain knowledge to increase its efficiency. This can be any hint, intuition, heuristic, or expert information that exists about the problem domain, or about the classifier under attack.

This section describes three such biases for the domain of image classification, discusses their motivation, and shows how to integrate them into the Biased Boundary Attack.

### 3.2.1 Low-frequency perturbations

When one examines typical adversarial examples, it quickly becomes apparent that most existing attacks create perturbations with high image frequency. But high-frequency patterns have a significant problem: they are easily identified and separated from the original image signal, or otherwise dampened by pre-processing defenses. Indeed, most winning defenses in the NeurIPS 2017 Adversarial Attacks and Defences Competition had components that performed denoising [84], median filtering [17], or random transforms [81]. In other words: many robust classifiers tend to filter high-frequency noise.

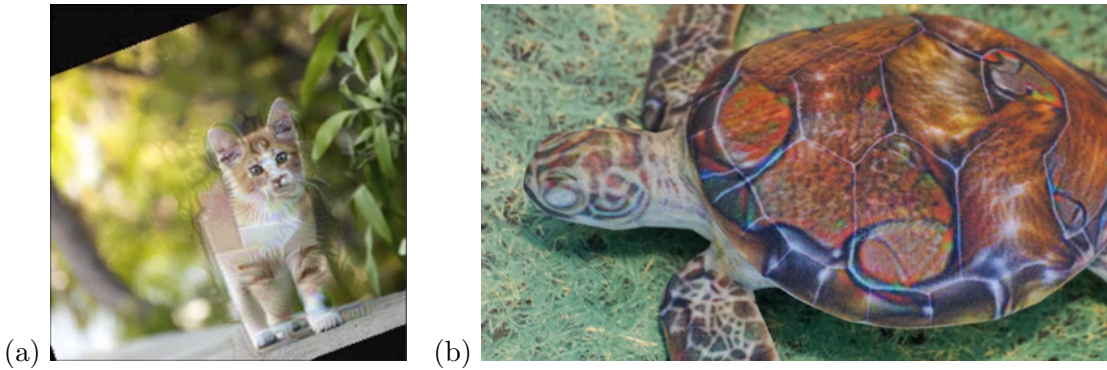
At the same time, it has recently been shown that strong white-box attacks can synthesize adversarial examples that are not easily filtered in this way [85]. Consider Figure 3.3, which shows adversarial examples that can be scaled, rotated, and even printed and photographed without losing their effect. It should be noted that such attacks are exotic and difficult – they require expensive optimization and white-box access. While they cannot be used for black-box attack, it is apparent that the final patterns contain little to no high-frequency noise.

This observation provides some evidence that strong perturbations seem to correlate with low image frequencies. Is this a causal relationship or a side effect? And can we somehow exploit this effect?

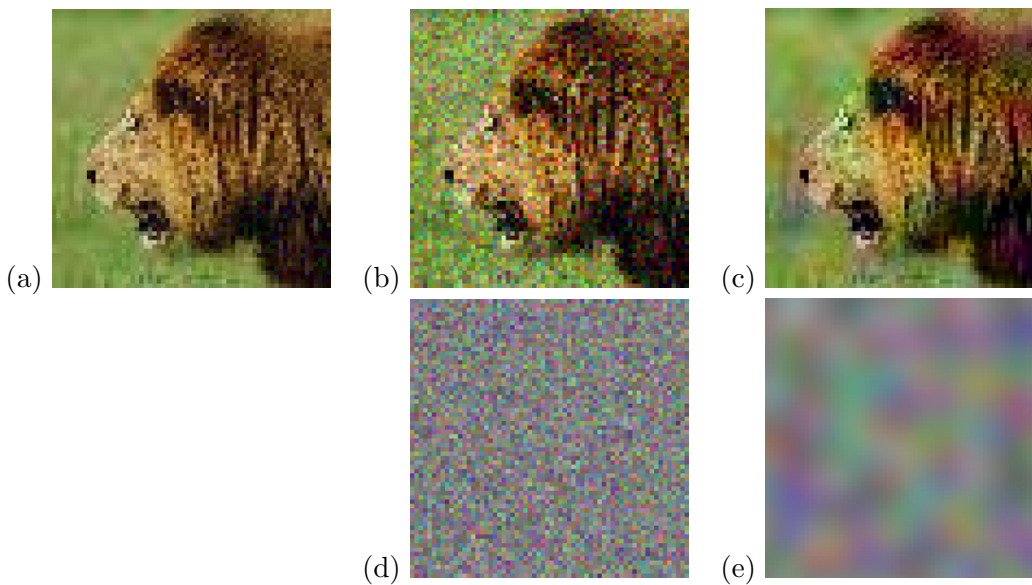
**Hypothesis.** Image frequency could play a large role in the effectiveness (and therefore efficiency) of adversarial attacks. Simply limiting perturbations to the low-frequency spectrum could increase the success chance of an attack, while incurring no extra cost.

**Creating low-frequency noise.** A straightforward way to generate parametrized, low-frequency patterns, is to use Perlin Noise [86]. Originally intended as a procedural texture generator for computer graphics, this function creates low-frequency noise patterns with a reasonably "natural" look. One such pattern can be seen in Figure 3.4(e). But how can it be incorporated into an attack?

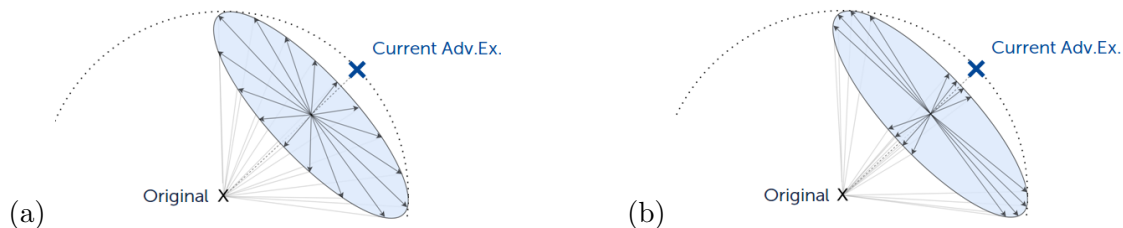
**Integration into the Biased Boundary Attack.** Let  $k$  be the dimensionality of the input space. The original boundary attack formulation applies an orthogonal perturbation  $\eta_{ortho}$  along the surface of a hypersphere around  $x_{orig}$ , in the hope of moving deeper into an adversarial region. From there, a second step  $\eta_{source}$  is taken towards  $x_{orig}$ . In the default configuration, candidates for  $\eta_{ortho}$  are generated from random samples  $r \sim \mathcal{N}(0, 1)^k$ , which are projected orthogonally to the source direction



**Figure 3.3:** Inspiration for low-frequency perturbations. Prior work [85] has demonstrated adversarial examples that can pass through many filters and transforms without losing their adversarial effect. Although they require white-box optimization at a prohibitive cost, they can still serve as intuition for improving black-box search. In both images, the perturbations are visible but do not resemble classic adversarial patterns – there is little to no high-frequency noise. Could this be a key feature of strong adversarial perturbations, and is it possible to emulate this effect? (a): Image of a cat, which is misclassified as "computer monitor". The image can be rotated and scaled while still retaining the adversarial effect. (b): A 3D-printed turtle with an adversarial pattern on its back. No matter from which angle the turtle is photographed, it is classified as "rifle". Taken from Athalye et al. [85].



**Figure 3.4:** Close-up toy example of low-frequency adversarial perturbation. (a): Original image. (b) and (c): Adversarial examples, obtained by adding randomized noise patterns until misclassification is achieved. (d) and (e): Isolated perturbation patterns. (d) is sampled from a normal distribution, and (e) is sampled from a distribution of Perlin noise patterns. Both (b) and (c) fool the classifier, but (c) can be obtained with fewer tries.



**Figure 3.5:** Sampling directions for the orthogonal step of the Biased Boundary Attack. (a) Original (unbiased) scheme: Uniformly distributed along the surface of the hypersphere. (b) With low-frequency bias: By limiting the samples to low-frequency patterns, sampling resolution increases in those directions.

and normalized to the desired step size. This leads to the directions being uniformly distributed along the hypersphere, as shown in Figure 3.5(a).

To introduce a low-frequency prior into the Biased Boundary Attack, the orthogonal direction is instead sampled from a distribution of Perlin noise patterns. Perlin noise is typically parameterized with a permutation vector  $v$  of size 256, the elements of which can be randomly shuffled on every call. Effectively, this allows us to sample pseudorandom noise images from the function  $Perlin_{h,w}(v)$ , where  $h$  and  $w$  are the image dimensions<sup>2</sup>. With the biased sample, the orthogonal perturbation is then calculated in the usual way:

$$r \sim Perlin_{h,w}(v) \quad (3.1)$$

$$\eta_{ortho,bias} = \frac{r}{\|r\|_2} - \delta \frac{r \cdot \delta}{\|r\|_2 \cdot \|\delta\|_2}$$

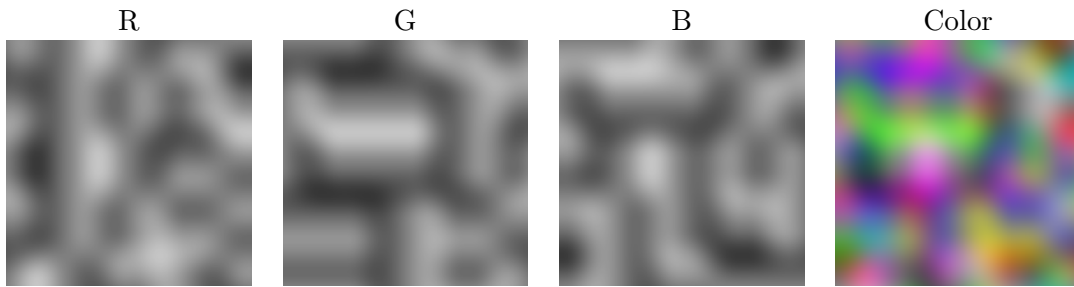
As a result, the samples are now strongly concentrated in low-frequency directions (see Figure 3.5(b)).

**Choosing a suitable frequency.** Naturally, noise frequency is a hyperparameter that needs to be chosen for optimal efficiency. Figure 3.7 shows a range of examples, with frequencies from 1 to 128. It seems intuitive that very low frequencies ( $\leq 1$ ) might not offer enough degrees of freedom for finding adversarial examples, while very high frequencies ( $\geq 128$ ) would hold little benefit over standard white noise patterns. This is confirmed by experiments in Chapter 4, which show that 32 delivers the highest efficiency boost for adversarial attacks on ImageNet.

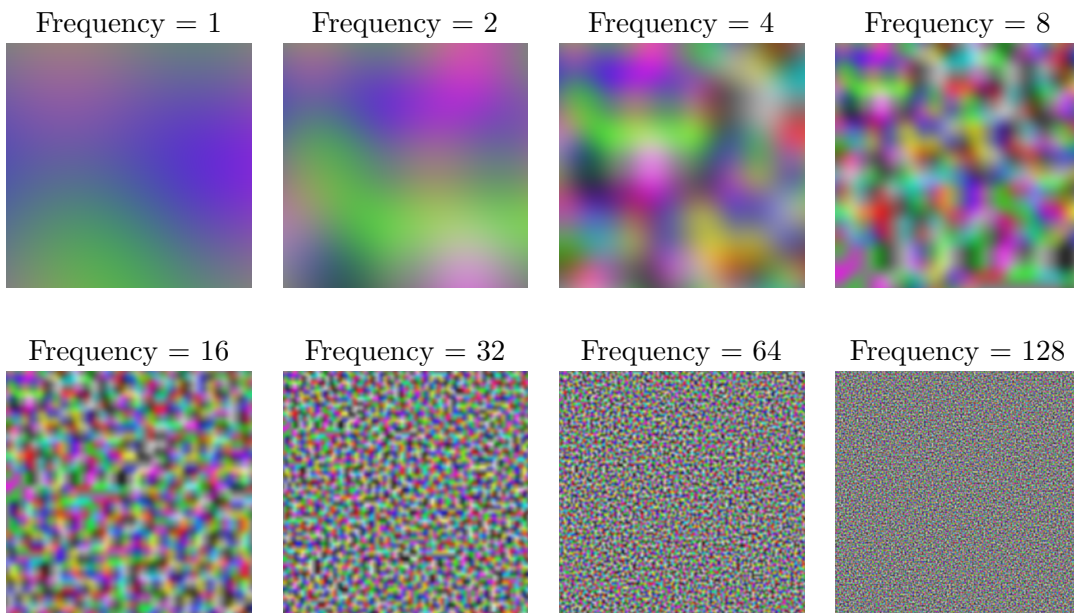
**Evaluation.** The experiments in Chapter 4 show that this low-frequency bias reduces the median number of queries required to fool an ImageNet classifier by roughly 70% (compared with an unbiased boundary attack). This empirically confirms the hypothesis and shows that the distribution of low-frequency noise patterns contains a higher concentration of adversarial directions than the normal distribution.

<sup>2</sup>To keep the notation simple,  $Perlin_{h,w}(\cdot)$  treats only the case of grayscale images. For color images of dimension  $k = h \cdot w \cdot c$ ,  $c$  a separate Perlin sample can be drawn for each channel, which results in a color image. See Figure 3.6 for an example.

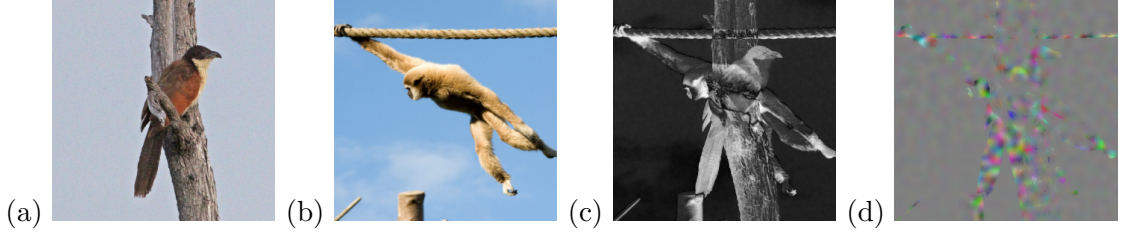




**Figure 3.6:** To obtain colored noise images, Perlin noise can be created separately for each channel and then combined into a color image. Each color channel is created from a different pseudo-random permutation vector.



**Figure 3.7:** Perlin noise with different frequencies. All images are noise vectors of magnitude  $\|\eta\|_2 = 1$  against a gray background, amplified 75-fold to make them visible. At very low frequencies, the pattern shows large blobs of nearly uniform color, while at very high frequencies the pattern looks similar to noise from a normal distribution (although not exactly the same). The maximum frequency is limited by image resolution (here: 299x299). For illustrative purposes, all images have been created from the same permutation vector (fixed random seed), which makes them display the same pattern in different scales. When used in the Biased Boundary Attack, noise generation is of course fully randomized.



**Figure 3.8:** Masking based on per-pixel image difference. (a) shows the original image, (b) an image of the target class. (c) is the difference mask, and (d) is a perturbation to which the mask has been applied. The perturbation concentrates on the central region, as the background is already quite similar between the images.

### 3.2.2 Regional masking

Currently, perturbations are evenly applied across the entire image. No matter if low or high frequency, the orthogonal step of a boundary attack perturbs all pixels nearly equally (when averaged over a large number of samples). This seems to be a waste – could the attack benefit from limiting the perturbation to specific image regions?

Boundary attacks perform an interpolation from  $x_{target}$  towards  $x_{orig}$ . In some regions, the images might already be quite similar, while in others they might differ significantly. Intuitively, one would want an attack to take larger steps in those regions where the difference is high. At the same time, it seems wasteful to modify pixels that are already similar, as any such distortion would have to be undone in a later step.

**Hypothesis.** Limiting perturbations to those image regions that show the highest difference could increase attack efficiency because it reduces the dimensionality of the search space. This could be a promising strategy to maintain efficiency at high resolutions.

**Creating masked perturbations.** It is straightforward to create an image mask  $|\delta|$  by taking the element-wise absolute difference between adversarial and original image (see Figure 3.8):

$$|\delta| = |x_{orig} - x_{adv}| \quad (3.2)$$

$$|\delta|_i = |\delta_i|; \quad |\delta| \in \mathbb{R}^k \quad (3.3)$$

At each step of the attack, this mask is recalculated based on the current position of  $x_{adv}$  and applied element-wise to the previously sampled orthogonal perturbation  $\eta_{ortho}$ :

$$\eta_{ortho,biased} = \frac{|\delta| \odot \eta_{ortho}}{\| |\delta| \odot \eta_{ortho} \|_2} \quad (3.4)$$

It is worth paying attention to the normalization step: Since  $|\delta|$  removes or dampens some dimensions of  $\eta_{ortho}$ , it automatically reduces the overall magnitude. Normalization then has an upscaling effect which amplifies the perturbation in those dimensions that were not masked. As a result, the same step size now allows for much stronger perturbations – confined to smaller regions of the image.



Per-pixel difference masking reduces the dimensionality of the search space, while at the same time allowing for larger perturbations of individual pixels in a single step. As a result, attack efficiency could be greatly increased.

**Evaluation.** The experiments in Chapter 4 show that regional masking based on image difference does indeed reduce the number of required queries by roughly 55% (compared with an unbiased boundary attack). This empirically confirms the hypothesis and shows that concentrating on specific regions of an image is a promising way of accelerating high-resolution black-box attacks.

### 3.2.3 Surrogate gradients

What other source of knowledge contains strong hints about directions that are likely to point to an adversarial region? Naturally – gradients obtained on a surrogate model. Transfer attacks are known to be extremely powerful (albeit brittle), so it could be useful to exploit surrogate models whenever they are available.

Arguably, the main weakness of transfer attacks is that they fail when the decision boundary of the surrogate model does not closely match that of the black box classifier. However, even when this is the case, the boundary may still be somewhere in the vicinity (see discussion in Section 2.4.2 of Chapter 2). Some advanced transfer attacks try to improve their success chance by adding small random perturbations to the adversarial gradient [16]. With any luck, they can find the adversarial region if it is nearby. Essentially, they supplement a gradient-based attack with limited random sampling.

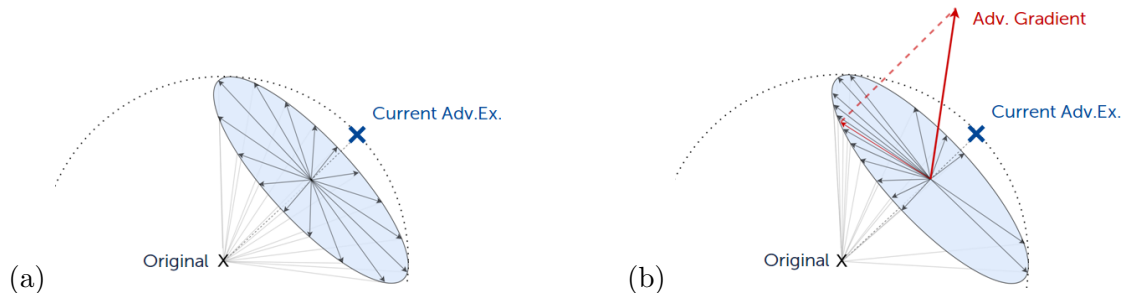
This work proposes to do exactly the opposite: to extend a sampling-based attack, such as the Biased Boundary Attack, with limited gradient information. Adversarial gradients can act as priors for the sampling procedure, increasing its efficiency whenever a surrogate is available. This should have the significant advantage that – in the case of low transferability – this approach would merely experience a slowdown where typical transfer attacks completely fail.

**Hypothesis (1).** Attack efficiency can be increased by biasing randomly sampled perturbations in the direction of an adversarial gradient calculated on a surrogate model.

**Hypothesis (2).** The attack can profit even from surrogate models that are considered too weak for direct transfer. While ”strong” surrogates – such as ensembles – should increase efficiency even further, they are not necessary to perform reliable attacks. In this way, the Biased Boundary Attack can benefit from a large collection of pre-trained models that are publicly available – without the need for costly fine-tuning.

**Implementing the gradient bias.** Gradient directions from surrogate models can be incorporated into the Biased Boundary Attack in the following way:

- An adversarial gradient is calculated at the position of  $x_{adv}$ , using the PGD method on a surrogate model (see Section 2.4 of Chapter 2). Since  $x_{adv}$  is already adversarial, it can be helpful to move a small distance towards the original image first,



**Figure 3.9:** Sampling directions for the orthogonal step of the Biased Boundary Attack. (a) Original (unbiased) scheme: Samples are uniformly distributed along the surface of the hypersphere. (b) With surrogate gradient bias: An adversarial gradient, obtained from a surrogate model, is projected onto the hypersphere. Samples are then biased so they concentrate towards this direction.

making sure to calculate the gradient from a point that is inside a non-adversarial region<sup>3</sup>.

- The adversarial gradient usually points away from  $x_{orig}$ , so a projection is performed to obtain the "sideways" component of the gradient direction. This vector, which is shown in Figure 3.9(b), can then be incorporated into the attack.
- As the projection is now on the same hyperplane as the randomly-sampled candidates for the orthogonal step, it becomes possible to bias the sampled directions towards the projected gradient by any method of our choosing. Provided all vectors are normalized, this can be done with simple addition:

$$\eta_{ortho,bias} = (1 - w) \cdot \eta_{ortho} + w \cdot Proj(\nabla L_{f_{surr}}(x_{adv}, y_{target})) \quad (3.5)$$

- The weight  $w$  controls the strength of the bias and is a hyperparameter that can be tuned according to the strength of the surrogate model. High values for  $w$  can be used when transferability is high, and vice versa. Were one to choose the maximum value,  $w = 1$ , then the orthogonal step would be equivalent to an iteration of the PGD transfer attack.

As a result, the sampling resolution is increased in the vicinity of the projected gradient. The rest of the search space is still covered, albeit with lower resolution. In this way, surrogate models are purely optional to the Biased Boundary Attack – instead of forming the central part, as is the case in classic transfer attacks.

It should be noted that at least *some* amount transferability should exist. Otherwise, the gradient will point in a bogus direction and using a high value for  $w$  would reduce

<sup>3</sup>Boundary attacks use an image of the target class as a starting point and then try to move towards the original image. Correspondingly, at each step, the current position is always classified as the adversarial target. PGD methods, in contrast, start with the original image and move away from it. It is therefore sensible to temporarily cross the boundary for performing a PGD-style gradient calculation.

efficiency instead of improving it. However, this seems rather unlikely. At the time of writing, no strategies were known that successfully eliminate transferability altogether [71]. Therefore, it can be expected that most state-of-the-art classifier models should deliver a benefit to the attack.

**Evaluation.** The experiments in Chapter 4 show that attack efficiency increases when incorporating gradients from surrogate models. Even when using a simple, publicly-available pre-trained classifier as surrogate model with very low  $w$ , the median number of queries is reduced by roughly 35%. This empirically confirms hypothesis (1). Crucially, the attack succeeds while a direct PGD transfer attack with the same surrogate fails in the majority of cases, which is evidence towards hypothesis (2). Additional experiments in Chapter 4 show that the Biased Boundary Attack can also benefit from stronger surrogates (such as ensembles), bringing the reduction up to 75% and more.

### 3.3 Efficient initialization

Apart from biasing the random search procedure, there is another opportunity for the Biased Boundary Attack to incorporate prior knowledge. Targeted boundary attacks are initialized with known images of the target class [15], but these are often far away from the image under attack. Could there be an easy way to reduce this distance?

It is evident that considerable effort has gone into the design of sophisticated search procedures for black-box adversarial attacks. Yet surprisingly, their initialization is not often discussed in detail. Surely, the choice of starting points could have an enormous impact on search efficiency?

Recent works offer only few pointers, most of which deal with PGD-style transfer attacks. There, small modifications such as random restarts [33] have been found to improve success rate. However, these are not directly applicable to boundary attacks, which require a starting point that already fulfills the adversarial criterion. Here, to the best of this author’s knowledge, only one method of initialization is known: from a data set of real images, pick a known image of the target class<sup>4</sup>. This is a rather important gap in current literature. Boundary attacks are known to suffer from low efficiency, and advanced initialization strategies have the potential to greatly improve it.

This Section discusses beneficial properties of starting points and how they can improve the efficiency of adversarial attacks. Drawing on these insights, a proof-of-concept initialization scheme is developed that synthesizes improved starting points from images of the target class. When combined with the Biased Boundary Attack, this reduces the number of required queries even further.

---

<sup>4</sup>For untargeted attacks, Brendel et al. [15] offer another strategy: They add gaussian noise to the original image in small increments until the label changes (the decision boundary is reached). While promising, this is not applicable to the targeted setting – it is very unlikely that the label will ever change to the desired class.

### 3.3.1 Criteria for suitable starting points

In image classification, an adversarial example is considered successful if it has a low distance to the original image and is at the same time classified as an adversarial label. Consequently, it seems reasonable to assume that these properties should be beneficial for attack efficiency:

- **Starting points should be close.** Intuitively, it makes sense to pick starting points that are already close to that goal. The optimization procedure would then merely refine them, requiring fewer iterations to arrive at an adversarial example or produce a better one in the same number of steps.
- **Starting points should reduce dimensionality.** Optimization often suffers from very large search spaces. An ImageNet example at a resolution of  $299 \times 299 \times 3$  has 268203 dimensions. It is therefore desirable to reduce this search space and concentrate only on specific dimensions.

The regional masking bias proposed in Section 3.2.2 already capitalizes on this insight. Therefore, a reduction in dimensionality should amplify the effect of the masking bias – ideally by not using a starting image entirely but only small regions of it. The mask bias would then limit the attack to a very small sub-region of the image, reducing the dimensionality of the search space even further.

Both are strong assumptions – certainly, they are not guaranteed to hold in general, as state-of-the-art classifiers have non-convex loss landscapes. Nevertheless, they can serve as intuition for a proof of concept, which can then be tested by experiment. As the rest of this Section shows, it is possible to synthesize images that aim to maximize these desirable traits, creating an initialization strategy that is more effective than simply picking the closest image.

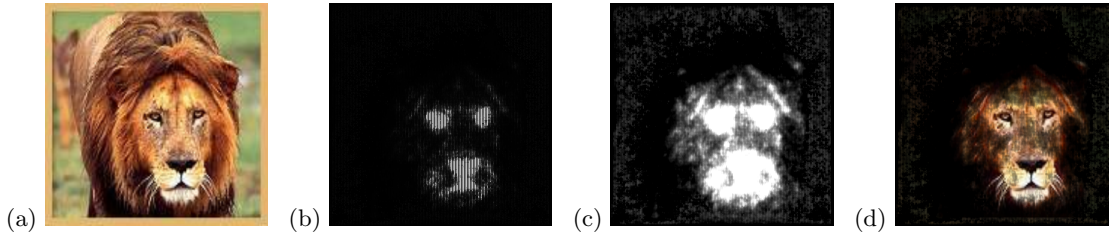
### 3.3.2 Initialization with salient patches

So far, starting points for boundary attacks have typically been found by searching a data set (e.g. the ImageNet validation set) for images of the target class and then picking the one that is closest to the image under attack:

$$x_{start} = \arg \min_x \|x - x_{orig}\|_2 \tag{3.6}$$

$$x \in \{x : f_{BB}(x) = y_{target}\}$$

This initialization is trivial whenever ground-truth annotations  $y_{gt} = y_{target}$  are available. As long as the classifier is reasonably accurate, there is a high chance that such an image will be classified as  $f_{BB}(x_{start}) = y_{target}$ , thereby fulfilling the adversarial criterion. However, full images of the target class typically score very badly on both closeness and dimensionality, so they are far from optimal. Therefore, it would be desirable to concentrate only on the most important regions of the image and to ignore others such as the background, which are unlikely to contribute to the classifier decision.



**Figure 3.10:** Segmentation of starting points. (a) An image of the target class, label "lion". (b) Saliency mask, calculated by an Inception-ResNet-v2 classifier [62]. (c) Smoothed and amplified mask to improve reliability. (d) The adversarial patch is formed by segmenting the image with the mask.

### 3.3.2.1 Segmentation by saliency

Naturally, an image of the target class contains features that classifiers react to strongly. In other words, these features are *saliient* for that class and image [87, 88]. Such features often tend to concentrate in small regions of an image (e.g. nose and eyes of an animal). The rest of the image matters little for the predicted class label and could therefore be discarded. Consider Figure 3.10, which shows an image of a lion on which a saliency map is computed. This map can be used as a mask that isolates the salient pixels of the target image. By blending only these pixels on top of the image under attack, misclassification can be achieved.

**Computing saliency maps.** Saliency maps are model-specific, and the salient pixels for a black-box classifier are unknown. It is therefore necessary to use a surrogate model for computing saliency maps. For this proof of concept, the simple gradient method proposed by Simonyan et al. [88] is used, together with SmoothGrad [89] for smoothing. The choice of method is not very important, as long as the saliency map covers pixels that are reasonably indicative of the target class – this can easily be confirmed by human inspection. To err on the side of safety, the saliency map is smoothed and amplified, as shown in Figure 3.10(c).

The computation of salient pixels offers potential for fine-tuning, however, this is out of scope for this work. The evaluation in Chapter 4 shows that even this simple method already increases attack efficiency by a considerable amount.

**Note on surrogates.** It is important to clarify that using surrogates to compute saliency is different from using them for adversarial transfer attacks. In adversarial transfer attacks, the attacker seeks to transfer tiny imperceptible perturbations between classifiers. Naturally, the success chance is relatively low. However, in the case of saliency maps, the goal is to transfer large contiguous image regions, which arguably contain the most striking features of the image. Any model that correctly classifies lions *must* react to the core features of a lion. Therefore, it is reasonable to assume that saliency maps, having been generously smoothed, transfer readily between classifiers.



**Figure 3.11:** Placing salient patches. (a) Original image, label "water bottle". (b) Starting point containing a salient patch of the target class, classified as the target label "lion". (c) Final adversarial example after running a Biased Boundary Attack, also classified as "lion". (d) Difference image.

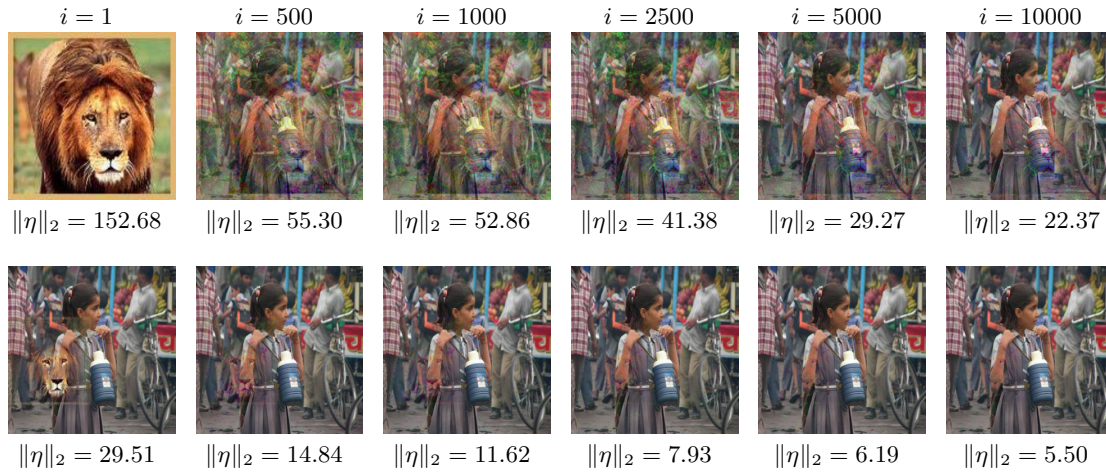
### 3.3.2.2 Placing salient patches

After the starting point has been segmented, the salient pixels form an "adversarial patch" that can change the classification of the entire image. The placement of such a patch offers another opportunity for reducing distance and dimensionality: It can be scaled to a smaller size and placed in a location where it replaces only a small number of pixels. Consider Figure 3.11: The adversarial patch is rather small, but still it causes the entire image to be classified as the target label.

To deliver a proof of concept, a simple brute-force method can be used: Salient patches are extracted from multiple images of the target class, randomly scaled and translated, and then blended over the original image. 50 such candidates are created and ranked by distance to the original image. The candidates are then tested against the surrogate model, and only those that are adversarially classified are then tried on the black box. In this way, it is possible to keep the number of queries to a minimum. The first image to be adversarially classified on the black box is then chosen as the final starting point.

All queries against the black box classifier must be added to the total query count of the attack. In the case that no success is found, an attacker may wish to fine-tune the procedure, increase patch size or fall back to full-sized images. However, as the evaluation in Chapter 4 shows, the method proposed here typically finds success within the first 10 candidates. These queries are well spent, as they can reduce perturbation size much more significantly than those spent on the subsequent attack, which often requires thousands of queries to cover the same distance. Figure 3.12 shows an example of this.

**Adversarial patches in other contexts.** The effect of adversarial patches is known and has been described by Brown et al. [90]. They construct small patches that are extremely salient and override all other features in the image, thereby changing classification. While this work exploits a similar effect, the adversarial patches of Brown et al. are strikingly visible to human observers. It would be possible to use them as starting points, but their strong visibility is contrary to the goal of this work – namely to make the perturbation imperceptible to humans.



**Figure 3.12:** Number of queries against perturbation magnitude for a Biased Boundary Attack. (Top): Initialized with a full image of the target class. (Bottom): Initialized with a salient patch. All images are classified as the adversarial label, "lion".

### 3.4 Summary

This Chapter has introduced the Biased Boundary Attack, a black-box adversarial attack that can incorporate various sources of prior knowledge about the target domain. Towards this end, three methods for biasing the random search procedure have been proposed, as well as an improved initialization scheme that synthesizes salient patches as starting points.

But how well do these methods work? How much efficiency can be gained, and how does the attack perform when compared to recent work in the field? In the next chapter, the methods proposed here are experimentally evaluated against a state-of-the-art image classifier, and systematic ablations are performed on their parameters.





## 4 Evaluation

Chapter 3 has described the methodology and intuition behind the Biased Boundary Attack – a black-box adversarial attack that aims to be both reliable and efficient. But how effective are the individual biases in detail? Can they be combined or do they interfere with each other? How strong is the effect of initialization? And how do the proposed methods compare to the state of the art?

This chapter performs an extensive experimental evaluation of the proposed techniques and tests them against a state-of-the-art image classifier. A total of 44 different attack configurations are evaluated, each generating hundreds of targeted black-box adversarial examples. Finally, the results are compared to recently proposed methods from literature, showing that the new attack outperforms prior work by a significant margin.

### 4.1 Setup

This section describes the experimental setup, the data and classifier on which the evaluation is performed, and finally the evaluation methodology and criteria for successful adversarial attack.

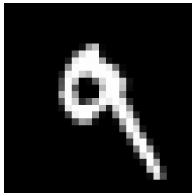
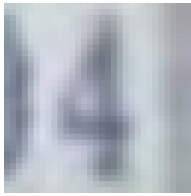


#### 4.1.1 Data set

All experiments are performed with images from the ImageNet classification data set [48]. It is one of the most widely used computer vision data sets and, together with MNIST [46], CIFAR10 [91] and SVHN[92], has become a staple of machine learning research [1, 15]. Table 4.1 shows a comparison of these data sets.

A common critique of adversarial example research is that attacks and defenses are often evaluated in scenarios that are unrealistic or trivial. It is therefore prudent to work with data that strikes a good balance between difficulty and realism:

- **ImageNet is difficult.** For a long time, research on adversarial examples focused on MNIST and CIFAR10, which are small data sets with low dimensionality. This makes any experiment significantly easier and faster to compute than on ImageNet. Some attacks can be performed quickly on MNIST, however on ImageNet, a single run may require millions of queries and several hours of time [15, 42]. Therefore, ImageNet can quickly reveal scalability issues in otherwise favorable approaches.
- **ImageNet is (reasonably) realistic.** The simplicity of MNIST, SVHN and CIFAR seems attractive, however it must be stressed that their low resolution and small number of classes do not represent real-world computer vision tasks too well.

## 4 Evaluation

	MNIST	SVHN	CIFAR10	ImageNet
				
Content	Scanned digits	House numbers	Various photos	Various photos
Resolution	$28 \times 28 \times 1$	$32 \times 32 \times 3$	$32 \times 32 \times 3$	$299 \times 299 \times 3$
Input dimensions	= 784	= 3,072	= 3,072	= 268,203
# Classes	10	10	10	1,000
# Images (train)	50,000	73,257	60,000	$\approx 1,200,000$
# Images (val)	10,000	26,032	10,000	50,000

**Table 4.1:** Comparison of popular computer vision data sets for machine learning. It is immediately apparent that ImageNet is the closest to data one might encounter in the real world. Similar data might be captured by the camera of a robot, or a user might submit similar photos to image processing services in the cloud. However, ImageNet also has high data dimensionality, which makes adversarial attack computationally expensive.

Both real-world robots and cloud image processing services rely on high-resolution images to recognize a wide range of objects.

Creating adversarial examples for ImageNet classifiers is both relevant and challenging. At the same time, the data is openly available (at least for academia) and has been in widespread use for several years already. This makes it an ideal dataset for benchmarking state-of-the-art adversarial attacks.

### 4.1.2 Evaluation procedure

First, an evaluation data set is created by randomly selecting 200 images from the ImageNet validation set. For each image  $x_{orig}$ , a random adversarial target class  $y_{target}$  is fixed. In this way, all experiments are performed on the same images and target classes. Appendix A contains a complete listing of all images and class labels.

Attacks must then produce an adversarial example  $x_{adv}$  for each pair  $(x_{orig}, y_{target})$  in the evaluation set. Attacks are allowed a maximum budget of 15000 queries per image, which amounts to roughly 5 minutes on the evaluation machine (see Section 4.1.7). Finally, the *success rate* and *query efficiency* over all images is calculated:

- **Success rate.** The number of successful adversarial attacks, divided by the total number of evaluation images. This measure can be determined for various query counts (e.g. 1000, 5000, 15000) and its development can be plotted, as the success rate is bound to increase together with the number of queries.

- **Query efficiency.** The median number of queries until success can serve as a single measure of efficiency. The median is preferable to the mean since any unsuccessful attack would greatly distort the mean (with the query count potentially approaching infinity). The median, however, can safely be calculated whenever the success rate is above 50%. Since any unsuccessful attack will have a higher query count than those that are already successful, it will not contribute to the median, no matter how much the number of queries. In this way, the median allows a direct comparison of attacks that have different success rates.

### 4.1.3 Success criterion

Boundary attacks always start with an image  $x_{adv} = x_{start}$  of the target class. Since  $f(x_{start}) = y_{target}$  by definition, this trivially satisfies the adversarial objective. Therefore, success must be characterized by low distance to the original image instead.

For this, a *success threshold*  $\epsilon_{success}$  is defined on the  $\ell^2$ -norm of the adversarial perturbation, and the adversarial attack is considered successful when misclassification is achieved with a perturbation smaller than this threshold:

$$\|\eta\|_2 = \|x_{adv} - x_{orig}\|_2 < \epsilon_{success} \quad (4.1)$$

How to choose  $\epsilon_{success}$ ? Since  $\ell^2$  scales with the number of dimensions, it depends on image resolution. As discussed in Chapter 2,  $\ell^2$  is preferable to  $\ell^\infty$ , but prior work that uses  $\ell^\infty$  can provide some intuition. Many established works [42, 34, 75] use values between 0.05 and 0.1 as  $\ell^\infty$ -threshold, and therefore a comparable  $\ell^2$ -threshold should be picked. For all evaluations,  $\epsilon_{success}$  is set to 25.89, which is the  $\ell^2$ -norm that corresponds to a worst-case perturbation with an  $\ell^\infty$ -norm of 0.05:

$$\|\eta\|_2 = \sqrt{\sum_{i=1}^k \eta_i^2} = \sqrt{299 \cdot 299 \cdot 3 \cdot (0.05)^2} \approx 25.89 \quad (4.2)$$

Figure 4.1 shows a number of perturbations with varying  $\|\eta\|_2$ . For all practical purposes, perturbations of norm  $\|\eta\|_2 < 25.89$  are difficult to spot and can be considered successful adversarial examples.

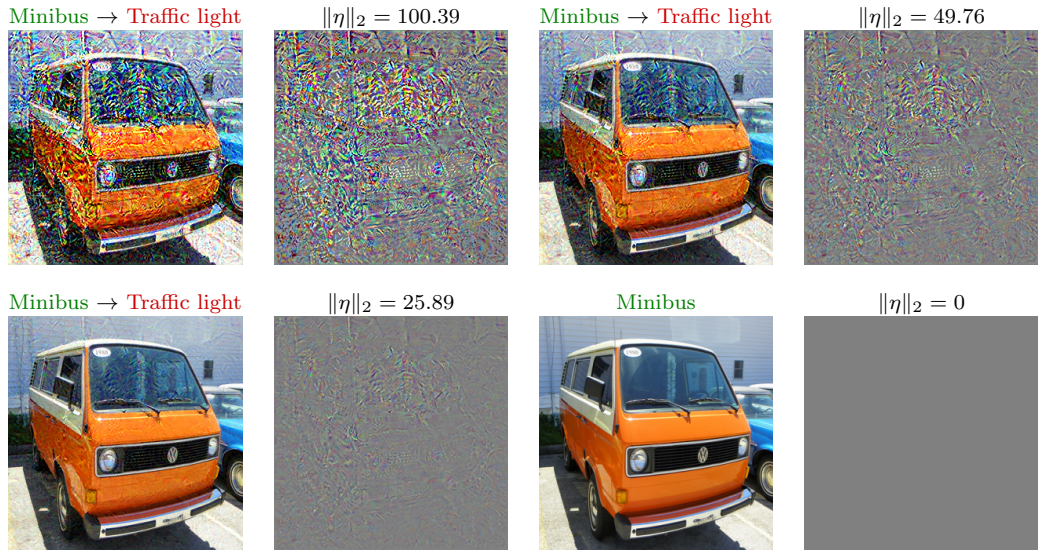
### 4.1.4 Choice of classifier under attack

In order to directly compare adversarial attacks, it is necessary to pick a classifier on which the attacks are performed. But which is the most suitable?

It is well known that adversarial vulnerabilities affect all state-of-the-art deep learning architectures [71] – therefore the choice of classifier architecture is not crucial. For this reason, it is admissible to pick a single architecture and use it as a common baseline on which the attacks are evaluated.

In this chapter, all attacks are conducted against an Inception-v3 CNN classifier [47], which has been popular in recent works on adversarial attack [15, 42]. To guarantee

## 4 Evaluation



**Figure 4.1:** Adversarial perturbations of varying magnitude. All images (except the last) are wrongly classified as "traffic light". While perturbations of magnitude 25.89 (the success threshold) can still be spotted by a careful observer, it is easy to disregard them as seemingly random noise. As such, they are sufficient for considering an attack successful.

reproducibility, no manual training is performed and pre-trained weights are used. These weights are available for download in most major deep learning frameworks; for the experiments in this chapter, they are taken from the official TensorFlow repository<sup>1</sup>. This model has a top-1 classification accuracy of 78.0%.

It should be noted that the evaluation data set (Section 4.1.2) is formed from the ImageNet validation set, not the training set. Since the classifier was trained only on the ImageNet training set, it has never seen the evaluation images before.

### 4.1.5 Choice of surrogate model

Some of the attacks compared in this evaluation use surrogate models to derive adversarial gradients. These include PGD-transfer attacks, but also the Biased Boundary Attack when using a surrogate bias. It is therefore necessary to specify a surrogate model for some of the experiments. To emulate attacks on unknown models, the surrogate must be sufficiently different to the model under attack. However, a number of reasonable assumptions can be made:

- **Unknown architecture.** The model under attack is a black box, so one must assume that its architecture and parameters are unknown and that they might differ significantly from any surrogate model that is available to the attacker. To simulate an attack in the real world, the surrogate must have a different architecture and be trained independently of the black box.

<sup>1</sup>See <https://github.com/tensorflow/models/tree/master/research/slim>.

- **Known model family.** On the other hand, it is safe to assume that most state-of-the-art image classifiers are CNNs of some sort. The surrogate can therefore be limited to the family of CNNs.
- **Known classification task.** It is also reasonable to assume that the broader task of classification is known, mapping from images to a certain set of classes. Therefore, the available classes are also known – either completely, or at least a subset that is relevant to the attacker.
- **Known data.** Finally, it is acceptable to use a surrogate that is trained with similar data as the black box. This seems a very strong assumption, but it can be justified from prior work that demonstrates adversarial transfer between classifiers that were trained on significantly different data [93]. As long as similar classes are contained, the exact choice of data set is not crucial.

For most real-world tasks, an attacker would surely be able to acquire data that is at least similar to that used by the victim. Also, ImageNet is a very popular general-purpose data set, so it is likely that many real-world systems use data that is similar to – or even directly incorporates parts of – ImageNet. Therefore, the choice of training data for the surrogate is not critical for transferability and it is permissible to use ImageNet-trained surrogates.

It should be noted that the evaluation data set (Section 4.1.2) is formed from the ImageNet validation set, not the training set. Care must be taken that the surrogates were trained only on the training set and that they have not seen the evaluation images before.

- **Simplicity.** It has been found that transferability increases with ensembling of surrogates [74]. In fact, this is often necessary to achieve the strong performance reported by PGD-transfer attacks [17]. While the surrogate bias certainly warrants a comparison of different surrogates (see experiment in Section 4.5), extensive fine-tuning of ensembles goes beyond the scope of this work. For the majority of experiments, any general-purpose classifier can be chosen that is reasonably accurate on ImageNet.

Following these assumptions, an Inception-ResNet-v2 classifier [94] is picked. Again, no manual training is performed, and the pre-trained weights made available by the original authors are used. The model used in this evaluation has a top-1 accuracy of 80.4% and is available in the official TensorFlow repository<sup>2</sup>.

#### 4.1.6 Enforcing the black-box setting

During the evaluation, both the classifier and the adversarial attack are running on the same machine. This means that, theoretically, the attacker could have white-box access to the classifier internals. To prevent any information leak, the following measures are taken:

<sup>2</sup>See <https://github.com/tensorflow/models/tree/master/research/slim>.

- **No access to classifier internals.** The classifier is isolated in a separate software component which offers only the classification function in a public interface, thereby emulating a black-box setting. The attack sends queries to the black box and receives classification results. It cannot query the classifier for gradients, parameters or any other internals.
- **Discrete input.** The black box component accepts as input only valid image data in the RGBU8 format. Here, each input dimension (pixel  $\times$  color channel) is an integer value in the interval of  $[0, 255]$ . It is necessary to enforce this constraint since RGBU8 is the predominant image format in most imaging applications and web services today. Any attack that might be used to attack real-world systems must be able to deal with this limitation.
- **Discrete output.** As discussed in Chapter 2, an attacker may gain large amounts of information if the classifier returns real-valued confidence scores. At the same time, this is not a reasonable assumption in the real world, so any attack should be able to succeed without them. Although Inception-v3 does produce confidence scores, the black box component hides them from the attacker and returns only the *argmax* of the confidence vector, which is the class label  $\hat{y}$  of the most confident prediction. This enforces a label-only classification setting.
- **One image per query.** In deep learning, it is common to process images in large batches. However, this does not necessarily fit a real-world attack setting. It may or may not be possible to bombard a web service with 128 images per second – in the end, any attack must be able to deal with strictly sequential requests.

In this way, the black-box software component guarantees a realistic query-limited label-only black-box setting in which attacks can be evaluated.

### 4.1.7 Software and hardware setup

The evaluation procedure is implemented using Python 3.6 [95] and NumPy 1.18.5 [96]. The neural network classifiers are run using TensorFlow 1.10 [97] with GPU support. All experiments are performed on a consumer-grade desktop machine with an AMD Ryzen 1300X quad-core CPU, 16GB RAM and an NVIDIA GeForce 1070 GPU.

On this machine, processing 15000 sequential classification queries takes roughly 5 minutes, which is a reasonable upper limit for an attack on a single image. Nevertheless, running all of the different attack configurations (44) on the entire evaluation set (200) results in a runtime of more than 30 days.

## 4.2 Ablation study of biases

The first experiment is an ablation study that gives an overview of all biases. This shows how effective they are individually, and also whether their gains can be combined. The experiment can also be seen as a sanity check: Without performing any fine-tuning, it verifies that the ideas proposed in Chapter 3 do indeed increase efficiency by a significant amount.

### 4.2.1 Setup

The experiment consists of 8 runs, each a Biased Boundary Attack with a different combination of biases. Each attack creates targeted adversarial examples for all 200 images in the evaluation set.

### 4.2.2 Hyperparameters

This study is performed with the following settings (see the following sections for ablations on individual hyperparameters):

- Perlin noise frequency: 32
- Mask strength: 1.0
- Mask channels: 3 (color)
- Surrogate model: Inception-Resnet-v2
- Surrogate weight  $w$ : 0.001
- Initialization: Closest image of the target class

### 4.2.3 Results

Table 4.2 and Figure 4.2 show the result: It is apparent that each bias significantly increases attack efficiency. The largest boost is obtained by the Perlin bias, followed by the mask bias, and finally the surrogate gradient bias.

It is worth noting that all biases can be combined and that they do not interfere with each other. When all three biases are active, the attack reaches 94% success after 15000 queries, with a median of only 3906 queries required until a satisfactory adversarial example is produced. This is a reduction of more than 80%!

**Note on extrapolated medians.** Some attacks have below 50% success rate after 15000 queries. It is therefore not possible to determine the median number of queries until success. To still get a rough overview, these values are linearly extrapolated from the slope between 10000 and 15000. While imprecise, linear extrapolation likely underestimates the true median (see flattening curves in Figure 4.2) and can therefore be used as a conservative estimate.

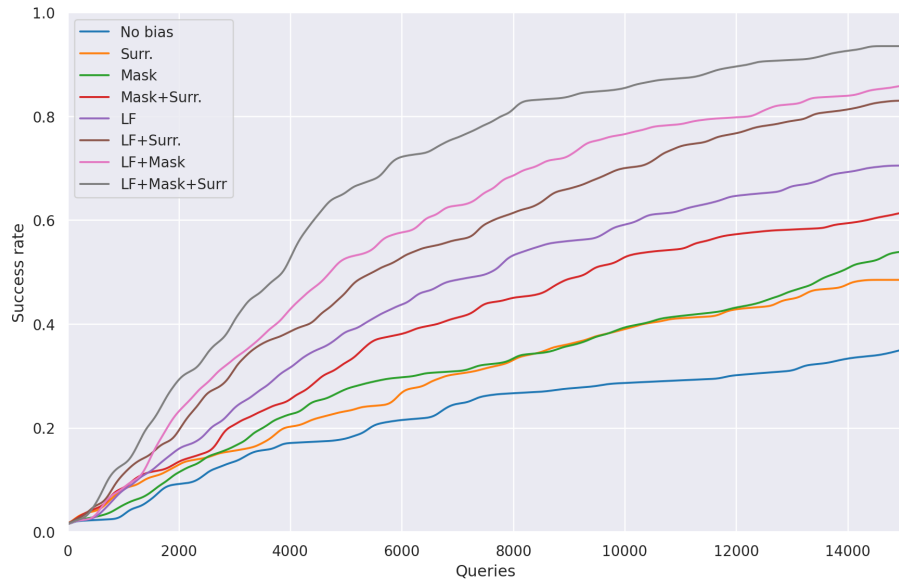
Figure 4.3 plots the distance to the original image against the number of queries for each attack configuration. It is clear that each bias delivers an improvement not only in the median number of queries but also in the mean norm of the perturbation.

## 4 Evaluation

Figure 4.4 shows adversarial examples created by different configurations. It is clear that image quality is drastically increased by the biases, and with all biases combined the adversarial perturbation is barely perceptible.

ACTIVE BIASES			SUCCESS RATE VS NUMBER OF QUERIES						MEDIAN QUERIES
LF	MASK	SURR.	500	1000	2500	5000	10000	15000	UNTIL SUCCESS
NO	NO	NO	0.02	0.03	0.11	0.17	0.28	0.35	$\approx 25357$
NO	NO	YES	0.04	0.08	0.14	0.23	0.39	0.49	$\approx 15789$
NO	YES	NO	0.03	0.05	0.14	0.28	0.39	0.54	13914
NO	YES	YES	0.04	0.08	0.15	0.32	0.53	0.62	9375
YES	NO	NO	0.03	0.08	0.19	0.39	0.59	0.70	7649
YES	NO	YES	<b>0.05</b>	0.11	0.27	0.46	0.70	0.83	5544
YES	YES	NO	0.03	0.08	0.28	0.53	0.76	0.86	4751
YES	YES	YES	<b>0.05</b>	<b>0.13</b>	<b>0.34</b>	<b>0.65</b>	<b>0.85</b>	<b>0.94</b>	<b>3906</b>

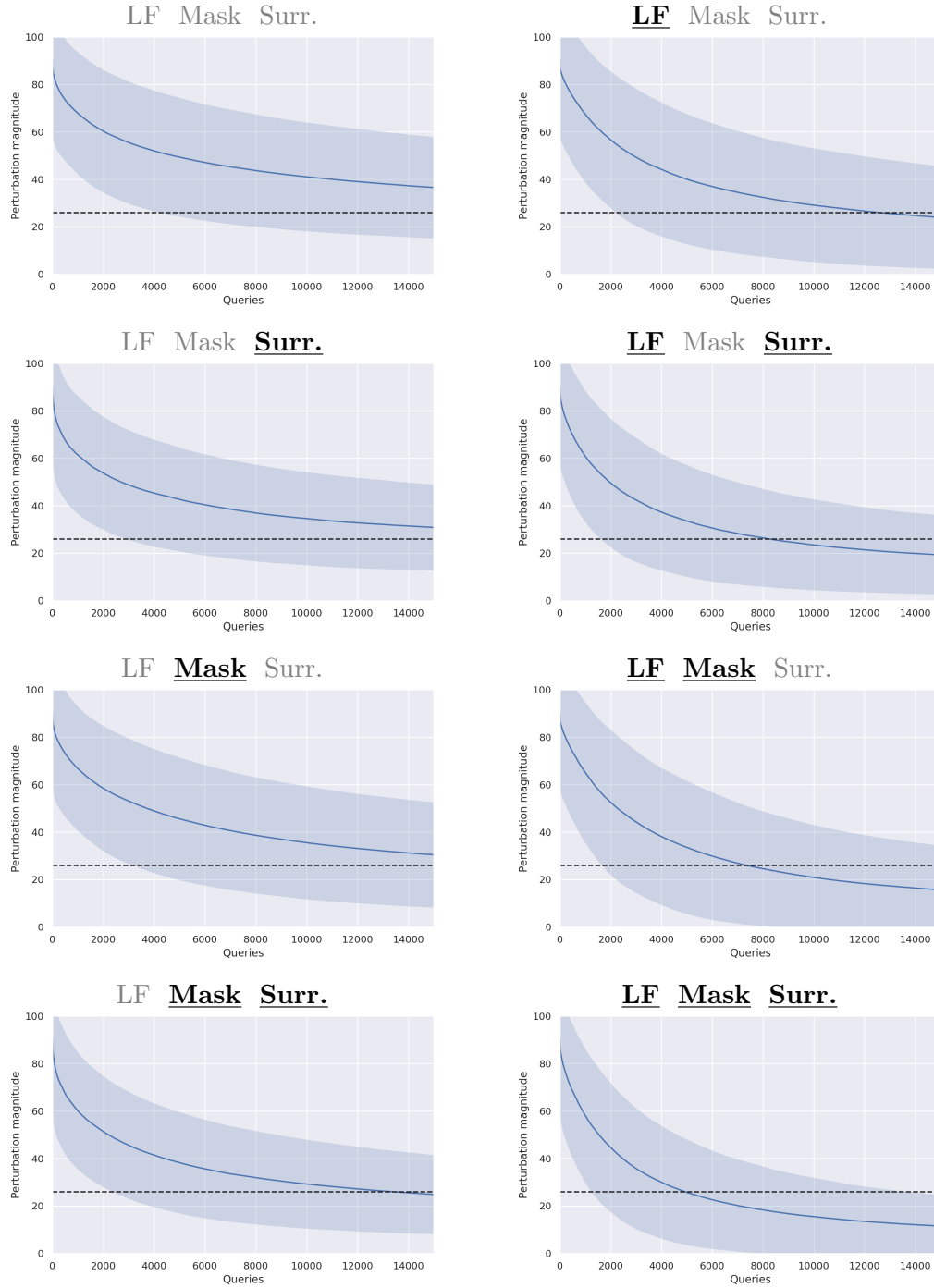
**Table 4.2:** Ablation study of biases. The low-frequency (LF) bias has the strongest effect, followed by the mask bias and finally the surrogate bias. Each bias improves efficiency on its own, and the combination of all biases delivers the strongest performance. *Median values over 15000 are extrapolated, as the median is only available for success rates  $\geq 0.5$ .*



**Figure 4.2:** Ablation study of biases. The success rate of all configurations is plotted against the number of queries. This data is the same as in Table 4.2. The runs do not start at zero, but slightly above - the reason is that, in some cases, an image of the target class could be found that is already very close to the image under attack. For these few images, the attack is considered successful after the first query.

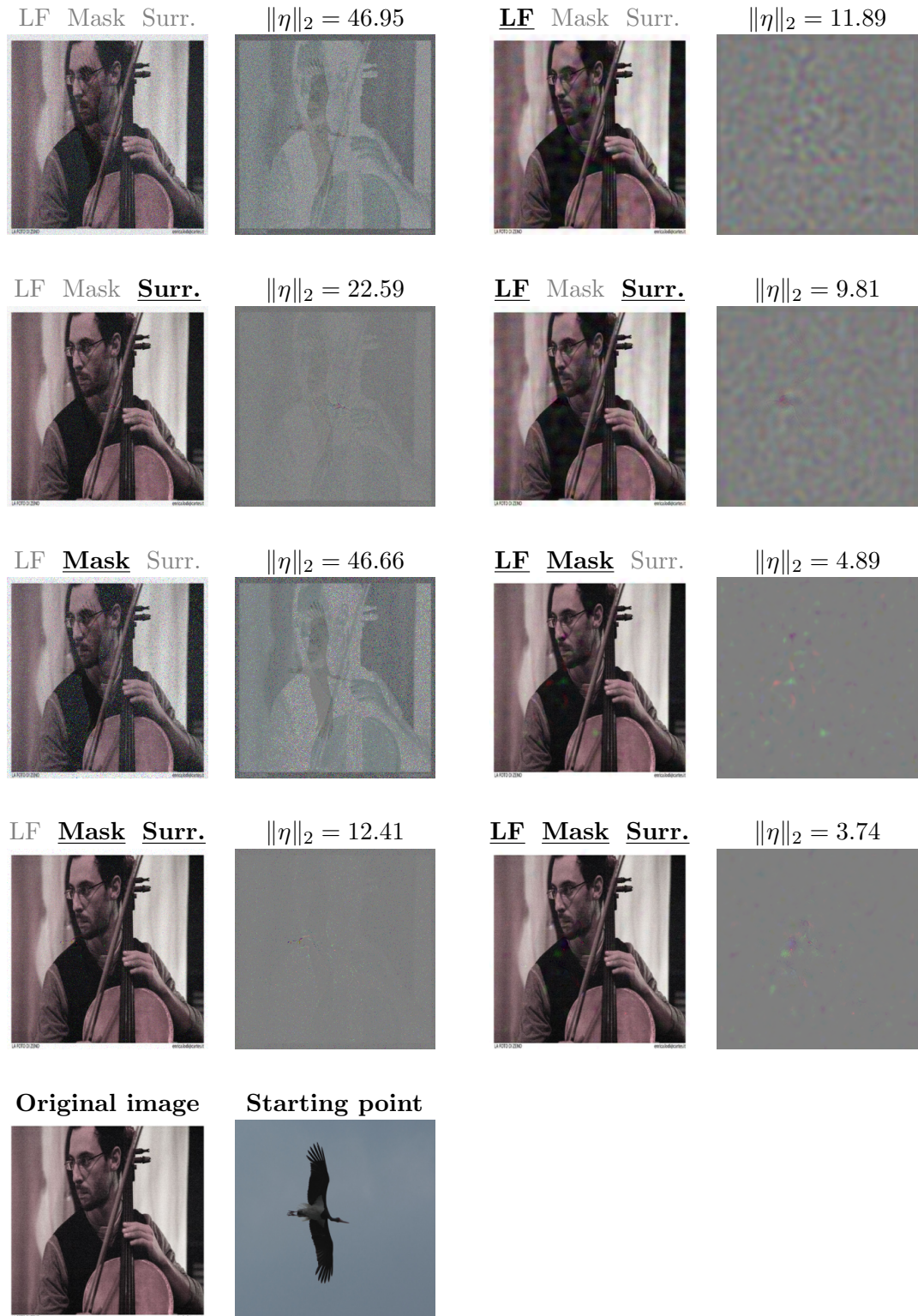


## 4.2 Ablation study of biases



**Figure 4.3:** Ablation study of biases. For each configuration, the perturbation magnitude  $\|\eta\|_2$  is plotted against the number of queries. The solid line shows the mean perturbation across all 200 images, while the shaded area shows the standard deviation. The dashed line is the success threshold. All three biases consistently enhance efficiency. Notably, mean queries until success are higher than the median reported in Table 4.2. This hints at a small number of outliers that require a very large number of queries, while in most cases an adversarial example is found quickly.

#### 4 Evaluation



**Figure 4.4:** Ablation study of biases. These are the final adversarial examples obtained after 15000 iterations for the same original image and target class. The original class is "cello" – all images are misclassified as the target "black stork".

## 4.3 Low-frequency bias

The ablation study shows that the low-frequency bias yields a very high improvement in efficiency. This is strong evidence towards the hypothesis presented in Section 3.2.1 of Chapter 3. In this section, two experiments are performed to determine the optimal frequency and to explore further opportunities for fine-tuning.

### 4.3.1 Comparison of frequencies

First, a simple grid search is performed to find the most efficient image frequency. 8 Biased Boundary Attacks are performed, with each run doubling the noise frequency from the previous (1 to 128). This amounts to a grid search on the frequency exponent:  $2^0$  to  $2^7$ .

### 4.3.2 Randomized combination of frequencies

While the first experiment determines a single frequency that is best on average, it is not certain that this is optimal in all cases. It could very well be that different frequencies are more efficient at various stages of the attack, or that different images might benefit from different frequencies.

To investigate this, another study is performed that chooses frequency randomly, effectively picking a different frequency for each subsequent step. Since each adversarial example is the cumulative result of up to 15000 such perturbations, this approximates a combination of all frequencies across the spectrum.

Two Biased Boundary Attacks are performed. First, the frequency exponent is sampled from a uniform distribution:  $freq \sim \mathcal{U}(0,7)$ . This attack combines all frequencies equally. However, anticipating the results of the first experiment, it is likely that some frequencies have a higher probability of success than others – therefore a weighted distribution could be more efficient. To test this, a second run is conducted with the frequency exponent sampled from a truncated normal distribution, centered around the strongest-performing frequency from the first experiment ( $32 = 2^5$ ):  $freq \sim \mathcal{N}(5,2)$ . For brevity, the two attacks are named "Rand-U" and "Rand-N". Figure 4.5 shows both distributions.

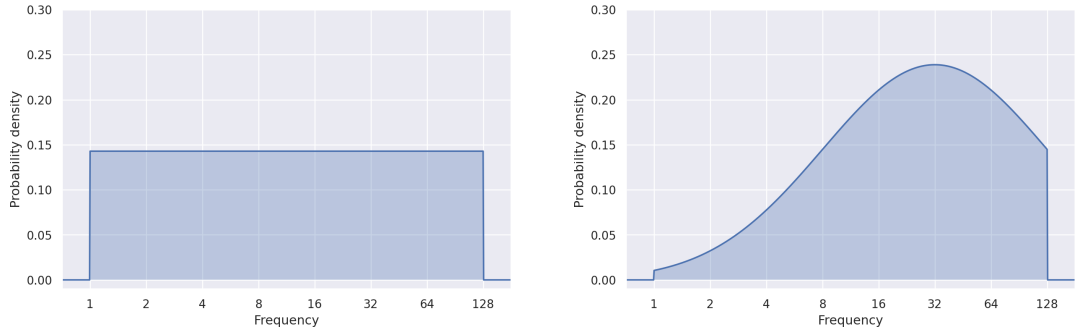
### 4.3.3 Hyperparameters

For this comparison, the low-frequency bias is studied in isolation. Activating the mask and surrogate bias would likely dilute the effects observed by comparing frequencies, and comparing all bias configurations with all frequencies would incur prohibitive costs. As before, all attacks are initialized with the closest image of the target class.

### 4.3.4 Results

Table 4.3 and Figure 4.6 show the results of all runs, and Figure 4.7 shows adversarial examples generated with different frequencies.

## 4 Evaluation



**Figure 4.5:** Probability distributions for randomized choice of frequency. At each step of the attack, the frequency for the next perturbation is drawn randomly. (Left): Uniform,  $2^{\mathcal{U}(0,7)}$ . (Right): Truncated normal,  $2^{\mathcal{N}(5,2)}$ , centered around the strongest-performing frequency ( $32 = 2^5$ ).

**Comparison of frequencies.** Mid-range frequencies, between 32-64, deliver the highest boost in efficiency. In particular, setting the noise frequency to 32 single-handedly doubles the final success rate when compared to an unbiased attack. This is also the frequency chosen for the bias ablation study in Section 4.2.

**Randomized combination of frequencies.** While the uniform distribution (Rand-U) performs quite well, it is slightly less effective than manually choosing the best frequency (32). However, the normal distribution (Rand-N) centered around 32 manages to yield even better results than the singular choice of 32. This hints at two things: Firstly, there may not be a single frequency that is always optimal – this is hardly a surprise. Secondly, it is possible to fine-tune perturbation patterns, and doing so can yield another substantial boost in efficiency.

It is important to note that the distribution (Rand-N) was retroactively modeled after the success rates obtained by the first experiment, which would be considered bad practice in statistics. In other words, the noise patterns are specifically engineered to work well in the same setting as they were first evaluated. It is highly likely that the attack is now overfitting to the model under attack, or perhaps the evaluation dataset. Therefore, no claim to generality can be made, and the Rand-N attack can serve only as a toy example of fine-tuning.

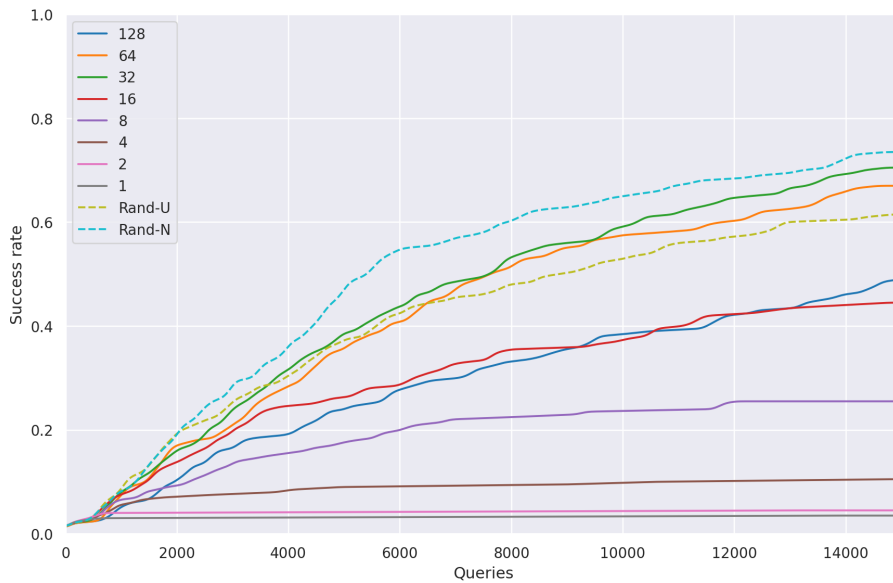
It could be interesting to further evaluate the generality of specific frequency distributions on hold-out testing datasets, but this is outside the scope of this work. Even without fine-tuning, low-to-mid-frequency noise drastically improves attack efficiency. An attacker may wish to perform fine-tuning, but the gains already obtained by uniform choice (Rand-U) are so large that it might not be necessary in many cases.

**Striking appearance of patterns.** Consider Figure 4.7: Although frequencies between 16 and 64 nominally achieve the lowest perturbation magnitude, their patterns may appear striking and repetitive to a human observer – perhaps more than higher frequencies. It can be seen that combining multiple frequencies (Rand-U/N) alleviates this issue and produces more naturally-looking patterns.

### 4.3 Low-frequency bias

PERLIN FREQ.	ACTIVE BIASES			SUCCESS RATE VS NUMBER OF QUERIES						MEDIAN QUERIES
	LF	MASK	SURR.	500	1000	2500	5000	10000	15000	
-	NO	NO	NO	0.02	0.03	0.11	0.17	0.28	0.35	≈25357
128	YES	NO	NO	0.02	0.05	0.13	0.23	0.38	0.49	≈15455
64	YES	NO	NO	0.02	0.06	0.18	0.35	0.57	0.67	7668
32	YES	NO	NO	0.03	<b>0.08</b>	<b>0.19</b>	<b>0.39</b>	<b>0.59</b>	<b>0.70</b>	<b>7649</b>
16	YES	NO	NO	0.03	<b>0.08</b>	0.16	0.26	0.37	0.44	≈18566
8	YES	NO	NO	0.03	0.06	0.11	0.17	0.23	0.25	≈76250
4	YES	NO	NO	0.03	0.05	0.07	0.09	0.09	0.10	>100000
2	YES	NO	NO	0.03	0.04	0.04	0.04	0.04	0.05	>100000
1	YES	NO	NO	0.03	0.03	0.03	0.03	0.03	0.04	>100000
RAND-U	YES	NO	NO	<b>0.03</b>	<b>0.09</b>	0.22	0.38	0.52	0.62	9013
RAND-N	YES	NO	NO	<b>0.03</b>	0.08	<b>0.25</b>	<b>0.47</b>	<b>0.64</b>	<b>0.74</b>	<b>5427</b>

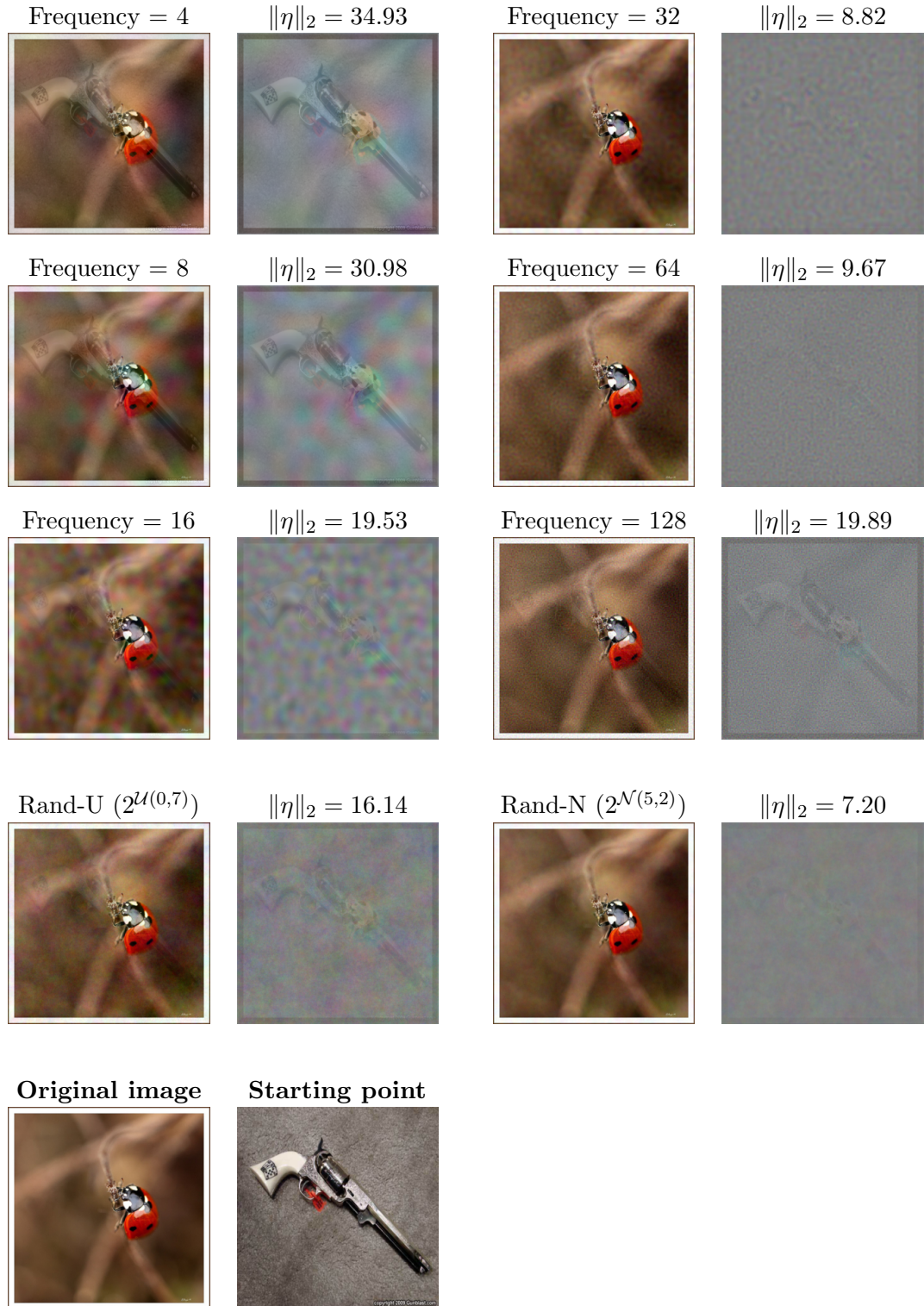
**Table 4.3:** Comparison of Perlin Noise frequency for the low-frequency bias. Upper segment: Unbiased attack as baseline. Central segment: Different Perlin frequencies. Mid-range frequencies (between 32-64) deliver a great boost in efficiency, while extremely low frequencies hurt performance instead. Lower segment: Randomized combination of frequencies. Rand-U: Frequency exponent sampled uniformly. Rand-N: Frequency exponent sampled from a truncated normal distribution.



**Figure 4.6:** Comparison of noise frequency (same data as in Table 4.3). It is evident that frequencies below 8 get stuck quickly and cannot proceed, no matter the amount of queries. The strongest frequencies, 32 and 64, exhibit similar performance at both low and high query counts. At low query counts, Rand-U and Rand-N strongly outperform all other attacks, but lose much of this advantage as the number of queries increases.



#### 4 Evaluation



**Figure 4.7:** Adversarial examples obtained by a low-frequency Biased Boundary Attack with different frequencies. 32 is the most effective frequency, however it is outperformed by Rand-N, which is a random combination of frequencies centered around 32. Frequencies 1 and 2 are omitted for brevity. The original label is "ladybug", but all images are classified as "revolver".

## 4.4 Regional masking bias

The ablation study in Section 4.2 has shown that the regional masking bias yields a considerable improvement in efficiency. With this, the hypothesis presented in Section 3.2.2 of Chapter 3 is confirmed. In this section, two additional experiments are performed to determine the optimal mask strength and to explore the relationship between masking and low-frequency biases.

### 4.4.1 Mask strength

The first experiment compares different mask strengths. Section 3.2.2 of Chapter 3 introduces the masking bias as a binary option, i.e. the perturbation is either masked according to the current image difference or it is not. However, instead of simply activating the mask and observing an improvement, it would be more instructive to compare multiple masks at different strengths.

The simplest way to achieve this is to interpolate between a perturbation with the bias enabled and one without. If  $\eta$  is an unbiased perturbation and  $\eta_{biased}$  is already masked, then:

$$\eta_{interp} = \frac{\eta + s(\eta_{biased} - \eta)}{\|\eta + s(\eta_{biased} - \eta)\|_2} \quad (4.3)$$

With this, the regional masking bias can be scaled between 0 and 1 via the hyperparameter  $s$ . For the rest of this section,  $s$  is called *mask strength*.

### 4.4.2 Color channels

From the ablation study in Section 4.2, it is clear that the masking bias synergizes well with the low-frequency bias. However, on closer observation, it becomes apparent that the masking bias tends to amplify a small number of brightly-colored blobs, which then become very noticeable. This happens in spite of a very small perturbation norm. Consider the examples where both biases are enabled in Section 4.2, Figure 4.4: Although the masking bias reduces perturbation magnitude, it makes a select few blobs more noticeable.

This might be caused by the difference mask  $|\delta|$  being calculated not per pixel, but per dimension: At a single pixel, the mask can have different values in each of the three color channels. If, for example, the difference is near-zero for the red and green channels of a pixel, but large for the blue channel, then the resulting perturbation will almost always be blue. Since the attack performs many such steps, it can develop a feedback loop where existing color differences are reinforced.

## 4 Evaluation

To address this imbalance, an experiment is conducted in which the difference mask  $|\delta_{gray}|$  is calculated in grayscale:

$$|\delta_{x,y,gray}| = \max(|\delta_{x,y,R}|, |\delta_{x,y,G}|, |\delta_{x,y,B}|) \quad (4.4)$$

As a result, the colored blobs should become less prominent – possibly at the cost of query efficiency. This experiment compares the performance of per-channel (color) masks with single-channel (grayscale) masks. All runs in this experiment are conducted with full mask strength,  $s=1.0$ .

### 4.4.3 Hyperparameters

For this comparison, the masking bias is studied first in isolation, and then in combination with the low-frequency bias (Perlin frequency 32). The surrogate bias is disabled, as it would likely dilute the observed differences. As before, all attacks are initialized with the closest image of the target class.

### 4.4.4 Results

Tables 4.4 and 4.5 contain the results, while Figure 4.8 shows a plot of all runs combined.

**Mask strength.** The attack with full mask strength performs the best, both on its own and when combined with the low-frequency bias. It therefore seems to make little sense to perform tuning on this hyperparameter – as opposed to Perlin frequency, where there exists a sweet spot (32) in the middle of the spectrum.

**Color channels.** The results in Table 4.5 suggest that using a grayscale mask performs significantly worse than a color mask. However, qualitative inspection paints a different picture. Consider Figure 4.9, which shows a picture of a mushroom with grayscale and color masks. It is plain to see that, while the grayscale mask performs worse in terms of perturbation norm, it also contains more muted colors that are less noticeable.

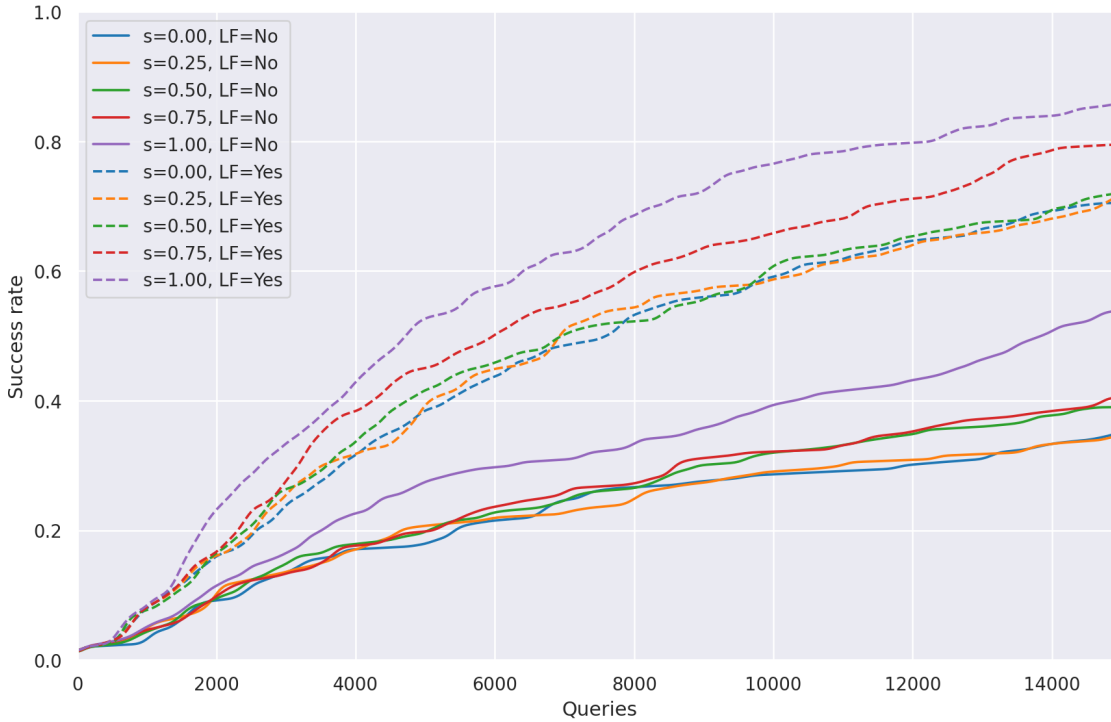
How to assess this result? On the one hand, the problem of the bright blobs is addressed, but on the other hand, this perceptual difference cannot easily be measured. Grayscale masks therefore can be a tool to increase the visual fidelity of adversarial examples, but should not be activated by default since they can be detrimental to performance as measured in this work.



#### 4.4 Regional masking bias

MASK STRENGTH $s$	ACTIVE BIASES			SUCCESS RATE VS NUMBER OF QUERIES						MEDIAN QUERIES
	LF	MASK	SURR.	500	1000	2500	5000	10000	15000	
0.00	NO	NO	NO	0.02	0.03	0.11	0.17	0.28	0.35	$\approx 25357$
0.25	NO	YES	NO	0.02	<b>0.05</b>	0.12	0.21	0.29	0.35	$\approx 29090$
0.50	NO	YES	NO	0.02	0.04	0.13	0.20	0.32	0.39	$\approx 22857$
0.75	NO	YES	NO	<b>0.03</b>	<b>0.05</b>	0.12	0.20	0.32	0.41	$\approx 20588$
1.00	NO	YES	NO	<b>0.03</b>	<b>0.05</b>	<b>0.14</b>	<b>0.28</b>	<b>0.39</b>	<b>0.54</b>	<b>13914</b>
0.00	YES	NO	NO	<b>0.03</b>	<b>0.08</b>	0.19	0.39	0.59	0.70	7649
0.25	YES	YES	NO	<b>0.03</b>	<b>0.08</b>	0.19	0.40	0.59	0.72	6940
0.50	YES	YES	NO	0.02	<b>0.08</b>	0.20	0.42	0.61	0.72	7008
0.75	YES	YES	NO	<b>0.03</b>	<b>0.08</b>	0.24	0.45	0.66	0.80	5983
1.00	YES	YES	NO	<b>0.03</b>	<b>0.08</b>	<b>0.28</b>	<b>0.53</b>	<b>0.76</b>	<b>0.86</b>	<b>4751</b>

**Table 4.4:** Comparison of different mask strengths. The attack with full mask strength is the most efficient, both on its own and when combined with the low-frequency bias.

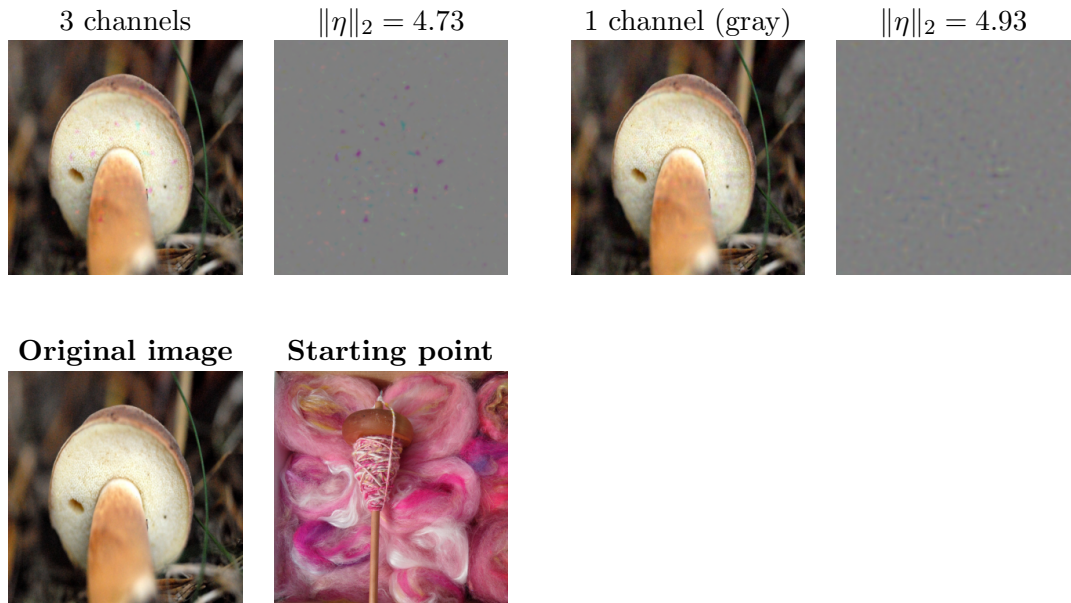


**Figure 4.8:** Comparison of different mask strengths (plot of Table 4.4). The attack with full mask strength is most efficient, both on its own and when combined with the low-frequency bias. The relative improvement in success rate is less pronounced when the low-frequency bias is active, but still significant.

#### 4 Evaluation

MASK CHANNELS	ACTIVE BIASES			SUCCESS RATE VS NUMBER OF QUERIES						MEDIAN QUERIES
	LF	MASK	SURR.	500	1000	2500	5000	10000	15000	
3 (COLOR)	NO	YES	NO	<b>0.03</b>	<b>0.05</b>	<b>0.14</b>	<b>0.28</b>	<b>0.39</b>	<b>0.54</b>	<b>13914</b>
1 (GRAY)	NO	YES	NO	0.02	<b>0.05</b>	0.12	0.22	0.33	0.41	$\approx 20294$
3 (COLOR)	YES	YES	NO	0.03	0.08	<b>0.28</b>	<b>0.53</b>	<b>0.76</b>	<b>0.86</b>	<b>4751</b>
1 (GRAY)	YES	YES	NO	<b>0.04</b>	<b>0.09</b>	0.24	0.45	0.68	0.81	5607

**Table 4.5:** Comparison of color and grayscale masks. The grayscale mask is less efficient than the color mask, but can still be beneficial as shown in Figure 4.9. All runs in this table are performed with  $s=1.0$ .



**Figure 4.9:** Comparison of color and grayscale masks. Although the perturbation with the grayscale mask has a slightly higher  $\ell^2$ -norm, it is far less noticeable than the one with the color mask. The original class is "bolete", but both adversarial examples are classified as "spindle".

## 4.5 Surrogate gradient bias

The ablation study in Section 4.2 has shown that the regional masking bias yields a considerable improvement in efficiency. Again, this experimentally confirms the hypothesis presented in Section 3.2.3 of Chapter 3. Here, two additional experiments are performed to investigate the effect of the surrogate bias strength, and to compare a number of different surrogate models.

### 4.5.1 Surrogate bias strength

The first experiment performs a study of the surrogate bias strength, or weight,  $w$ . As postulated in Chapter 3,  $w$  should be set to a rather low value, otherwise the attack might degenerate into a full transfer attack – which typically only succeeds if the surrogate model has been carefully tuned. The ablation study was performed with  $w = 0.001$ , but can this value be tuned to exploit the surrogate model to the fullest?

### 4.5.2 Surrogate models

This experiment investigates a range of different surrogate models. Until now, all experiments were performed with Inception-Resnet-v2, pre-trained in standard fashion on ImageNet. The following models are compared:

- Inception-ResNet-v2 [94]: This is the surrogate model used in the ablation study in Section 4.2. No manual training is performed, and pre-trained weights are taken from the official TensorFlow repository<sup>3</sup>. This model has a top-1 classification accuracy of 80.4%.
- ResNet-101 [72]: This is another well-known model architecture. Again, no manual training is performed and the pre-trained weights provided by TF-Slim are used. This model has a top-1 classification accuracy of 77.0%.
- Ensemble: Here, both Inception-ResNet-v2 and ResNet-101 are combined into an ensemble. This is achieved simply by adding the logit outputs of both models – the classification result is then the argmax of the sum. Prior work [17] has shown that this form of ensembling drastically improves transferability between models, and therefore this should benefit attack efficiency. This model has a top-1 classification accuracy of 80.1%.
- Inception-ResNet-v2 (AT): This model has the same architecture as the original, but adversarial training has been performed in order to make it more resilient against adversarial examples. Prior work has indicated that – although adversarially-trained models are more resistant to attacks – gradients calculated on them tend to transfer *less* well than those calculated on undefended models [34]. Therefore it can be expected that using such a model as surrogate would decrease

<sup>3</sup>See <https://github.com/tensorflow/models/tree/master/research/slim>.

## 4 Evaluation

attack efficiency, confirming the previously observed effect. For this experiment, the pre-trained weights provided by Kurakin et al. [17] are used. This model has a top-1 classification accuracy of 78.4%.

Calculating gradients on surrogate models is computationally expensive, especially for ensembles. Therefore, the model comparison is performed with only one bias strength,  $w = 0.001$ .

### 4.5.3 Hyperparameters

In both experiments, the surrogate bias is investigated first in isolation and then with both low-frequency and mask biases combined. Perlin frequency is 32, and mask strength is 1.0.

### 4.5.4 Results

Tables 4.6 and 4.7 contain the results, while Figures 4.10 and 4.11 show plots of all runs combined. Figure 4.12 is a qualitative inspection of examples: The surrogate bias often causes characteristic patterns that visibly contain salient features of the target class.

**Surrogate bias strength.** It is apparent from Table 4.6 that there exists a "sweet spot" at  $w = 0.01$ , where the efficiency gain is largest. However, this may be specific to the particular combination of classifier, dataset, and surrogate model. Since an expensive hyperparameter search is required to obtain the optimal value, in unknown situations it is better to pick  $w$  conservatively. A very low value such as  $w = 0.001$  is unlikely to be too high, while still providing a reasonable benefit in efficiency. This is also the reason why  $w = 0.001$  was chosen for the ablation study, and this is also the value that is used for attacks on unknown models in Chapter 5.

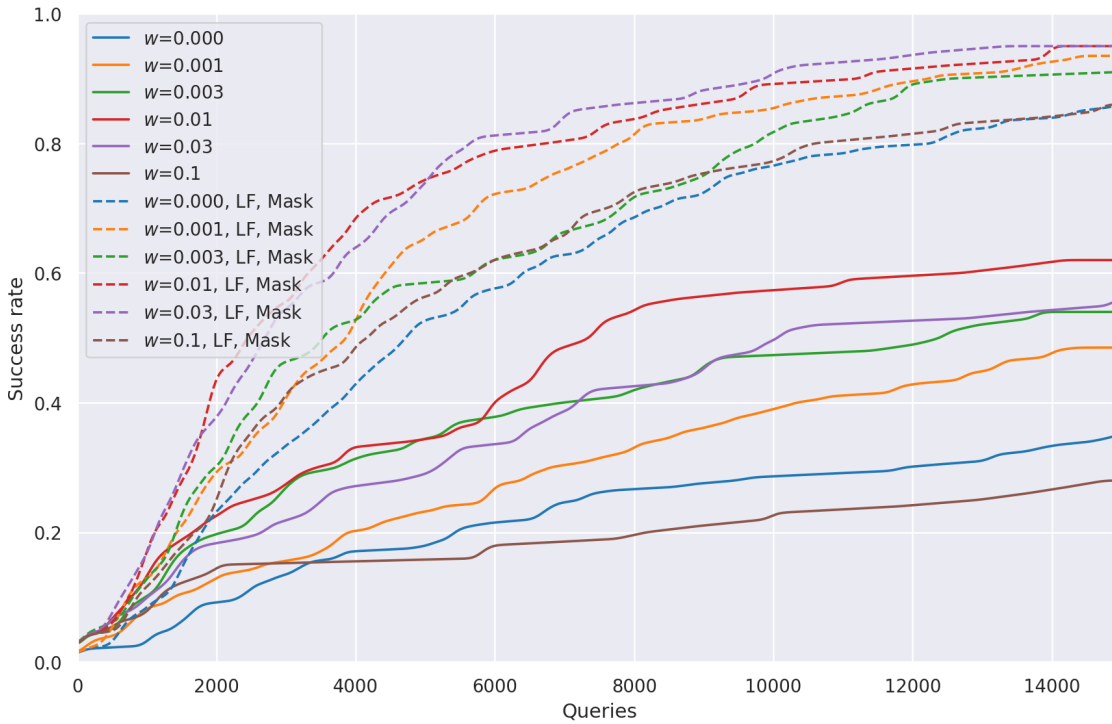
**Surrogate models.** Table 4.7 shows that an ensemble of surrogates drastically outperforms individual models. This is in line with expectations and prior work [17]. The ensemble is a perfect example of a "strong" surrogate, providing adversarial gradients that transfer well, while at the same time being computationally expensive. The methods presented in this thesis place an emphasis on simplicity, speed, and ease of access. Although their performance is impressive, large ensembles are neither simple nor fast – state-of-the-art transfer attacks routinely use ensembles of five and more models [18]! For this reason, ensembling is not pursued any further in the scope of this thesis.

It is also apparent that surrogates do not seem to benefit from adversarial training. This, again, confirms prior work on transfer attacks [34].

Overall, the experiments in this Section have shown that the surrogate bias can reduce the number of queries by a significant amount, even when not fine-tuned (very low  $w$ ) and without using strong surrogates (ensembles). The results indicate that surrogate models behave similarly when used as bias for the Biased Boundary Attack to when they are used in a PGD-style transfer attack. Thus, any advance in that line of work might be easily adopted.

SURROGATE STRENGTH $w$	ACTIVE BIASES			SUCCESS RATE VS NUMBER OF QUERIES						MEDIAN QUERIES
	LF	MASK	SURR.	500	1000	2500	5000	10000	15000	
0.000	NO	NO	NO	0.02	0.03	0.11	0.17	0.28	0.35	$\approx 25357$
0.001	NO	NO	YES	0.04	0.08	0.14	0.23	0.39	0.49	$\approx 15789$
0.003	NO	NO	YES	0.04	0.10	0.22	<b>0.34</b>	0.47	0.54	12475
0.01	NO	NO	YES	<b>0.07</b>	<b>0.13</b>	<b>0.25</b>	<b>0.34</b>	<b>0.57</b>	<b>0.62</b>	<b>7424</b>
0.03	NO	NO	YES	0.06	0.10	0.19	0.28	0.49	0.56	10119
0.1	NO	NO	YES	0.05	0.07	0.15	0.15	0.22	0.28	$\approx 33333$
0.000	YES	YES	NO	0.03	0.08	0.28	0.53	0.76	0.86	4751
0.001	YES	YES	YES	0.05	0.13	0.34	0.65	0.85	0.94	3906
0.003	YES	YES	YES	0.06	0.13	0.38	0.58	0.81	0.91	3540
0.01	YES	YES	YES	0.05	<b>0.16</b>	<b>0.50</b>	<b>0.74</b>	0.89	<b>0.95</b>	<b>2463</b>
0.03	YES	YES	YES	<b>0.07</b>	<b>0.16</b>	0.46	0.73	<b>0.91</b>	<b>0.95</b>	2733
0.1	YES	YES	YES	0.04	0.11	0.35	0.56	0.77	0.86	4240

**Table 4.6:** Comparison of different surrogate strengths  $w$ . There exists a sweet spot at  $w = 0.01$ , where the surrogate bias has the highest effect. While 0.1 appears a good value for our particular setup of classifier, dataset and surrogate model, it cannot be expected to generalize to other setups. For this reason, it is advisable to pick  $w$  conservatively (e.g. 0.001), and to only increase it if fine-tuning is desired.

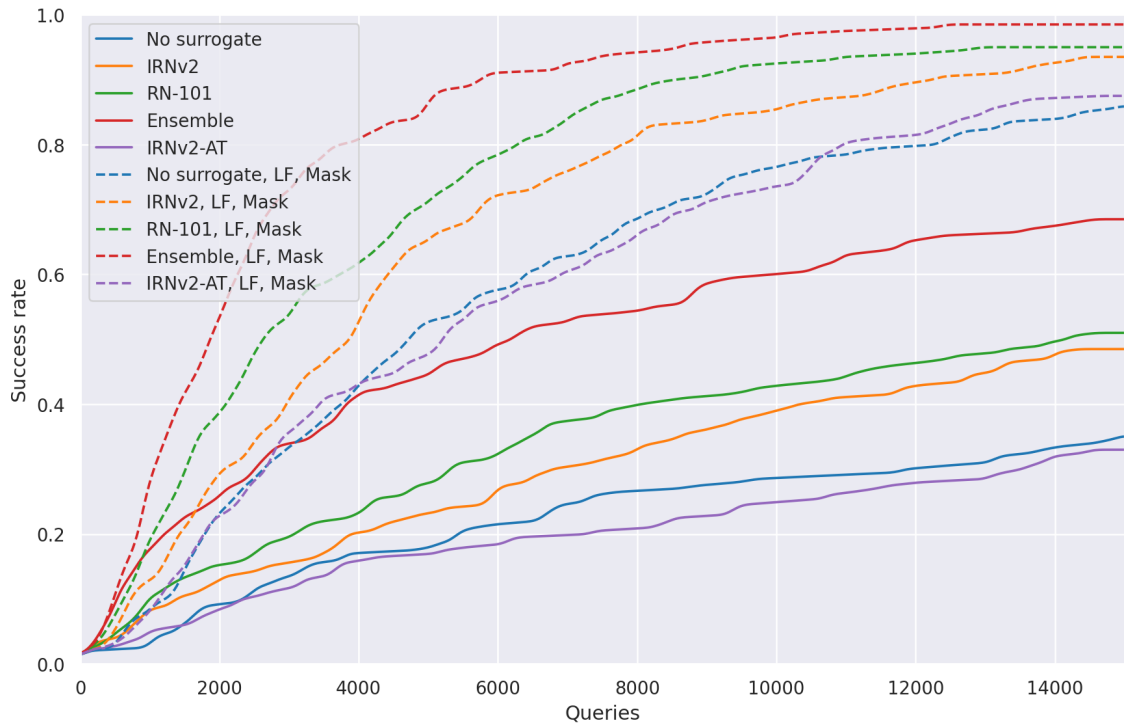


**Figure 4.10:** Comparison of surrogate strengths  $w$  (same data as in Table 4.4).

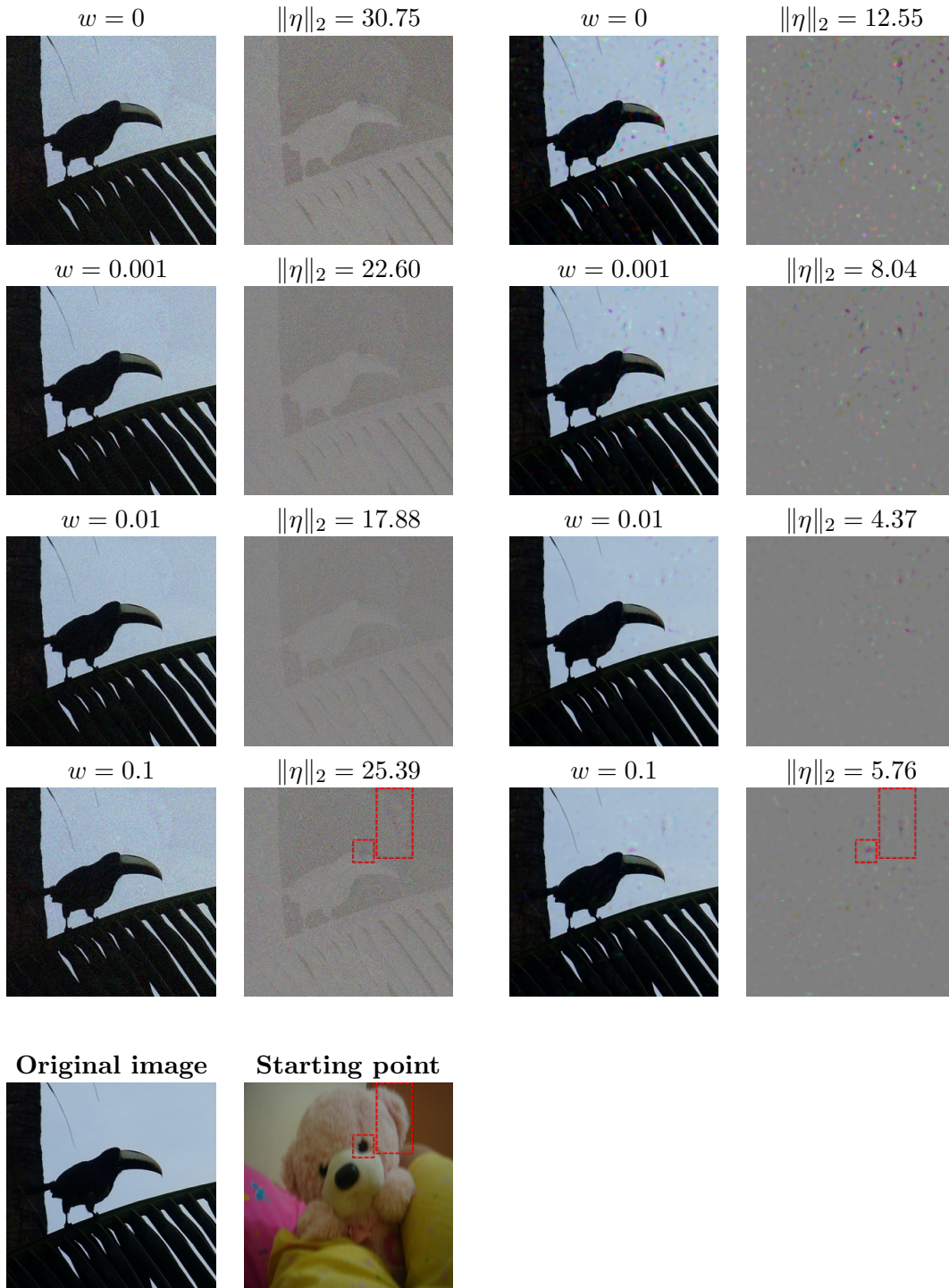
#### 4 Evaluation

SURROGATE MODEL	ACTIVE BIASES			SUCCESS RATE VS NUMBER OF QUERIES						MEDIAN QUERIES
	LF	MASK	SURR.	500	1000	2500	5000	10000	15000	
-	NO	NO	NO	0.02	0.03	0.11	0.17	0.28	0.35	≈25357
IRNv2	NO	NO	YES	0.04	0.08	0.14	0.23	0.39	0.49	≈15789
RN-101	NO	NO	YES	0.05	0.10	0.17	0.28	0.43	0.51	14207
ENSEMBLE	NO	NO	YES	<b>0.09</b>	<b>0.17</b>	<b>0.29</b>	<b>0.44</b>	<b>0.60</b>	<b>0.69</b>	<b>6243</b>
IRNv2-AT	NO	NO	YES	0.03	0.05	0.10	0.17	0.25	0.33	≈25000
-	YES	YES	NO	0.03	0.08	0.28	0.53	0.76	0.86	4751
IRNv2	YES	YES	YES	0.05	0.13	0.34	0.65	0.85	0.94	3906
RN-101	YES	YES	YES	0.08	0.19	0.48	0.71	0.93	0.95	2645
ENSEMBLE	YES	YES	YES	<b>0.10</b>	<b>0.29</b>	<b>0.67</b>	<b>0.87</b>	<b>0.96</b>	<b>0.99</b>	<b>1863</b>
IRNv2-AT	YES	YES	YES	0.03	0.08	0.29	0.48	0.74	0.88	5214

**Table 4.7:** Comparison of surrogate models. ResNet-101 performs similarly to Inception-ResNet-v2, and forming an ensemble of both drastically increases efficiency. As expected, adversarial training seems to have a detrimental effect on transferability.



**Figure 4.11:** Comparison of surrogate models. The ensemble drastically outperforms individual models, while the adversarially trained model hurts performance instead of increasing it.



**Figure 4.12:** Comparison of different surrogate strengths  $w$ . Left: surrogate bias only. Right: all biases combined. The original class is "toucan", but all images are classified as "teddy bear". In some cases, the surrogate bias visibly reinforces salient features of the target class (such as the eye and ear of the teddy bear, highlighted in red).



## 4.6 Initialization with synthetic starting points

So far, all experiments have been conducted with the same initialization method: Searching the ImageNet validation set for images of the target class, and selecting the image  $x_{start}$  that is closest to the image under attack  $x_{orig}$ . Surely it is possible to find better starting points than this?

### 4.6.1 Creating salient patches of the target class

As described in Section 3.3 of Chapter 3, it is possible to find images of the target class, segment them with saliency masks, and then copy the salient patches into the original image  $x_{orig}$ . This alone is often enough to change the label to the desired class, and the resulting image can be used as a starting point  $x_{start}$ .

While there could be many methods of synthesizing starting points, this one is extremely simple and serves as a direct proof of concept – namely that better starting points significantly increase attack efficiency, and that it is in fact very easy to construct them.

This section consists of a single experiment: The ”salient patch” initialization method proposed in Section 3.3 of Chapter 3 is tested and compared against the classic ”closest image” method in a number of attack configurations (biased and unbiased). This should suffice to deliver evidence whether initialization is a key driver of attack efficiency.

### 4.6.2 Hyperparameters

All attack hyperparameters are at the same values as the earlier ablation study in Section 4.2:

- Perlin noise frequency: 32
- Mask strength: 1.0
- Mask channels: 3 (color)
- Surrogate model: Inception-Resnet-v2
- Surrogate weight  $w$ : 0.001

### 4.6.3 Results

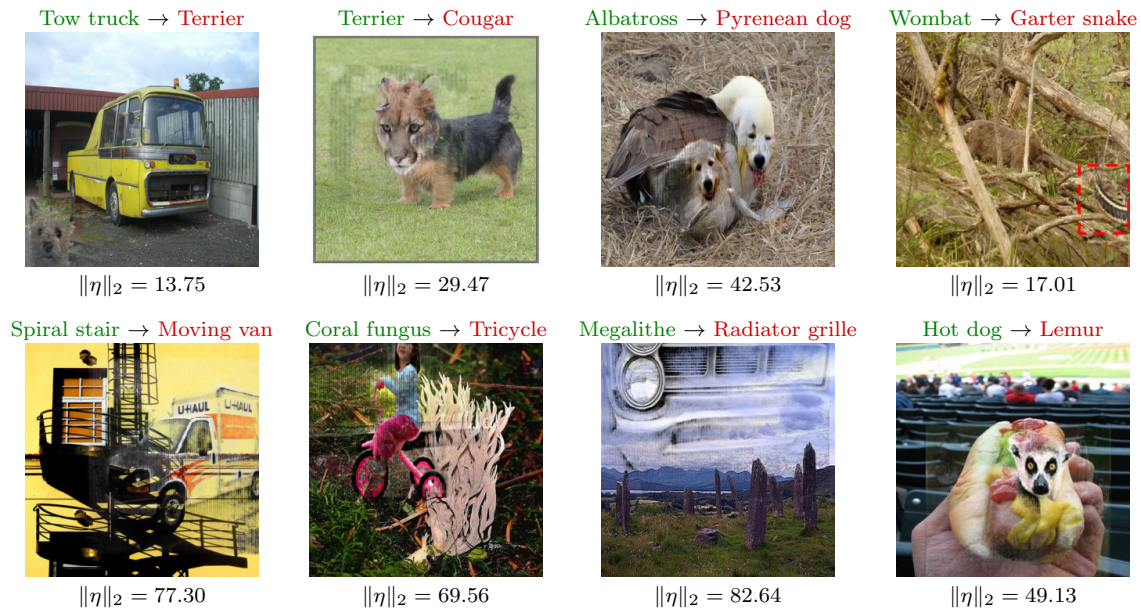
Table 4.8 and Figure 4.14 show the result: In every configuration, ”salient patch” initialization reduces the number of queries by more than 60%. When all biases are enabled, little more than 1000 queries are required to succeed on more than half of the evaluation images. Notably, the ”salient patch” method drastically increases success rates early in the attack. This is expected, as synthesized starting points are specifically designed to have very small perturbation norms.

Figure 4.15 shows a side-by-side comparison of a single attack. It is easy to see the superiority of synthesized starting points: In this particular example, the salient features of the terrier (its head) have been extracted, down-sized and placed into a corner of the



image. Evidently, this small patch is so salient that it overrides the classification result! In all cases, it was possible to synthesize these images while performing less than 10 queries on the black box model<sup>4</sup>.

**Success on the first try?** Figure 4.13 shows more starting points created by the method, highlighting a startling phenomenon: In some cases, the starting point is already below the success threshold ( $\|\eta\|_2 < 25.89$ ). Worse even, in some cases (see the image of a snake, top right) the perturbation is so well hidden that it can be used as an adversarial example straight away, without even conducting an attack! The classifier under attack (Inception-v3) is considered relatively accurate [47]. While vulnerability to adversarial patterns is not surprising, it is troublesome that simply copying small patches confuses the classifier so strongly.



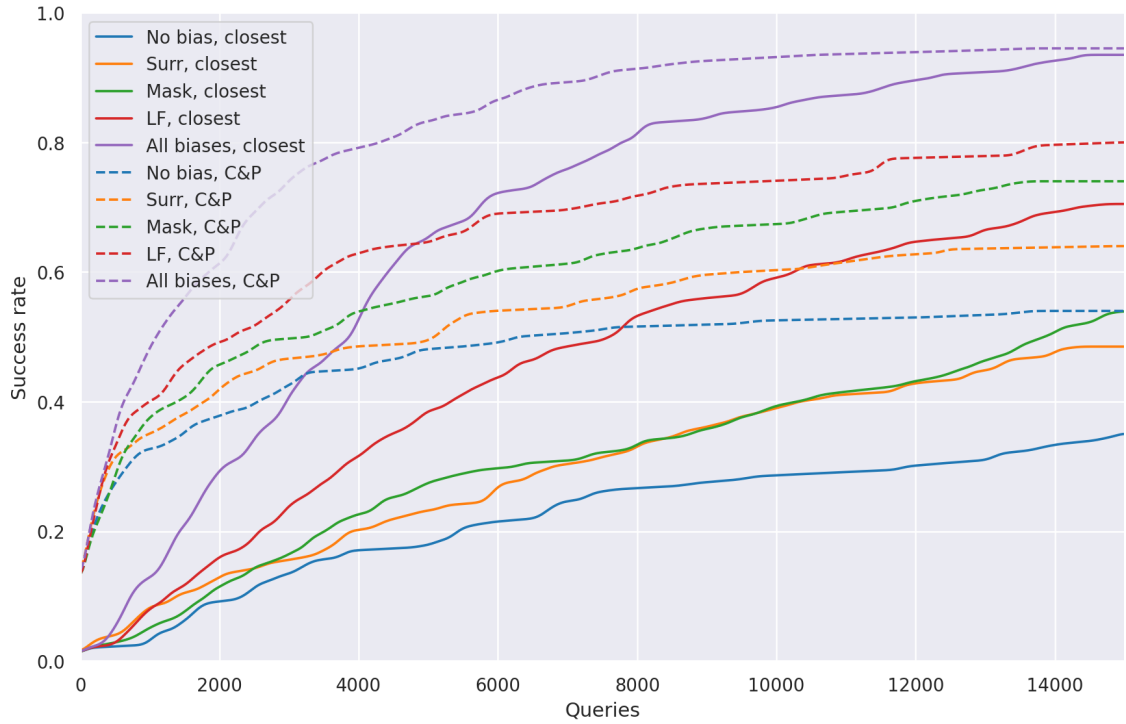
**Figure 4.13:** Starting points generated by salient patches. All images are classified as the adversarial label (red). Upper row: examples where the adversarial features have been placed well, in some cases so well that they are hardly noticeable (top right, highlighted). Lower row: examples where the method did not place the patches well. As a consequence, the perturbations are rather large - but still smaller than when using full images of the target class (where  $\|\eta\|_2$  routinely is greater than 100).

<sup>4</sup>See Section 3.3 of Chapter 3: The algorithm creates a large number of candidate images, applying random translation and scaling to the adversarial patch. These are tested against a surrogate model, and only those that are adversarially classified are sorted by perturbation magnitude and tested against the black box. In this way, it is possible to keep the number of queries to a minimum.

#### 4 Evaluation

INIT METHOD	ACTIVE BIASES			SUCCESS RATE VS NUMBER OF QUERIES						MEDIAN QUERIES
	LF	MASK	SURR.	500	1000	2500	5000	10000	15000	
CLOSEST	NO	NO	NO	0.02	0.03	0.11	0.17	0.28	0.35	≈25357
PATCH	NO	NO	NO	0.28	0.32	0.40	0.48	0.52	0.54	6574
CLOSEST	NO	NO	YES	0.04	0.08	0.14	0.23	0.39	0.49	≈15789
PATCH	NO	NO	YES	0.32	0.35	0.45	0.49	0.60	0.64	5134
CLOSEST	NO	YES	NO	0.03	0.05	0.14	0.28	0.39	0.54	13914
PATCH	NO	YES	NO	0.28	0.38	0.48	0.56	0.67	0.74	3367
CLOSEST	YES	NO	NO	0.03	0.08	0.19	0.39	0.59	0.70	7649
PATCH	YES	NO	NO	0.33	0.40	0.51	0.65	0.74	0.80	2197
CLOSEST	YES	YES	YES	0.05	0.13	0.34	0.65	0.85	0.94	3906
PATCH	YES	YES	YES	<b>0.37</b>	<b>0.49</b>	<b>0.69</b>	<b>0.83</b>	<b>0.93</b>	<b>0.95</b>	<b>1093</b>

**Table 4.8:** Comparison of initialization methods. Runs with "closest" are initialized with the closest image of the target class, while "patch" runs are initialized with salient patches. Changing the initialization method delivers a large boost in efficiency – the number of queries is reduced by more than 60%, no matter which biases are active.



**Figure 4.14:** Comparison of initialization methods. Synthesized starting points synergize well with all biases and enhance attacks in every configuration.

#### 4.6 Initialization with synthetic starting points



**Figure 4.15:** Comparison of initialization methods (all biases enabled). When using a full image of the target class, a large number of iterations is required, and the resulting perturbation is still noticeable. With the synthesized starting point, it is possible to limit the attack to a small region in the image, and after few iterations the perturbation is barely noticeable. The original label is "tow truck", but all images are classified as "cairn terrier".

## 4.7 Comparison with other state-of-the-art attacks

So far, the evaluations in this Chapter have shown that the Biased Boundary Attack significantly outperforms the original boundary attack formulation as proposed by Brendel et al. [15]. However, as described in Section 2.4.2 of Chapter 2, boundary attacks represent just one of three major families of black-box adversarial attacks – the others being transfer attacks and gradient estimation attacks. To put the performance of the new method into perspective, it is necessary to compare it against state-of-the-art attacks from other families.

In this Section, the Biased Boundary Attack is benchmarked against recently proposed transfer, gradient estimation, and boundary attacks that were considered state-of-the-art at the time of writing. For all methods, publicly available implementations are used, together with the hyperparameters recommended for ImageNet by the original authors. The respective implementations are modified only to use the common evaluation set, therefore all attacks are run on the same 200 images and use the same target labels (and starting points, where applicable).

To allow a fair comparison, the Biased Boundary Attack is run with the same hyperparameters as in the ablation study in Section 4.2. No further tuning is performed, and the attack is initialized with the default choice – the closest image of the target class.

Table 4.9 shows an overview of the results, and Figure 4.16 plots the success rate. The following subsections discuss the individual attacks and results in detail:

### 4.7.1 Gradient estimation attacks

Most gradient estimation attacks rely on confidence scores [27, 44, 42, 15], which renders them unusable in the label-only setting. However, it is still possible to estimate gradients via a number of clever optimizations. Two methods are compared:

- **Ilyas et al. [42]: Natural Evolutionary Strategies (NES).** At the time of writing, this was a rather popular gradient estimation attack: In their publication, Ilyas et al. were among the first to demonstrate a viable targeted attack against the Google Cloud API. Normally, their method would require confidence scores, but Ilyas et al. also propose a label-only version of the same method. It works, but at greatly reduced efficiency: As Table 4.9 shows, the attack does not produce any adversarial example within 15000 queries (in fact, no run succeeds before 276000 queries)<sup>5</sup>. This is in line with their published work, which states a median of 2.7 million queries required until success [42].
- **Cheng et al. [45]: Directional Estimation Attack.** Similarly, Cheng et al. re-frame the setting as a real-valued optimization problem which is then solved with gradient estimation. In their publication, they report higher efficiency than the

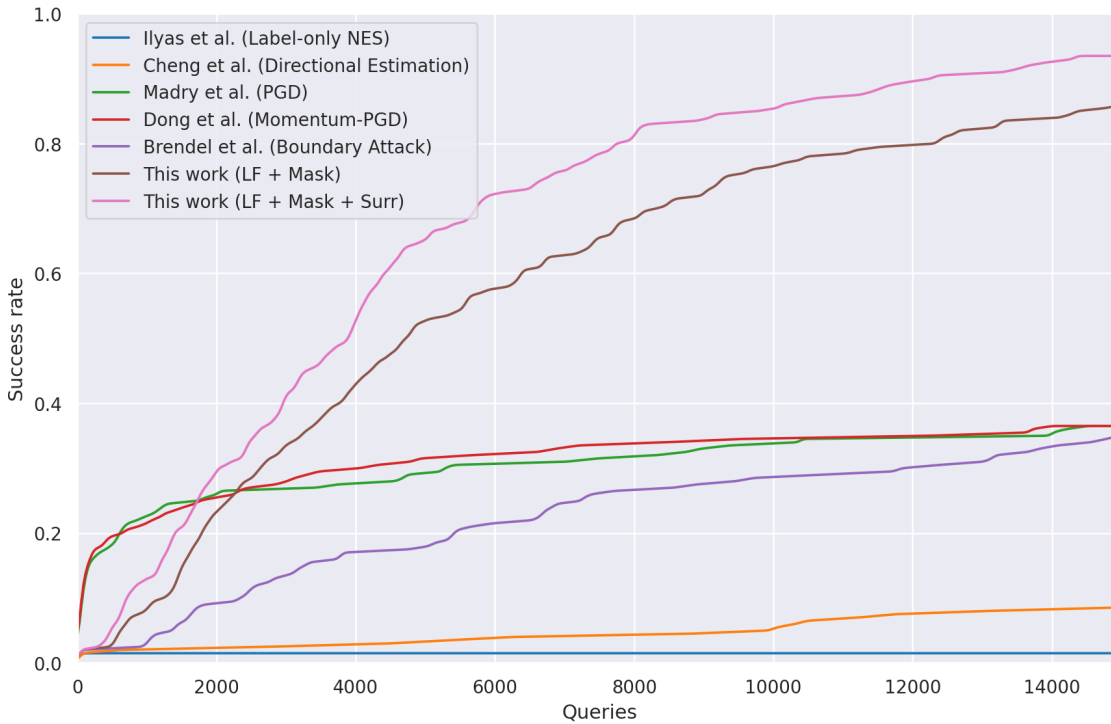
---

<sup>5</sup>The table lists 0.02 instead of 0.00 as minimal success rate. This is due to the choice of initialization (closest image of the target class): For these particular examples, the ImageNet validation set contained an image of the target class that was already very close to the image under attack. In these cases, the attack is automatically considered successful.

#### 4.7 Comparison with other state-of-the-art attacks

ATTACK	FAMILY	SUCCESS RATE VS NUMBER OF QUERIES					
		500	1000	2500	5000	10000	15000
Ilyas et al. [42] (Label-only NES)	GE	0.02	0.02	0.02	0.02	0.02	0.02
Cheng et al. [45] (Directional Estimation)	GE	0.02	0.02	0.02	0.03	0.05	0.09
Madry et al. [33] (PGD)	T	0.18	<b>0.23</b>	0.27	0.29	0.34	0.37
Dong et al. [75] (Momentum-PGD)	T	<b>0.20</b>	0.22	0.27	0.32	0.35	0.37
Brendel et al. [15] (Boundary Attack)	B	0.02	0.03	0.11	0.17	0.28	0.35
This work (LF + Mask biases)	B	0.03	0.08	0.28	0.53	0.76	0.86
This work (LF + Mask + Surr. biases)	B+T	0.05	0.13	<b>0.34</b>	<b>0.65</b>	<b>0.85</b>	<b>0.94</b>

**Table 4.9:** Comparison with recently proposed label-only black-box attacks. Attack families are denoted by GE (gradient estimation), T (transfer), and B (boundary). The original source code provided by the respective authors is used and all attacks are run on the evaluation data set. The Biased Boundary Attack with all biases enabled has the highest success rate after 15000 queries. While transfer attacks often succeed very early, they fail to produce adversarial examples for more than half of the evaluation images. Notably, even the model-free version (without a surrogate) outperforms prior work in mid-query scenarios. It is evident that gradient estimation is very inefficient in a label-only setting. In the case of Ilyas et al. [42], the number of required queries is so large that the attack requires far more than 15000 queries (over one million, according to their publication) to make a difference.



**Figure 4.16:** Comparison with recently proposed label-only black-box attacks. While transfer attacks have an early advantage, they are quickly outperformed by the more reliable Biased Boundary Attack.



original boundary attack formulation for untargeted attacks on ImageNet (a setting that is of limited value, see discussion in Section 2.3.2.1 of Chapter 2). Notably, their publication does not evaluate targeted attacks on ImageNet, although it does so on other datasets. When conducting a targeted attack on ImageNet, the results are different (see Table 4.9): Contrary to expectations, the attack performs significantly worse than even an unbiased boundary attack. It should be stressed that the original source code provided by the authors in [45] was used, together with the hyperparameters also made available by them.

### 4.7.2 Boundary attacks

To the best of this author’s knowledge, at the time of writing there existed only the original (unbiased) Boundary Attack:

- **Brendel et al. [15]: Boundary Attack.** Naturally, the performance of this method is the same as reported in the ablation study Section 4.2, with all biases deactivated. It outperforms both gradient estimation attacks, but is less efficient than transfer attacks – it only starts to approach their success rate after 15000 queries.

### 4.7.3 Transfer attacks (simple surrogate)

Perhaps the most interesting attacks to compare to are transfer attacks. These are currently the most popular methods, having dominated open competitions in recent years [17, 18]. To allow a fair comparison, all attacks in Table 4.9 use the same surrogate model as the Biased Boundary Attack, Inception-ResNet-v2.

- **Madry et al. [33]: PGD.** This is a typical PGD attack as described in Section 2.4 of Chapter 2, the only difference to the white-box variant being that the adversarial gradient is calculated on a surrogate. After every step, the adversarial candidate is then tested against the black-box classifier.

Typically, this attack performs only a small number of steps and then either succeeds or fails [33, 34]. However, Madry et al. also introduce the concept of ”random restarts”, where an attack can be conducted multiple times, starting from a randomly perturbed version of the original image. This is a useful extension, as an otherwise unsuccessful attack may still succeed when starting from a slightly different initialization. For this experiment, the source code and hyperparameters provided by the authors are used, but the number of restarts is greatly increased – the attack is allowed to try again and again until the query budget is exhausted.

Looking at Table 4.9, it is clear that the performance of PGD transfer is hit-and-miss: When a transfer succeeds it does so very early, but in many cases it never succeeds, even after many restarts. This is somewhat expected, as no fine-tuning or ensembling has been used for the surrogate model in this experiment.

- **Dong et al. [75]: Momentum-PGD.** This is an extension of PGD transfer which adds a simple momentum term to the adversarial gradient, reportedly improving attack success chance. It was the winning attack in the NeurIPS'17 Adversarial Vision Challenge [17] and therefore is a good example of a strong attack.

Considering Table 4.9, however, it is apparent that momentum delivers only a small improvement over the original formulation – at least for targeted attack on ImageNet. Dong et al. also note this in the appendix of their paper [75]. Again, perhaps a stronger surrogate model is needed to fully show the strength of these attacks?

#### 4.7.4 Transfer attacks (strong surrogate)

Most prior results for strong transfer attacks have been obtained on large, carefully-crafted ensembles of surrogate models [34, 75, 17]. It is not entirely fair to benchmark transfer attacks with a simple pre-trained model such as Inception-ResNet-v2. Therefore, the previous experiment is repeated with a stronger surrogate, namely an ensemble of Inception-ResNet-v2 and ResNet-101. This is the same model that was evaluated for the Biased Boundary Attack in Section 4.5.

And indeed, this drastically increases the performance of both PGD and Momentum-PGD – as Table 4.10 shows, from 37% to 62% success rate after 15000 queries. Nevertheless, the Biased Boundary Attack also profits from the strong surrogate model, increasing the success rate from 94% to 99%. Figure 4.17 shows the result more clearly: No matter if a simple or a strong surrogate is used, the Biased Boundary Attack starts outperforming PGD transfer attacks after 2000 queries.

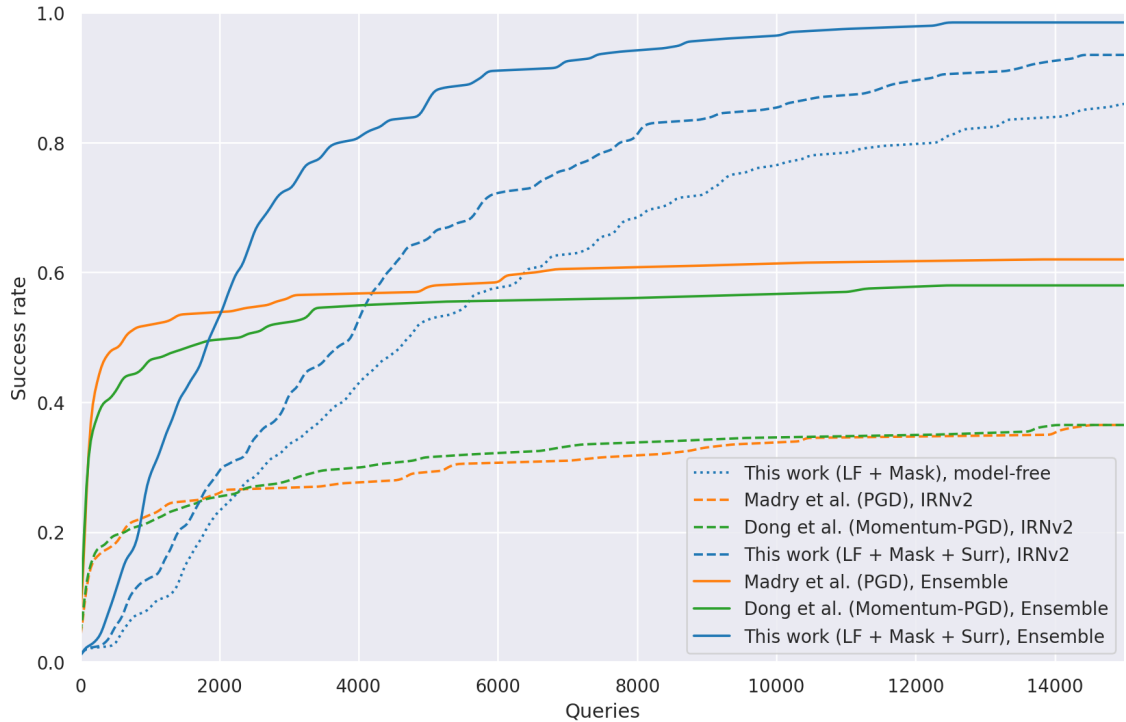
Crucially, even the model-free version (low-frequency and mask bias, but no surrogate) outperforms strong transfer attacks after roughly 6000 queries. This is important to note, as creating ensembles of more than two models is very slow and requires large amounts of memory. While this might increase the success rate even more, it is outside the scope of this work as it would put a prohibitive cost on the experiments.

Nevertheless, the evaluation in this Chapter clearly shows that the Biased Boundary Attack decidedly outperforms prior work in one of the most difficult and realistic attack settings, even when no surrogate models are available. At the same time, if they are indeed available, they can be exploited by the attack to great effect.

## 4 Evaluation

ATTACK	SURRE. MODEL	SUCCESS RATE VS NUMBER OF QUERIES					
		500	1000	2500	5000	10000	15000
This work (LF + Mask)	(none)	0.03	0.08	0.28	0.53	0.76	0.86
Madry et al. [33] (PGD)	IRNv2	0.18	0.23	0.27	0.29	0.34	0.37
Dong et al. [75] (Momentum-PGD)	IRNv2	0.20	0.22	0.27	0.32	0.35	0.37
This work (LF + Mask + Surr.)	IRNv2	0.05	0.13	0.34	0.65	0.85	0.94
Madry et al. [33] (PGD)	Ensemble	<b>0.48</b>	<b>0.52</b>	0.55	0.58	0.61	0.62
Dong et al. [75] (Momentum-PGD)	Ensemble	0.42	0.47	0.51	0.55	0.57	0.58
This work (LF + Mask + Surr.)	Ensemble	0.10	0.29	<b>0.67</b>	<b>0.87</b>	<b>0.96</b>	<b>0.99</b>

**Table 4.10:** Comparison with transfer attacks when using a stronger surrogate model. State-of-the-art transfer attacks crucially rely on hand-tuned ensembles of classifiers to maximize transferability. To make a fair comparison, this Table shows the same attacks when using an ensemble of Inception-ResNet-V2 and ResNet-101 as a surrogate. The transfer attacks drastically benefit from a stronger surrogate, but the Biased Boundary Attack does so as well.



**Figure 4.17:** Comparison with transfer attacks when using a stronger surrogate model. Both when using a weak (IRNv2) and strong (Ensemble) surrogate, the Biased Boundary Attack starts outperforming the transfer attacks after 2000 queries. Interestingly, even the model-free version (without surrogate bias) manages to outperform strong transfer attacks after 6000 queries.



## 4.8 **NeurIPS 2018 Adversarial Vision Challenge**

As part of this work, an implementation of the Biased Boundary Attack was submitted to the NeurIPS 2018 Adversarial Vision Challenge, an open competition of adversarial attacks and defenses.

The submission won second place in the targeted attack track, which shows that the method performs well against state-of-the-art defense methods and robust classifiers. Appendix B describes the submission and the competition format in detail.

## 4.9 **Summary**

It is evident that the methods proposed in Chapter 3 greatly improve the efficiency of boundary attacks. The resulting attack, the Biased Boundary Attack, decidedly outperforms the state of the art in targeted black-box adversarial attacks. Crucially, it not only outperforms the original Boundary Attack but also methods from other families such as PGD transfer attacks, which have dominated the field in recent years [17, 18]. This shows that Biased Boundary Attacks are both powerful and easy to use, and this significantly lowers the bar for conducting attacks on unknown systems.

This Chapter has focused only on adversarial attacks, applying them to a single image classifier, Inception-v3. But what about other classifiers? And what about adversarial defenses? Armed with a black-box attack that is both efficient and reliable, it is now possible to evaluate the robustness of a broad range of image classifiers, machine learning models, defense mechanisms, and real-world commercial services.



## 5 Benchmarking the robustness of image classifiers

With an efficient attack now available, it becomes possible to attack a broad range of image classifiers and to perform a side-by-side comparison of their (empirical) robustness.

First, state-of-the-art deep learning architectures are investigated. This is contrasted with traditional techniques such as Support Vector Machines and Decision Trees, combined with various feature extractors. The results show that these methods have similar vulnerabilities as deep learning classifiers. While some combinations of feature extractors and encoding schemes seem to be slightly more robust than CNNs, the attack is still successful in the vast majority of cases.

Next, several recently proposed defenses against adversarial attacks are benchmarked. The results show that many of them do in fact not work when confronted with an adaptive black-box attack. A select few approaches, such as Adversarial Training, show great promise as they significantly hamper attack progress. However, it is still possible to create adversarial examples in more than 45% of cases.

The results indicate that, in the ImageNet evaluation setting, no classifier currently exists that is consistently robust against adversarial attacks. Finally, the attack developed in this work is also used to successfully generate adversarial examples for a number of commercial cloud-based image classifiers. This demonstrates that even these services can be easily attacked with only a minimal investment of time and computational resources.

### 5.1 Setup

Much of the experimental setup in this Chapter is identical to that of Chapter 4, which compared the effectiveness of different adversarial attacks on a fixed image classifier. In this Chapter, the same evaluation is performed, but from the other side: A fixed attack is selected and performed on a broad range of image classifiers. The resulting attack success rate can then be interpreted as an empirical measurement of robustness.

#### 5.1.1 Data set

The evaluation data set is the same as in Chapter 4: 200 images, randomly selected from the ImageNet validation set, with a random adversarial target class. In this way, all experiments are performed on the same images and target classes as in Chapter 4.

The only exception to this is the experiment in Section 5.3, which investigates traditional computer vision pipelines. Since it is difficult to scale these methods to process all

of ImageNet (1.2M training examples with 1000 classes), a 10-class subset of ImageNet is selected for this particular experiment. See Section 5.3 for details.

### 5.1.2 Image pre-processing

The original ImageNet database contains images of various sizes. However, CNN image classifiers typically process images in a single fixed resolution. As in Chapter 4, all images in the evaluation set are resized to a resolution of  $299 \times 299$ , which is the input format for a typical Inception-v3 classifier. Consequently, the adversarial attacks in Chapter 4 also produce perturbations in the same resolution.

However, this Chapter compares several classifiers that operate on different image sizes, ranging from  $224 \times 224$  (VGG-16) up to  $600 \times 600$  (EfficientNet-B7). To allow for a side-by-side comparison, all attacks are performed in the same resolution,  $299 \times 299$ , and images are then up- or downsampled before feeding them into the classifier. In this way, the true input resolution of the classifier stays inside the black box and is obscured from the attacker.

### 5.1.3 Evaluation procedure

As in Chapter 4, attacks must produce an adversarial example  $x_{adv}$  for each pair  $(x_{orig}, y_{target})$  in the evaluation set. Attacks are allowed a maximum budget of 15000 queries per image, and success is registered as soon as an image of the target class is produced which is close to the original image. Same as in Chapter 4, this closeness is measured by the magnitude of the image difference:

$$\|\eta\|_2 = \|x_{adv} - x_{orig}\|_2 < \epsilon_{success} \quad (5.1)$$

The success threshold  $\epsilon_{success}$  is, again, set to 25.89. After the attack has tried to produce 200 adversarial examples for a classifier, the attack success rate and query efficiency are calculated. Then, the following statements about robustness can be made:

- **Successful attacks disprove robustness.** Whenever an adversarial example is found, the classifier is proven to be un-robust in the radius of  $\epsilon_{success}$ . In this way,  $\epsilon_{success}$  acts as an upper bound for the true robustness of the classifier<sup>1</sup>. A high success rate therefore disproves robustness.
- **Unsuccessful attacks do not prove robustness.** The opposite is not necessarily true: Where one attack fails, another might succeed. While the attack is designed to be both adaptive and efficient, it is unlikely to be optimal against all defenses. Therefore, a low success rate is not a proof of robustness.
- **Unsuccessful attacks demonstrate *empirical robustness*.** While not a proof of robustness, a low success rate measurement is still valuable for practical purposes: If a classifier does well in the benchmark, it is shown to be more robust against state-of-the-art black-box attacks than other classifiers.

<sup>1</sup>See Section 2.3.3 of Chapter 2 for an illustration, where the empirical bound is called  $\epsilon_{adv}$ .

In the real world, it is reasonable to assume that most attackers do not have access to white-box information, unlimited queries, or extreme computational power. Therefore, they are unlikely to craft special-purpose attacks against a single classifier or defense. Instead, they would prefer adaptive black-box attacks for their ease of access. Any demonstration of empirical robustness to this family of attacks is therefore valuable for real-life computer vision applications.

#### 5.1.4 Choice of attacks

The empirical robustness of all classifiers is evaluated with the Biased Boundary Attack introduced in Chapter 3. From the results obtained in Chapter 4, the two most promising attack configurations are picked:

- **Model-free attack:** First, it is promising to evaluate robustness in the most generic scenario, using a model-free attack without surrogates. Since surrogate models contain information that may or may not be applicable to the classifier under attack, relying on them always poses the risk of catastrophic failure when confronted with a new type of classifier. For this reason, it is worth using a simple model-free attack as a robustness "sanity check". These are the hyperparameters:
  - Low-frequency bias: Perlin noise frequency = 32
  - Mask bias: Strength = 1.0; Channels = 3 (color)
  - No surrogate bias
  - Initialization: Closest image of the target class
- **Strongest attack:** While the model-free attack should perform reasonably well, it is also fairly conservative and does not make use of some very potent optimizations developed in Chapter 3. To really test the limits of robustness, another attack is performed that uses the strongest hyperparameter settings found by the evaluation in Chapter 4:
  - Low-frequency bias: Perlin noise frequency =  $2^{\mathcal{N}(5,2)}$ , bounded in [1, 128]
  - Mask bias: Strength = 1.0; Channels = 3 (color)
  - Surrogate model: ResNet-101<sup>2 3</sup>
  - Surrogate weight  $w$ : 0.01
  - Initialization: Synthetic (salient patches)

---

<sup>2</sup>Although the ensemble was found to be strongest, it also greatly reduces the speed of the attack. Therefore, only a single surrogate is used.

<sup>3</sup>To stay consistent with the black-box setting, the surrogate must always be different from the model under attack. Therefore, for evaluating the ResNet-101 model itself, an Inception-ResNet-v2 surrogate is used instead. In all other cases, ResNet-101 is used as surrogate.

### 5.1.5 Software and hardware setup

All experiments are performed on a consumer-grade desktop machine with an AMD Ryzen 1300X quad-core CPU, 16GB RAM, and an NVIDIA GeForce 1070 GPU.

For the adversarial attacks, the implementation described in Section 4.1.7 of Chapter 4 is reused (TensorFlow 1.10 and Python 3.6). However, some of the classifiers compared in this Chapter are designed to use different machine learning frameworks (e.g. PyTorch or JAX), which are incompatible with the evaluation environment.

**An isolated environment for classifiers.** For this reason, the experiments in this Chapter utilize a client-server architecture: Both classifier (server) and adversarial attack (client) run in separate processes and communicate via a lightweight HTTP interface. The attacker sends a prediction request containing an image  $x$  to the classifier, which responds with the prediction  $\hat{y} = f(x)$ . This also mimics a typical black-box setup in the real world, where an attacker would query a web service in order to create adversarial examples for it.

Running all classifiers in separate processes enables the use of publicly-available reference implementations. This guarantees their correct usage and prevents any errors possibly introduced by re-implementation.

**Runtime considerations.** Some of the classifiers and adversarial defenses evaluated in this Chapter are very slow (e.g. ensembles and randomized defenses). Performing 15000 queries per image may vastly exceed the 5 minutes described in Chapter 4. For this reason, the time limit for a single image is increased to 1 hour, which means that each classifier must process 15000 inputs in this time ( $\approx 4$  images/s). This is not unrealistic – state-of-the-art classifiers routinely manage more than 100 images/s on the evaluation hardware. Classifiers that do not work at all within these constraints are considered unrealistic and excluded from the benchmark.

## 5.2 Deep learning architectures

In this section, a number of popular deep learning image classifiers are evaluated. These range from tested-and-true architectures like VGG-16 over ResNets and DenseNets to recent models like EfficientNets and Vision Transformers, which held the record for accuracy on ImageNet at the time of writing.

### 5.2.1 Selection of classifiers

The classifiers can be split into two categories. So far, CNN classifiers have mostly been trained in a purely supervised manner. However, recent work [98, 99] has found that higher accuracy can be achieved by using self-supervised pre-training on much larger amounts of unlabeled data before fine-tuning the model on the final data set (ImageNet).

#### 5.2.1.1 Trained normally

The following classifiers have been trained normally (supervised only):

- **VGG-16** [88]: By today’s standards, this is a rather shallow CNN with 16 layers. However, it came to fame after winning the ImageNet 2014 competition and is still in use as a simple baseline for CNN classifiers. The publicly available implementation and pre-trained model provided in Keras (TensorFlow 2, [97]) are used.
- **Inception-v3** [47]: Utilizing ”Inception” blocks, this CNN architecture connects convolutional layers in a sparse manner. This reduces the number of connections between layers and, hence, the number of parameters – allowing the addition of many more layers at the same computational budget. This version, consisting of 48 layers, held the record on ImageNet for some time. It is also the same classifier that was evaluated in Chapter 4. The publicly available implementation and pre-trained model provided in TensorFlow 1 [97] are used.
- **ResNet-101** [72]: The introduction of residual connections proved a breakthrough in CNNs. Effectively, their use mitigates the vanishing gradient problem, therefore allowing the addition of even more convolutional layers. This model uses 101 layers, but versions exist that contain more than 1000 layers. The publicly available implementation and pre-trained model provided in TensorFlow 1 [97] are used.
- **Inception-ResNet-v2** [94]: In a natural progression, this CNN classifier aims to combine the advantages of both Inception blocks and residual connections. It has 164 layers. The publicly available implementation and pre-trained model provided in TensorFlow 1 [97] are used.
- **DenseNet-201** [100]: This CNN architecture follows the concept of residual connections to its obvious conclusion: Simply connect each block to *every* other block. Perhaps counter-intuitively, this can reduce the number of parameters, since the

increased information flow allows layers to be smaller (fewer redundant convolutional filters). As a result, this architecture is considered more efficient than others, achieving high accuracy at low computational cost. This version has 201 layers. The publicly available implementation and pre-trained model provided in Keras (TensorFlow 2, [97]) are used.

### 5.2.1.2 With self-supervised pre-training

Additionally, the following classifiers were constructed with self-supervised pre-training:

- **Noisy Student (EfficientNet)** [98]: This CNN classifier was pre-trained in self-supervised fashion on 300 million unlabeled images from the (unpublished) JFT-300M data set. After this very expensive procedure, it was fine-tuned on the final 1.2 million labeled examples of ImageNet. This form of pre-training has been found to greatly boost accuracy.

The "noisy student" procedure augments unlabeled images with noise before distilling them multiple times in a pre-training loop. As such, it has been found to increase the quality of learned features, leading to higher accuracy. Notably, this procedure is not unlike the Adversarial Training defense (see Section 2.4.3.6 of Chapter 2) and indeed, the authors claim higher robustness against adversarial attacks. This makes the model even more interesting to evaluate.

In the benchmark, the smallest (EfficientNet-B0) and a very large (EfficientNet-B7) model are compared. The reportedly best-performing model (EfficientNet-L2) is even larger and cannot be evaluated since it runs very slowly on the evaluation hardware. However, this should not be a problem as EfficientNet-B7 is reported to possess excellent accuracy [98], which is also confirmed by this evaluation. The publicly available TensorFlow implementation and pre-trained models provided by the authors in [98] are used.

- **Vision Transformer** [99]: This classifier is remarkable in that it not only scores state-of-the-art accuracy on ImageNet, but also that it is *not a CNN*. Instead, it successfully applies Transformers to image processing – an architecture that has recently taken the domain of Natural Language Processing by storm. It should be interesting to see if a model that consists purely of Transformers has the same adversarial vulnerabilities as CNN classifiers.

Same as Noisy Student EfficientNet, this classifier was pre-trained in self-supervised fashion on the JFT-300M data set before being fine-tuned on ImageNet.

In the benchmark, both a small (ViT-B/16) and large (ViT-L/16) model are compared. Again, the reportedly best-performing model (ViT-H/14) cannot be evaluated, since it runs extremely slowly due to its size. The publicly available JAX implementation and pre-trained models provided by the authors in [99] are used.

All classifiers are evaluated with both the "model-free" and the "strong" version of the Biased Boundary Attack, as described in Section 5.1.4. Since the strong attack derives



gradients and synthesizes starting points from a ResNet-101 surrogate model, it cannot fairly be performed on ResNet-101 itself. For this reason, ResNet-101 is evaluated using an Inception-ResNet-v2 surrogate instead.

### 5.2.2 Results

Tables 5.1 and 5.2 show the evaluation results and Figures 5.1 and 5.2 plot success rate against query count. It is apparent that all of the evaluated classifiers are vulnerable to adversarial attack – including the Vision Transformer (ViT), which is not a CNN.

Models pre-trained on large amounts of data seem to enjoy a slight advantage – most of all Noisy Student, which claims adversarial robustness. However, this advantage quickly melts away when confronted with a stronger attack, which consistently scores over 90% success rate against all classifiers. While Noisy Student can boast the lowest attack success rate, 94%, this is very close to the competition and therefore a very small win. Compare Figure 5.2, where the success line is barely below the others. It goes to show that Noisy Student pre-training can improve robustness against weak attacks, but not against a strong attack in this setting.

Perhaps interestingly, ViT is less impacted by the strong attack than other classifiers, especially at low query counts. The strong attack uses gradients from a CNN surrogate (ResNet-101) and creates starting points from salient patches that were extracted from the same surrogate. The low attack success rate at the beginning shows that the initialization method is not as effective against ViT as it is against CNNs, indicating a lower transferability between CNNs and ViT. Still, a limited degree of transferability exists – otherwise, the strong attack could not have cut the median number of required queries in half.

All in all, it can be said that state-of-the-art deep learning image classifiers are not robust against adaptive black-box attacks. More data seems to help, but the difference is so small as to be negligible when faced with a strong attack.

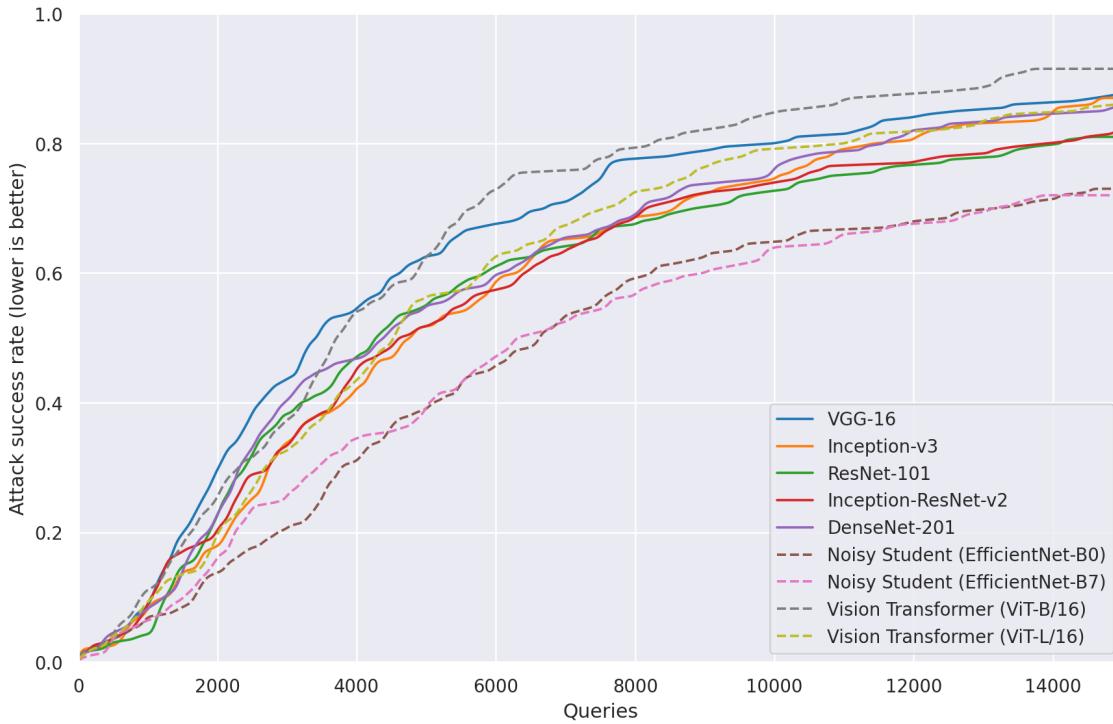
## 5 Benchmarking the robustness of image classifiers

CLASSIFIER	CLEAN ACC.	ATTACK SUCCESS RATE VS QUERY COUNT						MEDIAN QUERIES
		500	1000	2500	5000	10000	15000	
VGG-16 [88]	0.69	0.04	0.09	0.39	0.63	0.80	0.88	3451
Inception-v3 [47]	0.77	<b>0.03</b>	0.08	0.28	0.53	0.76	0.86	4751
ResNet-101 [72]	0.76	<b>0.03</b>	<b>0.04</b>	0.32	0.55	0.73	0.81	4297
Inception-ResNet-v2 [94]	0.79	0.04	0.10	0.29	0.52	0.74	0.82	4675
DenseNet-201 [100]	0.77	0.05	0.08	0.33	0.55	0.76	0.86	4415
Noisy Student (EfficientNet-B0) [98]	0.78	0.04	0.07	<b>0.17</b>	<b>0.39</b>	<b>0.65</b>	0.73	<b>6631</b>
Noisy Student (EfficientNet-B7) [98]	<b>0.86</b>	0.04	0.06	0.24	<b>0.39</b>	<b>0.65</b>	<b>0.72</b>	6374
Vision Transformer (ViT-B/16) [99]	0.83	0.04	0.11	0.31	0.63	0.85	0.92	3724
Vision Transformer (ViT-L/16) [99]	0.84	0.04	0.09	0.27	0.56	0.79	0.86	4549

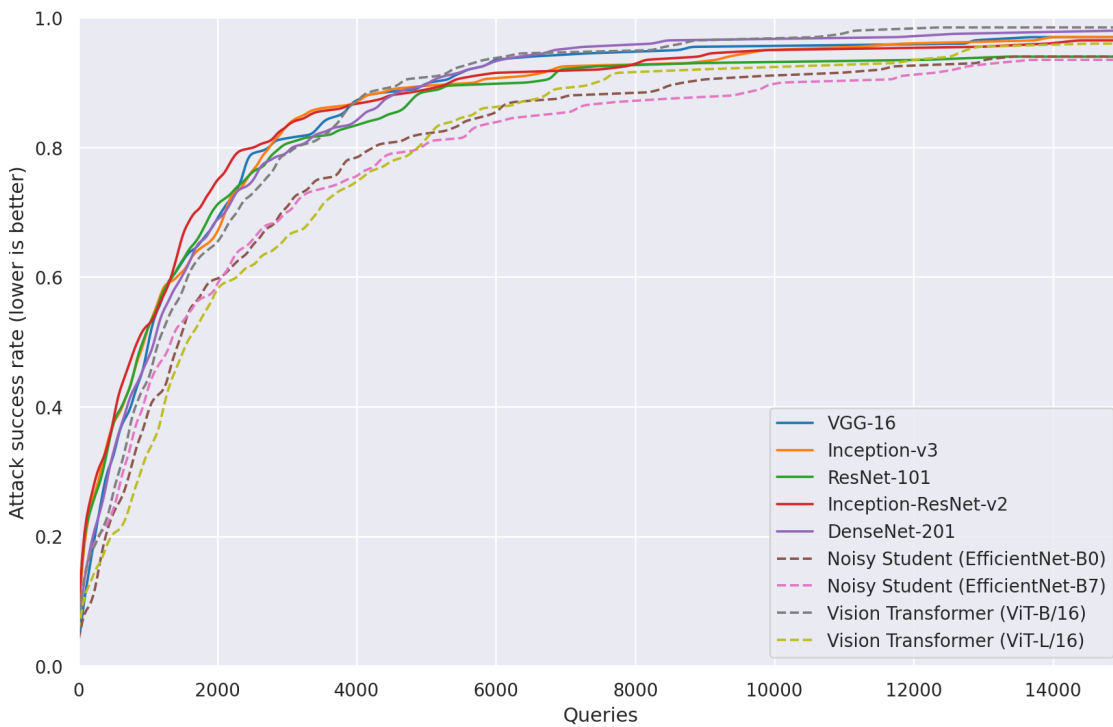
**Table 5.1:** Empirical robustness of deep learning image classifiers, measured with the Biased Boundary Attack (model-free version). Top section: trained on ImageNet in standard supervised manner. Bottom section: trained in self-supervised fashion on multiple, much larger image datasets. Even though there are large discrepancies in accuracy on clean images, the difference in robustness is rather small. The notable exception to this is EfficientNet with Noisy-Student self-supervised training, for which success rate is lower than for others, with the attack requiring a median of 40% more queries to achieve it.

CLASSIFIER	CLEAN ACC.	ATTACK SUCCESS RATE VS QUERY COUNT						MEDIAN QUERIES
		500	1000	2500	5000	10000	15000	
VGG-16 [88]	0.69	0.32	0.50	0.79	0.90	0.96	0.97	1016
Inception-v3 [47]	0.77	0.38	0.53	0.77	0.90	0.95	0.97	954
ResNet-101 [72]	0.76	0.37	0.53	0.76	0.89	0.93	<b>0.94</b>	941
Inception-ResNet-v2 [94]	0.79	0.37	0.53	0.80	0.89	0.95	0.97	866
DenseNet-201 [100]	0.77	0.33	0.48	0.75	0.90	0.97	0.98	1105
Noisy Student (EfficientNet-B0) [98]	0.78	0.24	0.39	0.65	0.82	0.91	<b>0.94</b>	1454
Noisy Student (EfficientNet-B7) [98]	<b>0.86</b>	0.23	0.43	0.66	<b>0.80</b>	<b>0.90</b>	<b>0.94</b>	1335
Vision Transformer (ViT-B/16) [99]	0.83	0.27	0.44	0.73	0.91	0.97	0.99	1163
Vision Transformer (ViT-L/16) [99]	0.84	<b>0.21</b>	<b>0.33</b>	<b>0.62</b>	0.82	0.92	0.96	<b>1565</b>

**Table 5.2:** Empirical robustness of deep learning image classifiers, measured with the Biased Boundary Attack (strong version). All classifiers are much less robust under the stronger attack. Although the Noisy-Student-pretrained models still boast lower success rates, curiously, the large Vision Transformer (ViT-L/16) has the highest median required queries. This difference may be due limited transferability of the surrogate - after all, the Vision Transformer is the only deep learning classifier without a CNN architecture.



**Figure 5.1:** Attack success rate against various deep learning image classifiers, measured with the Biased Boundary Attack (model-free version, plot of Table 5.1).



**Figure 5.2:** Attack success rate against various deep learning image classifiers, measured with the Biased Boundary Attack (strong version, plot of Table 5.2).

### 5.3 Classic computer vision methods

It is well known that deep neural networks are vulnerable to adversarial examples and the previous experiment has confirmed that even state-of-the-art classifiers (ViT, 2021, [99]) are little more robust than those designed years ago (VGG-16, 2014, [88]). Adversarial examples have led some to speculate that there might be something wrong with Deep Learning – that deep neural networks are flawed in some mysterious and fundamental way, and that they may even be inherently unsafe for use in the real world [101, 102].

But what could be the alternative? Before the rise of deep learning, more traditional computer vision techniques like the SIFT feature descriptor [103] and classifiers like the Support Vector Machine (SVM) [28] were in widespread use across various industries [104, 105]. These were considered reasonably safe, and the phenomenon of adversarial examples was discussed very little. Although adversarial classification was already known at the time [59], it was considered a mere curiosity and far from a serious threat.

And why would it be one? Traditional image classification pipelines typically consist of hand-engineered feature extractors and encoding schemes. Once an image is encoded, it is fed to simple and well-researched classifiers such as Logistic Regression, Decision Trees, or SVMs.

It would seem that – in such well-defined pipelines – everything is under control. After all, descriptors such as SIFT are specifically designed to extract robust features. And classifiers such as the SVM offer convex optimization, global minima, and large margins – robustness by design, together with proofs and theoretical guarantees. This is a stark contrast to Deep Learning, which offers none of these things [106]. Surely, tested-and-true methods would be much more robust to adversarial perturbations?

This section evaluates the robustness of traditional image classification pipelines. They are subjected to the Biased Boundary Attack, and the results are compared against a state-of-the-art CNN classifier.

*Acknowledgement of prior work:* The implementations of some traditional CV methods in this section are adapted from unpublished work by Ege Özsoy<sup>4</sup>.

#### 5.3.1 Simplifying the task: a 10-class subset of ImageNet

Unfortunately, it is not easy to evaluate classic CV methods directly on ImageNet. To the best of this author’s knowledge, there are no pre-trained classifiers available for public download, and the only attempts to scale traditional pipelines to ImageNet were done in the context of the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [77].

Even so, starting in 2012 the ILSVRC became dominated by CNN classifiers, rendering traditional CV obsolete in the context of that competition. The only reliable accounts on classic CV for ImageNet are therefore from the winners of the 2011 and 2010 competition, which report the extreme difficulty of scaling up their methods to 1000 classes and

---

<sup>4</sup>Ege Özsoy: Adversarial examples that fool classic computer vision methods. *Bachelor thesis (unpublished)*. Technical University of Munich, 2019. The author of this dissertation advised Mr. Özsoy during the creation of his thesis.

millions of examples [77]. Intricate fine-tuning and large computational resources were necessary, which is out of scope for the evaluation in this thesis.

Therefore, the robustness of classic CV methods must be evaluated on a simpler problem. Towards this end, a 10-class subset of ImageNet is created. To make sure that targeted adversarial attacks on these classes are not trivial, they each represent meaningfully different semantic concepts:

- Number of classes: 10
- Class labels: *tench, English springer, cassette player, chain saw, church, French horn, garbage truck, gas pump, golf ball, parachute*<sup>5</sup>.
- Image resolution: 299×299, RGB color
- Total number of training images: 12894
- Total number of validation images: 500

On this simple data set, classic CV pipelines can be trained and evaluated rather quickly. For the robustness benchmark, an evaluation set with 200 images is created from the validation data, together with random adversarial target labels. In this way, the benchmark setting is the same as for Deep Learning classifiers (Section 5.2), but at a reduced scale.

### 5.3.2 Building ImageNet classifiers with traditional CV methods

Traditional image classification pipelines typically consist of at least two stages: feature extraction and classification. Depending on the complexity of the extracted features, an additional encoding stage is often added in the middle, which reduces feature density. The resulting feature vectors are then processed with comparatively simple classifiers.

#### 5.3.2.1 Feature extractors

The evaluation compares two popular feature extractors:

- **Scale Invariant Feature Transform (SIFT)** [103]: One of the most popular feature descriptors, SIFT has been a staple in image processing pipelines for many years. SIFT features are designed to be invariant to scaling, rotation and translation, and robust to illumination changes. SIFT identifies a variable number of keypoints in an image and assigns a feature histogram to each, resulting in feature vectors of 128 dimensions for each keypoint.

The implementation available in OpenCV 4.4 [108] is used with default parameters. Since an arbitrary number of keypoints can be detected per image (often more than 100), SIFT features require further encoding to reduce their dimensionality.

---

<sup>5</sup>These are the same classes as used in previous work on the similar "ImageNette" dataset [107].

- **Histogram of Oriented Gradients (HOG)** [109]: Conceptually simple, but often considered equally powerful, HOG splits an image into blocks and then stores histograms of the image gradients for each, together with the gradient orientations. An additional normalization step increases robustness to illumination changes.

The implementation available in Scikit-image 0.17.2 [110] is used, with 10 pixels per block and 2 cells per block. This vector can be directly fed to a classifier without the need for further encoding.

### 5.3.2.2 Encodings

For the encoding stage, two clustering schemes are considered:

- **Bag of Visual Words (BoVW)** [111]: Similar to the popular Bag of Words (BoW) encoding used in natural language processing, the BoVW interprets feature descriptors as "visual words" and organizes them in a dictionary. Typically, this dictionary is created by performing k-means clustering on all features extracted from the training data set. Any image can then be expressed as a combination of these clusters, or a histogram of visual words. The resulting representation is sparse, making it well-suited for simple classifiers.

The implementation available in OpenCV 4.4 [108] is used with default parameters (k-means clustering). The vocabulary size is set to 2500.

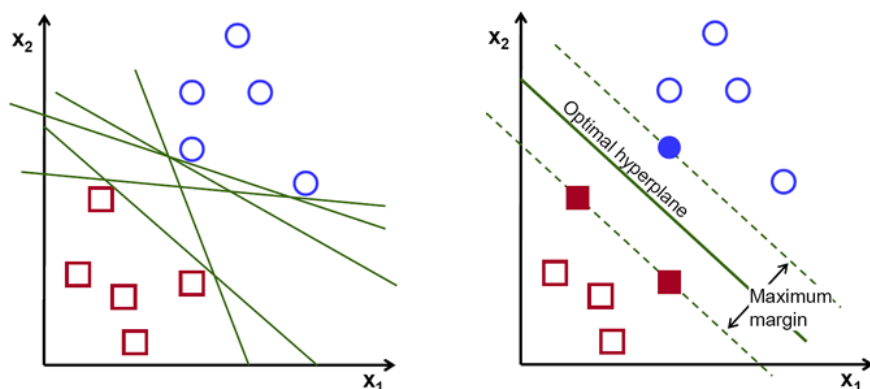
- **Fisher Vector (FV)** [112]: Fisher Vector encoding performs clustering with a Gaussian Mixture Model (GMM) instead of k-means, which effectively makes it a probabilistic version of the simpler BoVW encoding. This has the advantage that, for each encoded example, not only the visual words are stored, but also the example's deviation from the cluster centers. This advanced encoding can greatly improve accuracy and has played a pivotal role in winning submissions to ILSVRC in 2010 and 2011 [77]. In other words, it was the state of the art on ImageNet before the advent of Deep Learning.

The implementation available in the fishervector library [113] is used with default parameters, the number of GMM components is set to 256. Following the example of the ILSVRC2011 winners, Principal Component Analysis (PCA) is used to reduce the dimensionality of SIFT features from 128 to 64 before creating the encoding.

### 5.3.2.3 Classification methods

Finally, three classifiers are compared:

- **Logistic Regression (LR)** [114]: Possibly the simplest way to separate data into distinct classes, LR is a linear classifier that uses the logistic function to produce class probabilities. The class with the highest probability is picked as the prediction.



**Figure 5.3:** Max-margin classification in Support Vector Machines (SVM). Two classes (circles and squares) are linearly separated with a hyperplane. (Left): There exist many ways to perform classification with perfect accuracy, but not all of them have a large margin. (Right): A linear SVM picks the hyperplane that maximizes the distance (margin) between data points of different classes. This margin directly corresponds to  $\epsilon_{adv}$ , the adversarial robustness. Figures taken from Garcia et al. [30].

Interestingly enough, this simple classifier is also employed in the last layer of most CNN architectures<sup>6</sup>.

The implementation available in Scikit-learn 0.23.2 [115] is used, along with default parameters (multinomial regression for multi-class prediction).

- **Support Vector Machine (SVM)** [28]: In its linear form, the Support Vector Machine is very similar to LR. However, being motivated from a geometrical perspective, it does not produce probabilities but a hyperplane that separates data points of different classes by the maximum possible margin (See Figure 5.3). Being a linear classifier, LR also produces such a hyperplane, but without the margin.

The SVM can also produce highly nonlinear decision boundaries by applying the kernel trick. However, kernel SVMs are known to scale badly with the number of data points, which makes them unsuited for large-scale classification tasks. Consequently, traditional CV pipelines for ImageNet have focused on linear SVMs (see results of the ILSVRC challenge 2010 and 2011 [77]).

The LinearSVC module available in Scikit-learn 0.23.2 [115] is used. The regularization weight  $C$  is set to 3, which was found to produce the highest accuracy. The SVM is used in one-vs-all mode to enable multiclass predictions.

- **Decision Trees with AdaBoost (AB)** [116]: Last but not least, a decision tree classifier is evaluated. A single decision tree is typically not enough for accurate image classification, but multiple trees can be combined into an ensemble

<sup>6</sup>CNNs employ multiple convolutional layers, followed by a single fully-connected layer that outputs the classification result. Being a linear softmax classifier, this last layer performs the same function as multinomial logistic regression. In essence, CNNs follow the same recipe as successful classic CV pipelines: Combine powerful feature extractors (convolutional layers) with a simple classifier (LR).

## 5 Benchmarking the robustness of image classifiers

by boosting. The popular AdaBoost algorithm increases accuracy and promotes diversity by training additional trees specifically on examples where previous trees did badly. In this way, it adaptively increases the size of the ensemble to match challenging data.

The implementation available in Scikit-learn 0.23.2 [115] is used, along with default parameters (maximum number of trees = 50). This implementation natively supports multi-class prediction.

### 5.3.2.4 Comparison to CNNs

Since the classic CV pipelines are now evaluated on a much simpler task (smaller data set), they cannot be directly compared to the results on Deep Learning classifiers in Section 5.2. For this reason, an additional Inception-v3 CNN classifier is trained on the same 10-class ImageNet subset and added to the evaluation. This allows a relative comparison between classic and CNN methods.

The model uses the Inception-v3 architecture, replacing only the final layer with a 10-way classifier (instead of the original 1000). The network is randomly initialized and trained for 50 epochs on the 10-class ImageNet subset. The Adam optimizer is used with a minibatch size of 32, a learning rate of 0.001, and exponential decay (0.95 per epoch).

No data augmentation or hyperparameter search is used. This makes the setup very simple and comparable to the classic CV pipelines, which have not been extensively fine-tuned either.

### 5.3.3 Results

This Section first compares the accuracy of the various classification pipelines on clean data and then evaluates their robustness with the Biased Boundary Attack. Finally, the results are put into perspective with regard to modern CNN image classifiers.

#### 5.3.3.1 Accuracy

Figure 5.4 plots the classification accuracy (top-1) of the various combinations. Most pipelines struggle to reach 50%. The strongest is SIFT+PCA+FV+SVM (55%), which is also the configuration most similar to the winning submission in the ILSVRC 2011 competition [77]. Even though the numbers could surely be improved by fine-tuning, this shows that the choice of traditional CV techniques is fairly representative.

The boosted decision tree ensemble classifier (AdaBoost) has the lowest accuracy, never achieving more than 36%. For comparison, a random choice would yield 10%<sup>7</sup>.

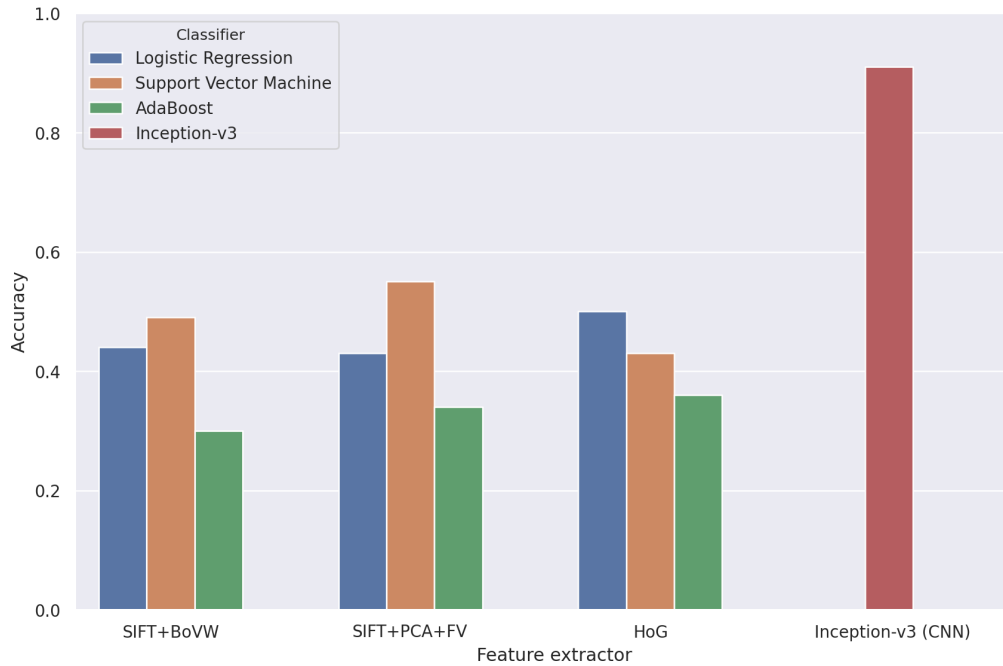
The CNN reaches 91%, which is significantly higher than the classic methods. This is to be expected but still impressive, considering no data augmentation or other fine-tuning was used for training.

---

<sup>7</sup>Random choice corresponds to 10% accuracy on the 10-class subset of ImageNet, but 0.1% on the 1000-class full data set.



### 5.3 Classic computer vision methods



**Figure 5.4:** Classification accuracy of traditional computer vision pipelines, compared with a CNN classifier. Even the strongest combination, SIFT+PCA+FV+SVM, does not surpass 55%, while the CNN classifier reaches 91%. All are evaluated on a 10-class subset of ImageNet, a task that is significantly easier than classification on the full ImageNet data set.

	(all)	Tench	English springer	Cassette player	Chain saw	Church	French horn	Garbage truck	Gas pump	Golf ball	Parachute
SIFT+BoVW+LR	0.44	0.32	0.66	0.42	0.10	0.16	0.68	0.42	0.56	0.50	0.62
SIFT+BoVW+SVM	0.49	0.48	0.66	0.46	0.30	0.28	0.66	0.40	0.52	0.58	0.56
SIFT+BoVW+AB	0.30	0.38	0.44	0.20	0.02	0.06	0.42	0.28	0.38	0.38	0.40
SIFT+PCA+FV+LR	0.43	0.26	0.68	0.38	0.00	0.08	0.74	0.44	0.58	0.56	0.62
SIFT+PCA+FV+SVM	0.55	0.40	0.84	0.63	0.32	0.22	0.66	0.62	0.58	0.68	0.58
SIFT+PCA+FV+AB	0.34	0.26	0.40	0.38	0.08	0.12	0.38	0.36	0.44	0.44	0.50
HOG+LR	0.50	0.60	0.50	0.66	0.32	0.50	0.34	0.62	0.48	0.34	0.62
HOG+SVM	0.43	0.36	0.34	0.68	0.18	0.42	0.30	0.58	0.48	0.36	0.64
HOG+AB	0.36	0.60	0.30	0.58	0.06	0.32	0.16	0.56	0.36	0.06	0.60
Inception-v3	0.91	0.92	1.00	0.94	0.80	0.90	0.92	0.94	0.86	0.82	0.96

**Table 5.3:** Accuracy for each class of the evaluation data set. Accuracies below 20% are marked in red. It is apparent that most of the traditional CV classifiers have near-zero accuracy for "chain saw" and "church". An attack with these classes as adversarial target would be pointless. To allow a meaningful comparison, adversarial attacks in the benchmark target the 8 remaining classes exclusively.

**Excluding underperforming classes.** The overall accuracy of some classifiers is low, which could endanger the validity of the robustness evaluation. After all, if a model gives mostly wrong answers, what point is there in being robust? Since there are only 10 classes in the data set, it is possible to investigate the models more in detail and plot their accuracy per class.

See Table 5.3: It is apparent that several of the traditional CV pipelines have near-zero accuracy for classes "chain saw" and "church". Indeed, if a model fails to correctly classify even a single real example, then it would be trivial to make this model robust to targeted adversarial examples (with that specific class as  $y_{target}$ ) as well. Further evaluation would then be pointless.

To remedy the issue and level the playing field, the problematic classes "chain saw" and "church" are excluded from the target labels of the evaluation data set. As a result, adversarial attacks are limited to the 8 remaining labels.

### 5.3.3.2 Adversarial robustness

Tables 5.4 and 5.5 show the evaluation results, while Figures 5.5 and 5.5 plot success rate against query count.

**Classic CV pipelines are vulnerable to adversarial attack.** For all combinations, adversarial examples could be found, with success rates ranging between 76-100%. Even the simple model-free attack was successful for 76% of the evaluation images against the most robust combination (SIFT+PCA+FV+LR). This demonstrates that traditional CV image classifiers are not significantly more robust than Deep Learning methods. It is true that most of the classic CV configurations demonstrate slightly higher robustness than the CNN. However, this small increase is not enough to defend against the attack and comes at the cost of much lower accuracy.

**Adversarial features can transfer from CNNs to classic CV.** The strong attack greatly reduces the number of queries needed for success when compared to the model-free attack. Although the difference is not as drastic as for the Deep Learning classifiers evaluated in Section 5.2, the strong attack still succeeds in cutting the number of queries in half even for most classic CV models. It does so by using gradients (for the surrogate bias) and saliency masks (for initialization) extracted from a CNN surrogate model, providing evidence that some degree of transferability between CNNs and traditional image classifiers exists.

**Robust combinations.** Combinations of SIFT and LR seem to be more robust than the rest. While even the model-free attack routinely reaches >90% success on other classifiers, SIFT+BoVW+LR admits only a 76% success rate. Even with the stronger attack, the success rate increases only to 80%. This seems peculiar, especially since LR is not robust at all when combined with HOG. At the same time, HOG+AB reaches similar levels of robustness, while AB performs badly with SIFT.

Classic CV methods do not offer automatic robustness by design and they must be combined in very specific ways to obtain it. In any case, even the most robust classifier in this benchmark could still be attacked with 80% success rate, which makes these robustness gains incremental at best.

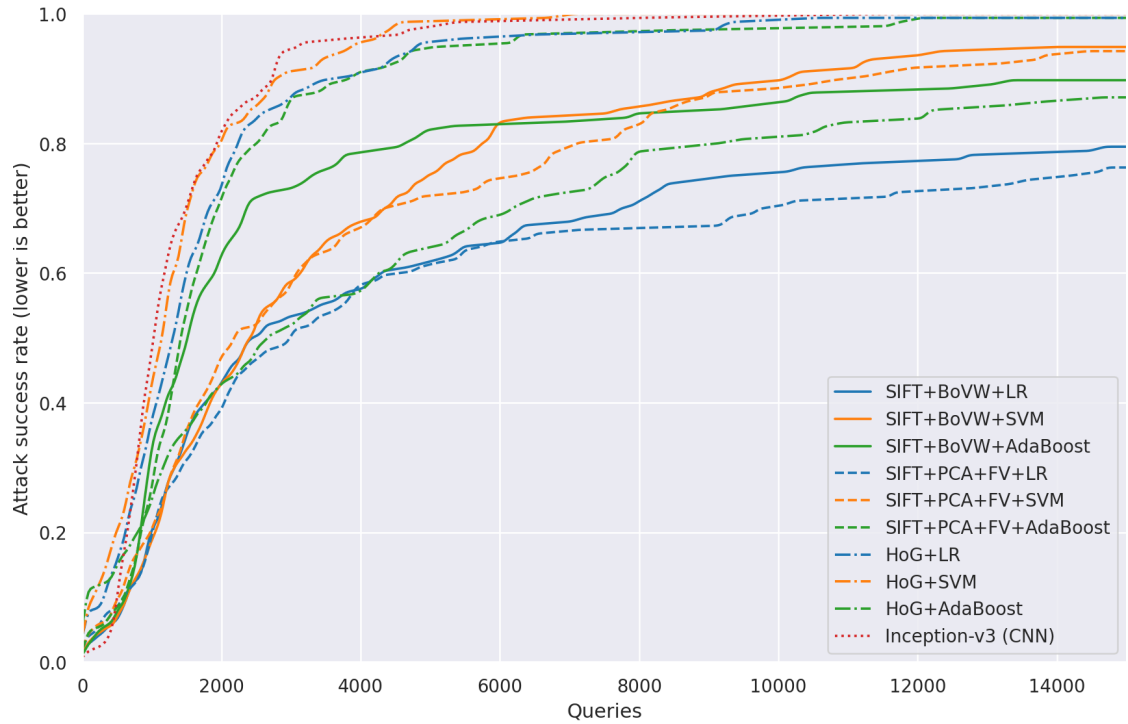
CLASSIFIER	CLEAN ACC.	ATTACK SUCCESS RATE VS QUERY COUNT						MEDIAN QUERIES
		500	1000	2500	5000	10000	15000	
SIFT+BoVW+LR	0.44	<b>0.06</b>	0.21	0.50	0.62	0.77	0.79	2507
SIFT+BoVW+SVM	0.49	<b>0.06</b>	<b>0.19</b>	0.52	0.75	0.89	0.95	2442
SIFT+BoVW+AB	0.30	0.08	0.33	0.71	0.82	0.86	0.90	1525
SIFT+PCA+FV+LR	0.43	0.08	<b>0.19</b>	<b>0.46</b>	<b>0.61</b>	<b>0.70</b>	<b>0.76</b>	<b>3002</b>
SIFT+PCA+FV+SVM	0.55	0.10	<b>0.19</b>	0.51	0.72	0.88	0.94	2188
SIFT+PCA+FV+AB	0.34	0.08	0.26	0.79	0.94	0.97	0.99	1407
HOG+LR	0.50	0.15	0.37	0.83	0.96	0.99	0.99	1309
HOG+SVM	0.43	0.21	0.43	0.85	0.99	1.00	1.00	1157
HOG+AB	0.36	0.14	0.24	0.48	0.64	0.81	0.87	2655
Inception-v3 (CNN)	<b>0.91</b>	0.09	0.49	0.87	0.97	0.99	1.00	1037

**Table 5.4:** Empirical robustness of traditional image classification pipelines, measured with the Biased Boundary Attack (model-free version). Combinations of SIFT and LR are the most robust, while the CNN is the least robust. Interestingly, the most (SIFT+PCA+FV+LR) and least (HOG+SVM) robust pipelines have the same accuracy.

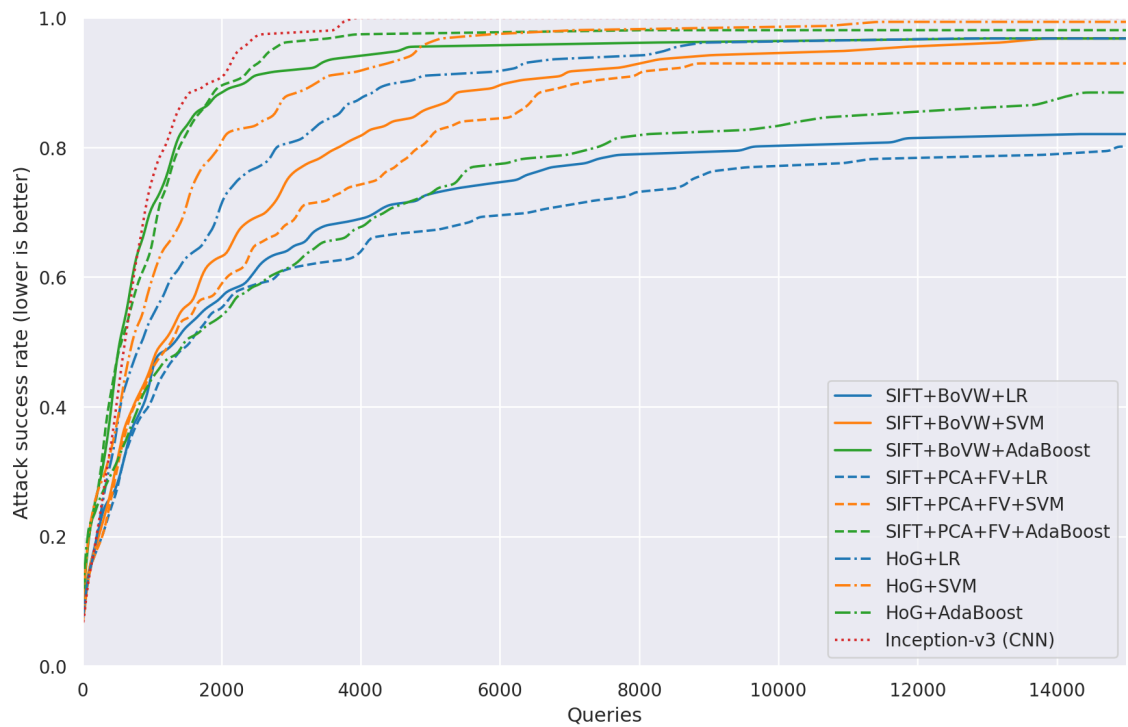
CLASSIFIER	CLEAN ACC.	ATTACK SUCCESS RATE VS QUERY COUNT						MEDIAN QUERIES
		500	1000	2500	5000	10000	15000	
SIFT+BoVW+LR	0.44	0.28	0.46	0.61	0.72	0.80	0.82	1356
SIFT+BoVW+SVM	0.49	0.30	0.46	0.69	0.86	0.94	0.97	1189
SIFT+BoVW+AB	0.30	0.47	0.71	0.91	0.96	0.96	0.97	531
SIFT+PCA+FV+LR	0.43	<b>0.26</b>	<b>0.40</b>	<b>0.58</b>	<b>0.67</b>	<b>0.77</b>	<b>0.80</b>	<b>1578</b>
SIFT+PCA+FV+SVM	0.55	0.29	0.46	0.65	0.81	0.93	0.93	1302
SIFT+PCA+FV+AB	0.34	0.48	0.64	0.92	0.97	0.98	0.98	579
HOG+LR	0.50	0.36	0.53	0.76	0.91	0.96	0.97	876
HOG+SVM	0.43	0.37	0.59	0.83	0.96	0.98	0.99	719
HOG+AB	0.36	0.31	0.44	<b>0.58</b>	0.72	0.83	0.88	1490
Inception-v3 (CNN)	<b>0.91</b>	0.41	0.74	0.97	1.00	1.00	1.00	610

**Table 5.5:** Empirical robustness of traditional image classification pipelines, measured with the Biased Boundary Attack (strong version). Even though the attack relies on a CNN surrogate model, it is still effective against non-CNN classifiers, cutting the number of required queries roughly in half. Again, a combination of SIFT and LR is the most robust, with the least robust being combinations of SIFT and AdaBoost. Still, even against the most robust classifier, the attack succeeds for 80% of the evaluation images and requires only a median of 1578 queries. This shows that classic CV pipelines are very vulnerable to adversarial examples, offering only a marginal improvement in robustness over the much more accurate CNN.

## 5 Benchmarking the robustness of image classifiers



**Figure 5.5:** Attack success rate against various traditional computer vision pipelines, measured with the Biased Boundary Attack (model-free version, plot of Table 5.4).



**Figure 5.6:** Attack success rate against various traditional computer vision pipelines, measured with the Biased Boundary Attack (strong version, plot of Table 5.5).

CLASSIFIER	# CLASSES		CLEAN ACC.	ATTACK SUCCESS RATE VS QUERY COUNT						MEDIAN QUERIES
	TRAIN	EVAL		500	1000	2500	5000	10000	15000	
SIFT+PCA+FV+LR	10	10	0.43	0.08	0.19	0.46	0.61	<b>0.70</b>	<b>0.76</b>	3002
Inception-v3 (1)	10	10	0.91	0.09	0.49	0.87	0.97	0.99	1.00	1037
Inception-v3 (2)	1000	10	<b>0.99</b>	0.07	0.22	0.66	0.90	0.97	1.00	1841
Inception-v3 (3)	1000	10+990	0.80	<b>0.01</b>	<b>0.02</b>	<b>0.17</b>	<b>0.43</b>	0.71	0.85	<b>5987</b>

**Table 5.6:** CNNs can leverage additional data to increase accuracy and robustness. (1): An Inception-v3 classifier, trained with data from 10 classes, with 10 outputs – the same as in Table 5.4. (2): Trained with the full ImageNet data set (1000 classes); the 1000 outputs are rounded to the 10 evaluation classes. (3): Trained with the full ImageNet data set (1000 classes); the 1000 outputs are not rounded. Since the evaluation data set contains only 10 classes, any prediction of the remaining 990 is automatically invalid. With this information, it is trivial to convert a large pre-trained model into an anomaly detector that increases robustness. This is easy to perform for CNNs, which scale readily to large amounts of data. The same cannot be said for traditional CV methods, where models of that size are known to underperform while being very difficult to create [77]. At the time of writing, no 1000-class traditional CV model could be freely obtained for ImageNet.

CLASSIFIER	# CLASSES		CLEAN ACC.	ATTACK SUCCESS RATE VS QUERY COUNT						MEDIAN QUERIES
	TRAIN	EVAL		500	1000	2500	5000	10000	15000	
SIFT+PCA+FV+LR	10	10	0.43	<b>0.26</b>	0.40	<b>0.58</b>	<b>0.67</b>	<b>0.77</b>	<b>0.80</b>	<b>1578</b>
Inception-v3 (1)	10	10	0.91	0.41	0.74	0.97	1.00	1.00	1.00	610
Inception-v3 (2)	1000	10	<b>0.99</b>	0.31	0.56	0.87	0.97	1.00	1.00	834
Inception-v3 (3)	1000	10+990	0.80	<b>0.26</b>	<b>0.38</b>	0.73	0.88	0.96	0.99	1421

**Table 5.7:** The same experiment, evaluated against the stronger attack. It is apparent that the attack is more efficient against the CNNs than against the combination of SIFT and LR. However, the additional data still allows the CNN in (3) to achieve comparable levels of robustness while retaining much higher accuracy.

### 5.3.3.3 A second look at CNNs: Increasing robustness with more data

How to interpret these results? Classic CV methods seem slightly more robust than Deep Learning methods, so could they be a promising alternative?

First and foremost, the traditional CV classifiers have significantly worse accuracy than the CNN. A drop from 91% (CNN) to 43% (the most robust classic model) cannot be justified by small gains in robustness (most attacks are still successful).

But there is another advantage to Deep Learning methods: They can easily be improved by adding more data. The same cannot be said for traditional CV pipelines, which have so far struggled to scale to ImageNet dimensions [77] and beyond. At the time of writing, no full ImageNet model was publicly available to the best of the author’s knowledge.

**Repurposing a 1000-class model.** For CNNs, it is trivial to extend the training set from 10 to 1000 classes and use the full ImageNet data set. Such classifiers are easily trained on consumer-grade hardware, and a wealth of them is publicly available. In an

additional experiment, an Inception-v3 CNN is trained on all 1000 classes but evaluated only on the 10-class subset. As Tables 5.6 and 5.7 show, this results in large gains of both accuracy and robustness:

- **Increasing accuracy:** It is trivial to take a 1000-class model and remove the 990 class outputs that are not in the evaluation data. As a result, the predicted class is then the arg max of the remaining 10. This has a rounding effect that increases accuracy on those 10 classes (from 91% to 99%). Notably, the robustness of this model is also slightly higher than that of the naively-trained CNN.
- **Increasing robustness:** As a variation, it is also possible to *not* remove the 990 outputs and instead treat them as "anomalous" classes – turning the 1000-class model into an anomaly detector on 10 classes without additional cost. Naturally, with only 10 of 1000 classes being valid, the space of adversarial inputs becomes substantially smaller. This increases the difficulty of attack, more than doubling the median number of required queries.

The results obtained in this Section indicate that traditional CV classifiers are slightly more robust than CNNs under empirical evaluation. However, similar robustness can be achieved in CNNs by adding more data – which is easy for Deep Learning methods, but not for classic approaches. Additionally, CNNs still vastly outperform classic methods in terms of accuracy. It may be worth pursuing traditional CV techniques on low-dimensional tasks for which very little data is available, but in the domain of large-scale image classification, Deep Learning firmly remains the state of the art.

## 5.4 Defenses against adversarial attack

So far, all of the evaluated classifiers have been shown vulnerable. However, none of these models were specifically designed to resist adversarial attacks – as such, it may be slightly unfair to conduct powerful attacks against undefended classifiers. There exists a large body of literature on how to make classifiers more robust, and this study would not be complete without a thorough comparison of such defenses.

### 5.4.1 Selecting suitable methods

As described in Section 2.4.3 of Chapter 2, adversarial defenses can be divided into six categories: Gradient obfuscation, ensembling, randomization, anomaly detection, pre-processing and adversarial training. As gradient obfuscation is irrelevant in the black-box label-only setting, only the remaining five are considered. For each category, at least one recently proposed defense is picked and evaluated in the same setting as the Deep Learning classifiers of Section 5.2 (full ImageNet, 1000 classes).

**Availability of reference implementations.** With the exception of ensembling, the code provided by the original authors of each defense is used. Since many methods are sensitive to hyperparameter choices, this is necessary to make sure all approaches are correctly implemented.

Unfortunately, this also limits the choice of defenses: Many proposed methods – even recent works – focus on much simpler data sets such as CIFAR10 and do not provide reference implementations for ImageNet. Many of these approaches do not scale well [117, 118, 82] and therefore cannot be performed for ImageNet, whose dimensionality is several orders of magnitude higher (268,203 vs 3,072). Therefore, this study can only evaluate defenses for which source code and hyperparameters on ImageNet have been made available.

**Runtime considerations.** Some defenses, especially those based on randomization and anomaly detection, have very high computational cost. Random Transforms [81], in their original implementation, require each input to be processed 16 times before classification. Some detectors based on statistical tests [119] even sample the same image over 1000 times. Naturally, such approaches are infeasible to evaluate and, surely, will have to be considerably sped up to be suitable for real-world applications.

### 5.4.2 Description of defenses

Defenses from the following five categories are compared:

#### 5.4.2.1 Ensembling

Ensembling of models has been known as a promising strategy for increasing robustness [17]. In a first and simple experiment, an ensemble of previously evaluated classifiers is formed:

- **Majority vote of classifiers.** It is straightforward to choose a few pre-trained classifiers that are known to perform well and combine them into an ensemble. Here, the ensemble is limited to three classifiers: Inception-v3, ResNet-101, and Inception-ResNet-v2. The models each process the same input image and then conduct a majority vote on the predicted class. The winner is chosen randomly in case of a tie.

*Note:* Against this model, the strong attack is modified to use DenseNet-201 as surrogate instead of ResNet-101 (which would violate the black box assumption).

#### 5.4.2.2 Randomization

Another popular scheme is to randomly perturb inputs before feeding them to the classifier. As is customary for stochastic defenses, multiple perturbed images are generated and the classification result is then averaged. This effectively emulates a randomized ensemble of classifiers. For the experiment, one of the most simple yet powerful approaches is considered:

- **Random Transforms** [81]: With this method, Xie et al. achieved second place in defenses in the NeurIPS 2017 Adversarial Vision Challenge. They make 16 copies of an input image, apply random upscaling and translation to the copies and then feed them to the classifier before averaging the result.

Notably, in their winning submission to the competition, they used this randomization in combination with an ensemble of strong adversarially-trained models. While evidently effective, this is problematic since it is not clear whether the robustness increase stems from randomization, ensembling, or adversarial training. To observe the effects separately, this experiment applies random transforms only to an otherwise undefended Inception-v3 model. In this way, it can be seen whether random transforms alone have a significant effect on robustness.

The reference implementation provided by Xie et al. is used. Since this method is extremely slow, the number of copies is reduced to 8 and these are passed to a simple Inception-v3 classifier. The classification result is obtained by averaging the logits.

#### 5.4.2.3 Anomaly detection

While a good number of approaches of detection-based approaches have been proposed recently, most of them are only available for much simpler data sets such as CIFAR10 [82, 120] and therefore cannot be included in the benchmark.

Others rely on excessive amounts of computational power, collecting statistics on the input space in the vicinity of an input to determine if it could be adversarial. For example, the popular detector called "The Odds Are Odd" [119] evaluates each image more than 1000 times. This would be far too slow. At the same time, reducing the number of samples to an acceptable level (e.g. 8) would likely render this type of detector ineffective.



This severely limits the choice of anomaly detectors on ImageNet. However, there is one method that claims robustness on ImageNet, provides a reference implementation and is reasonably fast to evaluate:

- **”Turning a Weakness Into a Strength” (TWIS)** [121]: In their paper, Hu et al. describe a combination of multiple criteria to detect adversarial examples: (1) They perturb the input with random noise. According to them, this is likely to flip the label of adversarial examples, allowing for their easy detection. (2) They also perform a multi-step PGD attack on every input to detect whether the input is close to the decision boundary of other classes.

The second criterion is expensive to evaluate – however, the authors claim that the first criterion is enough to reliably detect black-box attacks<sup>8</sup>. Therefore, this anomaly detector can be evaluated cheaply using only criterion (1).

The implementation provided by the original authors is used and applied to an Inception-v3 classifier.

Anomaly detectors are typically used for selective classification, i.e. they refuse service when an adversarial example is detected. It is easy to model a positive detection result as an additional output or class label  $y_{detected}$ : Whenever the attack is detected, the detector returns this additional label instead of the classifier’s prediction. For the targeted attack,  $y_{detected}$  is then simply another label to avoid while trying to achieve the adversarial label  $y_{target}$ .

#### 5.4.2.4 Pre-processing

Broadly speaking, pre-processing defenses work by applying a filter function to the input image. This filter is designed to remove the adversarial perturbation or, at least, diminish its effect. The pre-processed image is then fed to the classifier. In the evaluation, three notable pre-processing approaches are compared:

- **JPEG filter** [122, 83]: JPEG is a lossy image compression technique. An image that is compressed using JPEG and then decompressed again cannot be completely reconstructed and is often missing small, barely perceptible details. Recent work has found that this can also apply to adversarial perturbations [122] – therefore, pre-processing an image with JPEG is thought to be a simple but powerful defense.

Following the implementation by Guo et al. [83], images are compressed with a JPEG quality setting of 75 and decompressed again before feeding them to an Inception-v3 classifier.

- **Median filter** [17]: In the NeurIPS 2017 Adversarial Vision Challenge, the team that achieved 4th place in the defense track claimed that their success was based on median filtering.

---

<sup>8</sup>Specifically, they claim this for what they name gray-box attacks – these are subsumed by black-box attacks in our setting.

Specifically, Berdibekov et al. applied a median filter to images before feeding them to an ensemble of adversarially-trained classifiers. It is known that adversarial training and ensembling are strong defenses on their own [34], so it is unclear if the filter contributed much to the final result.

Based on the authors' description in the competition report [17], images are pre-processed with a 3x3 median filter. Afterward, they are fed to a standard Inception-v3 classifier so the effect of the filter can be observed in isolation.

- **U-net denoiser** [84]: With this method, Liao et al. achieved first place in the defense track of the NeurIPS 2017 competition. Their approach is to pre-process images with a convolutional U-net denoiser. Similar to a denoising autoencoder, this network compresses the image and then reconstructs it without the adversarial perturbation.

To be able to do so, the denoiser must be trained on adversarial examples. Liao et al. report training their winning model on a large number of perturbations created with PGD-style attacks.

In their reference implementation, Liao et al. provide a pre-trained denoiser for the Inception-v3 classifier. This model can be evaluated without any modification.

### 5.4.2.5 Adversarial Training

Adversarial Training aims to achieve robustness by adding adversarial examples to the training data. In this way, the model can be trained to become robust to worst-case perturbations. However, this process is far from simple as adversarial perturbations can take many forms and shapes.

The standard formula of Adversarial Training is to create a large number of adversarial examples using the very fast FGM method on the model that is being trained [33]. However, in practice, this leads to overfitting on a weak attack in a specific threat setting. While Adversarial Training has a powerful theoretical motivation, the devil is – as always – in the details.

The evaluation compares four implementations of Adversarial Training:

- **Ensemble Adversarial Training** [34]: This is one of the earliest works that claims improved black-box robustness on ImageNet. Specifically, Tramèr et al. deviate from the standard formula and create adversarial examples not for the model being trained, but for an ensemble of *different* classifiers. Since this is exactly the method employed by black-box transfer attacks, it would help train a model to be robust against them. This leads the authors to claim improved robustness in a black-box threat setting.

The authors provide an adversarially-trained version of Inception-ResNet-v2, which is used in the evaluation.

- **Feature Denoising Adversarial Training** [123]: Guided by the observation that adversarial perturbations are amplified by each layer of a neural network

until they become large enough to change the model’s output, Xie et al. propose to apply denoising operations not to the input image, but to the features found at various locations in the network.

They claim that this strategy greatly enhances adversarially-trained classifiers. Consequently, they provide only adversarially-trained models in their reference implementation, making it difficult to separate the idea of Feature Denoising from Adversarial Training.

The authors provide an adversarially-trained ResNet-152 classifier with added denoising blocks. This combination is used in the evaluation.

- **Large-scale Adversarial Training** [123]: Xie et al. also provide a baseline without Feature Denoising, an adversarially-trained but otherwise unmodified ResNet-152.

This classifier is interesting on its own since Xie et al. perform Adversarial Training on a very large scale: They generate perturbations with a 30-step PGD attack, which is very expensive as it increases the amount of computation needed for training by roughly a factor of 30. They report training for 52 hours on 128 NVIDIA V100 GPUs, which amounts to almost an entire year on a single GPU.

It should be interesting to see how well this classifier performs on its own (without Feature Denoising) and if it is more robust than Ensemble Adversarial Training, which utilizes a faster but weaker attack.

The adversarially-trained ResNet-152 classifier provided by Xie et al. is used.

- **Local Linearity Regularization (LLR)** [124]: Strictly speaking, this approach does not directly perform Adversarial Training but employs a regularization scheme that aims to achieve the same effect.

Specifically, Qin et al. find that Adversarial Training with strong attacks reduces the curvature of the adversarial loss function, causing its surface to become locally linear around training examples. Capitalizing on this insight, they propose a regularizer that enforces local linearity during training.

Similar to Adversarial Training, this additional loss term requires multiple inner optimization steps. However, the LLR term can be calculated much quicker than a strong adversarial example. According to the authors, LLR regularization achieves competitive robustness while offering a 5x speedup over large-scale Adversarial Training.

The authors provide a pre-trained ResNet-152 classifier, which is used in the evaluation.

### 5.4.3 Evaluation setting

All methods are evaluated in the same manner as undefended classifiers (Section 5.2). The Biased Boundary Attack is applied to all defenses, trying to generate adversarial examples for the 200 images in the evaluation set.

Since some of the attacks are very slow to evaluate (e.g. randomization and ensembling), only a single experiment is performed using the strong attack configuration (see Section 5.1.4).

As before, the surrogate model is a standard ResNet-101, which is replaced by Inception-ResNet-v2 for the Feature Denoising, large-scale AT and LLR defenses. This is done to prevent the usage of a surrogate that might be too similar to the classifier inside the black box.

### 5.4.4 Results

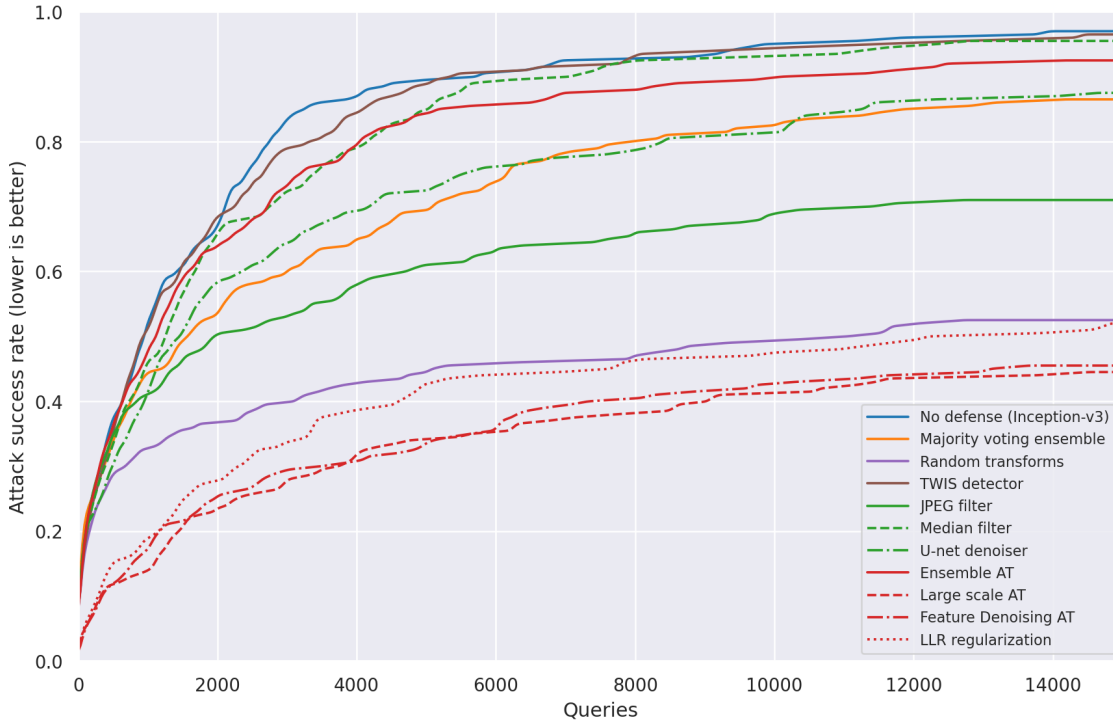
Table 5.8 shows the evaluation results and Figure 5.7 plots success rate against query count.

- **Majority voting ensemble:** This method offers a small but noticeable robustness increase (success rate reduces to 87% from 94-97% ), at the cost of tripled computational cost during inference. Notably, this is method has the highest accuracy on non-adversarial images.
- **Random transforms:** This defense seems to work well against the Biased Boundary Attack, reducing the success rate from 97% to 53%. However, it comes at the cost of eight-fold computational cost at inference time.
- **TWIS detector:** The detector appears to have no effect. Clearly, the assumption of being able to detect black-box adversarial examples only by proximity to the decision boundary does not hold against an adaptive attack. Even worse, the detector reduces accuracy on clean images by false positives.
- **JPEG filter:** This image filter offers a moderate improvement in robustness, reducing attack success rate to 71%. This is somewhat surprising, given that JPEG is largely a low-pass filter. Perhaps an even lower frequency in the low-frequency bias might be more effective against this defense. In any case, JPEG filtering seems reasonably effective at negligible cost.
- **Median filter:** This image filter seems to be ineffective. Quite possibly, it may be circumvented by the low-frequency bias. In any case, the success of the original implementation in the NeurIPS 2017 competition may be due to the use of Adversarial Training, but not due to median filtering as claimed.
- **U-net denoiser:** This pre-processor slightly improves robustness (87% vs 97% success rate) but does not have the effect that might be expected from the winner of the NeurIPS 2017 competition. Since the denoiser was trained on PGD-style attacks, it has likely overfitted to that particular style of perturbation. Quite possibly, it might be circumvented by the low-frequency bias. This demonstrates yet again that it is very difficult to "learn" the appearance of adversarial perturbations and that any defense that tries to do so suffers from overfitting.

## 5.4 Defenses against adversarial attack

DEFENSE	TYPE	BASE CLASSIFIER	CLEAN ACC.	ATTACK SUCCESS RATE VS QUERY COUNT						MEDIAN QUERIES
				500	1000	2500	5000	10000	15000	
(none)	-	Iv3	0.77	0.38	0.53	0.77	0.90	0.95	0.97	954
(none)	-	RN101	0.76	0.37	0.53	0.76	0.89	0.93	0.94	941
(none)	-	IRNv2	<b>0.79</b>	0.37	0.53	0.80	0.89	0.95	0.97	866
Majority voting ensemble	ENS	(the 3 above)	<b>0.79</b>	0.34	0.45	0.58	0.69	0.82	0.87	1597
Random transforms [81]	RND	Iv3	0.78	0.29	0.33	0.39	0.44	0.49	0.53	11291
TWIS detector [121]	DET	Iv3	0.75	0.35	0.51	0.74	0.89	0.94	0.97	951
JPEG filter [122, 83]	PRE	Iv3	0.74	0.35	0.41	0.51	0.61	0.69	0.71	1986
Median filter [17]	PRE	Iv3	0.74	0.33	0.46	0.68	0.85	0.93	0.96	1263
U-net denoiser [84]	PRE	Iv3	0.76	0.30	0.41	0.61	0.72	0.81	0.88	1493
Ensemble AT [34]	AT	IRNv2	0.78	0.37	0.48	0.68	0.84	0.90	0.93	1139
Large scale AT [123]	AT	RN152	0.55	<b>0.12</b>	<b>0.14</b>	<b>0.26</b>	<b>0.34</b>	<b>0.41</b>	<b>0.45</b>	≈22857
Feature Denoising AT [123]	AT	RN152	0.59	<b>0.12</b>	0.17	0.27	<b>0.34</b>	0.43	0.46	≈22500
LLR regularization [124]	AT	RN152	0.66	0.15	0.19	0.32	0.43	0.47	0.52	13028

**Table 5.8:** Empirical robustness of various proposed defense methods, measured with the Biased Boundary Attack (strong version). While a good number of defenses seem to be completely ineffective in the attack setting, both Random Transforms and Adversarial Training seem to significantly increase robustness. Although adversarial attack is still successful for 45% of the evaluation images, these techniques seem to hold the most potential.



**Figure 5.7:** Attack success rate against various defense methods, measured with the Biased Boundary Attack (strong version, plot of Table 5.8). While adversarially-trained classifiers are the most robust overall, they also seem to be much less vulnerable to the initialization method (starting points synthesized from salient patches). This is demonstrated by the much lower success rate early on – the other defenses merely succeed at slowing down the attack by various degrees.

- **Ensemble AT:** Perhaps surprisingly, this implementation of Adversarial Training seems to have little effect (93% vs 97%). Although it was considered the state of the art for some time, this model was trained only on very fast and weak FGM attacks transferred from unrelated models. Apparently, it has strongly overfitted on this kind of attack and is defenseless against other, adaptive, methods.
- **Large-scale AT:** This is the strongest defense in the evaluation, reducing the success rate to 45%. Clearly, large-scale AT demonstrates that training on strong multi-step attacks (PGD) is much more effective and also generalizes well to other, possibly unknown, attack methods. However, the accuracy on clean examples suffers greatly, being reduced to 55%.
- **Feature Denoising AT:** Interestingly, this model performs almost identically to large-scale AT. Apparently, Feature Denoising does not work against an adaptive attacker, while AT itself does. Again, it seems that the learned denoising operations suffer from overfitting to the noise they were trained on. However, the accuracy on clean examples does seem to benefit slightly (59%).
- **LLR regularization:** While slightly less robust than large-scale AT (52% success rate), this method has higher accuracy on clean images (66%). Considering that LLR is much cheaper at training time than large-scale AT, this approach seems very promising for future development.

#### 5.4.5 Suitability of defenses for real-world application

The evaluation has shown that some defenses can significantly improve the robustness of a classifier. However, they each have their drawbacks which may prevent their widespread adoption in real-world applications.

Consider Table 5.9: The strongest defense, large-scale Adversarial Training, requires an immense computational budget during training. It causes a significant drop in accuracy, which likely needs to be addressed before this method can enter into widespread use. At the same time, "weaker" methods of Adversarial Training (Ensemble AT) do not increase robustness enough to be of interest.

Random transforms and ensembles can offer significant robustness without requiring additional training. However, they increase the computational cost during inference – requiring a single image to be processed at least 8 times in the case of Random transforms. This makes these defenses unwieldy for real-world use – such a high computational budget could instead be used to run larger and more accurate models.

Interestingly enough, the simple JPEG filter seems to offer an attractive cost-benefit ratio. It offers a small but noticeable gain in robustness while incurring no cost at training time and very little at inference time. While this robustness is indeed limited, it could be added to almost any computer vision system in a simple and straightforward manner.

Another method that might be suited for the real world is LLR: It offers comparable robustness to large-scale AT at a much lower training cost. The drop in accuracy is

## 5.4 Defenses against adversarial attack

DEFENSE	TRAINING COST	INFERENCE COST	ACCURACY	ROBUSTNESS
Majority vote ensemble	++ (free)	--	++	+
Random Transforms	++	--- (expensive)	+	++
TWIS detector	++	-	-	---
JPEG filter	++	+	-	+
Median filter	++	+	-	---
U-net denoiser	-	-	+	+
Ensemble AT	-	++ (free)	+	-
Large scale AT	--- (expensive)	++	---	+++
Feature Denoising AT	---	++	---	+++
LLR regularization	-	++	--	++

**Table 5.9:** Comparison of the positive ( + ) and negative ( - ) aspects of defense methods. While Adversarial Training approaches offer strong robustness, they are very expensive to train. However, once trained, they do not incur additional cost at inference time. This stands in contrast to Random Transforms and Ensembles, which do not require additional training, but are expensive at inference time.

also less pronounced. This makes it a very promising direction for future research into adversarial robustness.

All things considered, such research will definitely be required. In this evaluation, no defense managed to reduce attack success rate below 45% – the Biased Boundary Attack was still successful in almost half of the cases. This is certainly worrying and shows how far even the best defenses are from truly preventing adversarial examples.

## 5.5 Real-world commercial services

So far, no classifiers and no defenses have been found to be entirely robust against adversarial examples. It is only natural to assume that real-world applications of computer vision also share the same vulnerabilities.

While any system that uses computer vision could be attacked in theory, realistic evaluations can prove difficult and expensive. For example, in the attack on Tesla Autopilot, it was necessary to reverse engineer several system components before gaining reliable access to the prediction output [26]. While the attack was ultimately successful, such an investigation is beyond the scope of this work. Instead, this Section focuses on another type of real-world application that is both widely used and well-suited for adaptive adversarial attacks: commercial image classification in the cloud.

Services such as Amazon Rekognition or Google Cloud Vision are very popular and often used by third parties that outsource their own computer vision needs. In some cases, even law enforcement authorities have been known to directly use these services [125]. At the same time, the providers offer direct access to their APIs for only a moderate fee (roughly 1 US dollar per 1000 queries). This makes them perfectly suited for adaptive black-box attacks such as the one developed in this work.

### 5.5.1 Setup

While the attack developed in this work is reasonably query-efficient, it is not possible to evaluate the entire set of 200 images. After all, 200 images for 15000 queries would result in 3 million queries, costing several thousand US dollars. While this is much less than the cost incurred in prior work [42], it is still a considerable amount and therefore considered out of scope for this work.

Also, the exact class labels used by commercial providers are not known to the attacker. Depending on the classifier, it might be impossible to force the prediction of a specific label in the evaluation data set. In other words, a targeted attack might not have the desired effect against all classifiers. For this reason, only a single adversarial example is constructed that demonstrates the dangers of adversarial attack:

**Making pedestrians disappear.** An image containing pedestrians is manipulated so that the classifier does not see them anymore. While not a classic targeted attack, this setup is equally challenging, since it is not enough to remove the class label "pedestrians" from the list of predictions – all related concepts must be removed as well<sup>9</sup>. A random ImageNet image from the class "bald eagle" is chosen as starting point as it satisfies this condition.

Some of the APIs evaluated in this Section are rather slow, taking up to a second for a single prediction. For this reason, the number of queries is limited to 3000, making sure that no experiment takes more than an hour.

---

<sup>9</sup>The image is considered adversarial if none of these concepts are in the list of predictions: Pedestrians, people, person, human, woman, man, girl, boy, adult, child, walking, couple, crowd, group, head, face, legs, running, walking, clothing, footwear, apparel.



### 5.5.2 Microsoft Azure

The first classifier to be evaluated is the image tagging API provided by Microsoft Azure Cognitive Services<sup>10</sup>. It accepts an image and returns a number of tags with confidence scores – in other words, a list of top-k predictions.

Figure 5.8 shows the result, evaluated on the Microsoft website. (Top): The API reports a multitude of concepts that are related to humans. For the adversarial criterion to be fulfilled, none of these tags must appear in the prediction list. (Bottom): After 3000 iterations of the Biased Boundary Attack, the pedestrians have become effectively invisible to the computer vision service. At the same time, the perturbation is barely noticeable - a successful adversarial attack.



**Figure 5.8:** Making pedestrians disappear for Microsoft Azure. (Top): Original image. The list of predictions contains "person", "footwear", "clothing", "woman", "people", "legs", "girl". (Bottom): Adversarial example, obtained after 3000 queries with the Biased Boundary Attack. All labels that hint at pedestrians are gone from the prediction list.

*Note on reproducibility:* This experiment was performed in February 2021. It is likely that all cloud service providers regularly update their classifiers, which may cause this particular adversarial example to lose its effect in the future. In that case, the attack can be run again to produce a new image.

<sup>10</sup> Available at <https://azure.microsoft.com/en-us/services/cognitive-services/computer-vision/>

### 5.5.3 Amazon Rekognition

Next, the same attack is attempted on Amazon AWS, using the Rekognition API<sup>11</sup>. Figure 5.9 shows the result, evaluated on the AWS console website: Although the perturbation is not completely invisible after 3000 queries, it is nevertheless effective at fooling the classifier into ignoring the humans. Interestingly, the API seems to regard the colorful perturbation as features of a painting, completely disregarding any humans that might be present regardless.



**Figure 5.9:** Making pedestrians disappear for Amazon Rekognition. (Top): Original image. The list of predictions contains "pedestrian", "person" and "human". (Bottom): Adversarial example, obtained after 3000 queries with the Biased Boundary Attack. All labels that hint at pedestrians are gone from the prediction list. Although the perturbation is visible to the human eye, it still causes the service to completely disregard the humans present in the picture.

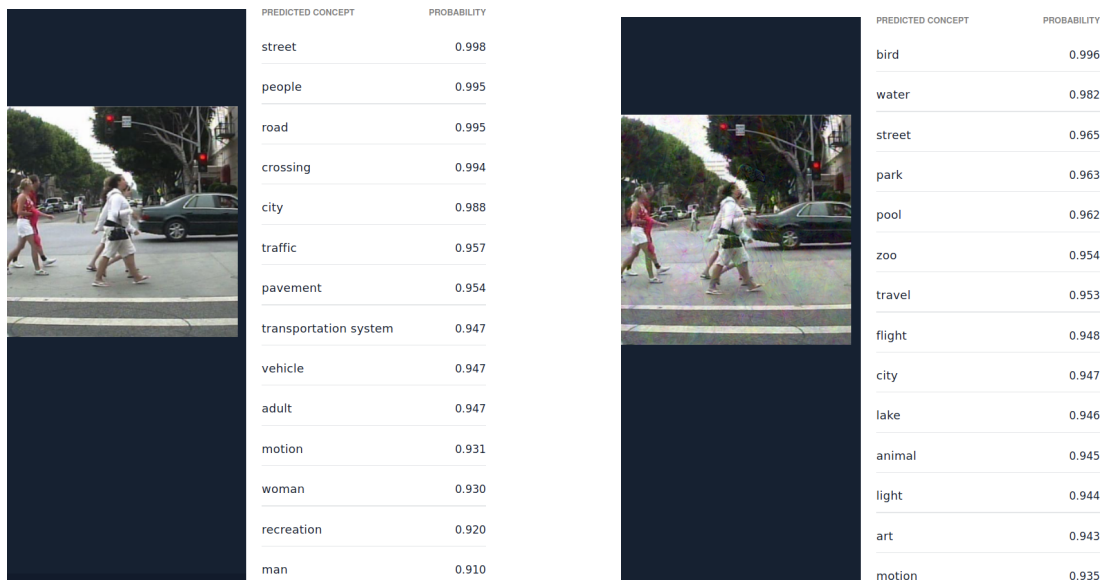
*Note on reproducibility:* This experiment was performed in February 2021. It is likely that all cloud service providers regularly update their classifiers, which may cause this particular adversarial example to lose its effect in the future. In that case, the attack can be run again to produce a new image.

<sup>11</sup> Available at <https://aws.amazon.com/rekognition/>

### 5.5.4 Clarifai

The attack is also performed on the Clarifai Image Recognition API<sup>12</sup>. Figure 5.10 shows a screenshot from the Clarifai website: The API fails to recognize the humans in the adversarially manipulated image.

In contrast to other services, Clarifai reports features related to birds (bird, zoo, animal), which are likely remnants from the starting image (bald eagle). In this way, the perturbation has a curious side effect: It removes humans and adds birds.



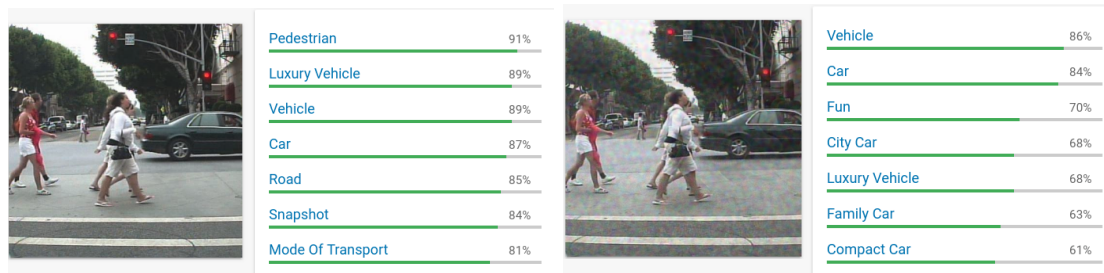
**Figure 5.10:** Making pedestrians disappear for Clarifai. (Left): Original image. The list of predictions contains "people", "adult", "woman", "man". (Right): Adversarial example, obtained after 3000 queries with the Biased Boundary Attack. All labels hinting at humans are gone (this includes additional labels further down in the list).

*Note on reproducibility:* This experiment was performed in February 2021. It is likely that all cloud service providers regularly update their classifiers, which may cause this particular adversarial example to lose its effect in the future. In that case, the attack can be run again to produce a new image.

<sup>12</sup> Available at <https://www.clarifai.com/models/general-image-recognition>

### 5.5.5 Google Cloud Vision

Finally, the attack is also successful against the Google Cloud Vision API<sup>13</sup>:



**Figure 5.11:** Making pedestrians disappear on Google Cloud Vision. (Left): Original image. The list of predictions contains "pedestrian". (Right): Adversarial example, obtained after 1000 queries with the Biased Boundary Attack. Perhaps interestingly, the label "pedestrian" was replaced by "fun". Images taken from [126].

*Possible conflict of interest:* The author of this thesis took up a position with Alphabet in 2020 and was therefore affiliated with Google at the time of writing. To prevent any conflict of interest, Figure 5.11 cites an older version of this experiment, which was published separately in 2019 [126]. The author had no knowledge of Google’s systems at that time.

*Note on reproducibility:* This experiment was conducted in January 2019. It used an older version of the Biased Boundary Attack and performed only 1000 queries. Also, Google Cloud Vision has likely undergone significant changes since that time. Therefore, this result cannot be directly compared to the other services.

## 5.6 Summary

The experiments in this Chapter have shown that virtually all image classifiers are vulnerable to adversarial examples in the evaluation setting. This includes not only state-of-the-art deep learning methods, but also traditional computer vision pipelines.

While there exist some defense methods that show promise, most fail completely against an adaptive attacker. But even the best defenses come with their own crippling weaknesses: The strongest methods reduce accuracy and drastically increase either training or inference cost.

Consequently, real-world commercial providers may be reluctant to adopt any sort of defense mechanism at all. The last experiment appears to support this: It is not difficult to create adversarial examples that cause potentially dangerous failures in popular commercial services.

<sup>13</sup>Available at <https://cloud.google.com/vision>

## 6 Conclusion

This chapter summarizes the main results of the thesis. Based on them, a number of recommendations for improving the robustness of real-world image processing systems are given. Finally, limitations of this work are discussed and promising directions for future work are identified.

### 6.1 Adversarial robustness is not easily achieved

The evaluations in this thesis have shown that all investigated classifiers are vulnerable to adversarial examples. The measured robustness of deep learning classifiers is not significantly impacted by architecture. Some traditional computer vision pipelines are found to be slightly more robust than others, but not enough to defend against a strong attack. Many defense approaches exist, but most offer only a marginal increase in robustness – if at all.

These findings are in line with prior works [16, 127] that have obtained similar results in carefully controlled white-box settings. It is worth noting that these studies were more of a theoretical nature since an attacker is unlikely to have white-box access in the real world. Going one step further, the results presented in this thesis confirm that the same vulnerabilities can easily be exploited in a more realistic black-box scenario. Even more, the efficient adaptive attack developed in Chapter 3 requires almost no knowledge about the defender and can be performed with only a minimal investment of time and computational resources.

### 6.2 Recommendations for real-world systems

These results indicate that adversarial attacks may be a realistic threat for any image processing system that admits a moderate amount of queries. This includes popular cloud-based services, such as those provided by Microsoft, Amazon, Clarifai and Google – the experiments in Section 5.5 of Chapter 5 have shown that these systems can currently be fooled in just 3000 queries. Therefore, it is reasonable to assume that, at the time of writing, they did not utilize very strong defenses if at all.

Judging from the results obtained in Chapter 5, it is possible to make the following recommendations:

- **No reason to avoid deep learning.** Adversarial examples are sometimes described as a problem introduced by and inherent to deep learning [101, 102]. However, the results in Section 5.3 of Chapter 5 provide evidence to the contrary: Traditional computer vision classifiers have similar vulnerabilities and are just as

## 6 Conclusion

easily exploited, while at the same time offering much lower accuracy. Therefore it is more promising to focus on deep learning and try to improve its robustness further.

- **Ensembling is useful, but expensive at inference time.** Randomization and ensembling offer tangible robustness benefits at little accuracy loss but are very expensive at inference time. As such, any hardware that performs inference must be scaled accordingly (often 8-fold and beyond). This is an investment that may often be prohibitive but still warranted in some cases.
- **Adversarial training is useful, but expensive at training time.** Large-scale adversarial training has yielded the highest increase in robustness. It incurs no additional cost at inference time but does so during training. However, this approach may still scale well for large services – a model needs to be trained only once and can then be distributed and applied on many devices and for many customers.

Adversarial training also struggles with accuracy. Addressing this, new developments such as the LLR regularizer seem very promising.

- **Low-hanging fruit should be used where possible.** The experiments in Section 5.4 of Chapter 5 have shown that some simple defenses like JPEG filtering can offer a small robustness increase at near-zero cost. Such defenses are easy to implement and could benefit almost any image processing system, even if that benefit is not very large.

### 6.3 Limitations and future work

This thesis has investigated black-box adversarial attacks against individual classifiers and defense mechanisms in a query-limited setting. Looking forward, this scenario could be extended in a number of ways:

- **Stronger alignment with human perception.** All attacks considered in this study minimize the  $\ell^2$ -distance between  $x_{adv}$  and  $x_{orig}$ . As outlined in Section 2.3 of Chapter 2, this only aligns with human perception when the distance is very low. However, this limits the choice of perturbations. For example, attackers may consider adversarial patches that are clearly visible but still fool humans [90] or perform spatial transformations that are known to reduce classification performance [128]. It is likely that attacks could be made even more effective by relaxing the  $\ell^p$  constraint.
- **Combination of defenses.** It is reasonable to expect that a combination of strong defenses would be even more robust. This has been seen in open competitions, where the strongest submissions combined randomization and filtering with ensembles of adversarially-trained models [17].

- **Intrusion detection.** The threat setting of this work has assumed that a defender will admit thousands of queries in short succession. However, attacks such as the one developed in Chapter 3 could easily be registered by a stateful detection system, as an attacker typically submits an unusual amount of images that are only slight variations of each other. See Chen et al. [129] for an early exploration of this topic.

Such intrusion detection systems could be a promising defense for real-world applications. In turn, attacks may try to circumvent them by creating fewer but stronger perturbations. This is a more general issue of computer security that is not specific to adversarial examples – it should be interesting to explore this connection further.

## 6.4 Summary of contributions

This thesis has studied the robustness of image classifiers against adaptive black-box adversarial attacks. In summary, the following contributions were made:

- A novel method for generating black-box adversarial examples was proposed that is both reliable and efficient. This Biased Boundary Attack can be performed on virtually any classifier and gains efficiency by incorporating prior knowledge about the problem domain. Three such sources of knowledge were discussed: image frequency, regional masking, and surrogate models.
- The attack method was experimentally evaluated and an ablation study of hyperparameters was performed. When compared with previous work, it was found that the Biased Boundary Attack significantly outperformed existing black-box attacks.
- The attack method was submitted to the NeurIPS 2018 Adversarial Vision Challenge, an open competition of adversarial attacks and defenses, where it won second place in the targeted attack track. This further underscores its performance.
- The attack method was used to perform a comprehensive evaluation of the robustness of deep learning architectures, showing that all known architectures are similarly vulnerable to adversarial examples.
- It was shown that traditional computer vision pipelines are also vulnerable in the evaluation setting. A select few configurations were found to be marginally more robust than deep learning architectures, however not enough to repel adversarial attacks in the majority of cases.
- A number of recently proposed defense mechanisms were evaluated. It was found that most of them do not offer significant robustness against adaptive attacks. Only two approaches, adversarial training and randomized ensembling, were found to be effective. Still, the attack was able to fool even the strongest defense in 45% of cases.

## 6 Conclusion

- Finally, it was demonstrated that real-world commercial image processing services from Microsoft, Amazon, Clarifai and Google can also be attacked quickly and effectively.

This work has shown that adaptive black-box attacks are powerful tools for evaluating the adversarial robustness of image classifiers. They also pose a potential threat for real-world systems as they can be performed quickly, without much knowledge of the system under attack, and do not require a significant amount of computational resources. Therefore, it is important that future image processing systems are tested against them. Adaptive black-box attacks may soon become commonplace in the wild – they should form an integral part of any serious analysis of adversarial robustness in the future.



# Bibliography

- [1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, ACM, 2017.
- [2] 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.
- [3] Nicholas Carlini and David Wagner. Audio adversarial examples: Targeted attacks on speech-to-text. In *2018 IEEE Security and Privacy Workshops, SPW*, pages 1–7. IEEE, 2018.
- [4] Robin Jia and Percy Liang. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031. ACL, 2017.
- [5] Vahid Behzadan and Arslan Munir. Vulnerability of deep reinforcement learning to policy induction attacks. In *International Conference on Machine Learning and Data Mining in Pattern Recognition, MLDM*, pages 262–275. Springer, 2017.
- [6] Jacques Bughin, Jeongmin Seong, James Manyika, Michael Chui, and Raoul Joshi. Notes from the AI frontier: Modeling the impact of AI on the world. *McKinsey Global Institute*, 2018.
- [7] IEEE Standard Glossary of Software Engineering Terminology. *IEEE Std 610.12-1990*, page 64, 1990.
- [8] Christopher Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer, 2006.
- [9] Kemin Zhou and John C. Doyle. *Essentials of Robust Control*. Prentice Hall, 1998.
- [10] Samuel F. Dodge and Lina J. Karam. A study and comparison of human and deep learning recognition performance under visual distortions. In *26th International Conference on Computer Communication and Networks, ICCCN 2017*, pages 1–7. IEEE, 2017.
- [11] Chih-Hong Cheng, Chung-Hao Huang, and Georg Nührenberg. nn-dependability-kit: Engineering neural networks for safety-critical autonomous driving systems. In *2019 IEEE/ACM International Conference on Computer-Aided Design, ICCAD*, pages 1–6, 2019.

## Bibliography

- [12] Pornntiwa Pawara, Emmanuel Okafor, Lambert Schomaker, and Marco Wiering. Data augmentation for plant classification. In *Advanced Concepts for Intelligent Vision Systems*, pages 615–626. Springer, 2017.
- [13] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202, 1980.
- [14] Sara Sabour, Nicholas Frosst, and Geoffrey E. Hinton. Dynamic routing between capsules. In *Advances in Neural Information Processing Systems, NIPS*, pages 3856–3866, 2017.
- [15] Wieland Brendel, Jonas Rauber, and Matthias Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. In *International Conference on Learning Representations, ICLR*, 2018.
- [16] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 274–283. PMLR, 2018.
- [17] Alexey Kurakin, Ian Goodfellow, Samy Bengio, Yinpeng Dong, Fangzhou Liao, Ming Liang, Tianyu Pang, Jun Zhu, Xiaolin Hu, Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, Alan Yuille, Sangxia Huang, Yao Zhao, Yuzhe Zhao, Zhonglin Han, Junjiajia Long, Yerkebulan Berdibekov, Takuya Akiba, Seiya Tokui, and Motoki Abe. Adversarial Attacks and Defences Competition. In *The NIPS '17 Competition: Building Intelligent Systems*, pages 195–231. Springer, Cham, 2018.
- [18] Wieland Brendel, Jonas Rauber, Alexey Kurakin, Nicolas Papernot, Behar Veliqi, Sharada P. Mohanty, Florian Laurent, Marcel Salathé, Matthias Bethge, Yaodong Yu, Hongyang Zhang, Susu Xu, Hongbao Zhang, Pengtao Xie, Eric P. Xing, Thomas Brunner, Frederik Diehl, Jérôme Rony, Luiz Gustavo Hafemann, Shuyi Cheng, Yinpeng Dong, Xuefei Ning, Wenshuo Li, and Yu Wang. Adversarial Vision Challenge. In *The NeurIPS '18 Competition*, pages 129–153. Springer, Cham, 2020.
- [19] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In *Advances in Neural Information Processing Systems, NeurIPS*, pages 125–136, 2019.
- [20] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations, ICLR*, 2015.

- [21] Jiajun Lu, Hussein Sibai, Evan Fabry, and David A. Forsyth. NO need to worry about adversarial examples in object detection in autonomous vehicles. *arXiv preprint arXiv:1707.03501*, 2017.
- [22] Sibel Yenikaya, Gökhan Yenikaya, and Ekrem Düven. Keeping the vehicle on the road: A survey on on-road lane detection systems. *ACM Computing Surveys*, 46(1):2:1–2:43, ACM, 2013.
- [23] Sorin Grigorescu, Bogdan Trasnea, Tiberiu Cocias, and Gigel Macesanu. A survey of deep learning techniques for autonomous driving. *Journal of Field Robotics*, pages 1–25, 2019.
- [24] Sacha Arnoud. The rise of machine learning in self-driving cars. In *6.S094: Deep Learning for Self-Driving Cars*. Massachusetts Institute of Technology, 2018. Available at <https://youtu.be/LSX3qdy0dFg?t=4308>, accessed 28 May 2021.
- [25] Peter P. Swire. A model for when disclosure helps security: What is different about computer and network security? *Journal on Telecommunications and High Technology Law*, 3(1):163–208, 2004.
- [26] Tencent Keen Security Lab. Experimental Security Research of Tesla Autopilot, 2019. Available at [https://keenlab.tencent.com/en/whitepapers/Experimental\\_Security\\_Research\\_of\\_Tesla\\_Autopilot.pdf](https://keenlab.tencent.com/en/whitepapers/Experimental_Security_Research_of_Tesla_Autopilot.pdf), accessed 28 May 2021.
- [27] 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 *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, AISec 2017, AISec '17*, pages 15–26. ACM, 2017.
- [28] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.
- [29] Trevor Hastie, Robert Tibshirani, and Jerome H. Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer, 2009.
- [30] Fernando I. Garcia. Introduction to support vector machines. *The OpenCV library: Machine Learning Tutorials*, 2018. Available at [https://docs.opencv.org/3.4/d1/d73/tutorial\\_introduction\\_to\\_svm.html](https://docs.opencv.org/3.4/d1/d73/tutorial_introduction_to_svm.html), accessed 28 May 2021.
- [31] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *International Conference on Learning Representations, ICLR, Workshop Track Proceedings*, 2017.
- [32] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, ASIA CCS '17*, pages 506–519. ACM, 2017.

## Bibliography

- [33] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *International Conference on Learning Representations, ICLR*, 2018.
- [34] Florian Tramèr, 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.
- [35] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 580–587. IEEE, 2014.
- [36] Di Feng, Christian Haase-Schütz, Lars Rosenbaum, Heinz Hertlein, Claudius Gläser, Fabian Timm, Werner Wiesbeck, and Klaus Dietmayer. Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges. *IEEE Transactions on Intelligent Transportation Systems, T-ITS*, pages 1–20, IEEE, 2020.
- [37] Eric Wong and J. Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *Proceedings of the 35th International Conference on Machine Learning, ICML*, volume 80 of *Proceedings of Machine Learning Research*, pages 5283–5292. PMLR, 2018.
- [38] Guy Katz, Clark W. Barrett, David L. Dill, Kyle Julian, and Mykel J. Kochenderfer. Reluplex: An efficient SMT solver for verifying deep neural networks. In *Computer Aided Verification - 29th International Conference, CAV*, volume 10426 of *Lecture Notes in Computer Science*, pages 97–117. Springer, 2017.
- [39] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *IEEE Symposium on Security and Privacy, SP*, pages 39–57. IEEE, 2017.
- [40] Nicholas Carlini and David A. Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, AISec@CCS*, pages 3–14. ACM, 2017.
- [41] Dongyu Meng and Hao Chen. MagNet: A two-pronged defense against adversarial examples. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS*, pages 135–147. ACM, 2017.
- [42] Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. Black-box adversarial attacks with limited queries and information. In *Proceedings of the 35th International Conference on Machine Learning, ICML*, volume 80 of *Proceedings of Machine Learning Research*, pages 2142–2151. PMLR, 2018.

- [43] J. C. Spall. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE Transactions on Automatic Control*, 37(3):332–341, IEEE, 1992.
- [44] Chun-Chen Tu, Pai-Shun Ting, Pin-Yu Chen, Sijia Liu, Huan Zhang, Jinfeng Yi, Cho-Jui Hsieh, and Shin-Ming Cheng. AutoZOOM: Autoencoder-based zeroth order optimization method for attacking black-box neural networks. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence, AAAI*, pages 742–749. AAAI Press, 2019.
- [45] Minhao Cheng, Thong Le, Pin-Yu Chen, Huan Zhang, Jinfeng Yi, and Cho-Jui Hsieh. Query-efficient hard-label black-box attack: An optimization-based approach. In *International Conference on Learning Representations, ICLR*, 2019.
- [46] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *AT&T Labs*, 2010. Available at <http://yann.lecun.com/exdb/mnist>, accessed 28 May 2021.
- [47] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the Inception architecture for computer vision. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 2818–2826. IEEE, 2016.
- [48] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 248–255. IEEE, 2009.
- [49] Chuan Guo, Jared S. Frank, and Kilian Q. Weinberger. Low frequency adversarial perturbation. In *Proceedings of the 35th Conference on Uncertainty in Artificial Intelligence, UAI*, volume 115 of *Proceedings of Machine Learning Research*, pages 1127–1137. AUAI Press, 2019.
- [50] Andrew Ilyas, Logan Engstrom, and Aleksander Madry. Prior convictions: Black-box adversarial attacks with bandits and priors. In *International Conference on Learning Representations, ICLR*, 2019.
- [51] Michael P. Eckert and Andrew P. Bradley. Perceptual Quality Metrics Applied to Still Image Compression. *Signal Processing*, 70(3):177–200, Elsevier, 1998.
- [52] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial machine learning at scale. In *International Conference on Learning Representations, ICLR*, 2017.
- [53] Jiawei Su, Danilo V. Vargas, and Kouichi Sakurai. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 23(5):828–841, IEEE, 2019.

## Bibliography

- [54] Zhou Wang, Alan. C. Bovik, Hamid. R. Sheikh, and Eero. P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, IEEE, 2004.
- [55] Hamid. R. Sheikh and Alan. C. Bovik. Image information and visual quality. *IEEE Transactions on Image Processing*, 15(2):430–444, IEEE, 2006.
- [56] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems, NIPS*, pages 6629–6640, 2017.
- [57] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems, NIPS*, pages 2672–2680, 2014.
- [58] Florian Tramèr, Nicolas Papernot, Ian J. Goodfellow, Dan Boneh, and Patrick D. McDaniel. The space of transferable adversarial examples. *arXiv preprint arXiv:1704.03453*, 2017.
- [59] Nilesh N. Dalvi, Pedro M. Domingos, Mausam, Sumit K. Sanghai, and Deepak Verma. Adversarial classification. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 99–108. ACM, 2004.
- [60] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *IEEE International Conference on Computer Vision, ICCV*, pages 1026–1034. IEEE, 2015.
- [61] Richard H. Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208, 1995.
- [62] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 1–9. IEEE, 2015.
- [63] Yurii Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Applied Optimization. Springer, 2004.
- [64] 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.

- [65] Nicolas Papernot, Patrick D. McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *IEEE Symposium on Security and Privacy, SP*, pages 582–597. IEEE, 2016.
- [66] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR*, 2015.
- [67] Nicholas Carlini, Anish Athalye, Nicolas Papernot, Wieland Brendel, Jonas Rauber, Dimitris Tsipras, Ian J. Goodfellow, Aleksander Madry, and Alexey Kurakin. On evaluating adversarial robustness. *arXiv preprint arXiv:1902.06705*, 2019.
- [68] Pin-Yu Chen, Yash Sharma, Huan Zhang, Jinfeng Yi, and Cho-Jui Hsieh. Ead: Elastic-net attacks to deep neural networks via adversarial examples. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, AAAI*, pages 10–17. AAAI Press, 2018.
- [69] Jérôme Rony, Luiz G. Hafemann, Luiz S. Oliveira, Ismail Ben Ayed, Robert Sabourin, and Eric Granger. Decoupling direction and norm for efficient gradient-based L2 adversarial attacks and defenses. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 4322–4330. IEEE, 2019.
- [70] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR*, 2015.
- [71] Dong Su, Huan Zhang, Hongge Chen, Jinfeng Yi, Pin-Yu Chen, and Yupeng Gao. Is robustness the cost of accuracy? - A comprehensive study on the robustness of 18 deep image classification models. In *Proceedings of the European Conference on Computer Vision, ECCV*, pages 644–661, 2018.
- [72] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 770–778. IEEE, 2016.
- [73] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.
- [74] 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*, 2017.
- [75] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 9185–9193. IEEE, 2018.

## Bibliography

- [76] Jonathan Uesato, Brendan O’Donoghue, Pushmeet Kohli, and Aäron van den Oord. Adversarial risk and the dangers of evaluating against weak attacks. In *Proceedings of the 35th International Conference on Machine Learning, ICML*, pages 5032–5041, 2018.
- [77] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Fei-Fei Li. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, Springer, 2015.
- [78] Warren He, James Wei, Xinyun Chen, Nicholas Carlini, and Dawn Song. Adversarial example defense: Ensembles of weak defenses are not strong. In *11th USENIX Workshop on Offensive Technologies, WOOT*, 2017.
- [79] Guneet S. Dhillon, Kamyar Azizzadenesheli, Zachary C. Lipton, Jeremy Bernstein, Jean Kossaifi, Aran Khanna, and Animashree Anandkumar. Stochastic activation pruning for robust adversarial defense. In *International Conference on Learning Representations, ICLR*, 2018.
- [80] Yarín Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33rd International Conference on Machine Learning, ICML*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 1050–1059, 2016.
- [81] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, and Alan Yuille. Mitigating adversarial effects through randomization. In *International Conference on Learning Representations, ICLR*, 2018.
- [82] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *Advances in Neural Information Processing Systems, NeurIPS*, pages 7167–7177, 2018.
- [83] Chuan Guo, Mayank Rana, Moustapha Cissé, and Laurens van der Maaten. Countering adversarial images using input transformations. In *International Conference on Learning Representations, ICLR*, 2018.
- [84] Fangzhou Liao, Ming Liang, Yinpeng Dong, Tianyu Pang, Xiaolin Hu, and Jun Zhu. Defense against adversarial attacks using high-level representation guided denoiser. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 1778–1787. IEEE, 2018.
- [85] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing robust adversarial examples. In *Proceedings of the 35th International Conference on Machine Learning, ICML*, volume 80 of *Proceedings of Machine Learning Research*, pages 284–293. PMLR, 2018.



- [86] Ken Perlin. An image synthesizer. In *Proceedings of the 12th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '85*, pages 287–296. ACM, 1985.
- [87] David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert Müller. How to explain individual classification decisions. *Journal of Machine Learning Research*, 11:1803–1831, 2010.
- [88] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *International Conference on Learning Representations, ICLR, Workshop Track Proceedings*, 2014.
- [89] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda B. Viégas, and Martin Wattenberg. SmoothGrad: Removing Noise by Adding Noise. *arXiv preprint arXiv:1706.03825*, 2017.
- [90] Tom B. Brown, Dandelion Mané, Aurko Roy, Martín Abadi, and Justin Gilmer. Adversarial patch. *arXiv preprint arXiv:1812.09803*, 2018.
- [91] Alex Krizhevsky. Learning multiple layers of features from tiny images. *Technical report*, University of Toronto, 2012.
- [92] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Ng. Reading digits in natural images with unsupervised feature learning. *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.
- [93] Muzammal Naseer, Salman H. Khan, Muhammad Haris Khan, Fahad Shahbaz Khan, and Fatih Porikli. Cross-domain transferability of adversarial perturbations. In *Advances in Neural Information Processing Systems, NeurIPS*, pages 12885–12895, 2019.
- [94] Christian Szegedy, Sergey Ioffe, and Vincent Vanhoucke. Inception-v4, Inception-ResNet and the impact of residual connections on learning. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, AAAI*, pages 4278–4284. AAAI Press, 2017.
- [95] Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual*. CreateSpace, 2009.
- [96] Charles R. Harris, K. Jarrod Millman, Stéfan van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585:357–362, 2020.

## Bibliography

- [97] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek Gordon Murray, Benoit Steiner, Paul A. Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation, OSDI*, pages 265–283, 2016.
- [98] Qizhe Xie, Minh-Thang Luong, Eduard H. Hovy, and Quoc V. Le. Self-training with noisy student improves imagenet classification. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 10684–10695. IEEE, 2020.
- [99] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations, ICLR*, 2021.
- [100] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 2261–2269. IEEE, 2017.
- [101] Douglas Heaven. Why deep-learning AIs are so easy to fool. *Nature*, 574:163–166, 2019.
- [102] Gary Marcus. Deep learning: A critical appraisal. *arXiv preprint arXiv:1801.00631*, 2018.
- [103] David G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the International Conference on Computer Vision, ICCV*, pages 1150–1157. IEEE, 1999.
- [104] Thomas Moranduzzo and Farid Melgani. A SIFT-SVM method for detecting cars in UAV images. In *International Geoscience and Remote Sensing Symposium, IGARSS*, pages 6868–6871, 2012.
- [105] A. Kumar. Computer-vision-based fabric defect detection: A survey. *IEEE Transactions on Industrial Electronics*, 55(1):348–363, IEEE, 2008.
- [106] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.
- [107] Jeremy Howard. imagenette. Available at <https://github.com/fastai/imagenette/>, accessed 28 May 2021.
- [108] G. Bradski. The OpenCV Library. *Dr. Dobb’s Journal of Software Tools*, 2000.

- [109] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR*, pages 886–893. IEEE, 2005.
- [110] Stéfan van der Walt, Johannes L. Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D. Warner, Neil Yager, Emmanuelle Gouillart, Tony Yu, and the scikit-image contributors. Scikit-image: image processing in Python. *PeerJ*, 2:e453, 2014.
- [111] Josef Sivic and Andrew Zisserman. Video Google: A text retrieval approach to object matching in videos. In *IEEE International Conference on Computer Vision, ICCV*, pages 1470–1477. IEEE, 2003.
- [112] Florent Perronnin and Christopher R. Dance. Fisher kernels on visual vocabularies for image categorization. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*. IEEE, 2007.
- [113] Jonas Rothfuss. The fishervector library. Available at <https://github.com/jonasrothfuss/fishervector/>, accessed 28 May 2021.
- [114] John A. Nelder and Robert W. M. Wedderburn. Generalized Linear Models. *Journal of the Royal Statistical Society. Series A (General)*, 135(3):370–384, Wiley, 1972.
- [115] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- [116] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, Academic Press, Inc., 1997.
- [117] Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-GAN: Protecting classifiers against adversarial attacks using generative models. In *International Conference on Learning Representations, ICLR*, 2018.
- [118] Lukas Schott, Jonas Rauber, Matthias Bethge, and Wieland Brendel. Towards the first adversarially robust neural network model on MNIST. In *International Conference on Learning Representations, ICLR*, 2019.
- [119] Kevin Roth, Yannic Kilcher, and Thomas Hofmann. The odds are odd: A statistical test for detecting adversarial examples. In *Proceedings of the 36th International Conference on Machine Learning, ICML*, pages 5498–5507, 2019.
- [120] 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.

## Bibliography

- [121] Shengyuan Hu, Tao Yu, Chuan Guo, Wei-Lun Chao, and Kilian Q. Weinberger. A new defense against adversarial images: Turning a weakness into a strength. In *Advances in Neural Information Processing Systems, NeurIPS*, pages 1633–1644, 2019.
- [122] Gintare Karolina Dziugaite, Zoubin Ghahramani, and Daniel M. Roy. A study of the effect of JPG compression on adversarial images. *arXiv preprint arXiv:1608.00853*, 2016.
- [123] 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*, pages 501–509. IEEE, 2019.
- [124] Chongli Qin, James Martens, Sven Gowal, Dilip Krishnan, Krishnamurthy Dvijotham, Alhussein Fawzi, Soham De, Robert Stanforth, and Pushmeet Kohli. Adversarial robustness through local linearization. In *Advances in Neural Information Processing Systems, NeurIPS*, pages 13824–13833, 2019.
- [125] Russell Bandom. Amazon is selling police departments a real-time facial recognition system. *The Verge*, 2018. Available at <https://www.theverge.com/2018/5/22/17379968/amazon-rekognition-facial-recognition-surveillance-aclu>, accessed 28 May 2021.
- [126] Thomas Brunner, Frederik Diehl, Michael Truong Le, and Alois C. Knoll. Guessing smart: Biased sampling for efficient black-box adversarial attacks. In *IEEE International Conference on Computer Vision, ICCV*, pages 4957–4965. IEEE, 2019.
- [127] Florian Tramèr, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. In *Advances in Neural Information Processing Systems, NeurIPS*, volume 33, pages 1633–1645, 2020.
- [128] Logan Engstrom, Brandon Tran, Dimitris Tsipras, Ludwig Schmidt, and Aleksander Madry. Exploring the landscape of spatial robustness. In *Proceedings of the 36th International Conference on Machine Learning, ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 1802–1811. PMLR, 2019.
- [129] Steven Chen, Nicholas Carlini, and David Wagner. Stateful detection of black-box adversarial attacks. In *Proceedings of the 1st ACM Workshop on Security and Privacy on Artificial Intelligence, SPAI*, page 30–39. ACM, 2020.

# A Listing of evaluation images

As described in Section 4.1.2 of Chapter 4, all attacks are evaluated on a data set that is formed by pseudo-randomly selecting 200 images from the ImageNet validation set. This Appendix lists all image IDs, together with their ground truth and adversarial target labels.

#	ImageNet validation set ID	Correct class label ( $y_{gt}$ )	Adversarial class label ( $y_{target}$ )
1	11841	chime (494)	ocarina (684)
2	19602	papillon (157)	folding chair (559)
3	45519	goose (99)	lipstick (629)
4	25747	tow truck (864)	cairn (192)
5	42642	toilet tissue (999)	sundial (835)
6	31902	ladybug (301)	revolver (763)
7	30346	steel arch bridge (821)	pay-phone (707)
8	12363	Scotch terrier (199)	black-footed ferret (359)
9	32490	crib (520)	ostrich (9)
10	26128	balance beam (416)	pinwheel (723)
11	14227	cardigan (474)	red fox (277)
12	26376	tank (847)	radio (754)
13	44173	violin (889)	soap dispenser (804)
14	12968	spatula (813)	honeycomb (599)
15	32104	car mirror (475)	harvestman (70)
16	17844	cockroach (314)	canoe (472)
17	43460	notebook (681)	hook (600)
18	8369	vestment (887)	lionfish (396)
19	15055	long-horned beetle (303)	cockroach (314)
20	6338	bittern (133)	passenger car (705)
21	15301	confectionery (509)	cello (486)
22	46250	coucal (91)	face powder (551)
23	45580	water ouzel (20)	African grey (87)
24	24647	cannon (471)	Norwegian elkhound (174)
25	46712	loupe (633)	hook (600)
26	4150	black and gold garden spider (72)	teapot (849)
27	42460	cannon (471)	nail (677)
28	29079	bullet train (466)	dogsled (537)
29	19412	coral fungus (991)	syringe (845)
30	34839	mud turtle (35)	black and gold garden spider (72)
31	34478	buckle (464)	scabbard (777)
32	16956	bathing cap (433)	web site (916)
33	29342	manhole cover (640)	sea slug (115)
34	26351	Model T (661)	promontory (976)
35	271	bannister (421)	radio telescope (755)
36	24189	otter (360)	pencil box (709)
37	1399	gong (577)	tank (847)
38	5140	overskirt (689)	bassinet (431)
39	12649	reflex camera (759)	birdhouse (448)
40	19883	toucan (96)	teddy (850)
41	36238	pole (733)	goose (99)
42	34129	brown bear (294)	rapeseed (984)
43	39915	piggy bank (719)	Scottish deerhound (177)
44	39723	ptarmigan (81)	radio telescope (755)
45	7346	planetarium (727)	sleeping bag (797)
46	8346	magpie (18)	mixing bowl (659)
47	38676	lawn mower (621)	grey whale (147)
48	21388	keeshond (261)	wooden spoon (910)
49	2838	dough (961)	barber chair (423)
50	48141	bee eater (92)	leopard (288)
51	1984	fiddler crab (120)	dough (961)
52	22320	sea slug (115)	toy poodle (265)
53	1083	jay (17)	pajama (697)
54	30641	Blenheim spaniel (156)	maillot (639)
55	27739	scuba diver (983)	Dutch oven (544)
56	4622	keeshond (261)	dumbbell (543)
57	30351	tree frog (31)	pick (714)
58	40136	Model T (661)	Tibetan mastiff (244)
59	28428	papillon (157)	Chihuahua (151)
60	45494	Greater Swiss Mountain dog (238)	moving van (675)
61	7187	titi (380)	container ship (510)
62	10204	binoculars (447)	brassiere (459)

A Listing of evaluation images

63	26750	abacus (398)	vacuum (882)
64	44715	bloodhound (163)	Kerry blue terrier (183)
65	42944	yurt (915)	spotted salamander (28)
66	21015	English springer (217)	snowmobile (802)
67	33753	bow (456)	black stork (128)
68	913	tile roof (858)	black stork (128)
69	45359	perfume (711)	pretzel (932)
70	33325	mortar (666)	ringneck snake (53)
71	16670	pop bottle (737)	whiskey jug (901)
72	7496	Lhasa (204)	espresso maker (550)
73	49735	butcher shop (467)	chain (488)
74	2123	fig (952)	rain barrel (756)
75	39717	bagel (931)	dingo (273)
76	12167	mashed potato (935)	fox squirrel (335)
77	32266	African chameleon (47)	giant panda (388)
78	28900	Mexican hairless (268)	lab coat (617)
79	16788	pill bottle (720)	agama (42)
80	16537	goldfinch (11)	bell cote (442)
81	37875	sulphur butterfly (325)	dumbbell (543)
82	10632	wombat (106)	viaduct (888)
83	12186	albatross (146)	Great Pyrenees (257)
84	32712	mouse (673)	admiral (321)
85	2329	Great Dane (246)	toilet tissue (999)
86	4937	briard (226)	broccoli (937)
87	46058	wombat (106)	garter snake (57)
88	46458	water bottle (898)	lion (291)
89	9406	coral fungus (991)	tricycle (870)
90	23165	butcher shop (467)	rock crab (119)
91	42025	little blue heron (131)	school bus (779)
92	28518	mongoose (298)	basketball (430)
93	43515	miniature poodle (266)	ruffed grouse (82)
94	15009	gibbon (368)	coucal (91)
95	24457	rhinoceros beetle (306)	washbasin (896)
96	7229	box turtle (37)	abacus (398)
97	33733	alligator lizard (44)	jigsaw puzzle (611)
98	10998	brambling (10)	freight car (565)
99	13443	pinwheel (723)	wing (908)
100	43682	shovel (792)	loupe (633)
101	30014	jean (608)	cauliflower (938)
102	41806	printer (742)	peacock (84)
103	31462	balloon (417)	West Highland white terrier (203)
104	27468	piggy bank (719)	cabbage butterfly (324)
105	37803	basset (161)	sandal (774)
106	45114	water tower (900)	potpie (964)
107	25041	sax (776)	African chameleon (47)
108	29179	sandal (774)	maillot (639)
109	47356	ski mask (796)	little blue heron (131)
110	4006	carton (478)	cliff (972)
111	38324	lemon (951)	tray (868)
112	47427	poncho (735)	American Staffordshire terrier (180)
113	46960	water tower (900)	table lamp (846)
114	48059	sundial (835)	oystercatcher (143)
115	39118	notebook (681)	mobile home (660)
116	41256	stupa (832)	kelpie (227)
117	28556	acorn (988)	banana (954)
118	20056	maillot (639)	shopping cart (791)
119	21100	car mirror (475)	piggy bank (719)
120	30695	redbone (168)	wok (909)
121	30995	harp (594)	macaque (373)
122	3336	siamang (369)	thatch (853)
123	5665	hair slide (584)	football helmet (560)
124	28316	tarantula (76)	dung beetle (305)
125	12359	megalith (649)	grille (581)
126	44501	guillotine (583)	borzoi (169)
127	28716	coil (506)	moving van (675)
128	27316	radio telescope (755)	birdhouse (448)
129	41791	hamster (333)	jacamar (95)
130	38553	otterhound (175)	giant schnauzer (197)
131	32942	limpkin (135)	iron (606)
132	6101	cleaver (499)	Newfoundland (256)
133	36811	chickadee (19)	upright (881)
134	22326	cheetah (293)	oxcart (690)
135	46474	street sign (919)	tiger (292)
136	29590	Italian greyhound (171)	French loaf (930)
137	43079	bolete (997)	spindle (816)
138	5944	chambered nautilus (117)	toilet seat (861)
139	7107	water buffalo (346)	lesser panda (387)
140	20114	tick (78)	jersey (610)
141	29473	safety pin (772)	fireboat (554)
142	30098	mushroom (947)	coral reef (973)
143	49646	scoreboard (781)	gibbon (368)
144	22468	loggerhead (33)	toilet tissue (999)
145	44302	barrel (427)	comic book (917)
146	13919	passenger car (705)	silky terrier (201)

147	31275	feather boa (552)	Madagascar cat (383)
148	14991	gasmask (570)	corkscrew (512)
149	36373	street sign (919)	Windsor tie (906)
150	4130	velvet (885)	guenon (370)
151	5093	garbage truck (569)	fire engine (555)
152	16406	crib (520)	banana (954)
153	49529	hotdog (934)	Madagascar cat (383)
154	37287	oystercatcher (143)	vulture (23)
155	4775	Ibizan hound (173)	panpipe (699)
156	47529	chain (488)	flamingo (130)
157	12542	paddlewheel (694)	marmoset (377)
158	44970	Cardigan (264)	red-breasted merganser (98)
159	37492	space heater (811)	golf ball (574)
160	19139	pole (733)	bagel (931)
161	4727	cardoon (946)	police van (734)
162	15843	switch (844)	spiny lobster (123)
163	30943	slot (800)	pizza (963)
164	12367	knot (616)	harp (594)
165	5920	mushroom (947)	butternut squash (942)
166	31572	grasshopper (311)	potter's wheel (739)
167	37764	hermit crab (125)	killer whale (148)
168	25744	snowplow (803)	Chesapeake Bay retriever (209)
169	44605	daisy (985)	fountain (562)
170	34708	parking meter (704)	apron (411)
171	37787	bannister (421)	screen (782)
172	10551	red fox (277)	whiptail (41)
173	34194	thresher (856)	water snake (58)
174	5435	strainer (828)	passenger car (705)
175	37603	cash machine (480)	terrarin (36)
176	35259	Dandie Dinmont (194)	scale (778)
177	3985	hourglass (604)	partridge (86)
178	18564	harmonica (593)	frilled lizard (43)
179	20447	microwave (651)	tripod (872)
180	8157	violin (889)	goldfinch (11)
181	32696	barbell (422)	running shoe (770)
182	22887	Arctic fox (279)	weevil (307)
183	41603	shower cap (793)	black grouse (80)
184	34464	dalmatian (251)	tailed frog (32)
185	20208	hard disc (592)	Border terrier (182)
186	22466	cello (486)	black stork (128)
187	6716	common iguana (39)	sock (806)
188	6776	sundial (835)	African hunting dog (275)
189	13858	maillot (638)	Norwegian elkhound (174)
190	42172	sea urchin (328)	fireboat (554)
191	48563	snowmobile (802)	patas (371)
192	44596	viaduct (888)	Irish terrier (184)
193	42281	gondola (576)	bicycle-built-for-two (444)
194	3598	steam locomotive (820)	chain (488)
195	44275	hen (8)	hand blower (589)
196	26312	Norfolk terrier (185)	cougar (286)
197	41520	leopard (288)	grey fox (280)
198	29952	remote control (761)	mailbox (637)
199	12468	espresso (967)	running shoe (770)
200	3234	can opener (473)	cowboy hat (515)

Table A.1: The evaluation data set used in Chapters 4 and 5.

## B NeurIPS 2018 Adversarial Vision Challenge

When evaluating adversarial attacks and defenses, it is hard to obtain meaningful results. Very often, attacks are tested against weak defenses and vice versa, and results are cherry-picked. To sidestep this issue, an implementation of the efficient attack developed in this work was submitted to the NeurIPS 2018 Adversarial Vision Challenge (AVC) [18], where it was pitted against state-of-the-art robust models and defenses and won second place in the targeted attack track.

Details of the implementation have been described in the competition report [18], as well as in a separate publication on the Biased Boundary Attack [126] that was done as part of this work. This Appendix contains a summary of these results.

**Evaluation setting.** The AVC was an open competition between image classifiers and adversarial attacks in an iterative black-box decision-based setting. Participants could choose between three tracks:

- **Robust model:** The submitted code is a robust image classifier. The goal is to maximize the  $\ell^2$  norm of any successful adversarial perturbation.
- **Untargeted attack:** The submitted code must find a perturbation that changes classifier output, while minimizing the  $\ell^2$  distance to the original image.
- **Targeted attack:** Same as above, but the classification must be changed to a specific label.

For several months, attacks were continuously evaluated against the most robust models and vice versa. Each evaluation consisted of 200 images with a resolution of 64x64 – a task not unlike, but less complex than the evaluation in Chapters 4 and 5 – and the attacker was allowed to query the model 1000 times for each image. The final attack score was then determined by the median  $\ell^2$  norm of the perturbation over all 200 images and top-5 models (lower is better).

**Competitors.** Seeing as more than 60 teams competed in the challenge, it is reasonable to assume that the top-5 robust models largely depicted the state of the art in adversarial robustness. Most winning models used variations of adversarial training in combination with ensembling, while other types of defenses such as denoisers were notably absent. On the attack side, most winners employed PGD transfer attacks with surrogate ensembles.

**Dataset.** Classifiers were trained on the Tiny ImageNet dataset, which is a down-scaled version of the ImageNet classification dataset, limited to 200 classes with 500 images each. Model input consisted of color images with 64x64 pixels, with the output



## B.1 Random guessing with low frequency

DISTRIBUTION	MEDIAN $\ell^2$	BOUNDARY ATTACK BIAS	MEDIAN $\ell^2$
Normal	11.15	None	20.2
<b>Perlin noise</b>	<b>4.28</b>	LF	15.1
		<b>LF + Surrogate gradients</b>	<b>9.5</b>

**Table B.1:** Random guessing attack with low frequency (untargeted), evaluated against the top-5 robust models in the AVC.

**Table B.2:** Biases for the Boundary Attack (targeted), evaluated against the top-5 robust models in the AVC.

being one of 200 class labels. The evaluation was conducted on a secret hold-out set of images, which was not contained in the original dataset and was unknown to participants of the challenge.

## B.1 Random guessing with low frequency

Before implementing the biased Boundary Attack, a simple experiment was conducted to demonstrate the effectiveness of Perlin noise patterns against typical defenses. Specifically, a random-guessing attack was entered into the competition, sampling candidates uniformly from the surface of a  $\ell^2$ -hypersphere with radius  $\epsilon$  around the original image:

$$r \sim \mathcal{N}(0, 1)^k; x_{adv} = x_{orig} + \epsilon \cdot \frac{r}{\|r\|_2} \quad (\text{B.1})$$

With a total budget of 1000 queries to the model for each image, binary search was used to find the lowest radius  $\epsilon$  at which an adversarial example can be found. Since this naive approach might struggle in a targeted setting, the experiment was limited to the untargeted attack track, where the probability of randomly sampling *any of 199* adversarial labels is reasonably high.

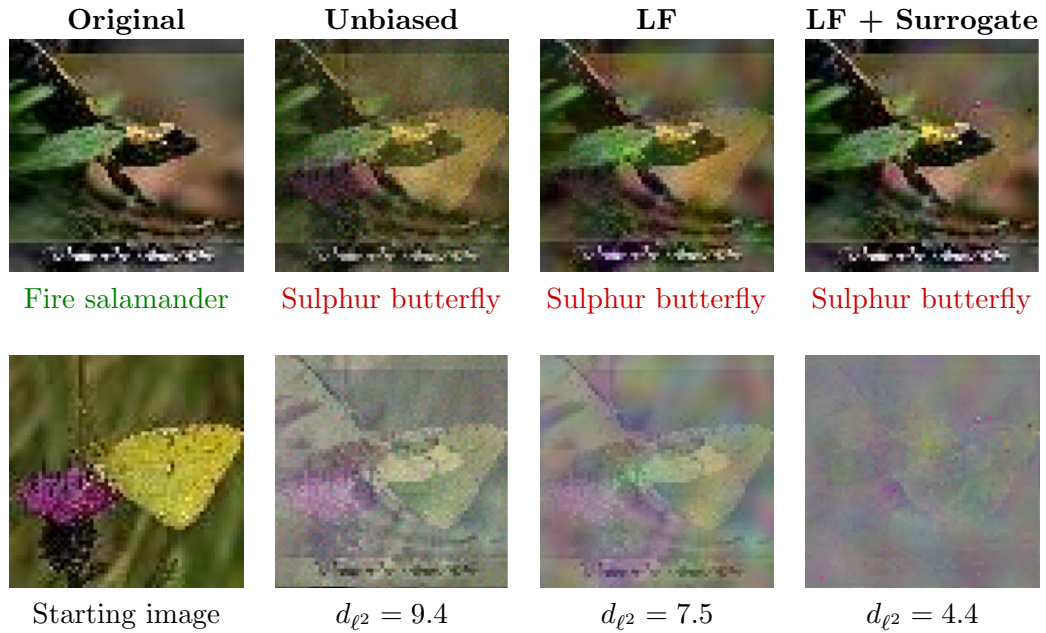
Next, the distribution was replaced with normalized Perlin noise (see Chapter 3 for notation):

$$r \sim \text{Perlin}_{64,64}(v); x_{adv} = x_{orig} + \epsilon \cdot \frac{r}{\|r\|_2} \quad (\text{B.2})$$

The Perlin frequency was set to 5 for all attacks on Tiny ImageNet. As Table B.1 shows, Perlin patterns were more efficient and the attack found adversarial perturbations with much lower distance (63% reduction). Although intended as a dummy submission to the competition, this attack was already strong enough for a top-10 placement in the untargeted track.

## B.2 Biased Boundary Attack

Next, the Biased Boundary Attack was evaluated in its intended setting, the targeted attack track in the AVC. To provide a point of reference, first the original Boundary



**Figure B.1:** Adversarial examples generated with different biases in the targeted attack submission to the AVC. All images were obtained after 1000 queries. The isolated perturbation is shown below each adversarial example.

Attack was implemented without biases. The attack worked, but was too inefficient to beat the competition. Compare Figure B.1, where the starting point is still clearly visible after 1000 iterations (in the unbiased case).

**Low-frequency bias.** As before, the distribution from which the attack samples the orthogonal step was replaced with Perlin patterns. Table B.2 shows that this alone decreased the median  $\ell^2$  distance by 25%.

**Surrogate gradient bias.** Next, projected gradients from a surrogate model were added. The bias strength  $w$  was set to 0.5. This further reduced the median  $\ell^2$  distance by another 37%, or a total of 53% when compared with the original Boundary Attack. Consider Figure B.1: 1000 iterations were enough to make the butterfly almost invisible to the human eye.

Interestingly, the attack worked well even though  $w$  was much higher than in the ImageNet evaluation in Chapter 4. During the competition there was not enough time to perform extensive ablations on hyperparameters – it is possible that a lower  $w$  would have yielded an even better result. Alternatively, transferability might have been better for Tiny ImageNet, which is a significantly easier task than ImageNet.

The submission to the AVC used an ensemble of ResNet-18 and ResNet-50 models, both of which were pre-trained baselines in the competition. While this may count as a strong surrogate, it was significantly weaker than the ones used by other winning AVC attack submissions, most of which much larger ensembles of carefully-trained models

[18]. Nevertheless, the Biased Boundary Attack outperformed most of them, yielding further evidence that it can exploit surrogate models more effectively than direct transfer attacks.

**Mask Bias.** The submission to the AVC did not use the mask bias, which was still in early development at that time. Judging from the results in Chapter 4, it could have further boosted the attack for an even better result.